Modélisation et simulation d'une architecture d'entreprise - Application aux Smart Grids

Rachida Seghiri

NNT : 2016SACLC053

THESE DE DOCTORAT
DE
L'UNIVERSITE PARIS-SACLAY
PREPAREE A
"CENTRALESUPELEC"

ECOLE DOCTORALE N° 580
Sciences et Technologies de l'Information et de la Communication

Informatique

Par

Rachida SEGHIRI

Modélisation et simulation d'une architecture d'entreprise
Application aux Smart Grids

Thèse présentée et soutenue à Gif-sur-Yvette, le 4 juillet 2016 :

Composition du Jury :

M, Nicolat Sabouret	Professeur à l'université Paris-Sud	Président
M, Yamine Aït-Ameur	Professeur à l'ENSEEIHT	Rapporteur
Mme, Samira Si-said Cherfi	Maître de conférences (HdR) au CNAM	Rapporteur
M, Imed Boughzala	Professeur à Télecom École de Management	Examinateur
M, Bruno Traverson	Ingénieur de recherche à EDF R&D	Examinateur
M, Frédéric Boulanger	Professeur à CentraleSupélec	Directeur de thèse
Mme, Claire Lecocq	Enseignante-chercheuse à Télécom SudParis	Encadrante
Mme, Anne Picault	Ingénieur de recherche à EDF R&D	Invitée

Remerciements

J'adresse mes vifs remerciements :

À mon directeur de thèse Frédéric Boulanger, professeur à CentraleSupélec, qui n'a eu de cesse de me faire bénéficier de sa grande culture scientifique, rehaussée de remarquables qualités humaines. Sa rigueur et son exigence scientifique n'excluent guère sa disponibilité et son écoute bienveillante.

À mes encadrants Claire Lecocq, enseignante-chercheuse à Télécom SudParis, et Vincent Godefroy, ingénieur-chercheur à EDF R&D, qui m'ont guidée et conseillée au quotidien. Claire m'a fait découvrir avec passion le domaine de la modélisation des Systèmes d'Information et a su m'enthousiasmer pour les problématiques des SI d'entreprise. Vincent a su faciliter mon intégration au groupe Interopérabilité des Application Métier (IAM) du département MIRE d'EDF R&D.

À Samira Si-said Cherfi, maître de conférences (HdR) au CNAM, et Yamine Aït-Ameur, professeur à l'ENSEEIHT, d'avoir accepté la lourde tâche de rapporter ce travail. Leur lecture attentive et leurs observations judicieuses ont permis de jeter un nouvel éclairage sur le contenu des ces travaux.

À Nicolas Sabouret, professeur à l'université Paris-Sud, Imed Boughzala, professeur à Télécom École de Management et Bruno Traverson, ingénieur-chercheur à EDF R&D, d'avoir accepté de faire partie du jury de cette thèse.

À Anne Picault, chef du groupe IAM du département MIRE de EDF R&D, de m'avoir permis de réaliser ces travaux de thèse dans d'excellentes conditions.

À Éric Suignard, Jean-Philippe Tavela et Charles Tan, ingénieurs-chercheurs du département MIRE, pour nos nombreux échanges sur les Smart Grids, et les réseaux électriques en général, dans le cadre du projet POMME.

À tous les membres du groupe IAM du département MIRE pour leur accueil chaleureux. Merci en particulier à Daniela Batista de sa constante bonne humeur et d'avoir facilité mes démarches administratives et rendu ainsi ces années de thèse d'autant plus agréables.

université PARIS-SACLAY | ÉCOLE DOCTORALE Sciences et technologies de l'information et de la communication (STIC)

Titre : Modélisation et simulation d'une architecture d'entreprise : application aux Smart Grids

Mots clés : Architecture d'Entreprise, Système d'Information, modélisation, simulation, Smart Grids

Résumé : Les Smart Grids sont des réseaux électriques intelligents permettant d'optimiser la production, la distribution et la consommation d'électricité grâce à l'introduction des technologies de l'information et de la communication sur le réseau électrique. Les Smart Grids impactent fortement l'ensemble de l'architecture d'entreprise des gestionnaires de réseaux électriques. Simuler une architecture d'entreprise permet aux acteurs concernés d'anticiper de tels impacts.
Dès lors, l'objectif de cette thèse est de fournir des modèles, méthodes et outils permettant de modéliser puis de simuler une architecture d'entreprise afin de la critiquer ou de la valider. Dans ce contexte, nous proposons un framework multi-vues, nommé ExecuteEA, pour faciliter la modélisation des architectures d'entreprise en automatisant l'analyse de leurs structures et de leurs comportements par la simulation. ExecuteEA traite chacune des vues métier, fonctionnelle et applicative selon trois aspects : informations, processus et objectifs. Pour répondre au besoin d'alignement métier/IT, nous introduisons une vue supplémentaire : la vue intégration. Dans cette vue nous proposons de modéliser les liens de cohérence inter et intra vues.
Nous mettons, par ailleurs, à profit des techniques issues de l'ingénierie dirigée par les modèles en tant que techniques support pour la modélisation et la simulation d'une architecture d'entreprise. Notre validons ensuite notre proposition à travers un cas métier Smart Grid relatif à la gestion d'une flotte de véhicules électriques.

Title : Modeling and simulation of Enterprise Architecture : application to Smart Grids

Keywords : Enterprise Architecture, Information System, modeling, simulation, Smart Grids

Abstract : In this thesis, we propose a framework that facilitates modeling Enterprise Architectures (EA) by automating analysis, prediction, and simulation, in order to address the key issue of business/IT alignment.
We present our approach in the context of Smart Grids, which are power grids enabled with Information and Communication Technologies. Extensive studies try to foresee the impact of Smart Grids on electric components, telecommunication infrastructure, and industrial automation and IT.
However, Smart Grids also have an impact on the overall EA of grids operators. Therefore, our framework enables stakeholders to validate and criticize their modeling choices for the EA in the context of Smart Grids. What we propose is a multi-view framework with three aspects – information, processes, and goals – for each view. In addition to the business, functional and application views, we add an integration view to ensure inter and intra-view consistency. We rely on Model Driven Engineering (MDE) techniques to ease the holistic modeling and simulation of enterprise systems. Finally, we show the utility of our approach by applying it on a Smart Grid case study: the management of an electric vehicles fleet.

Table des matières

Table des figures

1. Réseau Transport France (http ://www.rte-france.com/)

Liste des tableaux

Chapitre 1

Introduction

1.1 Contexte industriel

Les travaux de cette thèse s'inscrivent dans le contexte industriel des Smart Grids. Dès lors, nous commençons ce manuscrit par la présentation de ce paradigme émergent et des défis qu'il représente pour les entreprises fournissant et distribuant de l'électricité.

1.1.1 Qu'est ce qu'un Smart Grid ?

Le terme « Smart Grid » est une appellation générale désignant les technologies « intelligentes » qui « augmentent » [1] les réseaux électriques actuels en améliorant leurs performances.

Cette « augmentation » peut servir différents objectifs dépendant des limites du système électrique existant, du cadre de régulation, ainsi que de l'orientation politique des états. Il existe autant de définitions du terme « Smart Grid » que d'objectifs motivant son implémentation.

En Europe, les exigences de régulation et les volontés politiques des états membres en matière d'écologie favorisent l'intégration des énergies renouvelables et la participation active des consommateurs. Les Smart Grids européens sont donc synonymes de producteurs d'énergie renouvelables et de compteurs intelligents. Dès lors, la définition que donne *European Technology Platform Smart Grids* (ETP) des Smart Grids est la suivante :

Définition 1. *Les Smart Grids sont des réseaux électriques qui intègrent de manière intelligente les comportements et les actions de tous les acteurs connectés — les producteurs,*

1. À la manière de « l'augmentation de l'humain » qui désigne l'amélioration technique des performances humaines, aussi bien physiques, intellectuelles qu'émotionnelles [2013-9].

les consommateurs et ceux qui consomment et produisent en même temps — pour garantir une fourniture d'électricité efficiente, durable, économique et sûre [10].

Les Smart Grids américains mettent, quant à eux, l'accent sur la modernisation des réseaux de transport d'énergie souvent très mal entretenus. Les États-Unis doivent en effet faire face à un nombre de coupures de courant qui ne cesse d'augmenter, passant de 76 pannes en 2007 à plus de 300 en 2011 [2014-11]. Ces pannes sont aussi fréquentes qu'importantes. En 2003, un *blackout* dans l'Ohio prive 50 millions de personnes d'électricité et coûte 6 milliards de dollars [2005-12]. Ces coupures sont dues à la combinaison de trois facteurs [2014-13] :

- une infrastructure électrique vieillissante, en grande partie mécanisée et souffrant du manque d'investissement ;
- une forte croissance démographique ;
- des conditions climatiques de plus en plus extrêmes.

Aux États-Unis, moderniser le réseau électrique en investissant dans les Smart Grids est ainsi essentiellement guidé par un impératif de fiabilité. Pour qualifier les Smart Grids, le département de l'énergie américain (*United States Department of Energy*) dresse une liste d'exigences mettant en avant la sûreté des réseaux électriques [14] et en donne alors la définition suivante :

Définition 2. *Un Smart Grid est un réseau électrique qui répond aux exigences suivantes :*

- *auto-réparation en cas d'événements perturbateurs ;*
- *participation active des consommateurs ;*
- *résilience face aux attaques physiques et cybernétiques ;*
- *fourniture électricité de qualité face aux besoins du 21ème siècle ;*
- *intégration de moyens de production et de stockage d'électricité ;*
- *exploitation optimisée des infrastructures et conduite efficace des réseaux électriques.*

À partir de ces deux définitions, nous constatons que les impératifs régulatoires et écologiques (dans le cas de l'Europe) et l'obsolescence du système électrique (dans le cas des États-Unis) font de la mise à niveau des réseaux électriques un enjeu crucial. Cette mise à niveau passe d'abord par le déploiement des Technologies de l'Information et de la Communication (TIC) et donc par l'implémentation des Smart Grids, plutôt que par le remplacement massif des infrastructures électriques existantes.

Envisager uniquement le renforcement et le remplacement massif des infrastructures électriques du réseau n'est en effet pas une solution optimale et semble difficilement réalisable compte tenu de la croissance démographique constante des zones urbaines et du coût important des investissements à consentir [15].

1.1.2 Contexte économique, cadre législatif et modes de consommation en constante mutation

Le système électrique actuel est mis à l'épreuve par l'entrée en jeu de nouveaux acteurs économiques et de nouveaux cadres législatifs.

D'une part, la libéralisation du marché de l'électricité permet à un producteur quelconque de produire et de vendre son électricité après s'être convenablement raccordé au réseau électrique de distribution ou de transport (donc entre les centrales de production et les consommateurs). Contrairement aux centrales de production localisées en amont du réseau électrique de transport, ces sources d'énergie sont dites distribuées. Les sources d'énergie distribuées perturbent fortement la stabilité des réseaux électriques. En injectant du courant, elles peuvent déséquilibrer le niveau de tension du réseau, endommageant ainsi les équipements du système électrique tels que les transformateurs, les lignes et les protections.

D'autre part, les enjeux environnementaux encouragent le recours aux énergies renouvelables. Dans sa directive du 23 avril 2009, la commission européenne fixe à 20% la part de contribution des ressources renouvelables dans la production totale d'énergie à l'horizon de 2020. Les réseaux électriques sont de ce fait amenés à connaître une croissance constante de ces producteurs décentralisés dans les années à venir.

Cette même directive fixe à 20% la réduction des émissions de gaz à effet de serre par rapport à leur niveau en 1990 et à 20% l'augmentation de l'efficacité énergétique. Or les pics de consommation en période de pointe sont coûteux et émetteurs de CO_2. Ces pics apparaissent typiquement en fin de journée d'un jour de semaine lorsque les consommateurs allument leurs télévisions, plaques électriques et autres outils électroménagers en rentrant chez eux. En effet, pour maintenir l'équilibre entre l'offre et la demande d'électricité en cas de pics de consommation, les producteurs recourent à des centrales à charbon, à fioul et à gaz comme l'illustre la figure 1.1.

Seulement, ces pics s'intensifient significativement avec l'émergence de nouveaux usages de consommation d'énergie dont, notamment, la mobilité électrique. En France, les pouvoirs publics estiment à deux millions le nombre de véhicules électriques en circulation en 2020. L'impact de ces véhicules sur l'équilibre entre l'offre et la demande n'est pas négligeable. En effet, la recharge complète d'un véhicule électrique ayant 150 km d'autonomie est équivalente en termes d'appel de puissance à :

- un chauffe-eau si la recharge s'effectue en 8 h (recharge normale) ;
- un immeuble si la recharge s'effectue en 1 h (recharge accélérée) ;
- un quartier urbain si la recharge s'effectue en 3 min (recharge rapide).

Dès lors, pour des raisons environnementales législatives mais aussi financières, ces pics doivent être écrêtés. Les gestionnaires de réseaux électriques voient dans le déploiement des technologies Smart Grids un moyen d'y parvenir.

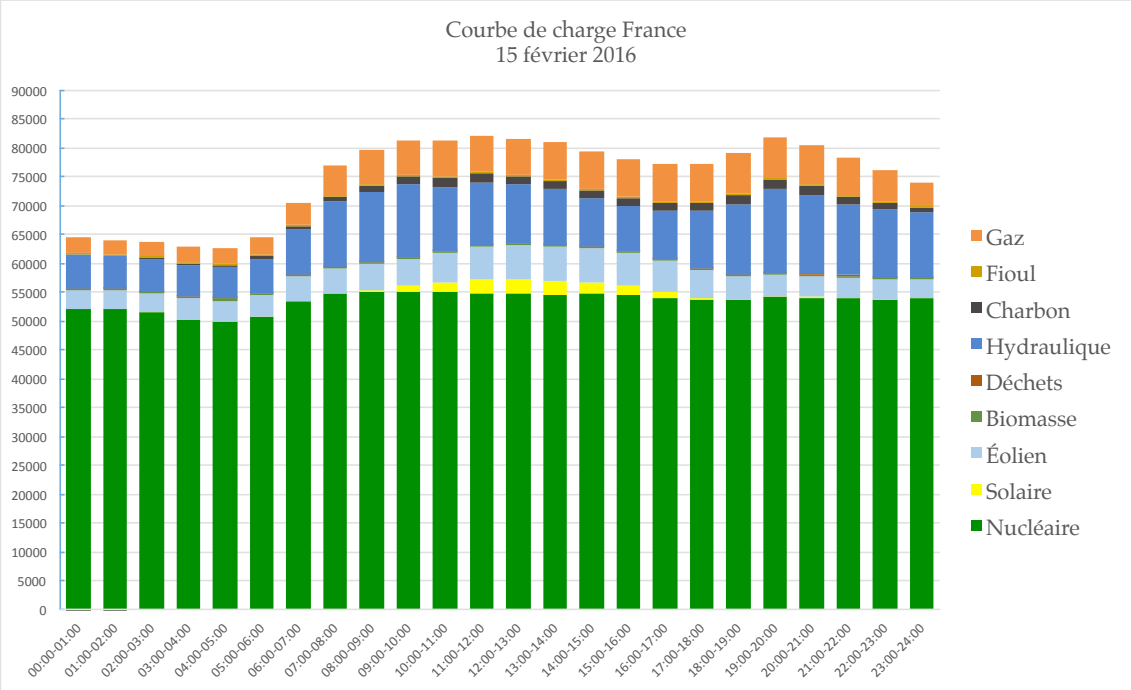

FIGURE 1.1 – Placement du type d'énergie sollicitée sur la courbe de charge française du lundi 15 février 2016 (source RTE [2])

1.1.3 Architecture des Smart Grids : vers des réseaux électriques flexibles et communicants

Pour faire face aux mutations du contexte énergétique, les gestionnaires de réseaux électriques ne peuvent plus compter uniquement sur la conduite prévisionnelle du réseau électrique (peu réactive face à l'intermittence des énergies renouvelables par exemple), ni envisager le redimensionnement du réseau (onéreux et non optimal).

La solution réside dans l'automatisation de la conduite des réseaux électriques, grâce à l'acquisition et l'exploitation en temps réel d'informations sur l'état des réseaux. Cela passe par le déploiement d'un réseau informatique au niveau des infrastructures électriques, et la mise en place dans le Système d'Information (SI) d'outils pour l'exploiter.

Ainsi équipés, les réseaux électriques s'apparentent à une toile d'araignée où les mailles interagissent constamment via des liens de communication. Ces mailles correspondent aux acteurs du système électrique : consommateurs, producteurs ou les deux à la fois. Outre l'électricité, ces acteurs produisent et consomment de l'information en temps réel grâce aux modules logiciels dont ils sont équipés et à divers moyens de télécommunication, tels que les réseaux mobiles ou le Courant Porteur de Ligne (CPL). Ce partage permanent et instantané d'informations entre les équipements préserve la stabilité du système électrique tout en augmentant son efficacité énergétique.

Pour illustrer les possibilités offertes par les TIC, prenons l'exemple du pic de consommation de la fin de journée d'un jour en semaine. Grâce aux TIC, il devient possible d'agir sur la demande plutôt que sur l'offre. Le distributeur d'électricité, s'appuyant sur des points de contrôle distants et des compteurs intelligents installés chez les clients pour envoyer et recevoir des informations et des consignes, peut alors adresser des demandes d'« effacement » aux consommateurs moyennant des incitations tarifaires. L'effacement peut prendre, par exemple, la forme d'une coupure du chauffage pendant 15 min à 30 min d'un foyer ou d'un bureau bien isolé. Sans incidence sur le confort des consommateurs, ces demandes d'effacement aident à lisser la courbe de charge aux heures de pointe tout en évitant de mobiliser des centrales de production.

Nés de la convergence des réseaux électriques et des TIC, les Smart Grids se composent ainsi de trois couches que nous retrouvons dans la figure 1.2 :

1. le premier niveau correspond à l'infrastructure et aux équipements électriques acheminant l'électricité tels que les lignes et les transformateurs ;
2. le second niveau correspond à l'infrastructure de communication composée de différentes technologies de télécommunication comme la fibre optique, le CPL, ou encore la Troisième Génération (3G) ;
3. le troisième niveau correspond aux applications informatiques qui incarnent « l'intelligence » du réseau. En utilisant des informations délivrées en temps réel, ces applications calculent des consignes à envoyer aux équipements concernés et automatisent ainsi la conduite du système électrique. Cette intelligence est centralisée au niveau des centres de conduite du réseau ou distribuée sur les équipements électriques.

1.1.4 Alignement du Système d'Information à une stratégie orientée Smart Grids

En traitant les données envoyées en temps réel par les capteurs installés sur les équipements électriques et chez les consommateurs, les SI calculent des consignes destinées à des organes télécommandés permettant ainsi de piloter les réseaux électriques à distance. Cette automatisation de la gestion des réseaux est une solution pour les adapter rapidement face aux contraintes qu'introduit l'intégration des énergies renouvelables et des nouveaux usages[15]. Les SI sont donc au cœur des enjeux des Smart Grids.

L'implémentation des Smart Grids va ainsi de pair avec la mise à niveau des SI des gestionnaires du réseau électrique. En effet, ces SI doivent pleinement intégrer les évolutions qu'induisent les Smart Grids au niveau des processus métier du gestionnaire de réseau, des acteurs impactés, des informations échangées ainsi que des applications informatiques et des infrastructures techniques sous-jacentes. Parmi ces évolutions nous citons :

– les nouveaux flux d'information provenant du réseau électrique ;

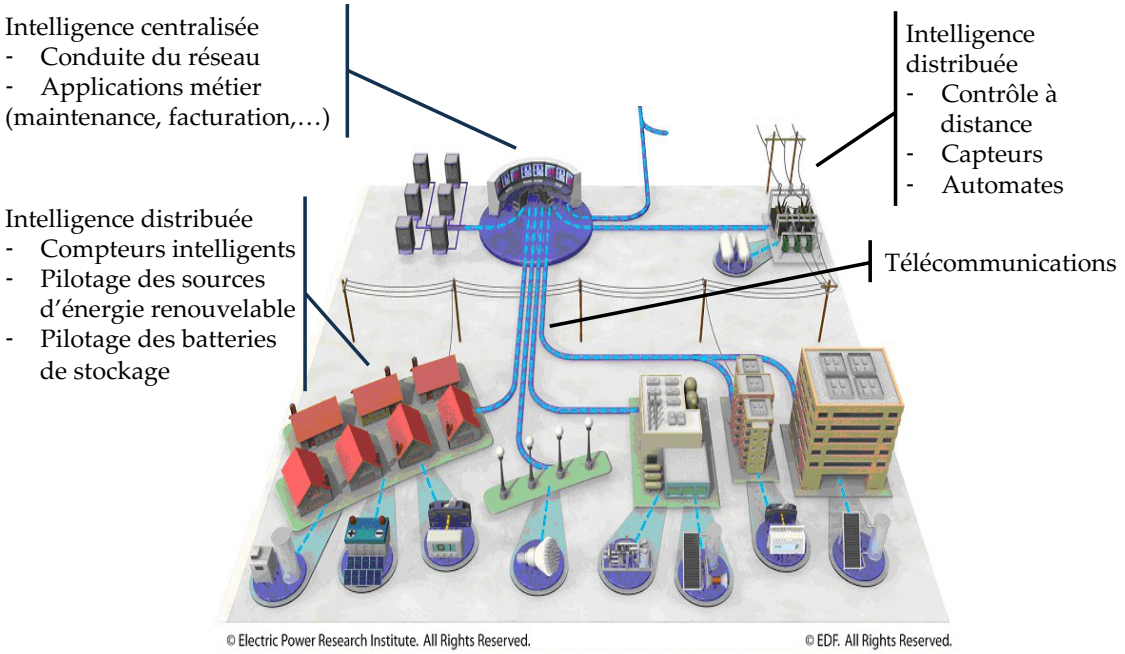

FIGURE 1.2 – Architecture des Smart Grids [2006-1]

- l'entrée en jeu de nouveaux acteurs tels que les producteurs décentralisés (éolien, photovoltaïque) ;
- les nouveaux équipements communicants comme le compteur Linky (Électricite Réseau de Distribution France (ERDF) annonce le déploiement de 30 millions de compteurs d'ici 2020 [3]) ;
- les nouvelles réglementations et directives européennes (dans le cas des gestionnaires de réseaux européens) ;
- les nouveaux usages comme les véhicules électriques ou encore les maisons connectées.

Outre leurs SI, les gestionnaires de réseaux électriques doivent faire évoluer leurs stratégies de développement en envisageant de nouveaux modèles métier et de nouveaux partenaires, tout en tenant compte de l'émergence des nouvelles technologies et des exigences du législateur. Une étude américaine, menée par IBM, CISCO, EPRI et South Carolina Edison, fait état de cinq thèmes stratégiques clés pour l'implémentation des Smart Grids :

- **permettre au consommateur de contrôler sa consommation d'énergie** et de réduire son empreinte carbone en utilisant des équipements intelligents et des véhicules électriques et en produisant de l'énergie renouvelable à domicile ;

3. www.erdf.fr/Linky

- **améliorer la sécurité et la productivité des employés** en mettant par exemple à leur disposition des outils performants pour le contrôle à distance, des équipements de protection, et des applications mobiles ;
- **intégrer des sources d'énergie renouvelables** distribuées sur le réseau en assurant la protection des équipements électriques, le stockage de l'énergie et la stabilité du réseau ;
- **améliorer l'efficience et la résilience du réseau** à travers les systèmes de mesure en temps réel, l'analyse et le contrôle à distance ;
- **fournir les informations et la connectivité nécessaires** en développant une infrastructure TIC pour répondre aux besoins d'informatisation du réseau électrique. Ce dernier point est une condition *sine qua non* de la réalisation des quatre précédents.

Compte tenu de ces thèmes clés, les gestionnaires de réseaux envisagent des stratégies impliquant l'adoption de technologies Smart Grids. La mise en œuvre effective de ces stratégies demande l'automatisation des actions sur le réseau et du traitement des données, ce qui entraîne donc le déploiement de nouveaux SI. Afin d'appréhender ces paradigmes naissants, des scénarios métier sont élaborés mais il est indispensable de les éprouver et de les valider avant d'envisager leur implémentation finale.

Plusieurs démonstrateurs physiques ont été déployés sur le terrain [4]. Ces projets pilotes permettent de mener des expérimentations en conditions réelles pour tester des fonctions et des services, comme par exemple le démonstrateur InfiniDrive [5] pour le pilotage des infrastructures de recharge de véhicules électriques ou le démonstrateur Venteea [6] pour l'intégration d'une forte capacité éolienne dans un réseau rural. Cependant, les démonstrateurs nécessitent que le gestionnaire de réseau de distribution recrute des clients industriels et/ou domestiques qui acceptent d'avoir du matériel à tester chez eux. De plus, leur exploitation reste limitée par les réglementations en cours. Enfin, leur mise en place se révèle souvent longue et coûteuse.

En plus de ces démonstrateurs, des réseaux de distribution d'expérimentation grandeur nature tel que Concept Grid [7], implanté à Électricité De France (EDF) Lab, permettent de tester les nouveaux équipements avant leur installation sur les réseaux du distributeur ERDF. Ces réseaux ont l'avantage de permettre la conduite de *stress tests* en conditions perturbées, impossibles à réaliser dans le cadre de démonstrateurs, ceux-ci impliquant de vrais clients. Cependant, la taille réduite de ces réseaux reste limitante.

Pour pallier toutes ces limitations, une troisième voie est la simulation. La simulation intègre les trois couches qui composent les Smart Grids : l'infrastructure électrique (transformateurs, lignes, charges, sources), l'infrastructure de télécommunication (réseaux mobiles, CPL) et enfin les SI qui les pilotent. Des simulateurs spécialisés dans la simulation de réseaux

4. www.erdf.fr/Carte_demonstrateurs_Smart_Grids
5. avem.fr/actualite-erdf-et-le-groupe-la-poste-lancent-le-projet-infini-drive-a-nice-3450.html
6. www.venteea.fr
7. chercheurs.edf.com

électriques (EMTP-RV, Dymola, PowerFactory, Eurostag, etc.) ainsi que des simulateurs de réseaux de télécommunication (OPNET, NS-3, OMNeT ++, etc.) ont déjà validé l'apport de la simulation dans leurs domaines respectifs. Toutefois, les SI sont souvent relégués à de simples modèles de calcul de consignes souvent développés en Matlab ou en C++[2014-16].

Dans ce contexte, la problématique industrielle dans laquelle s'inscrivent nos travaux de recherche est donc la suivante :

Comment valider une stratégie de développement orientée Smart Grids à travers la simulation de sa déclinaison au niveau du SI du gestionnaire du réseau électrique ?

1.2 Problématique de recherche

Les technologies Smart Grids illustrent le défi que représente l'évolution des TIC et l'intégration des énergies renouvelables pour les gestionnaires de réseaux électriques. Jeremy Rifkin annonce même « une troisième révolution industrielle, fondée sur le couplage des technologies de l'Internet et des énergies nouvelles » [2012-17]. Dans ce contexte, nous identifions trois axes de recherche : (1) l'adaptation au changement, (2) l'alignement stratégique et (3) la simulation de systèmes dynamiques et complexes.

1.2.1 Adaptation au changment

À l'ère numérique, l'adaptation au changement relève des préoccupations de toute entreprise s'appuyant sur les TIC pour mener ses activités. Le très haut niveau de concurrence que connaît le secteur des technologies de l'information stimule l'innovation. Or ces technologies sont de plus en plus le moteur qui fait progresser les métiers de l'entreprise. Être capable de s'adapter continuellement à l'évolution rapide et constante des TIC représente donc un véritable challenge pour les entreprises d'aujourd'hui.

Pour mener à bien tout changement, il est primordial de commencer par une description représentative de « l'objet » à changer, qu'il s'agisse d'une nouvelle version d'un avion, d'une voiture, d'un ordinateur ou encore d'une entreprise [1997-18]. Cette description représentative revient à concevoir l'architecture de l'objet en question. L'architecture en tant qu'activité est centrale dans plusieurs disciplines, allant de l'architecture du bâtiment à l'architecture logicielle, en ce qu'elle est un outil indispensable à la construction d'artefacts respectant les qualités attendues de l'objet final. En précurseur, Zachman [1997-18] recommande d'appliquer les principes d'architecture à l'entreprise pour faire face aux changements dictés par l'innovation technologique.

1.2.2 Alignement stratégique

Les SI sont en première ligne dès qu'il s'agit de l'évolution des TIC. Nos travaux se sont donc d'abord portés sur l'évaluation de l'impact de cette évolution sur les SI de l'entreprise. De ce fait, nous nous sommes d'abord appliqués à décrire l'architecture de ces SI [2015-19].

Les changements apportés par ces technologies impactent cependant, non seulement les SI, mais l'entreprise dans son ensemble : de sa stratégie à ses partenaires, en passant par ses objectifs, ses clients et ses processus métier. Par exemple, l'utilisation des véhicules électriques fait évoluer le SI du gestionnaire du réseau électrique qui doit mettre en place de nouvelles applications capables de bien gérer leur recharge. Mais il doit aussi faire évoluer son modèle métier en mettant à disposition de nouveaux contrats client favorisant la recharge hors de la période des pics de consommation par exemple, ou encore créer de nouveaux partenariats avec les constructeurs automobiles comme dans le cas d'EDF et Renault-Nissan qui collaborent sur un système de recharge pour véhicule électrique permettant à ce dernier de communiquer avec les bornes de recharge.

Évaluer l'adoption de nouvelles technologies de l'information oblige à prendre en compte l'entreprise dans son ensemble afin de garantir une cohérence entre la stratégie d'évolution adoptée et les SI qui implémentent cette stratégie. L'alignement entre la stratégie métier et le SI étant au cœur de l'*Enterprise Architecture* (EA)[8] [1997-18], nous adoptons l'EA pour évaluer l'impact des technologies Smart Grids sur les gestionnaires des réseaux électriques. Il est en effet indispensable de concevoir une architecture cible pour avoir « une vision générale de comment une entreprise va mettre en œuvre sa stratégie » [2006-20].

L'EA participe à l'alignement des composants d'une entreprise en offrant une vision globale et transverse [1987-7]. Dans le contexte des Smart Grids par exemple, elle permet d'aligner efficacement les intérêts des acteurs impliqués dans leur implémentation tels que les experts métier, les conseillers stratégiques, les experts environnementaux ou encore les experts en normalisation.

L'exécution effective d'une stratégie est cependant confrontée à des barrières de communication au sein de l'entreprise [2010-21]. Le recours à l'EA est d'autant plus justifié qu'elle représente un outil pour la transmission des objectifs stratégiques à tous les niveaux hiérarchiques de l'organisation en question [2008-22].

Pour toutes ces raisons, nous souhaitons mettre à profit les principes d'EA pour évaluer l'impact de l'adoption des technologies Smart Grids sur les SI des gestionnaires de réseaux électriques, tout en garantissant la cohérence entre la stratégie adoptée et les SI qui les implémentent.

8. L'acronyme EA (*Enterprise Architecture*) est souvent utilisé même dans la communauté francophone.

1.2.3 La simulation de systèmes dynamiques et complexes

Les Smart Grids correspondent à des systèmes dynamiques et complexes [2010-23] étant donné le grand nombre de parties prenantes qui interagissent eux (tels que les producteurs d'énergie renouvelable ou les consommateurs actifs sur le réseau), tout en ayant des comportements autonomes et des objectifs différents. Les SI pilotant les Smart Grids correspondent par conséquent à des systèmes dynamiques aux comportements complexes. Borshchev et Filippov [2004-24] affirment que le seul moyen d'adresser cette complexité est de simuler ces systèmes. La simulation est une technique connue pour valider ou critiquer la conception d'un système dès les premières étapes de son cycle de développement. Les acteurs impliqués dans le déploiement des Smart Grids acquièrent, par la simulation, une connaissance approfondie et directe des modèles créés pour valider ou critiquer leur choix d'implémentation.

Néanmoins, les approches d'EA se focalisent le plus souvent sur des aspects statiques et structurels tels que les interconnexions entre les différentes applications métier [2008-25]. De plus, les modèles issus de ces approches sont ne sont pas exécutables et sont conçus en priorité pour la documentation et la communication entre parties-prenantes [2013-26].

Notre problématique de recherche se résume de ce fait dans la question suivante :

Quels méthodes, modèles et outils adopter pour simuler une architecture d'entreprise afin de la valider ?

Dans le cadre de ces travaux de thèse, nous nous sommes saisis de cette problématique de modélisation et de simulation d'architectures d'entreprise et nous avons souhaité mettre à contribution les méthodes et technologies de l'Ingénierie Dirigée par les Modèles (IDM) pour apporter une aide supplémentaire à l'analyse et à la validation de ces architectures.

1.3 Organisation du document

Ce document est composé de deux parties principales. La première partie dresse un état de l'art de l'EA et de l'IDM. La seconde présente l'ensemble de nos contributions dont principalement un *framework* baptisé ExecuteEA s'appuyant sur la combinaison et l'intégration de plusieurs techniques de l'IDM pour faciliter l'analyse d'architectures d'entreprise. Enfin, une dernière partie résume l'ensemble de nos contributions et dresse les perspectives de ce travail.

Partie 1 : « État de l'art »

Le chapitre 2 : « Architecture d'Entreprise » Ce chapitre présente les notions fondamentales de l'EA et les principes guidant la conception d'une architecture d'entreprise. L'objectif de ces travaux de thèse étant de valider les évolutions envisagées pour une

architecture d'entreprise avant leur implémentation, nous considérons dans ce chapitre les approches d'analyses existantes.

Le chapitre 3 : « Ingénieurie Dirigée par les Modèles » Ce chapitre présente les objectifs, les concepts fondamentaux et les techniques principales de l'IDM. Nous considérons ensuite les approches d'EA existantes qui recourent à l'IDM et nous en dressons les contributions et les limites.

Partie 2 : « Une approche unificatrice par les modèles »

Le chapitre 4 : « ExecuteEA » Dans ce chapitre, nous examinons tout d'abord les pratiques actuellles d'EA en nous appuyant sur une étude de la doucmentation de projets dédiés aux Smart Grids. Forts de ce bilan, nous justifions le recours à une démarche IDM aboutissant à l'identification des concepts relatifs au domaine de l'EA, et nécessaire à la création d'un métamodèle pour l'EA que nous baptisant EAT-ME. Ce métamodèle est la pierre angulaire du *framework* ExecuteEA que nous destinons à l'analyse de la structure et du comportement d'une architecture d'entreprise.

Le chapitre 5 : « Implémentation et validation » Ce chapitre décrit l'implémentation qui concrétise les propositions que nous faisons à travers ce *framework* et son application au cas d'étude de la gestion d'une flotte de véhicules électriques.

Le chapitre 6 : « Délimitation de l'objet de recherche » Dans ce chapitre, nous abordons la question de la délimitation de l'objet d'étude. Cette délimitation a été, en soi, à l'origine d'une démarche de recherche à part en entière. Nous avons ainsi pu mettre en évidence certaines caractéristiques de notre objet d'étude et jeter les bases des travaux présentés dans ce mémoire.

Partie 3 : « Conclusion et perspectives »

Le chapitre 7 : « Résumé des contributions et perspectives » Ce chapitre résume les contributions de cette thèse et dresse les perspectives qui lui sont associées.

1.4 Le projet POMME

Une partie du travail présenté dans ce mémoire a été effectué dans le cadre du projet POMME (Plate-formes Orientées Métiers de Multi-simulations Électriques) réalisé dans le département MIRE (Mesure et Systèmes d'Information des Réseaux Électriques) de la direction R&D d'EDF (Électricité De France). L'objectif de ce projet est de coupler la simulation de SI, celle d'un réseau de télécommunication et celle d'un réseau électrique, de manière à simuler de bout en bout l'ensemble d'un Smart Grid.

Nous avons participé à ce projet en fournissant des méthodes, modèles et outils permettant de simuler un SI et plus globalement une architecture d'entreprise, que nous présentons dans la deuxième partie de ce mémoire.

Première partie

État de l'Art

—

Architecture d'Entreprise et Ingénierie Dirigée par les Modèles

Chapitre 2

Architecture d'Entreprise

Sommaire

L'émergence des Smart Grids suscite de profonds changements non seulement au niveau des réseaux électriques et des SI qui les pilotent, mais aussi pour l'ensemble de l'entreprise qui fournit et distribue de l'électricité : sa stratégie de développement, ses processus métier, etc. L'EA a pour objectif d'accompagner ce type de changements en capturant les différents composants de l'entreprise et leurs relations. C'est à ce titre que notre état de l'art traite de l'EA.

Ce chapitre présente ainsi les notions fondamentales de l'EA dans la section 2.1 avant

de présenter les principes guidant la conception d'une architecture d'entreprise dans la section 2.2. L'objectif de ces travaux de thèse étant de valider les évolutions envisagées pour une architecture d'entreprise avant leur implémentation, nous considérons, dans la section 2.3, les approches d'analyses existantes.

2.1 Notions fondamentales de l'Architecture d'Entreprise

2.1.1 Terminologie

Pour cette première partie de l'état de l'art consacré à l'EA, nous commençons par définir les termes de SI et d'EA.

2.1.1.1 Système d'Information

Une définition communément admise du SI est donnée par Robert Reix [1995-27] :

Définition 3. *Le SI est un ensemble organisé de ressources : matériel, logiciel, personnel, données, procédures permettant d'acquérir, de traiter, de stocker des informations (sous forme de données, textes, images, sons, etc.) dans et entre des organisations*

Nous adoptons cette définition car elle a l'avantage de ne pas réduire le SI d'une organisation à son système informatique. Le système informatique est constitué de l'ensemble du patrimoine matériel (hardware) et applicatif (software) de la dite organisation et a pour objectif d'automatiser le traitement de l'information. Nous adoptons l'acronyme IT (*Information Technologies*) pour le différencier du SI.

On suppose souvent que les SI sont totalement informatisés et c'est une des raisons qui mènent à confondre SI et IT. Cependant, le SI comprend non seulement le système informatique mais aussi des ressources humaines telles que les partenaires ou le personnel et des ressources comme les procédures de gestion ou encore le savoir-faire métier.

2.1.1.2 Architecture d'Entreprise

Il existe une multitude de définitions de la notion d'architecture d'entreprise. Nous rapportons ici celle donnée par Zachman [1997-18], fondateur de l'EA.

Définition 4. *Une architecture d'entreprise est un ensemble pertinent d'artefacts de conception ou de représentations descriptives pour décrire une entreprise de manière à ce que cette entreprise soit créée en respectant certaines exigences et à ce qu'elle soit facilement maintenue tout au long de son cycle de vie.*

Zachman voit ainsi dans l'architecture un gage de qualité et de maintenabilité. L'EA revient à appliquer à l'entreprise les principes de l'architecture telle qu'elle est pratiquée dans de nombreuses autres disciplines comme par exemple l'architecture du bâtiment. Construire une maison en procédant chambre par chambre sans plan d'architecture général peut en effet mener à un résultat peu probant. De même, le développement préalable d'une organisation sans architecture de référence risque de mener à une duplication de ses ressources et altère par conséquent son efficacité, sa cohérence interne, et rend fastidieuse toute entreprise de changement [1997-18] [2012-28].

Il est important de noter que le terme architecture d'entreprise peut prêter à confusion car il est à la fois utilisé pour désigner (1) l'activité de conception d'une architecture i.e., la description des éléments composant l'organisation en question et leurs relations, mais aussi (2) l'ensemble des artefacts résultant de cette activité. Pour éviter toute confusion, nous désignons l'activité de conception par l'acronyme EA et le résultat de cette activité, c'est-à-dire les artefacts qui en sont issus, comme étant l'architecture de l'entreprise. L'EA est apparue en tant que discipline dans les années 1980 suite à l'informatisation accrue des entreprises, mais son périmètre ne cesse d'évoluer depuis. La section suivante décrit les raisons et les grandes étapes de cette évolution.

2.1.2 Évolution de l'Architecture d'Entreprise

L'EA est ancrée dans l'architecture de systèmes informatiques [2008-22]. Mais le périmètre de l'EA ne cesse de s'étendre en comprenant d'abord l'IT et certains aspects métier de l'entreprise [2006-29], évoluant ensuite en adressant l'ensemble de l'entreprise en intégrant sa stratégie et ses processus décisionnels [2006-20], et allant même jusqu'à aborder l'environnement dans lequel elle évolue [2012-6].

L'origine de l'EA [1] remonte en effet aux travaux de Zachman, souvent considérés comme précurseurs. Il y propose un cadre d'architecture pour l'IT [1987-7] afin d'optimiser la gestion du patrimoine applicatif et de l'infrastructure technique de l'entreprise. À ses débuts, l'EA se focalise donc sur des artefacts purement IT (data, logiciels, équipements) pour rationaliser l'utilisation des ressources informatiques [2006-29], tout en répondant aux besoins métier de l'entreprise. L'EA est alors guidée par les pratiques de l'ingénierie logicielle.

Cependant, l'accroissement de la complexité de l'IT et son rôle de plus en plus prégnant dans le cœur de métier des entreprises [2005-31] ont fait de l'architecture IT une problématique inhérente à l'ensemble des composants de l'entreprise. Pour y faire face, l'EA a commencé à inclure quelques aspects métier tels les processus des acteurs impliqués [2006-29]. En intégrant ainsi des problématiques métier, l'EA ne relève plus de l'architecture IT mais de l'architecture du SI dans son ensemble.

1. D'après Scott Bernard [2012-28], le terme Architecture d'Entreprise a fait sa première apparition dans le livre de Steven Spewak intitulé « Enterprise Architecture Planning : developping a blueprint for data, applications and technology » [1993-30].

Pour Scott Bernard, l'EA doit même aller plus loin en intégrant la stratégie de l'entreprise dans son périmètre pour offrir une vision globale de l'ensemble des ressources de l'entreprise [2012-28]. Le terme « entreprise » implique ainsi une vue à haut niveau de l'ensemble de l'organisation. Le terme « architecture » fait référence à la mise en place d'un cadre structuré et cohérent pour l'analyse, le planning, et l'exploitation de toutes les ressources dont dispose l'entreprise pour atteindre ses objectifs.

Enfin, certains auteurs insistent sur le fait que l'entreprise évolue dans un environnement inconstant [2012-6]. L'EA doit par conséquent inclure les relations de l'entreprise avec son environnement pour mesurer l'impact de ce dernier et faciliter les processus d'adaptation et d'innovation. Les différentes phases d'évolution de l'EA sont à l'origine de l'apparition de plusieurs écoles que nous détaillons dans la section suivante.

2.1.3 Écoles de pensée de l'Architecture d'Entreprise

Aucune définition de l'EA n'a été universellement adoptée [2012-32] [2005-31]. Il existe en effet une pléiade de définitions émanant aussi bien du milieu académique que du milieu industriel donnant ainsi lieu à plusieurs écoles de pensée. Ce manque de consensus est dû à la nature intrinsèque de l'activité d'EA : celle-ci est régie par un ensemble de préceptes et de bonnes pratiques que l'architecte reste libre d'adapter au contexte de l'entreprise. Il est toutefois possible d'identifier un thème commun à toutes les définitions proposées [2012-6] :

Définition 5. *L'architecture d'entreprise décrit les composants interdépendants d'une organisation et guide leurs évolutions.*

En revanche, le *périmètre* de cette description ainsi que les *préoccupations* adressées diffèrent d'une définition à l'autre.

S'agissant du *périmètre*, le terme « entreprise » peut couvrir uniquement l'IT ou s'étendre à tous ses composants humains, stratégiques, économiques et techniques. Les *préoccupations* sous-jacentes à l'EA peuvent quant à elles couvrir des objectifs allant de l'optimisation des investissements dans l'infrastructure technique à l'implémentation de la stratégie de l'entreprise en passant par l'alignement métier/IT.

Partant de ce constat, James Lapalme identifie trois écoles de pensée en EA [2012-6] : l'architecture de l'IT d'entreprise (*Enterprise IT Architecting*), l'architecture intégrative de l'entreprise (*Enterprise Integrating*) et l'architecture de l'entreprise dans son environnement (*Enterprise Ecological Adaptation*). Chaque école a son propre système de croyances (devise, préoccupations et objectifs, principes et postulats).

Il est important de noter que cette catégorisation est épurée voire idéalisée, dans la mesure où la majorité des auteurs gravitent autour d'une école plutôt que de se conformer complètement à une seule école. *Enterprise IT Architecting* réduit le périmètre de l'architecture à l'IT de l'entreprise. *Enterprise Integration* adopte une approche intégrative en incluant toute l'entreprise dans son périmètre, de sa stratégie métier à la gestion de son patrimoine

applicatif. *Enterprise Ecological Adaptation* inclut l'environnement dans lequel évolue l'entreprise pour en déterminer les impacts potentiels et mettre en place une stratégie d'adaptation adéquate. Le tableau 2.1 résume ces écoles de pensée en termes de devises, objectifs et principes.

	Enterprise IT Architecting	**Enterprise Integration**	**Enterprise Ecological Adaptation**
Devise	L'architecture d'entreprise raccorde l'IT au métier de l'entreprise	L'architecture d'entreprise lie la stratégie et son exécution	L'architecture d'entreprise est un moyen d'innovation organisationnelle et une garantie de durabilité
Objectifs et préoccupations	Planifier l'IT et en optimiser les coûts	Implémenter efficacement la stratégie de l'entreprise	Adapter et innover
	Appuyer le métier	Assurer la cohésion de l'organisation	Assurer la cohésion de l'organisation
			Encourager la co-évolution entre l'entreprise et son environnement
Principes et postulats	Appliquer une approche réductionniste	Appliquer une approche holistique	Appliquer une approche holistique
	Ne pas remettre en question la stratégie et les objectifs métier	Ne pas remettre en question la stratégie et les objectifs métier	Créer la stratégie de l'entreprise est une priorité
	Concevoir les composants de l'organisation de manière indépendante	Concevoir les différents aspects de l'entreprise de manière intégrative	Concevoir les différents aspects de l'entreprise de manière intégrative
	Ne pas se préoccuper des aspects non IT	Tenir compte de l'environnement comme source de changement	L'environnement peut être transformé

TABLE 2.1 – Écoles de pensée de l'Architecture d'Entreprise selon [2012-6]

En définissant sa taxonomie, Lapalme insiste sur le fait que ces écoles de pensée se sont formées par héritage : l'*Enterprise Ecological Adaptation* hérite de l'*Enterprise Integration*, qui elle-même hérite de l'*Enterprise IT architecting*. Cependant, cet l'héritage implique une transcendance car il existe des différences fondamentales entre ces trois écoles de pensée. Par exemple, l'approche réductionniste de l'*Enterprise IT architecting* est fondamentalement opposée à l'approche holistique des deux autres écoles. Mais quelle qu'en soit l'école de pensée, l'EA présente des avantages certains pour l'entreprise que nous rapportons dans la section suivante.

2.1.4 Avantages de l'Architecture d'Entreprise

L'EA organise et structure les informations à l'échelle de l'entreprise tout en fournissant les détails appropriés à chacune des parties prenantes et en définissant le schéma directeur nécessaire à la construction de SI évolutifs et pertinents pour le métier.

Pour Zachman, manier l'architecture de l'entreprise avec agilité et l'adapter rapidement à un contexte économique et technologique en constante évolution est un facteur de survie déterminant pour les entreprises du 21ème siècle [1997-18]. Afin de souligner l'importance de l'EA, Ross [33] donne l'exemple d'une problématique d'architecture que l'entreprise américaine *Jonhson&Johnson* a rencontré en 1995. Le succès international de cette entreprise repose surtout sur l'autonomie de ses 170 filiales. Ses managers sont parfaitement satisfaits des processus métier mis en place mais ce n'est pas le cas de tous ses clients. Les très grands clients reçoivent en effet de nombreux bons de commande et plusieurs factures des différentes filiales de *Johnson&Johnson* et doivent donc les traiter séparément. Ces clients exigent un jour de ne recevoir qu'une seule facture. Or *Johnson&Johnson* n'en est pas capable. Ni ses processus métier, ni sa structure organisationnelle et encore moins son IT ne lui permet d'accéder à la demande de ses clients. Ceci est un cas d'école typique que permet d'adresser l'EA.

L'EA présente plusieurs avantages liés autant aux aspects purement IT qu'aux aspects métier. D'abord, en capturant l'essence du métier, de l'IT et de son évolution [2013-3], l'EA permet d'abstraire la complexité d'un système telle qu'une entreprise. Ensuite, en tant que référentiel et support de communication, l'EA facilite la coordination entre les projets IT d'une entreprise, la supervision des ressources techniques ainsi que la suppression des redondances applicatives [2007-34]. Enfin, l'EA est un moyen efficace pour représenter les composants d'une entreprise dans son état courant et désiré. Comme tableau de bord, l'EA facilite l'accès à l'information nécessaire à l'optimisation des processus métier et à l'alignement effectif entre l'IT et la stratégie adoptée.

2.2 Conception d'une architecture d'entreprise

L'EA peut faciliter le processus de prise de décision si elle aboutit à une vision à la fois globale et adaptée aux décideurs. Dans la section suivante, nous présentons quelques approches et cadres d'EA prenant en compte le processus de prise de décision et les acteurs qu'il implique.

2.2.1 Approches orientées points de vue

Quelle qu'en soit l'école, l'EA reste une tâche complexe [2004-35] car elle implique un grand nombre de parties prenantes. Chaque partie prenante a des préoccupations et des

systèmes de notation propres et relatifs à son domaine d'expertise et aux responsabilités lui incombant.

L'EA doit capturer une grande variété de composants difficiles à représenter dans un seul et unique modèle. En procédant par analogie avec l'architecture d'une ville, la multitude d'acteurs concernés peut difficilement lire un plan où l'on représente à la fois les rues et les bâtiments ainsi que les réseaux de transports, d'électricité, de gaz et d'eau.

Il en va de même pour l'architecture d'entreprise. La nature mutli-facettes inhérente à l'entreprise rend inappropriée toute approche monolithique [1999-36]. En effet, un analyste métier est concerné par les processus et les fonctions métier tandis qu'un administrateur de bases de données est concerné par les données manipulées. Pour cette raison, la plupart des cadres d'EA adoptent une approche par points de vue.

Les approches orientées points de vue sont d'abord utilisées pour la spécification des besoins en ingénierie logicielle [1979-37]. Les chercheurs s'intéressent alors aux systèmes à « perspectives multiples » [1992-38] [1996-39] [1994-40] [1993-41]. Ces travaux précurseurs contribuent à l'émergence de plusieurs normes pour les systèmes logiciels, proposant des cadres d'architecture orientés points de vue. C'est le cas de la norme IEEE-1471 [2000-42], du standard *Reference Model of Open Distributed Processing* (RM-ODP) [1995-43] ou encore du standard MDA (*Model Driven Architecture*) [2003-44].

Dans la norme IEEE1471 [2000-42], une vue correspond à une représentation du système selon une certaine perspective à laquelle est associé un ensemble de préoccupations. Les vues permettent ainsi de séparer les préoccupations des différentes parties prenantes. Un point de vue correspond, quant à lui, à un *template* pour la création de vues. Il formalise les objectifs des parties prenantes concernées par la vue ainsi que les techniques qui permettent de la créer et de l'analyser. Pour décrire un point de vue, la norme IEEE1471 [2000-42] exige de spécifier les attributs suivants :

- le nom du point de vue ;
- la partie prenante ciblée ;
- les préoccupations de celle-ci ;
- le langage, les techniques de modélisation ou encore les méthodes d'analyse à utiliser pour la création de la vue.

2.2.2 Cadres d'Architecture d'Entreprise

Les approches orientées points de vue sont largement utilisées en ingénierie logicielle comme moyen d'adresser la complexité des architectures [2004-35]. Comme l'EA trouve ses racines dans l'architecture IT [2008-45], de nombreux cadres d'EA recourent aux points de vue en transférant les concepts développés pour architecture IT à l'EA. Parmi les cadres d'EA les plus utilisés nous citons le cadre Zachman, *The Open Group Architectural Framework* (TOGAF). Nous citons aussi d'autres cadres d'architecture spécifiques à leurs domaines comme RM-ODP et *Smart Grid Reference Architecture* (SGAM).

2.2.2.1 Le cadre Zachman

Zachman [1987-7] propose de structurer et d'organiser les différentes représentations intervenant dans la description d'une entreprise en les classant selon une matrice à deux dimensions. La dimension verticale correspond aux points de vue et la dimension horizontale aux abstractions. Comme l'illustre le tableau 2.2, chaque cellule de la matrice correspond à l'intersection entre une partie prenante impliquée dans le processus de conception de l'architecture et une abstraction présentée sous forme d'une question. Chaque représentation est alors adaptée aux acteurs impliqués.

Abstractions (colonnes) ⟶

Perspectives (lignes) ↕	**Données** *Quoi*	**Fonctions** *Comment*	**Personnel** *Qui*	**Temps** *Quand*	**Motivation** *Pourquoi*
Exécutif *Planification*	Identification des données	Identification des processus	Identification des responsabilités	Identification des échéances	Identification des motivations
Management *Définition*	Définition des données	Définition des processus	Définition des responsabilités	Définition des échéances	Définition des motivations
Architecte *Conception*	Conception des données	Conception des processus	Conception des responsabilités	Conception des échéances	Conception des motivations
Ingénieur *Spécification*	Spécification des données	Spécification des processus	Spécification des responsabilités	Spécification des échéances	Spécification des motivations
Technicien *Implémentation*	Implémentation des données	Implémentation des processus	Implémentation des responsabilités	Implémentation des échéances	Implémentation des motivations

TABLE 2.2 – Cadre Zachman [1987-7]

Encore largement utilisé, le cadre Zachman est le premier à adresser l'entreprise dans son ensemble. Il est de plus facile à comprendre et ne dépend ni d'une méthode ni d'un outil en particulier. Sa mise en pratique reste cependant fastidieuse à cause du grand nombre de cellules à modéliser. En outre, la mise en cohérence entre les différents artefacts n'est pas évidente car les relations entre les cellules ne sont pas explicitement spécifiées. Selon Lankhorst [2013-3], le cadre Zachman offre un schéma de classification bien structuré mais ne spécifie aucune méthode pour mener les différentes activités d'architecture.

2.2.2.2 TOGAF

TOGAF, le plus connu et le plus utilisé des cadres d'architecture [2008-45], est comme son nom l'indique un standard de l'*Open Group*. TOGAF est doté de quatre points de vue — métier, information, applicatif et technique — et d'une méthode de conception — *The Architecture Development Method* (ADM).

La méthode ADM correspond à un processus de conception cyclique. Elle préconise de piloter l'EA par la gestion des exigences. Les exigences sont dérivées de la stratégie et des objectifs métier de l'entreprise. Elles sont de ce fait considérées comme le centre névralgique des activités d'architecture et font le lien entre les différentes étapes, de la conception de l'architecture à sa mise en œuvre comme l'illustre la figure 2.1.

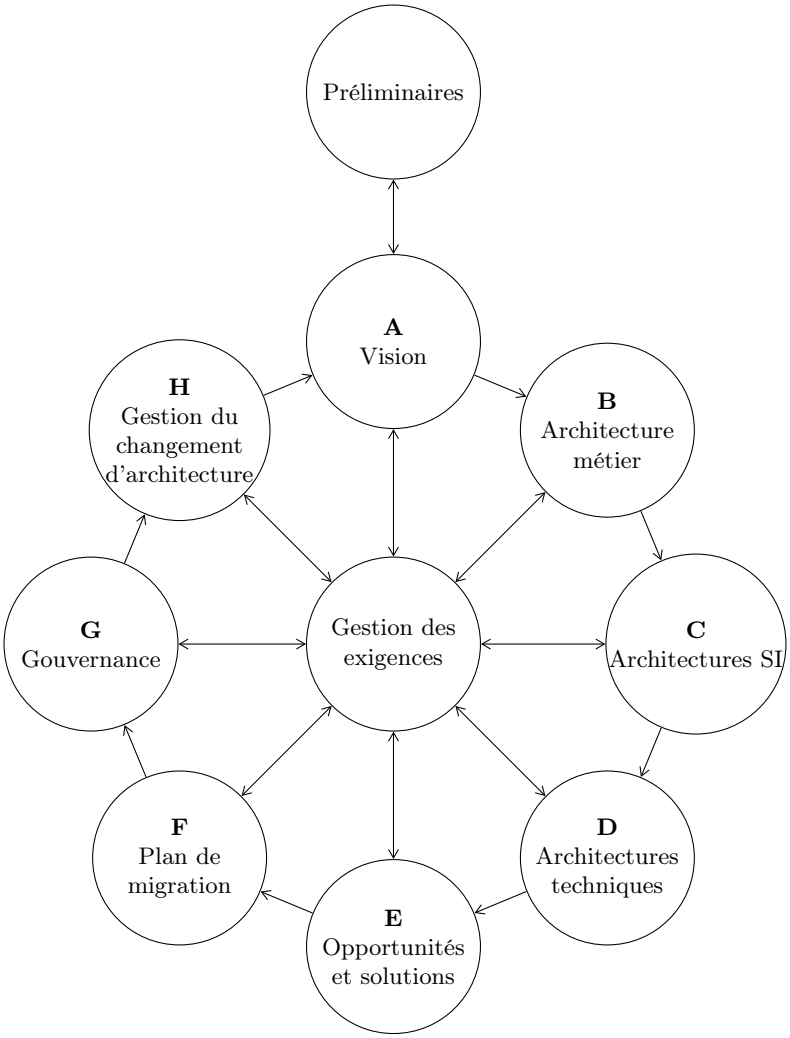

FIGURE 2.1 – TOGAF ADM [2009-2]

La méthode ADM commence par une phase préliminaire qui consiste à initialiser ou encore contextualiser l'EA. Pendant cette phase, la disposition de l'entreprise à engager une démarche d'EA est évaluée et les grands principes d'EA sont définis en accord avec le métier. Le cycle ADM se poursuit avec les huit phases, notées de A à G, relatives à la création des vues métier, applicative, information et technique. Il s'agit ensuite de planifier le déploiement de l'architecture avant de l'implémenter. La dernière phase (notée H) consiste à gérer les changements qui peuvent survenir suite aux évolutions métier ou technologiques pour assurer une mise à jour continue de l'architecture.

Générique, TOGAF peut être appliqué à des entreprises diverses indifféremment de leurs secteurs d'activité. Flexible et largement paramétrable, ce cadre d'architecture préconise un ensemble de bonnes pratiques à adapter selon les cas. Il est par exemple possible de

recourir au schéma de classification de Zachman dans le cadre d'une démarche ADM.

TOGAF comme Zachman laisse aux praticiens la liberté de choisir les langages de modélisation et le niveau de détail requis pour la conception des vues de l'architecture.

2.2.2.3 RM-ODP

RM-ODP est un standard ISO/ITU [2] qui définit un cadre d'architecture pour la spécification des systèmes distribués ouverts. Il se réfère au paradigme de l'Orienté Objet et identifie cinq points de vue :

- le point de vue Entreprise, qui décrit les activités métier du système ;
- le point de vue Information, qui définit l'information traitée par le système et la façon dont elle est traitée par les différents composants ;
- le point de vue Traitement, qui spécifie les traitements effectués par les différents composants en termes de fonctions et en faisant abstraction de toute plate-forme technique ;
- le point de vue Ingénierie, qui décrit les mécanismes logiciels permettant la distribution des composants et leur exécution sur les plates-formes d'exécution ;
- le point de vue Technologie, qui définit les technologies matérielles et logicielles utilisées pour l'infrastructure d'exécution, leur configuration.

RM-ODP préconise de découpler les préoccupations métier des contraintes liées à une plate-forme technique donnée. En effet, les cinq points de vue sont séparés mais corrélés. Chaque objet d'un point de vue donné correspond à un autre objet dans un autre point de vue.

RM-ODP offre un cadre de référence mais ne préconise pas de méthode d'architecture. Cette correspondance entre les différents points de vue n'est donc pas nécessairement respectée en spécifiant, par exemple, chaque point de vue de façon isolée. Pour y remédier, EDF R&D a proposé DASIBAO, une Démarche d'Architecture des Systèmes d'Information Basée sur RM-ODP. DASIBAO préconise le langage UML dans la spécification des cinq points de vues qu'il aborde dans l'ordre itératif suivant :

$$\text{Entreprise} \rightarrow \text{Information} \rightarrow \text{Traitement} \rightarrow \text{Ingénierie} \rightarrow \text{Technique}.$$

C'est donc par construction que DASIBAO se propose d'assurer la cohérence des différents points de vue.

2.2.3 Points de vue retenus

Les cadres d'EA n'utilisent pas tous les mêmes points de vue : leurs nombres, leurs noms ainsi que les préoccupations qu'ils adressent varient. Néanmoins, ces cadres recourent souvent, implicitement ou explicitement, aux points de vue ci-après.

2. International Organization for Standardization/International Telecommunication Union

Point de vue métier : ce point de vue reflète la vision métier de l'entreprise. On y retrouve ses objectifs ainsi que ses processus. Ces derniers sont représentés selon la structure organisationnelle de l'entreprise en termes d'acteurs internes et externes.

Point de vue fonctionnel : ce point de vue organise l'entreprise en blocs fonctionnels implémentant les processus de la vue métier. Cette structuration implique souvent une grande cohérence au sein d'un même bloc et une forte décorrélation entre blocs dans un souci de modularité et d'évolutivité. À l'échelle d'une entreprise, cette structuration devient vite complexe à cause du caractère étendu, transverse et interdépendant des processus métier impactés.

Point de vue applicatif : ce point de vue structure l'entreprise en blocs applicatifs. Chaque bloc implémente un ou plusieurs blocs fonctionnels. Il est aussi important de spécifier les échanges entre blocs applicatifs. Comme pour la vue fonctionnelle, la vue applicative des très grandes entreprises souffre souvent du syndrome du plat de spaghetti : les nombreuses applications fortement couplées deviennent difficiles et coûteuses à maintenir.

Point de vue technique : ce point de vue correspond à l'infrastructure technique nécessaire à l'exécution des blocs applicatifs. Le point de vue technique spécifie ainsi les machines physiques et liens de communication utiles au déploiement des applications informatiques.

En outre, ces cadres d'architecture sont orientés composants car ils utilisent des concepts tels que les macros processus, les blocs fonctionnels ou encore les blocs applicatifs. Les informations sont modélisées soit implicitement et de manière diffuse à l'intérieur des vues (TOGAF), soit séparément dans une vue dédiée et décorrélée des autres vues (RM-ODP, SGAM). Une troisième méthode consiste à les modéliser sous forme d'aspect pour chaque vue (Zachman).

De plus, ces cadres d'architecture organisent hiérarchiquement les différentes vues en appliquant *« IT follows business »* comme principe : commencer par la vue métier et la dériver progressivement jusqu'à l'infrastructure technique en passant par les fonctions et les applications [2006-29].

Les cadres d'architecture sont certes indispensables pour aboutir à une représentation pertinente des composants de l'entreprise mais ne suffisent pas à appréhender la complexité du système entreprise. Pour cela, il est primordial de disposer d'outils et de méthodes appropriés à l'analyse des modèles d'entreprise obtenus. La section suivante offre un tour d'horizon de l'activité d'analyse en EA.

2.3 Analyse en Architecture d'Entreprise

La valeur ajoutée de l'EA réside dans sa capacité à adresser le changement en offrant une vue holistique de l'entreprise. En effet, l'efficacité de l'entreprise dépend de l'orchestration

effective de ses différents composants et entités plutôt que d'optimisations locales et isolées [1992-46].

La documentation et la description ne suffisent cependant pas à assurer une architecture à la fois cohérente et pertinente pour le métier. Les techniques d'analyse de modèles sont indispensables à l'optimisation globale et effective d'une architecture [2013-3]. Les techniques d'analyse de modèles jouent donc un rôle crucial dans tout processus de changement affectant l'entreprise en éclairant efficacement la prise de décision. En effet, l'analyse de l'architecture d'entreprise ne doit pas se résumer pas à la revue mentale ou manuelle d'une vue d'ensemble étant données la taille et la complexité des architectures impliquées.

2.3.1 Au-delà des modèles « contemplatifs »

Les entreprises recourent aux cadres d'EA pour les guider dans la création et la maintenance de leurs architectures et pour avoir ainsi une vue globale et cohérente de leurs stratégies, leurs processus métier et leurs IT.

Les artefacts issus d'une démarche d'EA se résument souvent à un ensemble de documents utilisés comme supports de communication et comme schéma directeur au sein de l'organisation [2013-47] [2014-5]. Ces modèles fournissent certes un vocabulaire commun aux différentes parties prenantes mais ne sont ni manipulables ni interprétables par une machine. Ce sont des modèles purement « contemplatifs ». Cette terminologie est introduite par Bézivin [2001-48] en qualifiant les modèles de spécification utilisés pendant les premières phases de conception en génie logiciel.

Aussi, l'EA s'intéresse-t-elle davantage aux aspects structuraux de l'entreprise. Pour cette raison, les modèles utilisés, bien qu'offrant l'abstraction nécessaire pour adresser la complexité de l'entreprise, sont le plus souvent statiques. Ce genre de modèles ne permet pas d'appréhender les comportements de l'entreprise et sont donc insuffisants pour éclairer efficacement les prises de décision au niveau stratégique.

L'EA, à travers les différents cadres d'architecture proposés lors de ces trente dernières années, a certes contribué à traiter de manière intégrée les différents aspects d'une entreprise tels que les processus, le personnel, les services et l'IT. Cependant, la gestion des artefacts issus de l'EA reste un défi malgré l'existence d'outils sur étagère. En effet, les architectes d'entreprise expérimentés ainsi que les autres parties prenantes impliquées dans les activités d'architecture sont supposés se fier à leur bon jugement pour créer une architecture adéquate.

L'architecture créée est donc correcte par définition et dépend essentiellement des capacités de l'architecte et de son expertise. Certains travaux proposent de rendre l'EA moins dépendante de l'expertise de la personne en charge en traduisant les représentations d'architecture en ontologies [2013-49]. Les ontologies étant exécutables, elles permettent en effet de bénéficier des capacités d'analyse des raisonneurs disponibles.

Certaines méthodes d'EA sont accompagnées de langages de modélisation comme le langage

Archimate [3] pour TOGAF. Dans leur quête de généricité, ces langages deviennent rapidement très larges et difficiles à manipuler. Les modèles produits sont d'autant plus difficiles à gérer qu'ils ne sont pas manipulables par une machine. L'activité d'analyse en EA, bien que cruciale, est donc compromise par toutes ces limitations. Et bien que des modèles et des techniques destinés à l'analyse des architectures d'entreprise existent, le recours aux modèles exécutables reste encore marginal dans le domaine de l'EA [2013-26].

Parmi ces travaux nous citons le langage LEAP [2011-50]. Léger, générique et exécutable, LEAP est destiné à l'analyse des modèles issus de l'EA pour valider l'alignement des modèles métier et IT.

Les deux sections suivantes présentent deux schémas de classification des approches d'analyse en EA. Le premier est proposé par Lankhorst [2013-3] et le deuxième par Buckl et al. [2009-4]. Ces schémas de classification nous permettent de (1) comparer entre elles les différentes approches d'analyse recensées dans cet état de l'art et (2) positionner nos travaux.

2.3.2 Classification des approches d'analyse selon Lankhorst

Lankhorst [2013-3] utilise deux dimensions pour classifier les différentes approches d'analyse d'architecture d'entreprise : le type d'analyse et la technique employée. Ceci donne lieu à quatre catégories comme l'illustre la figure 2.2. La première dimension fait la distinction entre deux types d'analyse.

L'analyse fonctionnelle concerne les aspects fonctionnels de l'architecture. Elle permet par exemple de valider la structure ou de comprendre le comportement d'une architecture.

L'analyse quantitative concerne les aspects non fonctionnels de l'architecture comme la performance ou le coût.

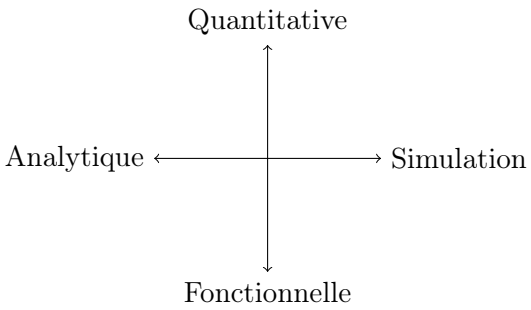

FIGURE 2.2 – Classification des approches d'analyse selon Lankhorst [2013-3]

La deuxième dimension identifie deux techniques pouvant être employées pour l'analyse fonctionnelle ou quantitative.

3. http ://www.opengroup.org/archimate

La technique de simulation des modèles revient à les exécuter. La simulation dans le cas de l'analyse fonctionnelle est utilisée pour mieux appréhender les aspects dynamiques d'une architecture. La simulation quantitative permet de mesurer des paramètres quantitatifs tel que le temps d'exécution d'un processus métier par exemple à travers plusieurs itérations de la simulation.

La technique analytique est plus formelle que la simulation. Ici le terme analytique signifie plutôt « mathématique ». Cette technique est plus efficace que la simulation quantitative pour fournir des indicateurs de performance.

2.3.3 Classification des approches d'analyse selon Buckl

Le schéma de classification de Lankhorst [2013-3] offre un premier aperçu des différentes approches d'analyse d'architecture cependant il ne révèle pas toutes les variantes et les subtilités d'une démarche d'analyse en EA. Nous considérons donc les travaux de Buckl et al. [2009-4] qui définissent un système de classification plus détaillé. Ce système classification peut même être considéré comme un cadre d'analyse d'architectures.

Buckl et al. [2009-4] font intervenir cinq dimensions pour catégoriser les approches d'analyse d'architecture d'entreprise : le sujet d'analyse, la référence temporelle, la technique d'analyse, les préoccupations de l'analyse, l'autoréférentialité. Ces dimensions sont illustrées par la figure 2.3.

2.3.3.1 Sujet de l'analyse

L'analyse peut concerner trois aspects différents de l'architecture : sa structure, son comportement dynamique ou son comportement statistique. D'abord, l'analyse de la structure est nécessaire pour appréhender la complexité structurelle des entreprises. Celle-ci est due à la densité des interconnexions entres ses composants. Ensuite, la complexité de la structure de l'entreprise induit une complexité au niveau de son comportement d'où la nécessité d'analyser l'aspect comportemental de l'architecture pour évaluer l'impact d'une anomalie sur le déroulement d'un processus par exemple. Enfin, l'analyse et l'agrégation des mesures statistiques provenant du comportement offrent une meilleure compréhension de l'architecture.

2.3.3.2 Référence temporelle

L'analyse peut se porter sur l'architecture d'une entreprise dans son état courant ou telle qu'elle est planifiée. Une analyse *ex post* concerne les modèles d'une architecture déjà mise en place alors qu'une analyse *ex ante* se réfère à différents scénarios élaborés pour une future implémentation.

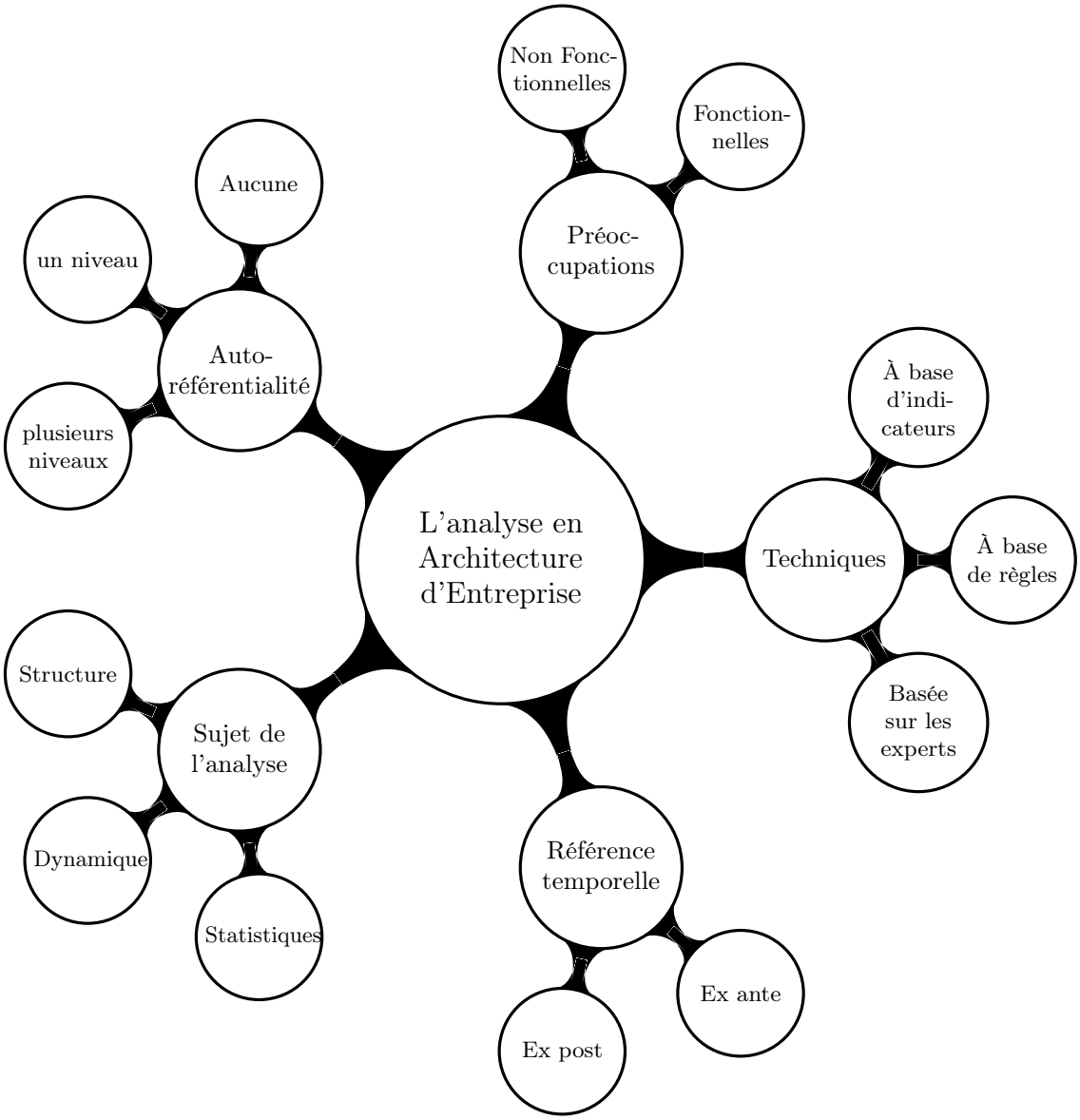

FIGURE 2.3 – Schéma de classification des approches d'analyse selon Buckl et al. [2009-4]

2.3.3.3 Technique d'analyse

Les techniques d'analyse employées peuvent s'appuyer sur des experts, sur des règles ou sur des indicateurs. Les analyses orientées experts sont les moins formelles et dépendent du niveau d'expertise de l'expert impliqué. Cette technique d'analyse est certes chronophage mais c'est celle qui offre le plus de flexibilité. Les résultats d'une telle analyse prennent la forme de conseils concrets ou d'idées et de directives générales concernant l'architecture en

question.

Les techniques d'analyse s'appuyant sur des règles sont plus formelles que celles s'appuyant sur des experts et peuvent être plus facilement automatisées. Ces règles décrivent les modèles de conception que l'architecture doit respecter ou éviter.

Les techniques d'analyse s'appuyant sur des indicateurs sont encore plus formelles que les deux précédentes et servent à évaluer en les quantifiant certaines propriétés de l'architecture. Les résultats d'une telle analyse doivent cependant être interprétés avec prudence car ils dépendent d'hypothèses souvent amenées à évoluer.

2.3.3.4 Préoccupations de l'analyse

Buckl et al. [2009-4] font la différence entre les analyses fonctionnelles et les analyses non fonctionnelles à la manière de Lankhorst [2013-3]. L'analyse fonctionnelle évalue si l'architecture remplit les fonctions métier de l'entreprise telles que la production ou la vente. L'analyse non fonctionnelle évalue des aspects comme le temps d'exécution d'un processus ou le coût de l'implémentation et de maintenance d'une architecture. Contrairement à Lankhorst [2013-3], Buckl et al. [2009-4] emploient le terme *non fonctionnelle* plutôt que *quantitative* en arguant que certains aspects non fonctionnels telle que la sécurité peuvent être analysés de manière non quantitative.

2.3.3.5 Autoréférentialité

Les personnes en charge de l'EA peuvent faire partie de l'architecture créée car ils appartiennent aussi à l'entreprise. Qui plus est, l'activité de décrire et de planifier l'architecture de l'entreprise peut elle-même être décrite et planifiée et fait par conséquent partie de l'activité d'EA. Selon Buckl et al. [1974-51] l'entreprise est en effet un système vivant capable de faire sa propre autopsie [1974-51].

L'analyse peut considérer un seul niveau d'autoréférentialité en intégrant les activités de management d'architecture. Une analyse à plusieurs niveaux d'autoréférentialité incorpore des activités de meta-management d'architecture comme par exemple la gouvernance des activités de management d'architecture. L'autoréférentialité augmente la complexité de l'analyse. Peu de travaux considèrent des niveaux d'autoréférentialité multiples [2014-52]. Nous citons parmi celles-ci les travaux de [2014-53] qui traitent la gestion et la gouvernance d'architecture en définissant des stratégies d'évolution pour les architectures actuelles vers les architectures cible dans un framework dédié.

2.3.4 Analyse de la structure

L'analyse de la structure est au cœur de la problématique de l'alignment métier/IT. Dans cette partie, nous présentons d'abord les finalités de cette activité d'analyse avant d'aborder

les techniques existantes et leurs limites.

2.3.4.1 Finalités

Maintenir une cohérence entre l'infrastructure informatique de l'entreprise et ses processus métier est au cœur des activités d'EA [2013-3]. L'objectif de cet alignement est d'améliorer l'efficacité de l'entreprise et de maximiser ses bénéfices. L'alignement métier/IT reste une problématique cruciale pour les entreprises qui s'appuient sur les technologies de l'information dans la réalisation de leurs objectifs métier [2005-54]. Zachman affirme même que seules les entreprises capables d'aligner rapidement leurs SI à leurs stratégies métier sont en mesure de survivre dans un environnement hautement concurrentiel [1997-18].

Pour Lankhorst [2013-3], l'EA a pour vocation d'offrir une vue générale et homogène du métier de l'entreprise, de son patrimoine applicatif, de son infrastructure technique et de son évolution pour en faciliter l'analyse via un ensemble cohérent de principes, méthodes et modèles. De plus, l'EA explicite et documente les relations entre les processus métier et l'IT de l'entreprise [2005-54].

En offrant une vision globale de l'entreprise et en documentant les relations entre ses différents composants, l'EA est un outil incontournable pour les architectes dans leur quête d'alignement métier/IT. Cependant, les méthodes et techniques actuelles offertes par l'EA ne suffisent pas à atteindre cet objectif [2013-55] car :

- les processus métier, les technologies, les structures organisationnelles ainsi que l'environnement sont en constante évolution. Les changements sont si rapides et nombreux qu'un réel alignement relève du vœu pieux [2013-3]. L'alignement métier/IT correspond plutôt à un idéal kantien vers lequel l'entreprise doit tendre à défaut de le réaliser complètement ;
- l'architecture en tant que discipline tient davantage de l'art que de la science. En effet, l'architecte d'entreprise analyse souvent de la documentation (tels que des tableaux Excel ou des illustrations Power point) même si quelques logiciels permettent désormais de visualiser ces modèles. Cette tâche devient rapidement ardue dès qu'il s'agit de grandes entreprises dont l'important patrimoine applicatif s'apparente à un plat de spaghetti.

2.3.4.2 Approches existantes d'analyse de la structure, et leurs limites

Les modèles exécutables peuvent assister les architectes d'entreprise à surmonter les obstacles cités ci-dessus. Les modèles exécutables augmentent l'agilité de l'architecture en détectant automatiquement les incohérences dès les premières phases de conception.

Les approches de modélisation en EA recourent souvent à des langages semi-formels qui ne permettent pas de vérifier dynamiquement et automatiquement certaines propriétés requises pour l'architecture comme la cohérence entres vues. Partant de ce constat, [2013-49]

propose de recourir aux ontologies, tirant ainsi profit des raisonneurs disponibles pour analyser la structure des architecture d'entreprise. Les auteurs arrivent à analyser l'impact du changement ou encore les relations et dépendances entre les opérations métier et les applications informatiques de l'entreprise. Cependant, les architectes d'entreprise sont peu familiarisés avec les ontologies. Les modèles d'architecture sont donc traduits en ontologies, risquant de faire apparaître un écart sémantique entre les modèles utilisés pour la représentation de l'architecture et les ontologies utilisées pour mener l'analyse.

La diversité des acteurs impliqués dans l'EA comme l'architecte d'entreprise, le manager ou encore l'ingénieur IT engendre une hétérogénéité au niveau des modèles utilisés. [2013-56] propose de créer un mapping entre des modèles hétérogènes (tels que des tableurs Excel, de la documentation ou des bases de données). Pour ce faire, les auteurs recourent à des modèles manipulables par machine et mettent à profit les techniques issus de l'IDM. Ces travaux adressent l'hétérogénéité des modèles et offrent une vue intégrée de l'architecture de l'entreprise en l'adaptant aux acteurs concernés. Mais ces travaux ne permettent pas de mener des analyses concernant l'impact du changement par exemple.

2.3.5 Analyse du comportement

Le comportement de l'architecture d'entreprise est peu abordé dans la littérature, en partie à cause de la nature statique des cadres d'EA. Cependant, l'analyse du comportement de l'entreprise peut offrir nombre d'avantages. La simulation est un moyen reconnu pour analyser le comportement d'un système. Dans cette partie, nous commençons par exposer les finalités de l'analyse par simulation du comportement des architectures d'entreprise avant de présenter les travaux qui traitent de cette problématique et leurs limites.

2.3.5.1 Finalités

Shannon [1975-57] donne la définition suivante du processus de simulation.

Définition 6. *La simulation est un processus consistant à modéliser un système réel et à mener des expérimentations sur le modèle obtenu dans le but de comprendre le comportement du système et/ou d'évaluer différentes stratégies concernant son fonctionnement.*

Quel qu'en soit le domaine d'application, la simulation est un moyen d'apprécier les choix des concepteurs sur le comportement du système modélisé. Elle peut se traduire par l'animation d'un modèle (représentant notre perception du système, qu'il soit existant ou à construire) et l'étude du comportement de ce modèle en fonction des variables en entrée.

La simulation des architectures d'entreprise permet de modifier localement des stratégies et d'observer l'impact de ces modification sur le comportement global du système [2008-25]. La simulation des architectures d'entreprise est d'autant plus cruciale dans le contexte des Smart Grids. Ces derniers sont en constante et rapide transformation : évolution des cadres

législatifs, apparition de nouveaux partenaires, hétérogénéité des interactions avec les clients finaux (les compteurs intelligents, Internet, les téléphones ou encore les tablettes).

Ainsi, le recours à la simulation dès les premières phases du cycle de vie des architectures d'entreprise dans le contexte des Smart Grids augmente leur évolutivité en apportant une aide supplémentaire à leur validation. La simulation des modèles facilite leur exploration par les experts métier et lève les ambiguïtés engendrées par les modèles purement contemplatifs. Elle permet en outre un prototypage rapide et une analyse itérative des modèles par les parties prenantes tels que les architectes d'entreprise, les experts métier, les analystes ou les architectes IT.

2.3.5.2 Approches de simulation existantes et leurs limites

[2015-58] recense quatorze approches d'analyse d'architectures d'entreprise et les classe selon les quatre dimensions du schéma de classification de Lankhorst [2013-3] (précédemment illustré par la figure 2.2). Seulement quatre approches parmi les quatorze recourent à la simulation comme outil d'analyse d'une architecture d'entreprise.

Le nombre limité d'approches utilisant la simulation pour l'analyse d'architecture d'entreprise est dû au fait que l'EA est initialement conçue comme une description statique des composants essentiels de l'entreprise et de leurs interconnexions [2013-59]. Les cadres d'EA standards ne prennent pas en compte les informations nécessaires pour analyser les aspects comportementaux d'une entreprise et mettent d'abord l'accent sur ses aspects structuraux tels que les liens entre les processus et les applications métier. Il en résulte que les outils d'EA n'adressent souvent que les aspects statiques de l'entreprise car ils se basent sur les mêmes cadres d'architecture.

Buckl et al. [2009-4] classifient les approches d'analyse d'architecture existantes selon leur propre système de classification illustré par la figure 2.3. Nous dressons le même constat que pointe l'état de l'art de l'analyse en EA de [2015-58].

Dans le présent état de l'art, nous nous intéressons donc aux travaux qui analysent la dimension dynamique d'une architecture d'entreprise. Parmi ces approches, celles développées par Glazner [2011-60], Ludwig et al. [2011-61] et Manzur et al. [2015-58] soulignent l'importance de simuler le comportement d'une architecture d'entreprise.

Glazner [2011-60] propose une approche de simulation hybride pour évaluer le comportement d'une entreprise en combinant une simulation à événements discrets et une simulation multi-agents. Il met à profit les capacités d'abstraction de l'EA pour adresser la complexité d'un système telle que l'entreprise, et s'en sert comme socle pour structurer les modèles de simulation. Les techniques et les langages utilisés pour la simulation sont décorrélés des langages de représentation utilisés par les architectes d'entreprise, entraînant ainsi un écart sémantique entre l'aspect statique de l'entreprise (les composants de l'entreprise et leurs relations) et son aspect dynamique (le comportement des composants).

Ludwig et al. [2011-61] simulent une architecture d'entreprise en fonction de la configuration organisationnelle d'une entreprise. Ils développent ainsi un langage exécutable pour décrire les relations entre des concepts d'ordre organisationnel comme l'entité, le rôle qu'elle joue, les services qu'elle procure et les processus dans lesquels elle intervient. La méthode et l'outil développés permettent de reconfigurer les modèles organisationnels lors de l'exécution. Ces travaux n'adressent cependant que la vue métier et une partie de la vue fonctionnelle d'une architecture d'entreprise.

Manzur et al. [2015-58] proposent une plate-forme de simulation qui s'appuie sur deux métamodèles différents : un premier métamodèle, correspondant au langage Archimate, pour spécifier les composants structurels de l'architecture et un deuxième métamodèle complémentaire pour spécifier les comportements des composants structurels. Les modèles sont ensuite simulés afin d'observer le comportement de l'architecture et de l'évaluer selon les indicateurs établis. Cependant cette approche évalue uniquement les aspects non fonctionnels d'une architecture d'entreprise.

2.4 Conclusion

Dans ce chapitre nous avons mis en évidence le rôle central de l'EA dans l'acquisition d'une vue holistique de l'ensemble des artefacts qui compose une entreprise. Nous avons présenté les principes fondamentaux de l'EA ainsi que son évolution depuis ses origines profondément ancrées dans l'architecture IT jusqu'à ses formes les plus actuelles qui traitent de l'ensemble de l'entreprise en temps que système irréductible.

Au cours de ce chapitre, nous nous sommes attachés à mettre en valeur les avantages de l'EA, autant liés aux aspects purement IT de l'entreprise qu'à ses aspects métier et stratégiques, dont notamment la mise à disposition de l'information nécessaire à l'optimisation des processus métier et l'alignement effectif entre la stratégie adoptée par l'entreprise et son IT.

Nous avons ensuite présenté différents cadres employés pour la conception d'architectures d'entreprise. Ces cadres adoptent une approche par points de vue. Une telle approche permet en effet de séparer les préoccupations des parties prenantes dans des vues différentes et de mieux appréhender la complexité de l'entreprise en tant que système. Nous avons identifié quatre points de vue principaux : métier, fonctionnel, applicatif et technique.

Enfin, nous avons introduit les différentes techniques existantes pour l'analyse d'entreprise, en détaillant l'analyse de la structure et l'analyse du comportement d'une architecture d'entreprise. Nous avons mis en lumière les limitations, en termes d'analyse d'architecture, de l'utilisation de modèles de représentation purement contemplatifs pour l'EA. Dans le chapitre suivant nous présentons l'IDM et proposons un état de l'art des techniques qui lui sont associées permettant de palier les limitations des modèles d'EA purement contemplatifs.

Chapitre 3

Ingénierie Dirigée par les Modèles

Sommaire

Dans le chapitre précédent, nous avons vu que les modèles utilisés pour l'EA, bien qu'offrant l'abstraction nécessaire pour adresser la complexité de l'entreprise, sont le plus souvent purement contemplatifs et ne permettent pas de mener une analyse automatisée et pertinente de la structure et du comportement d'une architecture d'entreprise. L'IDM se fait fort de traiter la problématique des modèles purement contemplatifs dans le contexte du génie logiciel. Dès lors, nous nous intéressons dans ce chapitre aux modèles, méthodes et techniques de l'IDM qui peuvent bénéficier à l'EA.

Ce chapitre présente ainsi la genèse et les objectifs de l'IDM dans la section 3.1 avant d'en présenter les concepts fondamentaux que sont les modèles et les métamodèles dans la section 3.2. Dans une approche IDM, l'automatisation de la manipulation des modèles passe en premier lieu par les techniques de transformation de modèle que nous introduisons dans la section 3.3. Enfin, nous considérons les approches d'EA existantes qui recourent à l'IDM et nous en dressons les contributions et les limites dans la section 3.4.

3.1 Genèse et objectifs

L'IDM est née du constat que le paradigme du « tout est objet », prôné dans les années 1980, a atteint ses limites avec ce début de siècle [2004-62]. En effet, face à la croissance de la complexité des systèmes logiciels, au coût de la main d'œuvre et de la maintenance, une approche centrée sur le code, jugé alors seul représentant fiable du système, suscitait de moins en moins l'adhésion des industriels et du milieu académique.

Partant de ce constat, l'*Object Management Group* (OMG) a proposé en novembre 2000, l'approche MDA qui s'inscrit dans le cadre plus général de l'IDM et se réalise autour d'un certain nombre de standards tels qu'UML, *Meta-Object Facility* (MOF), XML, QVT, etc. Le monde de la recherche s'y est aussitôt intéressé pour dégager les principes fondamentaux de l'IDM [2001-63, 2002-64, 2002-65] et déjouer le piège des définitions parfois trop floues qui mènent à confondre les concepts du paradigme objet et ceux de l'IDM [2004-66]. Par ailleurs, des industriels comme *International Business Machines* (IBM) [2004-67] et Microsoft [2004-62] ont aussi rendu publique leur vision de l'IDM. Ainsi, l'IDM prend son origine dans la convergence de toutes ces visions et des avancées techniques de chacun.

L'originalité de l'IDM ne réside pas dans le recours systématique aux modèles pour le développement logiciel comme le laisserait entendre sa terminologie [2004-68]. Plusieurs méthodes de modélisation, telles que Merise ou SSADM, préconisent aussi l'utilisation de modèles mais dont le rôle s'achève aux phases amont du développement logiciel : l'analyse et la conception. Les modèles servent alors à faciliter la communication et la compréhension entre les différents acteurs mais n'interviennent pas dans la phase de production, de maintien et d'évolution. Nous parlons dans ce cas de modèles « contemplatifs ».

L'IDM a pour objectif de rendre les modèles « productifs » sur tout le cycle de vie du système et à tous les niveaux d'abstraction. Pour y parvenir, les modèles doivent être décrits formellement afin d'être interprétés et exécutés par une machine. Dès lors, ces modèles permettent d'industrialiser la production logicielle, jusque-là centrée sur le code produit par l'informaticien [2005-69].

3.2 Concepts fondamentaux

En mettant à profit des disciplines telles que la modélisation par objets, l'ingénierie des langages, la compilation de langages, les méthodes formelles ou encore la programmation par composants, l'IDM offre un cadre intégrateur reposant sur quelques concepts fondamentaux : la notion de modèle et la relation *Représente*, la notion de métamodèle et la relation *ConformeÀ*.

3.2.1 Modèle et Représentation

La notion de modèle est centrale dans l'IDM car, comme nous venons de l'évoquer, l'enjeu de cette approche est de rendre les modèles productifs sur tout le cycle de vie du système. Il n'existe pas de définition universelle de la notion de modèle. En nous appuyant sur les définitions données dans les travaux [1967-70] [2001-63] et [2003-71], nous adoptons la définition suivante du terme modèle :

Définition 7. *Un modèle est une abstraction d'un système, selon le bon point de vue, qui permet de répondre à des questions prédéfinies sur ce système en lieu et place de celui-ci.*

De cette définition découle la première relation fondamentale de l'IDM qui lie le modèle et le système qu'il représente. Celle-ci est nommée *Représente* et notée μ. Bien que la relation *Représente* ne soit pas nouvelle dans l'ingénierie logicielle (Merise, UML), l'IDM a permis d'en définir les contours [2003-72] [2003-71] [2004-66].

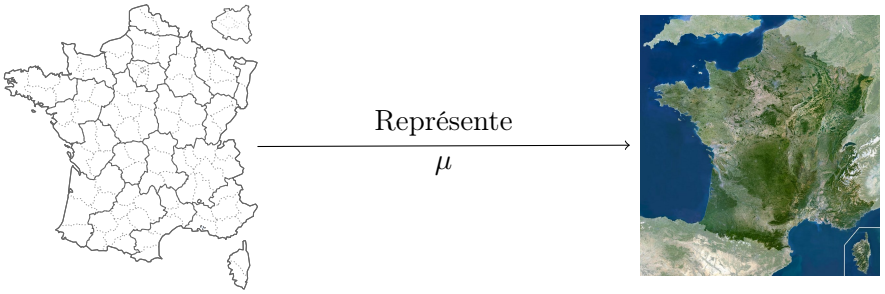

FIGURE 3.1 – Relation entre système et modèle [2006-1]

Cette définition n'est pas restreinte à l'informatique et pourrait s'appliquer à n'importe quel système. La figure 3.1 reprend l'exemple connu de la cartographie où une carte géographique joue le rôle de modèle pour le territoire français qui joue le rôle su système réel.

L'intérêt de l'IDM est de produire des modèles exploitables informatiquement. Ceci n'est possible que si ces modèles sont décrits par des langages formels. Il devient alors important de bien définir ces langages à l'aide de métamodèles.

3.2.2 Métamodèle et Conformité

L'originalité de l'IDM ne réside pas dans la relation de représentation qui trouve plutôt son origine dans les méthodes de modélisation telles que Merise. L'apport de l'IDM est dans l'utilisation systématique de métamodèles pour la description des langages de modélisation.

Il existe plusieurs définitions de la notion de métamodèle dans la littérature. Cependant la définition suivante est communément admise [2004-68].

Définition 8. *Un métamodèle est un modèle du langage de modélisation qui sert à exprimer les modèles.*

Une autre définition courante mais erronée de la notion de métamodèle suppose qu'un métamodèle est un modèle de modèle. La figure 3.2 reprend l'exemple de la cartographie évoquée plus haut. Nous appliquons récursivement la relation *Représente* (μ) au territoire français. Ici, une carte de la France joue le rôle de modèle du territoire français. Un fichier XML spécifiant l'éditeur, l'année d'édition et la langue de carte joue le rôle de modèle de cette carte dans une base de données. Dans ce contre-exemple, le fichier XML n'est pas un métamodèle de la carte administrative de la France car il ne décrit pas les concepts utilisées par la carte et leurs relations. Un métamodèle n'est donc pas un modèle d'un modèle.

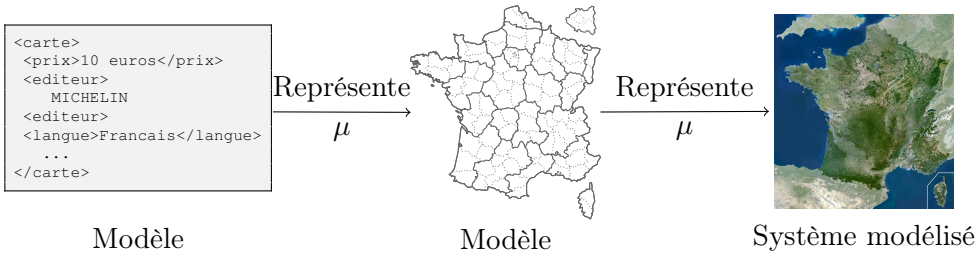

FIGURE 3.2 – Modèle de modèle selon l'exemple de la cartographie [2006-1]

Par ailleurs, le concept de métamodèle induit la deuxième relation fondamentale de l'IDM liant un modèle à son métamodèle. Cette relation est nommée *ConformeÀ* et notée χ [2004-66] [2004-73]. La figure 3.3 reprend l'exemple de la cartographie. La légende de la carte peut être vue comme le métamodèle de la carte de la France car elle spécifie les concepts utilisés pour modéliser la carte (département et région) et leurs relations (une région comporte des départements). Pour être lisible, la carte doit être conforme à la légende, c'est-à-dire utiliser uniquement ces concepts. Cependant, la légende est un cas particulier de métamodèle qui définit la manière dont sont représentés les concepts.

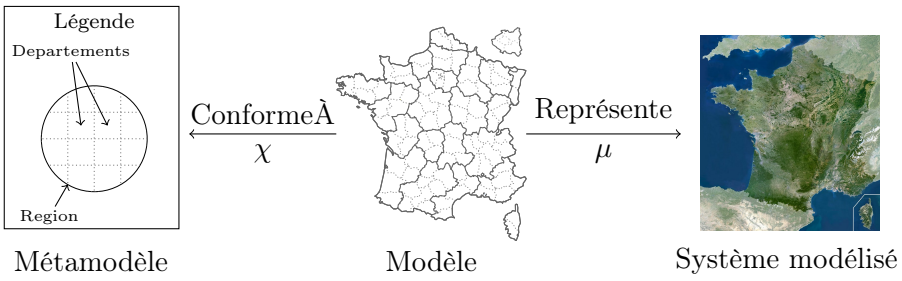

FIGURE 3.3 – Relations entre système, modèles, métamodèle et langage de modélisation [2006-1]

3.3 Transformation de modèle

Comme expliqué précédemment, la préoccupation majeure de l'IDM est de rendre les modèles opérationnels sur tout le cycle de vie des systèmes logiciels, depuis l'analyse et la conception jusqu'à la maintenance et l'évolution. La transformation de modèle se retrouve au cœur de l'IDM car c'est à travers elle que se fait l'automatisation des traitements apportés aux modèles. Dans cette section, nous donnons une définition de la transformation de modèle avant d'en présenter les types et les usages.

3.3.1 Définition de la transformation de modèle

L'OMG définit une transformation de modèle comme « le processus consistant à convertir un modèle en un autre modèle d'un même système » [2011-74].

Kleppe et al. [2003-44] proposent une définition moins générique en insistant sur l'aspect automatique de ce processus : « une transformation de modèle consiste en la transformation automatique d'un modèle source en un modèle cible, selon une description établie de cette transformation ». Cette définition implique qu'une transformation est décrite à un niveau d'abstraction au dessus de celui des modèles : au niveau d'un métamodèle auquel elle doit se conformer.

Mens et Van Gorp [2006-75] étendent cette définition en considérant qu'une transformation est une opération qui peut avoir en entrée un ou plusieurs modèles source et en sortie un ou plusieurs modèles cible :

Définition 9. *Une transformation génère automatiquement un ou plusieurs modèles cible à partir d'un ou plusieurs modèles source, selon une description établie de la transformation.*

C'est cette dernière définition que nous adoptons dans ce document. Par ailleurs, notons que, si les métamodèles source et cible sont différents, la transformation est dite exogène. Si les métamodèles source et cible sont identiques, la transformation est dite endogène. Ces termes sont introduits par Mens et Van Gorp [2006-75].

3.3.2 Composants d'une transformation de modèle

La figure 3.4 illustre les composants d'une transformation de modèle : les modèles source, les modèles cible, la définition de la transformation et le moteur qui va effectuer la transformation selon sa définition.

La description de la transformation définit la manière de transformer un ou plusieurs modèles source en un ou plusieurs modèles cible. Elle est créée dans un langage de transformation de modèle. Par exemple, si c'est un langage à base de règles, la description de la transformation consiste en un ensemble de règles de transformation à effectuer sur les modèles cible [2003-44].

Un moteur de transformation exécute ou interprète la description. Il applique donc la description aux modèles source pour produire les modèles cible en suivant les étapes ci-dessous [2005-76] :

- identifier l'élément du ou des modèles source à transformer ;
- pour chaque élément identifié, produire l'élément cible qui lui est associé dans le ou les modèles cible ;
- produire une trace de la transformation qui lie les éléments du ou des modèles cibles aux éléments du ou des modèles source.

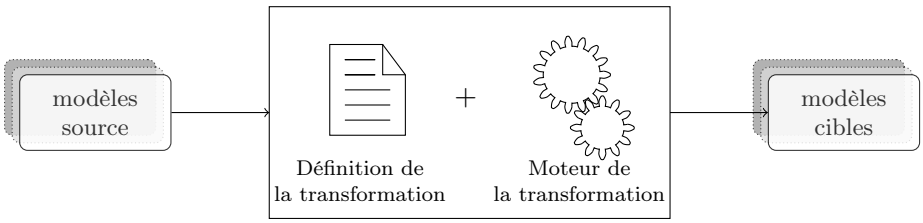

FIGURE 3.4 – Composants d'une transformation de modèle

3.3.3 Usages de la transformation de modèles

Les transformations de modèles sont au cœur d'une démarche dirigée par les modèles : elles permettent d'automatiser les manipulations subies par les modèles telles que la modification, la création, l'adaptation, la composition ou encore le filtrage de modèles, à travers la réutilisation systématique d'informations contenues dans les modèles existants.

Il est possible de recourir aux transformations de modèles sur tout le cycle de vie d'un système. Les usages les plus répandus sont le raffinement, l'intégration d'outils, la composition, l'analyse, la simulation et l'optimisation que nous présentons dans la suite de ce document.

3.3.3.1 Raffinement

Le raffinement consiste à rajouter plus de détails au modèle initial. Ce type de transformation peut aussi bien être endogène (métamodèles source et cible identiques) ou exogène (métamodèle source et cible différents). Le raffinement se prête parfaitement à toute la partie descendante du cycle en V où les modèles passent à des niveaux d'abstraction plus bas. Ceci revient à faire des transformations successives de type modèle-à-modèle et une transformation de type modèle-à-texte pour aboutir au code final.

Raffiner un modèle revient à décomposer des concepts de haut niveau, à choisir un algorithme particulier, à spécialiser un concept pour un contexte donné ou encore à le concrétiser sous forme d'une solution exécutable par une machine en générant le code à partir de modèles de plus haut niveau d'abstraction [2000-77].

3.3.3.2 Intégration d'outil

Il existe une panoplie d'outils disponibles pour créer, manipuler, analyser ou encore simuler des modèles. Souvent ces outils utilisent des métamodèles internes et des espaces techniques qui leurs sont propres. Ainsi, l'échange de modèles entre ces outils est compromis et l'interopérabilité est fortement entravée. L'utilisateur se trouve obligé d'utiliser un seul et même outil sur tout le cycle de vie du système et ne peut donc pas tirer avantage des possibilités offertes par d'autres outils plus adaptés à ses besoins à certaines étapes.

L'intégration d'outil est une solution pour pallier la divergence syntaxique et sémantique des outils et des langages de modélisation par le biais la transformation de modèle [2005-76]. Ce type de transformation permet de naviguer entre deux métamodèles, de synchroniser des modèles qui évoluent séparément sur des outils distincts, d'établir des correspondances entre métamodèles pour maintenir la cohérence des modèles conformes à ces métamodèles. Il sera donc possible de faire appel à des outils mieux adaptés à chaque étape du cycle de vie.

3.3.3.3 Composition

Pour réduire la complexité inhérente à la modélisation et à l'analyse de grands systèmes, tels que les Smart Grids par exemple, il est possible d'adopter une approche par points de vue qui permet de séparer les préoccupations. Les modèles produits correspondent donc à ces différents points de vue qu'on peut ainsi valider séparément dans un premier temps. A l'issue de cette approche modulaire, on pourra composer ces modèles, c'est-à-dire les assembler, pour aboutir un modèle global du système.

Dans le cas le plus simple, les deux modèles à composer sont conformes à un même métamodèle. Cependant, il est aussi possible de composer deux modèles conformes à deux métamodèles différents.

Les deux modèles à composer peuvent aussi présenter des concepts en commun. Deux techniques existent pour composer des modèles :

- la première technique consiste à les fusionner. Dans ce cas, le modèle final résultant de la composition doit contenir toutes les informations issues des modèles initiaux, sans duplication des informations communes [2006-78]. [2008-79] présente un framework générique capable de composer des modèles indépendamment de leurs langages de modélisation. L'approche consiste à identifier les éléments qui représentent le même concept dans les deux modèles à composer et à les fusionner dans un nouveau modèle qui représente une vue intégrée de ces concepts. Il est aussi possible de spécialiser le framework pour un métamodèle particulier mais qui reste conforme au MOF ;
- la deuxième technique consiste à les tisser. Dans ce cas, on crée des correspondances entre les éléments qui représentent un même concept. Un métamodèle générique est créé pour définir les correspondances qui sont donc modélisées dans le modèle final. On y retrouve donc les éléments en commun dupliqués mais liés par un lien de correspondance.

Il est à noter que le modèle issu du tissage de deux modèles M_A et M_B peut être utilisé comme modèle intermédiaire que l'on note M_T pour la fusion de M_A et M_B. Dans ce cas, l'opération de fusion consiste à produire un modèle M_{AB} en prenant comme entrée M_A, M_B et M_T. Cette technique est notamment utilisée par [2007-80] pour la composition semi-automatique de modèles.

3.3.3.4 Simulation

La transformation de modèle peut être utilisée pour simuler des modèles. En effet, une transformation de modèle peut mettre à jour le système modélisé. Dans ce cas, le modèle cible est une mise à jour du modèle source et la transformation est de type sur-place (modèles source et cible confondus).

Par exemple, [2011-81] simule un comportement simple d'un jeu de Pacman en utilisant la transformation de modèle. La transformation spécifie les règles de transition qu'une instance du jeu peut prendre (Pacman et fantôme se trouvant dans la même case, Pacman et pomme se trouvant dans la même case, etc.). En ingénierie des langages, ceci revient à définir la sémantique opérationnelle d'un langage de modélisation. L'exécution de la transformation anime le modèle en fonction du comportement qu'on lui confère.

La transformation peut aussi être utilisée comme intermédiaire dans la simulation de modèle. Des modèles en entrée d'un outil de simulation externe sont produits par une transformation des modèles que l'on souhaite simuler. Cette technique permet de tirer profit d'outils de simulation existant sur le marché en utilisant l'intégration d'outils.

3.3.3.5 Analyse et optimisation

La transformation de modèle peut être utilisée pour les activités d'analyse de modèle. Une analyse simple telle que le calcul de métrique de similarité entre deux modèles via la transformation de modèle est donnée dans [2007-80] avec un modèle de transformation écrit en *Atlas Transformation Language* (ATL) [2006-82].

Des analyses plus complexes sont possibles grâce à l'intégration d'outils d'analyse externes vers lesquels les modèles source sont transformés.

[2010-83] propose d'utiliser la transformation de modèle pour l'analyse de sûreté de fonctionnement dans le domaine de l'automobile. Les modèles source sont transformés en modèles conformes au métamodèle de l'outil d'analyse de sûreté de fonctionnement retenu.

L'optimisation vise à améliorer les propriétés non fonctionnelles des modèles telle que l'évolutivité, la fiabilité, la modularité, etc. L'optimisation est typiquement utilisée sur les modèles d'architecture. Les transformations utilisées pour l'optimisation sont de type endogène car on cherche à affiner la conception de modèles existants. La réingénierie est un exemple de transformation utilisée pour optimiser les modèles en cherchant à améliorer la maintenabilité, la lisibilité et l'évolutivité des modèles.

3.3.4 Approches existantes pour la transformation de modèle

Le recours à la transformation de modèle est l'objet de recherches informatiques antérieures à l'apparition de l'approche IDM. Par exemple, les compilateurs utilisent la transformation pour passer du code source au fichier binaire [1985-84]. Même si ce type de transformation trouve son origine dans le domaine de la programmation informatique, la transformation de modèle embrasse un champ d'application plus large encore.

Nous trouvons dans la littérature plus d'une trentaine d'approches différentes de transformation de modèle [2011-81]. Czarnecki et Helsen proposent une classification de ces approches selon plusieurs critères tels que le paradigme retenu pour définir la transformation, la relation entre les modèles sources et cibles, la directivité de la transformation, le nombre de modèles cible et source, l'orchestration et l'ordonnancement des règles de transformation, etc. [2006-85]. [2011-86] définit les trois grandes catégories d'approches suivantes.

Par programmation : Les modèles offrent une interface qui permet d'écrire les transformations dans un langage de programmation. Mais cette technique relève plus de la programmation que de la modélisation. Ce sont en fait des applications informatiques qui ont la particularité de manipuler des modèles. L'avantage de cette approche est qu'elle utilise un langage de programmation généraliste tel que Java ou C++ pour écrire les transformations. Ainsi le programmeur n'a pas besoin d'apprendre un nouveau langage. Cependant ces applications ont tendance à devenir difficilement maintenables.

Par template : Dans cette approche, des canevas des modèles cible sont définis. Ces modèles contiennent des paramètres qui sont remplacés par les informations contenues dans les modèles source. Ce type de transformation est souvent utilisé pour les transformations qui génèrent des modèles dont la syntaxe concrète est textuelle. Cette approche est associée au patron de conception *visitor* qui va traverser la structure interne du modèle source.

Par modélisation : Cette approche vise à appliquer les principes de l'IDM aux transformations de modèle elles-mêmes. Les modèles de transformation sont ainsi pérennes, réutilisables et indépendants des plates-formes d'exécution [2006-87]. Pour cela, des langages de modélisation dédiés à l'activité de transformation de modèle sont utilisés. Cette approche considère donc la transformation comme un modèle à part entière conforme à un métamodèle de transformation. La figure 3.5 illustre cette approche en positionnant la transformation par rapport aux niveaux d'abstraction de l'IDM. Elle corrobore ainsi la vision unificatrice de l'IDM à travers le paradigme du « tout est modèle » [2005-69].

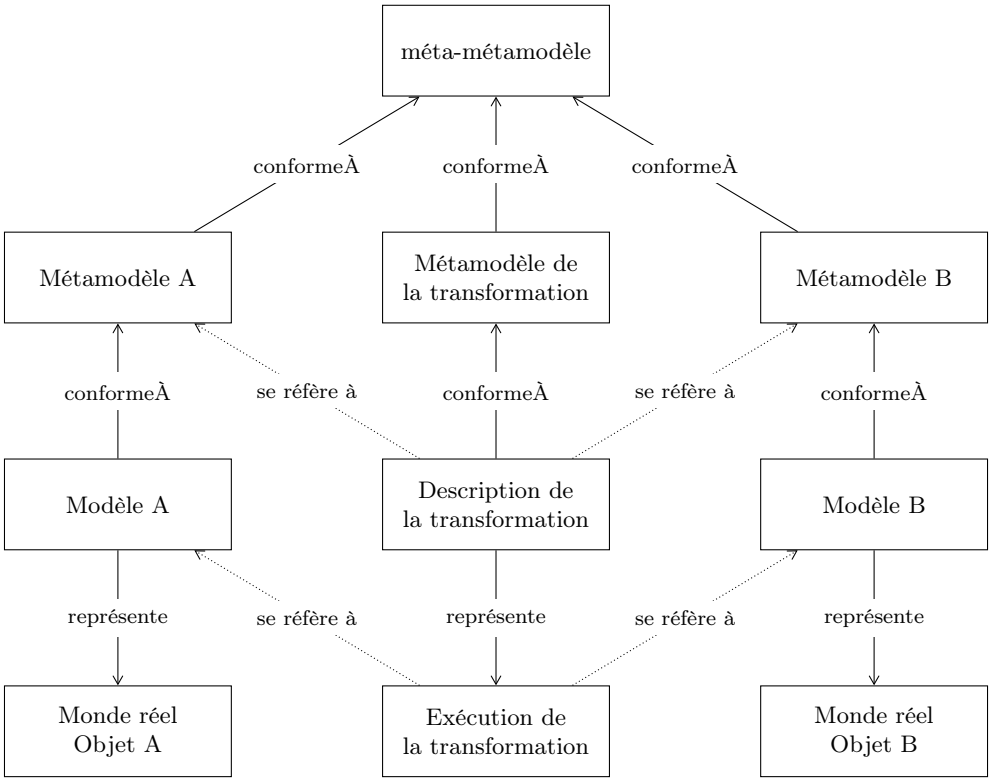

FIGURE 3.5 – Méta niveaux d'une transformation de modèle

3.3.5 Langages et outils pour la transformation de modèle

Sans viser l'exhaustivité, nous introduisons succinctement quelques langages et outils dédiés à la transformation de modèles.

3.3.5.1 ATL

ATL [1] est né de la volonté de proposer des langages de modélisation dédiés à la transformation de modèle en définissant un métamodèle et des outils pour l'exécution des transformations.

ATL est un langage hybride (déclaratif et impératif) à base de contraintes *Object Constraint Language* (OCL) [2006-82].

ATL peut réaliser des transformations sur place, c'est-à-dire, une transformation où le modèle source et le modèle cible sont confondus en utilisant le mode raffinement de modèle.

3.3.5.2 QVT

Query View Transformation (QVT) [2008-88] [2011-74] est un standard de l'OMG. Le métamodèle de QVT est conforme au MOF. Comme ATL, QVT se base sur OCL pour accéder aux éléments des modèles. QVT définit deux langages de transformation de type modèle-à-modèle. QVT *Declarative* (QVTd) [2] est un langage déclaratif. QVT *Operational* (QVTo) est un langage impératif [3].

3.3.5.3 Kermeta

Kermeta [4] est un langage généraliste de méta-modélisation exécutable et de méta-programmation orientée objet qui peut aussi décrire des transformations de modèle. Intégré à EMF, il est doté d'un métamodèle conforme au MOF qu'il étend avec un langage d'action impératif utilisé pour écrire le corps des opérations définies sur les concepts d'une syntaxe abstraite (ce qui revient à doter une syntaxe abstraite d'une sémantique opérationnelle). On peut ainsi décrire n'importe quel traitement sur un modèle, ce qui est assimilé à une transformation de modèle.

1. http ://www.eclipse.org/atl/
2. http ://projects.eclipse.org/projects/modeling.mmt.qvtd
3. http ://projects.eclipse.org/projects/modeling.mmt.qvt-oml
4. http ://www.kermeta.org/

3.3.5.4 Acceleo

Acceleo [5] est un langage de transformation de modèles pour lequel les modèles cible ont une syntaxe concrète textuelle. Il suit une approche de transformation par *template*. Le template correspond au modèle cible qui prend la forme d'un texte avec des espaces réservés aux informations provenant des modèles source. Accelo utilise OCL pour accéder au modèles source.

L'IDM se cantonne souvent au processus de développement logiciel en offrant une aide supplémentaire à l'analyse et la validation des modèles de spécification et en automatisant certaines tâches à travers la génération de code par exemple. Mais des disciplines plus orientées métier telle que l'EA peuvent aussi tirer partie des avantages qu'apporte l'IDM à l'ingénierie logicielle.

3.4 L'Ingénierie Dirigée par les Modèles pour l'Architecture d'Entreprise

La méta-modélisation et les transformations de modèle sont des pivots majeurs dans la recherche liée à l'automatisation de la manipulation des modèles en ingénierie logicielle. Nous nous sommes attachés dans la suite de ce chapitre à identifier les approches existantes de modélisation en EA recourant à ces deux techniques incontournables de l'IDM.

Nous présentons dans la section 3.4.1 les approches d'EA recourant à la méta-modélisation et en particulier le langage Archimate avant de présenter quelques approches d'EA recourant aux transformations de modèles dans la section 3.4.2. Enfin, l'IDM préconise le recours aux langages exécutables sur tout le cycle de vie du système logiciel (de la conception à l'implémentation). Nous avons dès lors établi, dans la section 3.4.3 un ensemble de critères qui nous ont permis d'identifier différents langages de modélisation exécutables pour représenter des architectures d'entreprise tout en automatisant leur analyse.

3.4.1 Méta-modélisation en Architecture d'Entreprise

Les cadres d'EA contribuent à réduire la complexité liée à un système telle que l'entreprise en décomposant celle-ci en plusieurs vues. Les architectes d'entreprise doivent ensuite représenter les artefacts qui composent ces différentes vues. Le recours aux techniques de modélisation n'est donc pas étranger à l'EA. Les architectes d'entreprise ont besoin d'une représentation claire et pertinente des composants de l'entreprise pour leur propre compréhension mais aussi pour faciliter la communication avec les autres parties prenantes comme les experts métier, les architectes fonctionnels, les architectes applicatifs, etc.

5. http ://www.eclipse.org/acceleo/

Souvent, les cadres d'architecture ne préconisent pas de langages de modélisation en particulier et se contentent d'ériger un ensemble de bonnes pratiques. Pour représenter l'entreprise, les architectes utilisent alors leurs propres systèmes de notations. Si deux architectes appartenant à des filiales différentes d'une même entreprise utilisent des notations différentes, des problèmes de communication, de collaboration et d'interopérabilité à l'échelle de l'entreprise risquent d'apparaître. L'architecture de l'entreprise perd alors de sa crédibilité en tant que référentiel d'entreprise.

Graphiques ou textuelles, ces notations manquent de plus d'une sémantique précise et formalisée, ce qui entraîne la création de modèles purement contemplatifs. Les outils de visualisation et d'analyse sont d'autant plus difficiles à développer.

L'*Open Group*[6] propose de mettre en cohérence les notations utilisées en EA en créant un langage de modélisation standardisé nommé Archimate. Archimate sert ainsi à décrire l'entreprise, sa structure organisationnelle, ses processus, ses règles métier ou encore son SI. Archimate est un langage de représentation graphique utilisé pour documenter et communiquer les architectures d'entreprise au sein d'une même organisation ou entre différentes organisations. Archimate est en outre étroitement lié au cadre d'architecture TOGAF.

Bien que concis, le métamodèle d'Archimate vise à couvrir la majorité des concepts intervenants dans la création d'une architecture d'entreprise. Archimate utilise trois vues : métier, applicative et technique. Pour chacune des vues, trois types d'éléments différents sont spécifiés : structure active, comportement et structure passive (voir figure 3.1). Les structures actives correspondent aux éléments dotés d'un comportement. Il peut s'agir d'un acteur métier, d'un module applicatif ou d'un élément de l'infrastructure technique. Le comportement est défini comme un ensemble d'activités ou de tâches poursuivies par un ou plusieurs éléments de la structure active. Il peut s'agir par exemple d'un processus métier. La structure passive comporte les éléments manipulés par les éléments de la structure active, comme par exemple les objets métier.

	Structure passive	**Comportement**	**Structure active**
Métier			
Application			
Techonologie			

Table 3.1 – Composants du langage Archimate

Le métamodèle du langage Archimate possède une syntaxe concrète graphique permettant visualiser les modèles. Elle associe par exemple des couleurs différentes aux éléments en fonction de leur type. Les éléments de type structure active sont bleus, ceux de type comportement sont jaunes et enfin ceux de type structure passive sont verts. Mais aucune

6. `http://www.opengroup.org/standards?tab=1`

sémantique standard d'exécution n'est formalisée. Les modèles d'architecture sont ainsi souvent purement contemplatifs et les outils implémentant Archimate n'offrent pas la possibilité d'exécuter ces modèles à des fins d'analyse de structure et de comportement.

3.4.2 Approches d'Architecture d'Entreprise recourant à l'Ingénierie Dirigée par les Modèles

L'IDM a prouvé sa capacité à traiter des systèmes complexes [2007-89]. L'application des techniques de l'IDM aux approches d'EA fait l'objet de quelques rares travaux [2013-56]. Les travaux de [2003-8] font figure de précurseurs. Ces derniers décrivent un mapping entre le cadre Zachman et les vues MDA (figure 3.2). Mais le périmètre de ces travaux se réduit à l'architecture IT de l'entreprise plutôt qu'à l'entreprise dans son ensemble.

Abstractions (colonnes) →

Perspectives (lignes) ↓

	Données *Quoi*	**Fonctions** *Comment*	**Personnel** *Qui*	**Temps** *Quand*	**Motivation** *Pourquoi*	
Exécutif *Planification*	Identification des données	Identification des processus	Identification des responsabilités	Identification des échéances	Identification des motivations	CIM
Management *Définition*	Définition des données	Définition des processus	Définition des responsabilités	Définition des échéances	Définition des motivations	
Architecte *Conception*	Conception des données	Conception des processus	Conception des responsabilités	Conception des échéances	Conception des motivations	PIM
Ingénieur *Spécification*	Spécification des données	Spécification des processus	Spécification des responsabilités	Spécification des échéances	Spécification des motivations	PSM
Technicien *Implémentation*	Implémentation des données	Implémentation des processus	Implémentation des responsabilités	Implémentation des échéances	Implémentation des motivations	Code

TABLE 3.2 – Mapping Zachman/MDA [2003-8]

[2014-5] propose d'étendre la démarche MDA, traditionnellement appliquée au développement de systèmes informatiques, à l'ensemble de l'entreprise pour mettre en place une organisation dirigée par les modèles ou MDO (*Model Driven Organisation*). Comme l'illustre la figure 3.6, une MDO comprend un modèle de l'organisation (*Model of the Organization*), un modèle spécifique à une plateforme (*Platform Specific Model*) et une plateforme d'entreprise (*Platform for Organization*).

Le modèle de l'organisation correspond à l'ensemble des modèles métier présentant les processus, les objectifs, les acteurs, les ressources. L'organisation fait appel à son IT pour réaliser ses objectifs métier. L'ensemble du système informatique correspond alors à la plateforme opérationnelle de l'entreprise. Comme pour MDA, le modèle spécifique à une plateforme est dérivé du modèle de l'organisation et sert de pivot pour générer automatiquement la plateforme de l'organisation par transformation de modèles.

Dans leurs travaux, Clark et al. [2014-5] proposent un cadre théorique et une étude d'opportunité quant à la généralisation du MDA à l'échelle de l'entreprise. Cependant MDA reflète la vision particulière de l'OMG concernant l'IDM. L'approche MDA est de ce

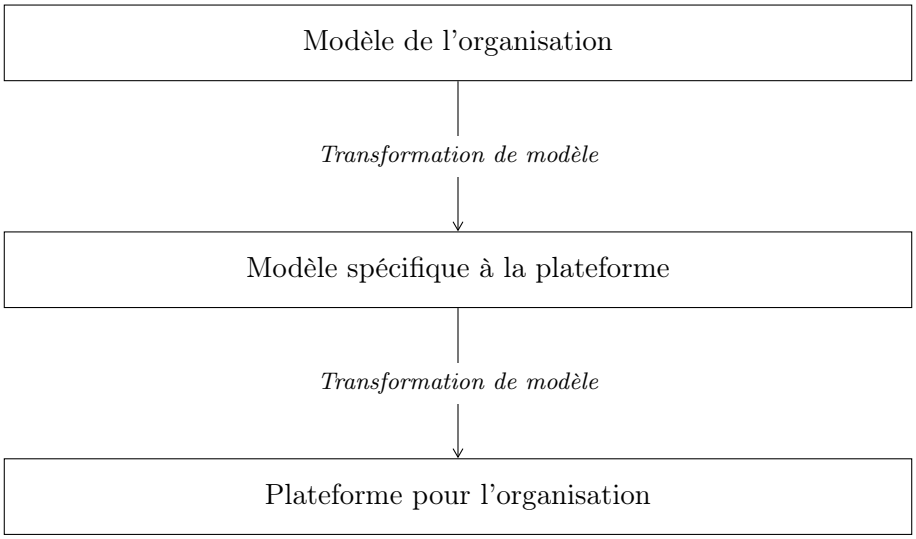

FIGURE 3.6 – Architecture d'une organisation dirigée par les modèles [2014-5]

fait restreinte aux standards de l'OMG. Les travaux de Clark et al. sont donc limités par l'approche MDA (celle-ci étant un sous-ensemble de l'IDM).

L'IDM préconise le recours aux modèles exécutables pendant tout le cycle de vie du système à implémenter. Cependant, à l'échelle d'une entreprise, l'utilisation de ce type de modèles est souvent limitée à l'implémentation des systèmes informatiques, c'est à dire à l'implémentation des vues applicative et technique d'une architecture d'entreprise. L'usage des modèles en tant qu'artefacts exécutables est peu répandue en EA [2013-47].

Parmi les travaux mettant à contribution les techniques de l'IDM nous citons ceux de Clark et al. [2011-50]. Ces derniers proposent un langage concis et exécutable pour modéliser et simuler les différentes vues d'une architecture d'entreprise. Cependant ce langage est principalement textuel. Il offre une visualisation graphique des aspects structuraux d'une organisation sous la forme d'un diagramme de classes. Mais la syntaxe concrète pour la spécification du comportement est essentiellement textuelle. L'adoption de ce langage par les parties prenantes à profil non technique n'est pas évidente. Or l'EA a pour vocation d'offrir un référentiel compréhensible et partagé par toutes les parties prenantes.

Les travaux de Brunelière et al. [2013-90] tirent profit des capacités de l'IDM à automatiser la manipulation des modèles pour offrir une assistance dans la création et la visualisation d'architectures d'entreprise. La plateforme proposée s'appuie sur TOGAF et fournit un support pour la gouvernance et la prise de décision souvent gérées manuellement. Ces travaux se focalisent sur les techniques IDM permettant 1°) de concevoir une cartographie de l'existant par rétro-ingénierie à partir des données disponibles dans le SI, 2°) d'adapter la représentation graphique des modèles au besoin des utilisateurs concernés, 3°) de gérer plusieurs vues d'un même système et d'automatiser leur manipulation. Brunelière et al. [2013-90] démontrent que les techniques de l'IDM apportent des réponses à certaines

limitations liées aux pratiques actuelles d'EA mais ne supportent pas la simulation des modèles obtenus.

Manzur et al. [2015-58] utilisent les techniques de l'IDM (en particulier la métamodélisation) pour analyser la composante dynamique des modèles Archimate à travers la simulation de propriétés non fonctionnelles. Dans un premier temps, ils réduisent le métamodèle d'Archimate en ne retenant que les concepts indispensables à la modélisation des propriétés non fonctionnelles. Comme Archimate est d'abord conçu pour modéliser la structure des architectures d'entreprise, Manzur et al. enrichissent dans un deuxième temps le métamodèle obtenu avec deux autres types de concepts 1°) des concepts pour décrire le comportement et 2°) des concepts uniquement liés à l'exécution.

Les travaux de Manzur et al. [2015-58] illustrent les avantages de la métamodélisation et la définition d'une sémantique formelle pour l'exécution de modèles d'architecture d'entreprise. Néanmoins, ces travaux ne concernent que les propriétés non fonctionnelles des modèles d'architecture. Ces travaux formalisent aussi bien le métamodèle de la vue métier que ceux des vues applicative et technique. Cependant la simulation des différentes vues se fait séparément. L'approche ne permet donc pas de simuler simultanément l'ensemble de l'architecture.

3.4.3 Langages exécutables pour l'Architecture d'Entreprise

L'automatisation de l'analyse des modèles d'architecture d'entreprise s'avère particulièrement cruciale dans le contexte particulier des Smart Grids : évolution des cadres législatifs, apparition de nouveaux partenaires, hétérogénéité des interactions avec les clients finaux via les compteurs intelligents, les smartphones, les tablettes tactiles, etc. Or l'exécution des modèles est indispensable à l'automatisation de l'analyse de leur structure et de leur comportement. Exécuter des modèles apporte ainsi une aide précieuse à la validation d'une architecture d'entreprise dès les premières phases de son cycle de vie et augmente son agilité et son évolutivité. D'une part, l'exécution des modèles facilite leur exploration par les experts métier en levant les ambiguïtés engendrées par les modèles purement contemplatifs. D'autre part, exécuter des modèles permet d'évaluer et de comparer des architectures de manière objective en définissant des métriques et des indicateurs d'évaluation.

L'exécution des modèles est rendue possible par la définition d'une sémantique exécutable du langage dans lequel ces modèles sont exprimés. La sémantique d'un langage correspond au sens que peuvent prendre les concepts manipulés et leurs agencements lorsqu'ils sont instanciés au niveau des modèles [2012-91]. La définition de la sémantique du langage dépend de l'objectif poursuivi : simulation, génération de code, vérification, compilation, etc. L'expression de la sémantique d'un langage fait l'objet d'intenses recherches en ingénierie des langages, et en particulier sous la thématique des langages formels [2007-92].

Comme l'atteste le manifeste d'IBM [2006-93], les axes principaux de l'IDM sont 1°) les standards ouverts, 2°) l'automatisation et 3°) la représentation directe. Compte tenu de ces axes, pour modéliser les différentes vues d'une architecture d'entreprise, nous préconisons

l'utilisation de **langages standardisés, exécutables et compréhensibles** par les acteurs concernés par la modélisation d'architectures d'entreprise pour les Smart Grids.

Nous identifions plusieurs langages, issus de l'ingénierie des langages et en particulier de l'IDM, pouvant satisfaire ces critères :

- Un sous-ensemble de UML [7] limité au diagramme de classes et au diagramme d'activité possède désormais une sémantique d'exécution décrite par le standard fUML [8] (*Foundational UML*). Les diagrammes de classes conviennent pour la description des modèles d'information tandis que les diagrammes d'activité sont adaptés à la description de la dynamique d'un modèle et au comportement attendu des fonctions ;
- Le standard BPMN [9] (*Business Process Model and Notation*) est un langage de modélisation graphique permettant de décrire tous les aspects d'un processus métier à l'aide d'un seul type de diagramme. Ce formalisme présente l'avantage d'avoir une sémantique d'exécution bien définie permettant le développement d'outils pour la simulation de modèles de processus métier. BPMN est parfaitement adapté à la description de processus métier ;
- Le standard OCL [10] est un langage textuel standard d'expression de contraintes. Il est adossé à UML pour exprimer les propriétés difficiles à capturer dans des diagrammes UML. L'exécution d'OCL se fait à travers la transformation de modèle en ciblant un langage d'expression de contraintes de plus bas niveau qui soit exécutable comme MiniZinc [2007-94], ou à travers son utilisation au niveau du métamodèle avec OCLinEcore [11].

3.5 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté les concepts et relations essentiels de l'IDM à savoir les concepts de modèle et de métamodèle ainsi que les relations de représentation (liant un modèle au système qu'il modélise) et de conformité (liant un modèle à son métamodèle). Nous avons ensuite passé en revue les principes de la transformation de modèle ainsi que quelques approches et langages employés pour transformer les modèles. Les transformations de modèle et la méta-modélisation sont au cœur de l'IDM. Ce sont les principaux pivots de recherche en IDM pour automatiser la manipulation de modèle et les rendre ainsi productifs. Dès lors, nous avons présenté un état de l'art des approches qui recourent à l'IDM pour mener les différentes activités d'EA. Enfin, nous avons présenté des critères permettant de sélectionner les langages exécutables issus de l'IDM pouvant bénéficier à l'EA.

7. http ://www.omg.org/spec/UML/
8. http ://www.omg.org/spec/FUML/
9. http ://www.omg.org/spec/BPMN/2.0/
10. http ://www.omg.org/spec/OCL/
11. wiki.eclipse.org/OCL/OCLinEcore

Ce chapitre clôt cette première partie consacrée à l'état de l'art de l'EA et de l'IDM. Nous proposons, dans la partie suivante, un *framework* pour mener des analyses de comportement et de structure d'architecture d'entreprise dénommé ExecuteEA. Ce *framework* s'appuie sur les techniques IDM que nous venons de présenter, dont la méta-modélisation, les transformations de modèles et les langages de modélisation exécutables.

Deuxième partie

Contribution — EAT-ME une approche unificatrice par les modèles

Chapitre 4

Le *framework ExecuteEA*

Sommaire

Dans la partie précédente nous avons présenté le domaine de l'EA en nous focalisant sur la revue de la littérature relative à l'analyse du comportement et de la structure d'une architecture d'entreprise. Nous avons aussi présenté le domaine de l'IDM en mettant en lumière les techniques mises en œuvre pour rendre productifs les modèles utilisés sur tout le cycle de vie d'un système logiciel, notamment la méta-modélisation et les transformations de modèle.

Dans ce chapitre, nous examinons tout d'abord les pratiques actuelles d'EA, dans la section 4.1, en nous appuyant sur une étude de la documentation de projets dédiés aux Smart Grids corroborant la revue de la littérature présentée dans la première partie de

ce mémoire. Forts de ce bilan, nous justifions le recours à une démarche IDM aboutissant à l'identification des concepts relatifs au domaine de l'EA (section 4.2), et nécessaire à la création d'un métamodèle pour l'EA que nous baptisons EAT-ME (section 4.3). Ce métamodèle est la pierre angulaire du *framework* ExecuteEA que nous destinons à l'analyse de la structure et du comportement d'une architecture d'entreprise (section 4.4).

4.1 Bilan des pratiques actuelles d'Architecture d'Entreprise et adoption d'une démarche d'IDM

4.1.1 Analyse des pratiques actuelles en Architecture d'Entreprise

L'emploi de l'EA peut servir différents desseins. Kurpjuweit et Winter [2007-95] classent les objectifs de l'EA en trois catégories : (1) la documentation et la communication, (2) l'analyse et la compréhension et (3) la conception et l'implémentation. Dans cette partie, nous exposons les limites des pratiques actuelles de l'EA au regard de ces trois catégories d'objectifs. L'identification de ces limites s'appuie essentiellement sur l'analyse de la documentation de projets Smart Grid à laquelle nous avons eu accès, mais aussi sur des entretiens avec les personnes impliquées dans la rédaction ou l'utilisation de cette documentation au sein du département Mesures et système d'Information des Réseaux Électriques (MIRE), ainsi qu'avec les experts en EA de EDF R&D. À ce titre, deux observations méritent d'être mentionnées :

- premièrement, le nombre restreint des documents consultés n'est pas représentatif des pratiques courantes, mais les limitations identifiées sont corroborées par les travaux académiques présentés dans le chapitre 2 portant sur l'état de l'art de l'EA ;
- deuxièmement, nous n'avons pas eu accès à des documents d'EA en tant que tels, pour des raisons de confidentialité et de réglementation interne à EDF. Cependant, nous avons estimé que la documentation à l'échelle d'un projet Smart Grid est suffisante pour appréhender une architecture d'entreprise dans sa globalité. En effet, l'ensemble des artefacts identifiés dans la taxonomie de Zachman [1987-7] pour une entreprise se retrouvent à l'échelle d'un projet mené au sein de cette même entreprise. Par exemple, ces projets impliquent souvent plusieurs acteurs, une multitude de processus métier, une décomposition de ce processus en fonctions, des applications à implémenter, des objectifs à accomplir, etc.

4.1.1.1 L'Architecture d'Entreprise pour la documentation

L'EA permet de capturer les composants de l'entreprise dans leur état courant et leur état cible. La manière la plus employée pour représenter ces composants reste la documentation [2013-55]. Ces documents couvrent des aspects tels que les processus métier, les fonctions, les applications et l'infrastructure technique de l'entreprise, ainsi que leurs

relations. Pour remplir leur rôle de référentiel d'entreprise, ces documents doivent être précis et régulièrement mis à jour. L'activité de documentation en EA, telle que pratiquée actuellement, est confrontée à des obstacles majeurs empêchant la production de documents suffisamment pertinents pour servir de feuille de route sur toute la trajectoire de l'entreprise.

Premièrement, la complexité de l'entreprise se retrouve dans sa représentation. En effet, une entreprise comme ERDF comporte des milliers de processus métier et des milliers d'applications supportant ses activités. Ces processus et ces applications sont, qui plus est, fortement interdépendants. Par conséquent, l'effort à fournir pour produire une documentation d'architecture pertinente est considérable. De plus, maintenir ces documents à jour n'est pas aisé du fait (1) du grand nombre d'artefacts à gérer et (2) du manque d'outils adéquats comme le confirment Shah et al. [2007-34] et Roth et al. [2013-96]. Or si la documentation n'est pas convenablement mise à jour, la valeur ajoutée et la crédibilité de l'EA sont remises en doute par l'ensemble de l'entreprise.

Deuxièmement, l'EA nécessite de collecter des informations provenant d'unités et de services différents. Cette collecte est bien souvent centralisée au niveau de l'entité IT de l'entreprise, où une seule équipe est chargée d'acquérir les informations provenant des autres entités. Cette étiquette IT porte souvent préjudice aux efforts d'EA au sein de l'entreprise en la réduisant à de l'architecture purement IT. Qualifiant l'EA d'« enjeu du siècle », Zachman déplore cet état de fait et soutient qu'il est nécessaire que l'EA transcende la sphère IT.

Troisièmement, les documents utilisés en EA ne sont pas homogènes. Selon leurs besoins, les membres de l'équipe responsable de la documentation ont en effet tendance à utiliser des représentations hétérogènes. Shah et al. [2007-34] font le constat que les équipes d'EA utilisent de surcroît des outils différents pour produire ces représentations, aboutissant à une documentation imprécise. Il est alors difficile d'expliciter et d'exprimer les relations entre les représentations. La cohérence de l'ensemble de l'architecture s'en trouve ainsi compromise.

Pour finir, la valeur ajoutée de l'EA diminue si la documentation produite n'est pas régulièrement mise à jour. Comme expliqué dans la section 2.3.1, les documents d'EA correspondent la plupart du temps à des modèles purement contemplatifs. C'est, de notre point de vue, l'une des raisons majeures qui expliquent le manque d'outils permettant d'automatiser cette mise à jour, voire même d'automatiser l'analyse de l'architecture d'entreprise comme expliqué dans la section suivante.

4.1.1.2 L'Architecture d'Entreprise pour l'analyse

L'analyse de la structure et du comportement de l'entreprise permet d'acquérir une connaissance plus approfondie de l'ensemble de l'architecture. L'analyse des composants de l'entreprise est limitée par la manière dont ces composants sont représentés, c'est-à-dire sous la forme de documentation. Comme ce type de représentation ne permet pas l'automatisation de l'analyse, dans la pratique, les choix d'architecture reposent entièrement sur le bon sens, l'expérience et l'expertise de l'architecte d'entreprise. Comme pour la documentation,

l'analyse de l'architecture d'une grande entreprise est une activité éprouvante et chronophage à cause du grand nombre d'artefacts manipulés et de leur forte interdépendance. Les architectes mènent souvent les activités d'analyse manuellement [2013-55], sans le support d'outils intégrant naturellement les cadres d'architecture.

Premièrement, la documentation de l'architecture telle que décrite dans la section précédente est souvent source d'erreurs à cause de la grande quantité et la forte interdépendance des artefacts manipulés [2005-54]. Analyser cette documentation est donc indispensable pour détecter ces erreurs de représentation.

Notre tour d'horizon des pratiques actuelles nous amène à constater que l'EA joue un rôle prépondérant durant les phases de description de l'existant et de spécification du système à mettre en œuvre, mais qu'elle n'est pas exploitée dans toute sa potentialité. L'entreprise gagnerait à promouvoir l'architecture d'entreprise au rang de référentiel accompagnant l'entreprise sur toute sa trajectoire. Ceci permettrait d'anticiper les incohérences potentielles entre les besoins formulés par le métier, et le SI qui est tenu de les satisfaire.

Pour ces travaux de recherches, plusieurs cas d'études ont été examinés, dont notamment la régulation de tension d'un réseau de distribution électrique [2012-97], le pilotage d'une batterie chez un particulier [2012-98] et la gestion d'une flotte de véhicules électriques [2015-19]. Qu'il s'agisse d'un projet de recherche européen (le pilotage d'une batterie chez un particulier), d'un projet interne à EDF (la régulation de tension), ou d'un projet opérationnel impliquant EDF ainsi que d'autres partenaires (la gestion d'une flotte de véhicules électriques), la documentation à laquelle nous avons eu accès ne va pas au delà de la description générale des fonctionnalités attendues. Cette documentation ne permet pas d'analyser afin de les valider la structure et le comportement du système avant son implémentation finale. Pour accéder à un niveau de détails suffisant à l'analyse de l'architecture de ces cas d'étude, une rétro-ingénierie et des échanges avec les développeurs et les concepteurs ont été nécessaires. Ces constatations tirées du terrain rejoignent les conclusions de l'état de l'art concernant les limites des « modèles contemplatifs » exposé dans le chapitre 3.

Deuxièmement, les mises en œuvre courantes de l'EA adoptent une approche descendante où l'architecte d'entreprise ne fait qu'aligner le SI à la stratégie de l'entreprise. Une démarche descendante implique que la vision et les grandes lignes de la stratégie sont conçues en amont de l'EA et indépendamment de la réalité du SI qui va l'implémenter. Les choix stratégiques sont alors faits dans « une tour d'ivoire » et sont ainsi déconnectés de la réalité du SI. Il nous paraît au contraire important que l'analyse de l'architecture d'entreprise permette d'évaluer et d'éclairer les choix stratégiques eux-mêmes, en plus de planifier leur déclinaison au niveau du SI.

Notre vision rejoint celle de l'école de pensée « *Enterprise System Architecting* » décrite par Lapalme [2012-6] et présentée dans la section 2.1.3. Cependant, les approches d'analyse telles que pratiquées actuellement ne permettent pas d'y parvenir. L'analyse et la validation de la stratégie est désincarnée de l'EA. Par exemple, les algorithmes d'optimisation et de simulation de processus métier s'adressent uniquement aux analystes métier et ne sont

pas couplés à la cartographie du patrimoine applicatif et aux capacités de l'infrastructure technique.

Inversement, l'analyse de l'architecture d'entreprise se cantonne à valider la robustesse et la résilience du SI en dressant l'inventaire des applications d'entreprise, des interfaces logicielles, des serveurs d'application, et de leur aptitude à mettre en œuvre la stratégie de l'entreprise, sans jamais chercher à l'adapter au SI. Appliquée à l'EA, l'approche descendante consiste à décomposer les éléments d'une vue supérieure en éléments plus détaillés dans la vue inférieure, présumant que la recomposition des éléments de la vue inférieure correspondra parfaitement aux éléments de la vue supérieure. Or, des travaux de recherche portant sur la complexité des SI actuels soulignent que des propriétés dites « émergentes » apparaissent quand un grand nombre de systèmes complexes sont fortement interconnectés [2004-99], comme c'est le cas de l'IT d'une entreprise de la taille d'EDF par exemple.

Une définition de l'*émergence* est donnée en sciences du Design par Gero [1992-100] :

Définition 10. *Une propriété qui est uniquement implicite, i.e. qui n'est pas représentée explicitement, est dite émergente si elle peut être explicitée.*

L'analyse de l'EA, doublée d'une approche ascendante (et non uniquement descendante), peut expliciter les propriétés émergentes de l'IT de l'entreprise et les porter au niveau du métier pour en extraire une plus-value pour l'entreprise et orienter avantageusement sa stratégie. L'EA est d'autant plus pertinente pour l'entreprise si l'analyse de l'architecture n'est pas uniquement destinée à exécuter la stratégie de l'entreprise, comme c'est souvent le cas, mais à mettre en place des relations bidirectionnelles (*i.e.* ascendantes et descendantes) entre les acteurs des différentes vues de l'architecture, pour mieux éclairer la prise de décision à tous les niveaux de l'entreprise. Ces relations bidirectionnelles seront alors gérées et orchestrées par l'architecte d'entreprise. [1]

4.1.1.3 L'Architecture d'Entreprise pour la conception et l'implémentation

L'observation des pratiques actuelles, à travers la revue de la documentation des projets Smart Grid évoqués dans la section 4.1.1.2 et à travers l'état de l'art présenté dans le chapitre 2, pointe vers le même constat : les artefacts de conception utilisées en EA se résument à une description de grands principes et à une liste de recommandations générales.

Ces descriptions, bien que nécessaires à la construction et au partage d'une vision globale de l'entreprise et des grandes étapes, ne sont pas suffisantes. Les limites que nous identifions rejoignent les constats de l'approche IDM pour le développement logiciel. Il s'agit là encore d'une représentation purement contemplative. Celle-ci n'est pas en mesure d'accompagner l'entreprise sur toute sa trajectoire en allant jusqu'à l'implémentation du système final, et ce pour plusieurs raisons.

1. c'est l'approche décrite dans la section 4.4, en particulier par la figure 4.12

Tout d'abord, les représentations purement contemplatives restent ambiguës et sujettes à des interprétations inadéquates. Les parties prenantes utilisent des langages hétérogènes et des systèmes de notations différents en fonction de leur besoin et de leur perspective. Un terme ou une annotation de la perspective métier peut vouloir dire quelque chose de différent s'il est utilisé dans une perspective IT. Par exemple, le terme « qualité » employé dans la vue métier peut vouloir dire « aptitude d'un processus à remplir un objectif métier ». Employé pour un processus applicatif, le terme « qualité » peut correspondre à « la bonne orchestration et l'interopérabilité des applications » sans considérer l'objectif métier qui leur a donné lieu. Ce décalage d'interprétation peut conduire à l'implémentation d'un système informatique performant mais qui ne répond pas aux objectifs métier de l'entreprise.

Ensuite, ces représentations ne sont pas assez réactives face à l'évolution de la stratégie de l'entreprise. L'effort de traduction du besoin métier en vision technique est essentiellement humain et consiste en la rédaction de spécifications sous la forme de listes, textes et figures. Mettre à jour ces spécifications suite à un changement de stratégie nécessite un effort et un temps non négligeables. D'ailleurs, les documents de spécification ne sont souvent pas mis à jour après les demandes d'évolution. Celles-ci sont directement implémentées dans le code, remettant en cause la traçabilité de l'évolution de l'entreprise.

Puis, nous estimons que l'ambiguïté et le manque de réactivité de ces représentations d'architecture expliquent, en partie, la raison pour laquelle elles ne sont pas suffisamment diffusées et partagées au sein de l'entreprise et ne l'accompagnent pas sur toute sa trajectoire. D'un point de vue organisationnel, les équipes des différents projets ne sont pas toujours conscientes de l'existence d'une architecture d'entreprise de référence [2007-34]. L'EA échoue par là à offrir un schéma directeur et une vue d'ensemble de l'état de l'entreprise et ne remplit pas son rôle de garant de la cohérence et de l'alignement métier/IT lors de l'implémentation du système final.

Enfin, les représentations purement contemplatives ne permettent pas d'exploiter les liens de traçabilité indispensables à la vérification de la cohérence de l'architecture sur toute la trajectoire de l'entreprise. La plupart des cadres d'EA ne représentent pas explicitement ces liens de cohérence. Ainsi, lorsque ces liens sont représentés, c'est sous une forme purement contemplative qui ne permet donc pas de les exploiter informatiquement *a posteriori* pour valider la cohérence globale de l'architecture. À titre d'exemple, ces liens sont utiles pour vérifier que tous les composants de la vue technique participent bien à l'exécution de la stratégie de l'entreprise.

4.1.1.4 Des représentations informelles, hétérogènes et incompatibles avec la constante évolution de l'entreprise

Qu'il s'agisse de la documentation et de la communication, de l'analyse et de la compréhension ou de la conception et de l'implémentation, l'EA est limitée par l'emploi et la production de représentations informelles et purement descriptives.

L'EA englobe ces différentes activités et les décline sur les différentes vues d'architecture

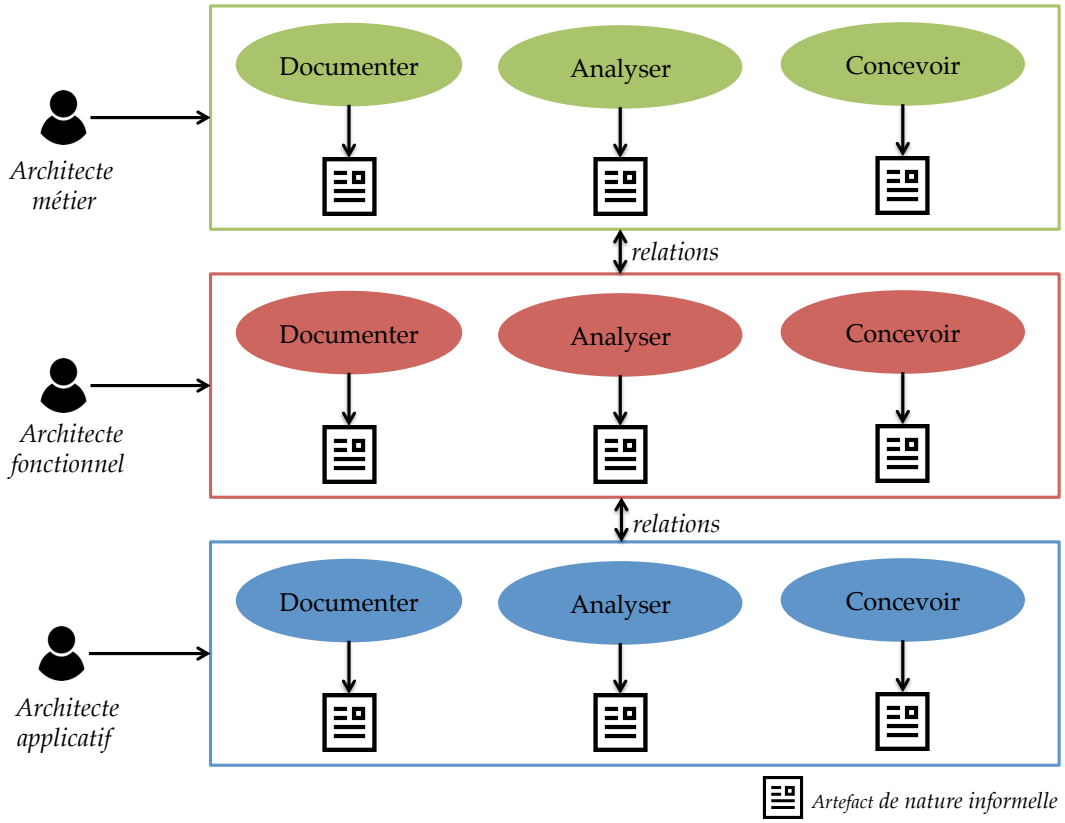

FIGURE 4.1 – Activités et artefacts produits pour les différentes vues d'une architecture d'entreprise

comme l'illustre la figure 4.1. Les artefacts produits au cours des différentes activités — documentation, analyse et conception — et pour les différentes vues — métier, applicative, technique —sont donc d'autant plus nombreux. Ces artefacts adressent les préoccupations de différentes parties prenantes.

La revue de la littérature relative à l'EA, nous a permis de catégoriser ces parties prenantes en identifiant un rôle par vue d'architecture : un architecte métier, un architecte fonctionnel, un architecte applicatif. Chaque rôles consiste à créer une vision holistique au regard de ses propres préoccupations. C'est donc pour cette raison que nous identifions un architecte par vue :

un architecte métier Le rôle de l'architecte métier consiste à documenter, analyser et concevoir des processus métier permettant à l'entreprise de réaliser sa stratégie. Il orchestre les processus métier et les flux d'information en cohérence avec la stratégie pour aboutir à l'architecture métier de l'entreprise ;

un architecte fonctionnel Le rôle de l'architecte fonctionnel consiste à documenter,

analyser et concevoir les blocs fonctionnels qui réalisent les processus de la vue métier. Il orchestre ces blocs fonctionnels et les flux d'informations qui circulent entre ces blocs pour aboutir à l'architecture fonctionnelle de l'entreprise ;

un architecte applicatif Le rôle de l'architecte applicatif consiste à documenter, analyser et concevoir les applications permettant d'implémenter les blocs de la vue fonctionnelle. Il doit assurer une bonne orchestration de ces applications et des informations échangées pour aboutir à l'architecture applicative de l'entreprise.

Les architectes sont essentiellement tenus de documenter, d'analyser et de concevoir des architectures selon leurs propres perspectives, en assurant les critères de *cohérence interne* à chacune des vues mais en respectant aussi les critères de *cohérence externe* avec les autres vues. Cependant, la nature informelle des artefacts impliqués dans les différentes activités d'architecture rend difficile le maintien de cette cohérence.

Comme souligné dans la dans l'introduction 1, les entreprises doivent constamment faire évoluer leurs stratégies. En conséquence, l'architecture doit satisfaire, en plus des contraintes de cohérence, des contraintes de souplesse et de réactivité. Mais souvent, la nature informelle des artefacts d'architecture ne permet pas de satisfaire les contraintes de souplesse et de réactivité.

Dans le cadre d'une démarche IDM, ce type d'artefacts est qualifié de purement contemplatif. L'IDM traite la question des modèles purement contemplatifs dans le contexte particulier du génie logiciel, en proposant une approche unificatrice par les modèles [2006-101]. Une revue comparative de la littérature relevant de l'IDM et des pratiques courantes de l'EA nous a amenés à faire le parallèle entre les deux domaines et à envisager l'IDM comme cadre méthodologique et technologique pour l'EA. Nous justifions le recours à une démarche IDM pour traiter les limites des pratiques actuelles dans la section suivante.

4.1.2 L'Ingénierie Dirigée par les Modèles : un cadre méthodologique et technologique

La problématique que traitent les présents travaux (voir chapitre 1) est résumée dans la question suivante :

Quels méthodes, modèles et outils adopter pour simuler une architecture d'entreprise afin de la valider ?

Pour répondre à cette problématique, nous avons adopté une démarche IDM. Nous commençons par démontrer la pertinence de ce choix de démarche au regard de la problématique traitée. Nous décrivons ensuite la dite démarche.

4.1.2.1 De la cohérence entre démarche adoptée et problématique traitée

Le recours à une démarche IDM tire sa légitimité de la mise en évidence des caractéristiques communes des deux disciplines que sont le génie logiciel et l'EA. L'IDM est certes originellement développée pour le génie logiciel, mais nous identifions un nombre de similarités entre les deux disciplines justifiant l'adéquation d'une telle démarche pour l'EA.

Premièrement, l'EA tout comme le génie logiciel, cherche à construire des systèmes manipulant de l'information. Il s'agit d'un système à logiciel prépondérant pour le génie logiciel [2012-91] et du système entreprise pour l'EA. Or les entreprises se mettent de plus en plus à produire et à consommer de l'information [1997-18]. Certains auteurs évoquent même une nouvelle ère de l'information qui sonnerait la fin de l'ère d'industrialisation en inscrivant les entreprises dans une économie de l'information [1981-102] [2014-103]. C'est aussi le cas des Smart Grids dont le fonctionnent repose sur la mise en réseau, au sens informatique, des équipements électriques.

Deuxièmement, l'EA et le génie logiciel comportent des activités similaires telles que l'expression des besoins et des objectifs, l'analyse, la conception, la maintenance, la documentation, l'implémentation ou encore la validation. Ces activités restent similaires dans leur nature même si elles ne portent pas sur le même système, en l'occurrence l'entreprise et le logiciel.

Troisièmement, en génie logiciel, les partie prenantes sont confrontées aux limites de ces artefacts hétérogènes et de nature informelle (mis à part le code) produits aux différentes étapes du cycle de vie du système (documentation, spécifications, etc.). Jézéquel et al. [2006-101] affirment que « ces artefacts donnent de multiples points de vue sur le logiciel en cours de développement et, en pratique, sont souvent indépendants les uns des autres », et que le problème est « d'être capable d'assurer une cohérence entre ces vues, ou au minimum une traçabilité entre les éléments des différents artefacts ». Comme nous l'avons souligné dans la partie précédente, le même problème se pose pour l'EA.

Similarité des domaines et communauté des objectifs légitiment donc pleinement le recours à l'IDM comme cadre méthodologique et technologique pour répondre à la problématique traitée par les présents travaux.

4.1.2.2 Contribution de l'Ingénierie Dirigée par les Modèles à l'Architecture d'Entreprise

La vision que nous souhaitons transmettre ici est la suivante : l'Ingénierie Dirigée par les Modèles est un moyen d'unifier les activités et les artefacts de l'Architecture d'Entreprise. Ainsi unifiée, l'EA parviendra à réaliser pleinement ses objectifs d'alignement effectif et d'accompagnement efficace de l'entreprise tout au long de son cycle de vie.

L'IDM offre à cet effet un nombre de méthodes et de technologies permettant d'homogénéiser les vues d'architecture tout en gardant la possibilité d'utiliser la technologie ou la représentation la mieux adaptée à chaque vue. C'est par la méta-modélisation que l'IDM

parvient à cette unification. Les métamodèles permettent de formaliser les langages de modélisation et d'automatiser ainsi le traitement des modèles. Il est alors possible de vérifier automatiquement la conformité de ces modèles au métamodèle. Le métamodèle renferme les règles d'association des concepts manipulés par les modèles. La relation de conformité peut alors servir à assurer la cohérence globale de l'EA. Les modèles d'architecture manipulés deviennent alors productifs, par opposition aux modèles contemplatifs habituellement utilisés. Les modèles productifs sont en réalité des modèles exécutables qui peuvent être transformés en d'autres modèles exécutables selon les besoins de modélisation.

Le recours intensif aux modèles exécutables dans les activités d'EA pour chacune des vues d'architecture permet de créer une architecture d'entreprise souple et réactive à l'évolution de la stratégie d'entreprise. Les modèles d'une vue peuvent être automatiquement raffinés en d'autres modèles pour la vue d'en dessous grâce aux transformations de modèles. Les modèles exécutables et les transformations de modèles permettront d'automatiser les activités d'EA telles que l'analyse de la structure et du comportement ou encore la mise à jour.

Nous détaillerons les possibilités d'utilisation des méthodes et techniques de l'IDM dans la suite de ce chapitre. Mais nous présentons d'abord la démarche IDM mise en œuvre pour la création du métamodèle EAT-ME.

4.1.2.3 Démarche mise en œuvre

S'il n'existe pas de démarche standardisée pour mettre en œuvre l'IDM [2013-104], les différents auteurs conviennent néanmoins de la nécessité de commencer par analyser le domaine d'application [2012-91] afin de l'appréhender. Dans notre contexte, il s'agit de la découverte des pratiques d'EA dans le contexte des Smart Grids. L'appréhension du domaine a pour objectif d'identifier les concepts essentiels en EA ainsi que leurs règles d'association afin de concevoir un métamodèle pour le domaine traité. La figure 4.2 est une représentation de la démarche IDM mise en œuvre pour mener les travaux de cette thèse.

Il est important de préciser que la définition du métamodèle EAT-ME s'est faite de manière itérative : commencer par quelques concepts en précisant leurs relations, créer des modèles conformes à ce métamodèle, tester le modèle obtenu par rapport aux attentes du domaine, enrichir à nouveau le métamodèle et ainsi de suite.

L'appréhension du domaine de l'EA a été menée de différentes manières :

- par l'analyse des documentations d'architecture utilisées dans divers projets Smart Grids. Nous avons tenu à examiner les pratiques d'EA dans le contexte des Smart Grids pour mieux mettre en œuvre notre approche par la suite ;
- par les entretiens menés avec les experts d'EA du département MIRE (dont notamment plusieurs personnes certifiées TOGAF) ainsi que des chefs de projets Smart Grid, des membres de groupes de normalisation Smart Grid de la Commission électrotechnique

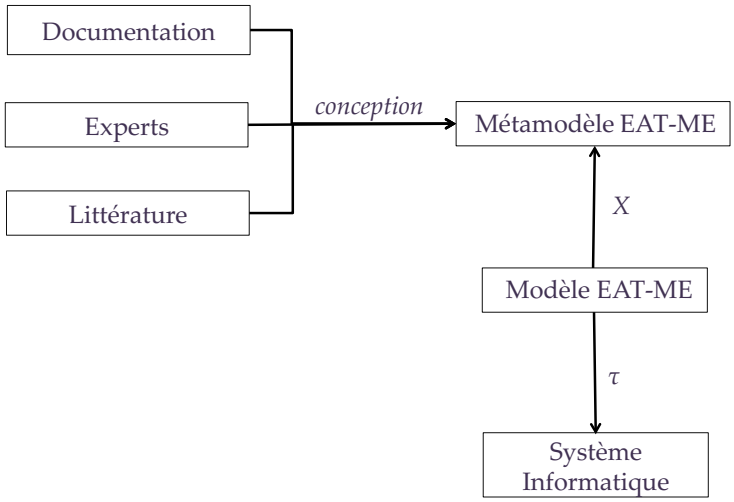

FIGURE 4.2 – Mise en œuvre d'une démarche IDM pour l'Architecture d'Entreprise

internationale (CEI) [2] ;

– par une revue de la littérature portant sur l'EA de manière générale, écrite par des chercheurs en EA et des architectes d'entreprise.

La revue de la littérature a été présentée dans le chapitre 2. Dans la suite de ce chapitre, nous présentons donc l'analyse du domaine à travers l'étude de la documentation à laquelle nous avons eu accès et les entretiens menés avec les différents experts du domaine.

4.2 Analyse du domaine

4.2.1 L'Architecture d'Entreprise pour les Smart Grids

4.2.1.1 Démonstrateurs européens

Tout d'abord, nous avons étudié les spécifications des démonstrateurs Smart Grids européens et en particulier celles de ADDRESS et PREMIO. Le choix de ces deux démonstrateurs s'explique par la participation du département MIRE à ces deux projets, ce qui a facilité l'accès aux documents officiels.

Le projet ADDRESS regroupe 25 partenaires dans 11 pays. L'objectif visé est d'améliorer l'efficacité, la sécurité et la qualité de la production et de la consommation d'électricité dans un contexte de forte croissante de la participation des énergies renouvelables à la production

2. http ://www.iec.ch

d'électricité. Pour cela, des technologies sont développées pour gérer la consommation d'électricité des clients résidentiels et professionnels.

Le projet PREMIO vise à optimiser la gestion globale du réseau électrique de la région Provence-Alpes-Côte d'Azur (PACA) en intégrant les TIC et en expérimentant :

- la production locale de l'énergie ;
- l'effacement : il s'agit de baisser ou de couper la consommation des utilisateurs sans en dégrader le confort ;
- stockage/déstockage de la chaleur ou du froid.

L'étude de la documentation des deux projets a été utile à la compréhension des enjeux Smart Grids et à l'exploration des pratiques de spécification de ce type de grand projet impliquant plusieurs partenaires. Les deux spécifications se présentent sous la forme de contenu textuel agrémenté de diagrammes de cas d'utilisation et de diagrammes de séquences UML. La spécification reste cependant essentiellement textuelle avec quelques dessins d'architecture, donnant lieu à une représentation purement contemplative. En outre, aucun lien de traçabilité entre l'architecture proposée dans la spécification et son implémentation réelle n'est explicité.

4.2.1.2 Normalisation de *Use Cases* Smart Grids

La normalisation de *Use Cases* ou de cas d'utilisation consiste à imposer un canevas pour la spécification des cas d'utilisation Smart Grid puis à faire le mapping entre cette spécification et le cadre d'architecture SGAM. Le SGAM est un cadre d'architecture spécialisé dans l'architecture des Smart Grids.

La normalisation de *Use Cases* est utilisée par le projet européen GRID4EU [3] qui rassemble six démonstrateurs Smart Grid différents. Les *Use Cases* ainsi normalisés, sont ensuite mis en correspondance avec le cadre d'architecture SGAM. La normalisation des *Use Cases* dans le cadre de GRID4EU, a été faite après l'implémentation des six démonstrateurs pour comparer leurs retours d'expérience. Nous avons retenu, en particulier, le démonstrateur de la société nationale d'électricité italienne (ENEL) et son *Use Case* portant sur la régulation de tension des réseaux de distribution présentant une forte pénétration de Producteur d'Énergie Décentralisé (DER). C'est le cas d'application de la norme et du cadre SGAM le plus fourni et le plus abouti du projet GRID4EU.

L'analyse des documents de normalisation et des spécifications de projets européens appliquant ces normes est à l'origine de plusieurs constatations. D'une part, dans le SGAM la traçabilité entre les vues n'est pas directe. Elle est au contraire assurée par la superposition des différentes vues d'architecture sur le *Smart Grid Plan*. Ce dernier décrit un réseau électrique selon deux composantes : physique et organisationnelle. La figure 4.3 illustre une projection de la vue fonctionnelle et de la vue technique (appelée vue composant dans le SGAM), telles que définies par le démonstrateur d'ENEL pour GRID4EU, sur le *Smart*

3. http ://www.grid4eu.eu/

Grid Plan. Pour tirer les liens de traçabilité entre deux vues, il faut donc les superposer en même temps sur le *Smart Grid Plan*. La normalisation de *Use Cases* Smart Grid préconise l'utilisation du langage UML mais une grande partie des *Use Cases* est spécifiée sous la forme de texte accompagné de quelques diagrammes d'activité ou de séquence.

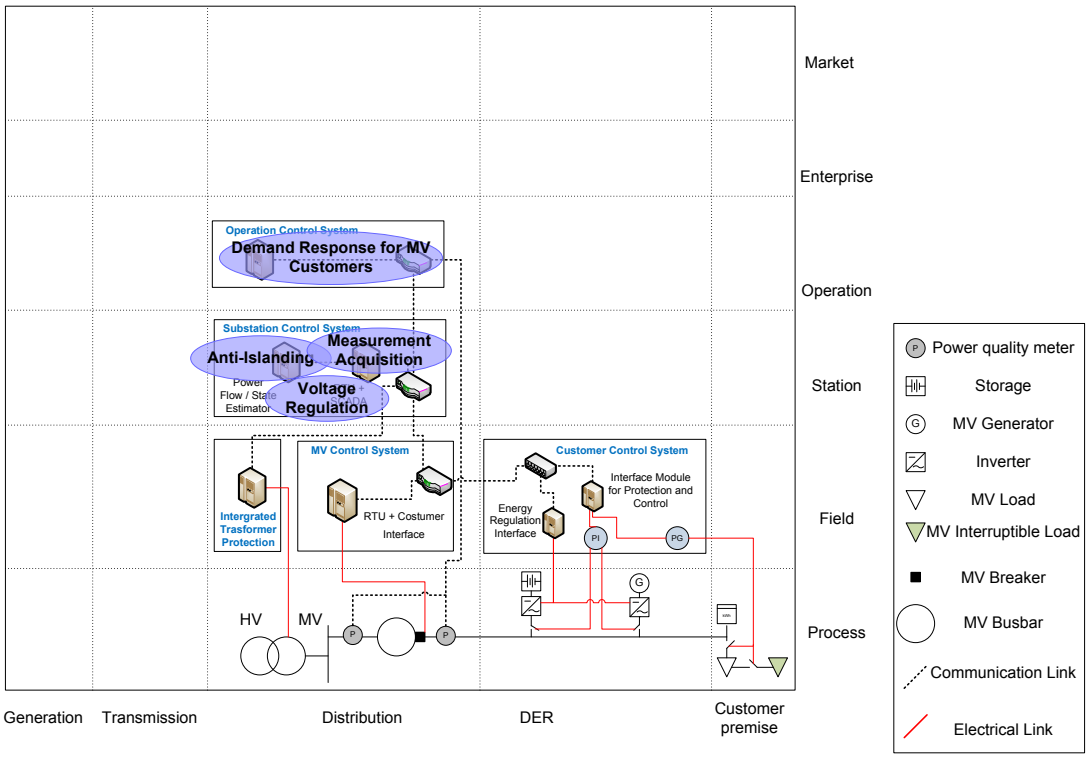

FIGURE 4.3 – Projection de la vue fonctionnelle du démonstrateur ENEL de GRID4EU sur le *Smart Grid Plan*

D'autre part, il existe un grand déséquilibre entre (1) la spécification des différents démonstrateurs (2) le soin apporté à la description d'une vue au sein d'un même démonstrateur. En effet, certaines équipes détaillent leur cas d'utilisation plus que d'autres. De plus, cet exercice ayant été mené par des équipes dont le profil est fortement technique, les vues composant et communication du SGAM sont souvent plus fournies que la vue métier par exemple.

Bien que le démonstrateur d'ENEL soit le plus en avance en terme de normalisation de *Use Cases*, l'analyse des spécifications de la régulation de tension sur un réseau électrique n'a pas été suffisante à l'appréhension de cette problématique sans connaissances préalables sur le fonctionnement des réseaux de distribution. Pour cette raison, et afin de collecter plus d'informations sur le sujet, les spécifications des projets auxquels a participé le département MIRE et qui abordent la régulation de tension, ont été analysés.

4.2.1.3 Projets de recherches et développement

Le contexte de cette thèse CIFRE a favorisé les échanges avec les experts et les ingénieurs-chercheurs de EDF R&D en général, et du département MIRE en particulier. Nous avons ainsi eu accès aux documents de spécification relatifs à un projet de régulation de tension de réseau de distribution. Ces documents sont pour l'essentiel textuels. Des schémas de réseaux électriques et des courbes de charges viennent illustrer les grands principes de la régulation de tension mise en place. Le projet R&D a abouti à l'implémentation d'une maquette de simulation développée sous Simulink [4]. Les algorithmes de régulation sont écrits en C++.

Outre la compréhension des principes de la régulation de tension, l'étude de terrain a pour objectif de mener une étude sociologique des pratiques d'architecture au niveau des projets Smart Grids. Quelques entretiens ont été menés avec le développeur C++ et des experts de la régulation de tension. Les experts techniques interrogés utilisent leurs propres langages de modélisation (automates programmables) et d'implémentation (C++). Ils sont de plus peu sensibles aux problématiques soulevées par l'EA. En effet, ils n'avaient pas connaissance du SGAM ni de la normalisation de *Use Cases* pour la régulation de tension des réseaux de distribution.

Partant de ce constat et de l'émergence des problématiques liées au véhicule électrique, notre étude terrain se porte alors sur le projet de la gestion d'une flotte de véhicules électriques. Ce projet est en effet transverse et touche plusieurs processus métier de l'entreprise. D'une part, la flotte de véhicules est indispensable à la tournée des agents EDF. D'autre part, leurs recharges impliquent un changement de paradigme sans précédent dans le fonctionnement des réseaux électriques, comme expliqué dans le chapitre 1.

Cependant, la recharge des véhicules électriques relève encore une fois du domaine purement électrotechnique. Le cas métier de la gestion d'une flotte de véhicules électrique a l'avantage de toucher l'ensemble de l'entreprise : du gestionnaire de la flotte, à l'agent qui utilise le véhicule pour sa tournée, en passant même par le responsable des tournées. De plus, même s'il s'agit d'un projet de R&D, celui-ci a abouti à une implémentation testée sur une petite agence de La Poste, contrairement à au projet R&D pour la régulation de tension qui se limite à la simulation.

Les documents de spécification du projet de gestion d'une flotte de véhicules sont pour l'essentiel textuels. Cependant, des *slides* de présentation détaillent (1) les blocs fonctionnels principaux et leur décompositions en fonctions (2) les flux d'information entre ces blocs (3) les équipes chargées de les implémenter. Cette présentation illustre l'importance pour les équipes de projet d'établir des liens clairs, d'une part entre les aspects « donnée » et « traitement », et d'autre part entre la vue fonctionnelle et la vue applicative.

Cependant, le support de présentation n'étant pas auto-portant, un entretien avec la chef de projet et avec d'autres membres de l'équipe a été nécessaire pour connaitre les objectifs

4. http ://fr.mathworks.com/products/simulink/

métier du projet car ces derniers ne sont pas explicités sur les supports de présentation. Il s'avère que l'objectif métier était d'obtenir le meilleur retour sur investissement (ROI [5]) possible de l'intégration de véhicules électriques dans la flotte de l'entreprise.

Ces entretiens ont en outre permis d'identifier quelques problèmes rencontrés sur le terrain que l'équipe de projet n'a pas pu anticiper. Par exemple, l'étude préalable n'a pas tenu compte d'un facteur sociologique important. En effet, les agents de La Poste ont l'habitude d'effectuer leur tournée quotidienne avec une voiture thermique qui leur est attribuée sur l'année de manière à ce qu'ils puissent garder leurs équipements de travail dans le véhicule. Or avec les contraintes d'autonomie associées à l'usage de véhicules électriques, l'affectation d'un véhicule à un agent dépend de la distance de sa tournée, l'obligeant ainsi à constamment déplacer son matériel et détériorant ses conditions de travail.

4.2.2 Identification des concepts

Un métamodèle est un modèle du langage de modélisation qui sert à décrire les modèles. Or l'activité de modélisation est guidée par l'intention du modélisateur. Par conséquent, il convient de définir clairement nos intentions de modélisation avant de caractériser les concepts retenus pour le métamodèle EAT-ME.

D'abord, nous cherchons à caractériser les concepts essentiels et indispensable à la modélisation d'une vue d'architecture en mettant en évidence les caractéristiques communes aux artefacts utilisés par les praticiens. En effet, le métamodèle est un moyen d'unifier l'ensemble des artefacts employés dans les diverses activités d'EA.

Ensuite, l'objectif de EAT-ME est de permettre de vérifier la cohérence interne à une vue mais aussi la cohérence entre deux vues d'architecture. Il est donc indispensable de représenter ces liens de cohérence dans le métamodèle. Nous expliquons dans la suite l'importance d'expliciter ces liens pour l'analyse de la structure et du comportement de l'architecture

Enfin, nous cherchons à automatiser la manipulation des modèles d'architecture à travers la mise en œuvre de transformations de modèle. Il convient de de spécifier dans le modèle d'architecture l'existence de ces transformations et leur utilité.

Les approches par points de vue sont adaptées à la modélisation de systèmes complexes comme les architectures d'entreprise, nous adoptons donc une approche par points de vue. Celle-ci facilite la conception des modèles par les acteurs impliqués en séparant leurs préoccupations respectives. Elle permet également de présenter les modèles obtenus, ainsi que les résultats d'analyse de manière plus compréhensible, car chaque point de vue n'utilise que les concepts métier propres à chaque acteur, selon sa perspective. Ceci nous amène à identifier le premier concept qui est celui de **vue**. Dans la suite, nous identifions les concepts nécessaires à la modélisation de chacune des vues traitées dans ces travaux.

5. *Return On Investment*

4.2.2.1 Concepts identifiés pour la vue métier

La vue métier reflète la perspective de l'architecte métier. Ce dernier décrit l'organisation de l'entreprise en termes de **processus métier**. La figure 4.4 illustre les concepts retenus pour la vue métier en les représentant sous un format libre.

Un processus est une séquence de **tâches** métier produisant un résultat qui participe à la réalisation d'un ou de plusieurs **objectifs métier** de l'entreprise. Ce résultat doit donc être mesurable au regard de l'objectif qu'il remplit afin de l'évaluer. De plus, les tâches métier produisent et utilisent des **concepts métier**.

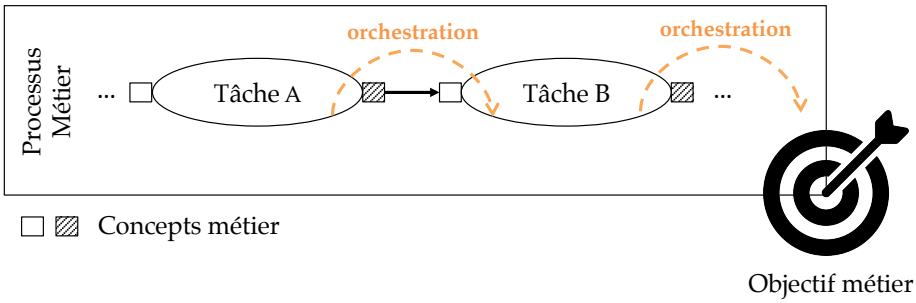

FIGURE 4.4 – Concepts identifiés pour le point de vue métier et représentés selon un formalisme libre

L'architecte métier doit organiser les différentes tâches métier en termes de processus cohérents et créateurs de valeur pour l'entreprise. À ce titre, les objectifs de chaque processus métier doivent être clairement identifiés et lui être associés. Si deux processus sont identifiés comme remplissant les mêmes objectifs métier, l'architecte métier peut alors les détecter grâce à ces relations et garantir ainsi la **cohérence interne** à la vue.

Il doit en outre veiller à la bonne orchestration des processus métier mais aussi des tâches composant un processus pour garantir une bonne exécution des processus métier. L'orchestration des tâches et des processus est en grande partie garantie par le partage et la diffusion de l'information, c'est-à-dire des concepts métier, au sein de l'entreprise participant ainsi à la cohérence interne de la vue métier.

Dans le cas des très grandes entreprises, il est possible qu'un même concept métier soit référencé par deux noms différents, à la suite d'une fusion ou d'une acquisition par exemple. Le rôle de l'architecte métier est d'identifier cette incohérence, de mettre en œuvre les solutions requises pour la corriger et de la communiquer au reste de l'entreprise. Le concept métier est donc une entité de première importance pour la vue métier.

4.2.2.2 Concepts identifiés pour la vue fonctionnelle

La vue fonctionnelle est la pierre angulaire d'une architecture d'entreprise. C'est par elle que se fait la transition entre la vue métier très abstraite, et la vue applicative qui représente le système informatique. Elle joue de ce fait un rôle de premier plan dans la problématique d'alignement métier/IT mais surtout en tant qu'interface et interprète entre le métier et l'IT.

La vue fonctionnelle reflète la perspective de l'architecte fonctionnel. Ce dernier doit spécifier les **fonctions** qui raffinent les tâches métier en les organisant en **processus fonctionnels**. Les tâches et les fonctions sont liés par des relations de **raffinement**. Les fonctions participent à atteindre des **objectifs fonctionnels**, qui eux mêmes raffinent les objectifs de la vue métier. Comme pour la vue métier, définir des relations explicites entre les processus fonctionnels et les objectifs qu'ils remplissent permet de vérifier qu'il n'y a pas de redondance ou de recouvrement et d'optimiser ainsi l'ensemble de l'architecture.

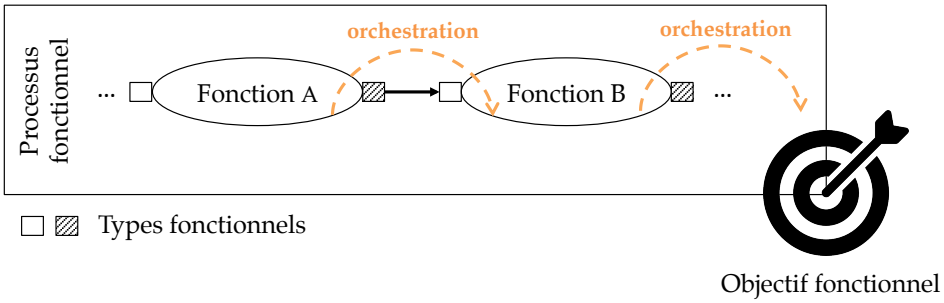

FIGURE 4.5 – Concepts identifiés pour le point de vue fonctionnel et représentés selon un formalisme libre

Les fonctions utilisent et produisent des **types fonctionnels**. L'orchestration d'un processus fonctionnel nécessite que le type fonctionnel en sortie d'une fonction soit compatible avec le type de donnée en entrée de la fonction suivante. La vue fonctionnelle doit donc permettre de vérifier cette compatibilité pour garantir une **cohérence interne**. Si un type T_A en sortie d'une fonction F_A n'est pas compatible avec le type en entrée T_B d'une fonction F_B, l'architecte doit pouvoir détecter cette incompatibilité de type. Une transformation de modèle $t_{T_A \to T_A}$ peut alors être envisagée.

Le types fonctionnels raffinent les concepts métier en donnant leur type, c'est-à-dire leurs différents attributs. Ce raffinement permet de vérifier que l'ensemble des concepts métier sont bien représentés dans la vue fonctionnelle et de vérifier ainsi que les informations de la vue fonctionnelle sont bien alignées avec celles de la vue métier et garantir ainsi une **cohérence externe** avec la vue métier.

4.2.2.3 Concepts identifiés pour la vue applicative

La vue applicative reflète la perspective de l'architecte applicatif. Ce dernier spécifie les **modules applicatifs** qui implémentent les fonctions de la vue fonctionnelle. L'objectif de cette vue est de garantir une **cohérence externe** entre la vue applicative et la vue fonctionnelle, en traçant les liens de **raffinement** entre fonctions et modules. Cette traçabilité est garantie entre la vue métier et la vue applicative par transitivité et participe ainsi à maintenir l'alignement métier/IT.

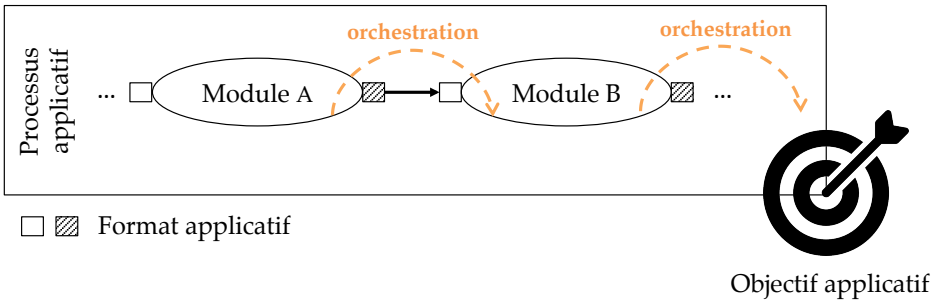

FIGURE 4.6 – Concepts identifiés pour le point de vue applicatif et représentés selon un formalisme libre

Comme l'illustre la figue 4.6, la vue applicative spécifie en outre les **formats applicatifs** manipulés par les modules. Expliciter ainsi les formats applicatifs permet de garantir une bonne orchestration des processus applicatifs. Par exemple, un module M_A produit un format F_A et un module M_B prend en entrée un format F_B. Si les deux formats ne sont pas compatibles alors l'orchestration des deux modules n'est pas possible. Dans ce cas, l'architecte applicatif peut le détecter et mettre en place une **transformation de modèles** $t_{F_A \to F_A}$ pour traiter cette incompatibilité de format sans pour autant changer ses modules.

La vue applicative permet de représenter l'architecture applicative d'une entreprise qui peut englober des centaines voire des milliers de modules et d'applications. Formaliser ainsi cette vue permet de mener des analyses d'impact et de changement. Par exemple, les liens de traçabilité entre la vue métier et la vue applicative (via la vue fonctionnelle) permettent d'identifier les processus métier susceptibles d'être impactés par un changement de module applicatif. À l'inverse, il est possible d'identifier les modules applicatif impactés par un changement de processus métier.

L'architecte fonctionnel doit spécifier les **objectifs applicatifs** que doit remplir chaque module. De cette manière il est possible de détecter les éventuelles applications redondantes ou celles qui ne participent plus à atteindre les objectifs de l'entreprise. Les objectifs applicatifs sont de plus dérivés des objectifs fonctionnels, eux mêmes dérivés des objectifs métier. Ainsi, il est possible de tracer explicitement la déclinaison des objectifs de l'entreprise sur l'ensemble de l'architecture pour contrôler la mise en œuvre d'une stratégie d'entreprise.

L'analyse des pratiques de l'EA a permis d'identifier ces éléments indispensables à la mise en œuvre de méthodes permettant de garantir la cohérence globale de l'architecture d'entreprise. Il est important de noter que nous avons repris l'approche ensembliste de Favre [2005-105] en procédant par identification des caractéristiques communes des artefacts habituellement manipulés par les architectes d'entreprise contrairement à une approche ontologique qui vise à dresser un inventaire des connaissances du domaine.

Cette démarche est d'autant plus justifiée que notre objectif premier n'est pas de modéliser une architecture d'entreprise dans ces moindres détails mais de prendre en comptes les éléments indispensables à la vérification de la cohérence de l'ensemble de l'architecture.

Les concepts identifiés, directement issus l'analyse du domaine de l'EA, ont servi à la conception du métamodèle EAT-ME pour *Enterprise ArchiTecture MEtamodel*.

4.3 Le métamodèle EAT-ME

La conception du métamodèle EAT-ME a été menée de manière itérative. En partant de quelques concepts et en enrichissant le métamodèle à chaque itération. Des versions successives de ce métamodèle ont fait l'objet de publications ([2015-19] et [2016-106]). L'analyse du domaine a été menée dans le cadre des projets Smart Grid, tout en confrontant systématiquement les concepts identifiés aux pratiques d'EA issues de la littérature. Le métamodèle EAT-ME a été conçu en gardant à l'esprit la possibilité de l'utiliser dans différents contextes et pas seulement pour les Smart Grids.

4.3.1 Éléments de base et cohérence *intra-vue*

L'identification des concepts met en évidence les éléments de base nécessaires à la modélisation d'une architecture d'entreprise : « Information », « Action » et « Objectif ». Ces trois entités élémentaires permettent de décrire le contenu de chacune des vues d'architecture (métier, fonctionnelle et applicative). La figure 4.7 illustre, sous la forme d'un diagramme de classes, la partie[6] du métamodèle EAT-ME dédiée à la représentation de ces éléments.

EAT-ME n'a pas pour objectif la modélisation extensive des artefacts d'architecture mais se focalise sur l'abstraction des artefacts d'architecture à leur plus petit dénominateur commun pour pouvoir raisonner non plus sur la sémantique des éléments mais sur leur cohérence en termes de relations d'interdépendance. Par exemple, l'architecte métier peut décrire des procédures composées de processus, eux-mêmes composés d'activités Le concept d'action peut aussi bien être utilisé pour décrire des procédures, des processus, des activités du moment qu'il s'agisse d'un élément d'architecture muni de comportement. Il suffit de se placer à la bonne échelle mais le concept d'action reste valable. Certains auteurs qualifient les systèmes obéissant à ce principe de systèmes à structure fractale [2004-107].

6. Nous fournissons des sous-parties du métamodèle pour des raisons de lisibilité

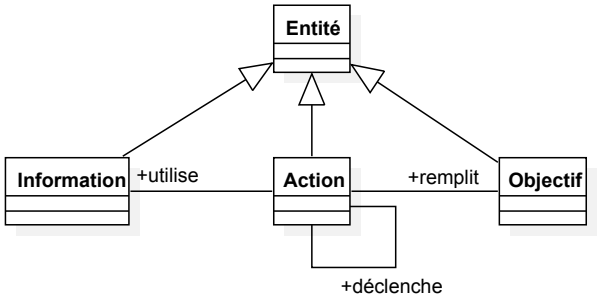

FIGURE 4.7 – Partie du métamodèle concernant les éléments de base et leurs relations

L'intérêt du métamodèle est de décrire la structure d'une architecture tout en maintenant une cohérence interne à chaque vue. Comme expliqué dans la section précédente, cette cohérence passe par la modélisation explicite de chaque entité afin de créer des liens de traçabilité indépendamment de l'exécution des processus. Par exemple, le fait de modéliser toutes les informations indépendamment des actions et de créer ensuite des liens de traçabilité entre chaque action et les informations qu'elle utilise permet de vérifier la compatibilité entre l'action et les informations impliquées, avant même d'exécuter le processus. Ainsi, si une action déclenche une autre action, il faut que l'information à la sortie de la première action soit compatible avec l'entrée de l'action déclenchée. En ce sens, les liens de traçabilité « déclenche » et « utilise » garantissent une bonne orchestration des processus.

La relation « remplit » permet de relier chaque action à ses objectifs et donne ainsi la possibilité de suivre efficacement l'évolution de l'architecture au regard des objectifs définis. L'exploitation des liens de traçabilité entre actions et objectifs permet de (1) détecter les actions redondantes en termes d'objectif (2) déterminer les actions (et par transitivité les informations) impactées par un changement d'objectif (3) et inversement, identifier les objectifs atteints par une évolution d'une action ou l'altération d'une information.

La figure 4.8 illustre la partie du métamodèle consacrée à la déclinaison des éléments de base (Action, Information, Objectif) en concepts spécialisés pour chacune des vues d'architecture (métier, fonctionnelle et applicative). Ces concepts ont d'abord été identifiés lors de l'analyse du domaine (voir section 4.2.2), ainsi :

- « Concept », « Type » et « Format » héritent de « Information » ;
- « Tâche », « Fonction » et « Module » héritent de « Action » ;
- « Objectif_Métier », « Objectif_Fonctionnel » et « Objectif_Applicatif » héritent de « Objectif ».

D'une part, les éléments de base du métamodèle nous ont amenés à identifier la notion d'**aspect** d'une vue. Un aspect d'une vue est un sous-ensemble d'une vue dédié à la modélisation d'informations, d'actions ou encore d'objectifs. D'autre part, l'identification de relations entre les éléments de base nous a conduit à introduire une nouvelle vue : la vue

intégration. La vue intégration est dédiée à la modélisation des liens de traçabilité entre les éléments de base – information, action et objectif — à l'intérieur d'une même vue.

En ce sens, la vue intégration explicite les liens de **cohérence intra-vue**, autrement dit, les liens de cohérence entre les aspects d'une même vue. Les notions de vue et d'aspect relèvent de la structure globale d'une architecture d'entreprise que nous détaillons dans la partie suivante.

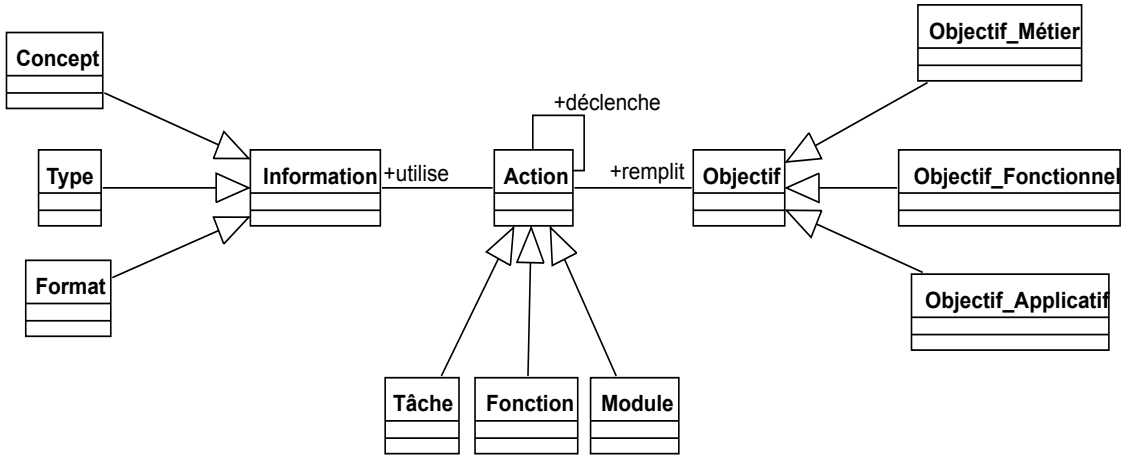

FIGURE 4.8 – Partie du métamodèle concernant la déclinaison des entités de base sur les vues métier, fonctionnelle et applicative

Dans la partie précédente, nous présentons les éléments constitutifs à chaque vue et leurs liens de cohérence.

4.3.2 Structure globale et cohérence *inter-vue*

La figure 4.9 illustre la partie du métamodèle dédiée à la représentation de la structure globale d'une architecture d'entreprise. Dans le métamodèle, une vue peut être spécialisée en vue « Métier », « Fonctionnelle » et « Applicative ». Dans cette thèse, nous avons uniquement abordé ces trois vues tout en gardant à l'esprit la nécessité d'étendre le métamodèle la vue technique, à laquelle mécanisme d'héritage se prête bien. Chaque vue est composée de trois aspect : « Information », « Processus » et « Objectif ».

L'aspect « Information » permet d'avoir un modèle explicite des données utilisées dans chacune des vues :

– **aspect information du point de vue métier**
  Cet aspect établit le modèle de données métier en regroupant les concepts métier manipulés par le processus métier. Ce modèle est peu sujet au changement, sauf

évolution importante des pratiques métier ;

– **aspect information du point de vue fonctionnel**
Cet aspect établit le modèle de données fonctionnelles en regroupant les types des données utilisées par les blocs fonctionnels nécessaires à la réalisation de processus métier ;

– **aspect information du point de vue applicatif**
Cet aspect établit le modèle de données applicatives en regroupant les formats de données compatibles avec les modules applicatifs.

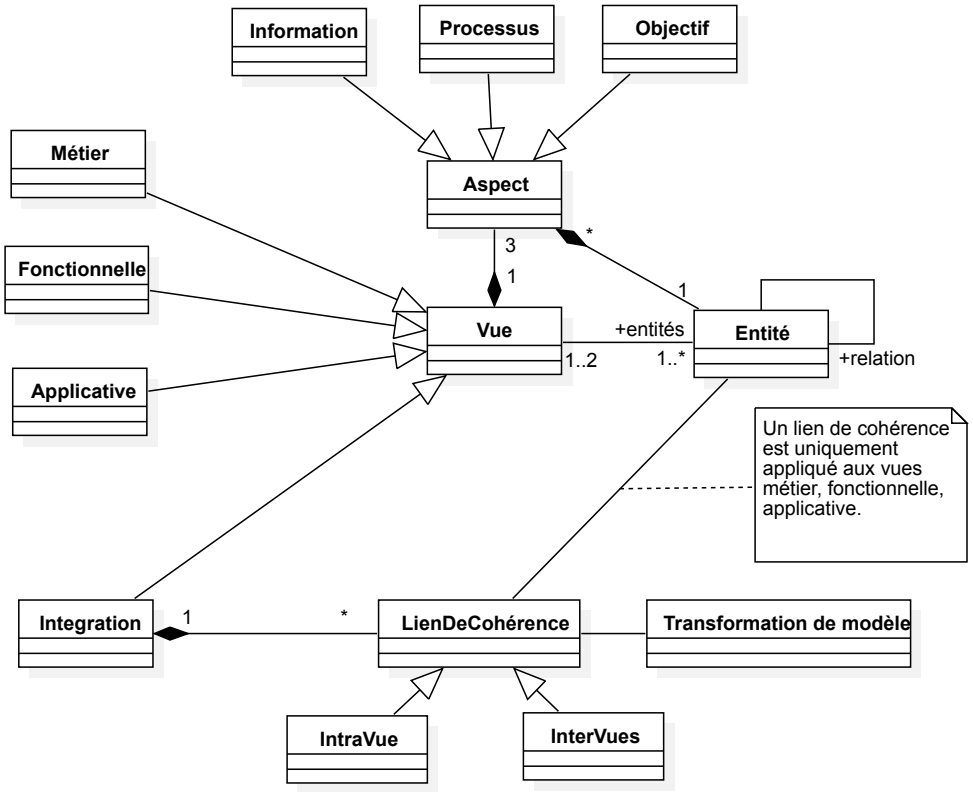

FIGURE 4.9 – Partie du métamodèle concernant la structure globale d'une architecture d'entreprise

De plus, nous modélisons le comportement de chaque vue dans l'aspect « Processus » où un processus est constitué d'un enchaînement d'actions :

– **aspect processus du point de vue métier**
Cet aspect correspond aux processus métier de l'entreprise, décrits en termes d'enchaînement de « Tâches » sans faire référence aux détails d'implémentation ;

– **aspect processus d'un point de vue fonctionnel**

Cet aspect correspond aux processus fonctionnels de l'entreprise, décrits en termes d'enchaînement de « Fonctions » ;

– **aspect processus du point de vue applicatif**

Cet aspect correspond aux processus applicatifs de l'entreprise, décrits en termes d'enchaînement de « Modules ».

De plus, nous étendons chacune des vues par l'aspect « Objectif » :

– **aspect objectif du point de vue métier**

Cet aspect regroupe les motivations métier pilotant les processus métier et modélisés par la classe « Objectif_métier » ;

– **aspect objectif du point de vue fonctionnelle**

Cet aspect regroupe les motivations fonctionnelles pilotant les processus fonctionnels et modélisés par la classe « Objectif_fonctionnel » ;

– **aspect objectif du point de vue applicative**

Cet aspect regroupe les motivations applicatives pilotant les processus applicatifs et modélisés par la classe « Objectif_applicatif ».

La préoccupation majeure de l'EA réside dans l'alignement métier/IT [2005-54], c'est-à-dire l'alignement d'un même aspect entre deux vues d'architecture différentes. Nous proposons d'y dédier une vue spécifique : la vue intégration. Ainsi, en plus d'expliciter les liens de cohérence à l'intérieur d'une même vue à travers la classe « IntraVue » abordée dans la section 4.3.1, la vue intégration permet de modéliser les liens de traçabilité entre des éléments relatifs à un même aspect et appartenant à des vues différentes en spécifiant les relations de raffinement à travers la classe « InterVues » du métamodèle.

La figure 4.10 illustre la partie du métamodèle dédiée à la vue intégration. En plus des liens de cohérence intra-vue, nous y modélisons les liens de cohérence inter-vues à travers la relation « raffine ». Ainsi, une tâche est raffinée par une ou plusieurs fonctions qui la réalisent, elles-mêmes raffinées par un ou plusieurs modules applicatifs qui les implémentent. De même, un objectif métier est raffiné par un ou plusieurs objectifs fonctionnels, eux-mêmes raffinés par un ou plusieurs objectifs applicatifs. De manière analogue, un concept métier est raffiné par un ou plusieurs types fonctionnels, eux-mêmes raffinés par un ou plusieurs formats applicatifs.

Qu'il s'agisse d'une cohérence intra-vue ou inter-vues, la vue intégration définit un *mapping* en spécifiant (1) les entités à aligner, (2) les liens de cohérence entre ces éléments et (3) des transformations de modèle associées à ces liens. Nous identifions plusieurs cas de figures où ces transformations de modèle s'avèrent commodes dans le contexte de l'EA dans la section 4.4. Parmi ces cas de figure nous citons, notamment, l'automatisation du passage d'une vue à l'autre en utilisant une transformation de modèle pour générer le code d'un module de la vue applicative à partir d'un modèle de fonction de la vue fonctionnelle.

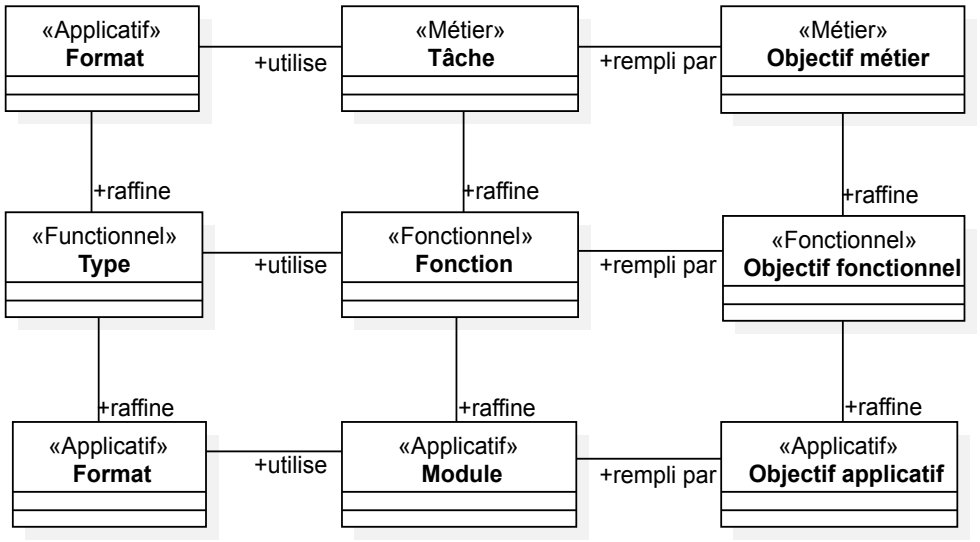

FIGURE 4.10 – Métamodèle de la vue intégration

4.4 Le *framework ExecuteEA*

Une approche par points de vue est indispensable à l'appréhension des systèmes complexes car elle permet de séparer les préoccupations des parties prenantes dans des vues distinctes. Chaque vue traduit donc la perspective d'une partie-prenante. De nombreux *frameworks* d'EA adoptent de ce fait une approche par points de vue. Cependant, les différentes préoccupations sont souvent traitées de manière séquentielle et plus ou moins indépendante des autres vues comme l'attestent S. Kurpjuweit et R. Winter [2007-95].

Nous proposons donc un *framewok* d'EA permettant de traiter les différentes vues d'architecture de manière intégrée tout en permettant à l'architecture d'entreprise d'accompagner l'entreprise sur toute sa trajectoire. Le métamodèle EAT-ME représente la pierre angulaire de ce *framework*. Nous recourons en effet à la méta-modélisation pour unifier les différentes activités d'architecture — documentation, analyse et conception — en les inscrivant dans le cadre méthodologique et technologique offert par l'IDM.

La vision unificatrice de l'IDM a pour *leitmotiv* l'usage systématique de modèles productifs [2005-108], c'est-à-dire de modèles **exécutables** *versus* les modèles purement contemplatifs habituellement utilisés pour mener les différentes activités d'EA [2013-26]. Nous avons donc choisi la dénomination *Execute Enterprise Architecture* (*ExecuteEA*) pour le *framework* que nous présentons dans cette section.

Le *framework ExecuteEA* repose sur trois piliers : (1) une approche unificatrice pour la modélisation d'architecture d'entreprise, (2) un cadre structurant orienté points de vue et

(3) l'identification d'un ensemble de langages et de techniques issus de l'IDM et adaptés à l'analyse d'architecture d'entreprise.

4.4.1 Approche conceptuelle et cadre structurant

Parmi les différentes activités d'EA, nous avons choisi de focaliser nos travaux de thèse sur l'activité d'analyse et ce pour plusieurs raisons.

D'abord, l'analyse fait partie des activités les moins traitées en EA [2008-109] [2013-55], quelle qu'en soit l'école de pensée comme relaté dans l'état de l'art concernant l'EA présenté au chapitre 2. Cet état de fait est en partie dû aux limitations des modèles contemplatifs qui font légion en EA. C'est pendant l'activité l'analyse que les modèles exécutables sont le plus valorisés et indispensables.

Ensuite, comme l'illustre la figure 4.11 l'analyse joue un rôle central dans une démarche d'EA. Les modèles issus de l'analyse peuvent directement servir à documenter l'EA (« modèles EA validés » dans la figure 4.11). Il s'agit de modèles qui recourent directement aux concepts du domaine de l'EA et qui ne nécessitent donc pas de changer de niveau d'abstraction pour être appréhendés par les parties prenants. En IDM, ces modèles sont qualifiés de *problem-level abstration models* [2007-89].

Enfin, l'analyse de modèles d'architecture est fortement liée à l'activité de conception. Le recours aux modèles exécutables pour la conception d'architectures d'entreprise permet de mener des analyses directement sur ces même modèles (« modèles EA à améliorer » et « modèles EA à valider » dans la figure 4.11). Dès lors que ces modèles sont validés à l'issue des activités d'analyse et de conception, il devient possible de les utiliser pour la documentation (en mettant automatiquement à jour les anciens modèles d'architecture) et pour l'implémentation. L'implémentation peut alors se faire par transformation de modèles, en transformant les « modèles EA validés » vers la plate-forme cible.

Nous entendons analyser deux aspects distincts mais corrélés d'une architecture d'entreprise : la structure et le comportement de la dite architecture. L'analyse de la structure repose essentiellement sur la formalisation du métamodèle EAT-ME et vise à intégrer des modèles provenant des différentes vues d'architecture afin de garantir la cohérence de l'ensemble de l'architecture. L'analyse du comportement de l'architecture repose quant à elle essentiellement sur la simulation des modèles issus de l'analyse de la structure.

Le *framework ExecuteEA* englobe une approche conceptuelle et un cadre structurant les différentes activités d'EA, dont notamment l'analyse. Nous commençons par présenter l'approche conceptuelle illustrée par la figure 4.12. L'approche consiste à définir un processus précis en identifiant quatre rôles participant à ce processus : l'architecte métier, l'architecte fonctionnel, l'architecte applicatif et enfin l'architecte d'entreprise. Les trois premiers rôles ont déjà été présentés dans la section 4.1.1.4.

Le processus commence par la modélisation de la vue métier par l'architecture métier. L'architecte fonctionnel traduit la vue métier en une vue fonctionnelle en collaborant avec

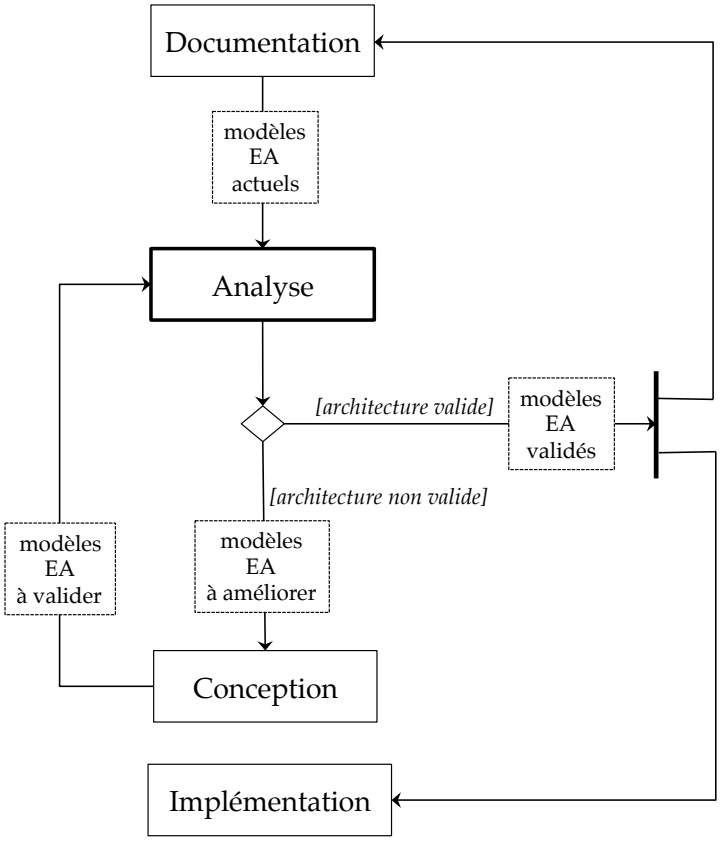

FIGURE 4.11 – Évolution des modèles d'architecture et rôle central de l'activité d'analyse en EA

l'architecte métier. L'architecte applicatif traduit la vue fonctionnelle en une vue applicative tout en collaborant avec l'architecte fonctionnel et par extension avec l'architecte métier. Le rôle de l'architecte d'entreprise consiste alors à orchestrer l'ensemble de ces collaborations.

L'architecture ainsi modélisée est soumise à une analyse structurelle menée par l'architecte d'entreprise en étroite collaboration avec les architectes des différentes vues. L'architecte d'entreprise est alors responsable de la vue intégration qui a pour mission de vérifier la cohérence globale des modèles d'architecture. Ainsi intégrés, les modèles sont soumis à une analyse comportementale via la simulation. L'approche proposée suit un processus itératif. À l'issue de la simulation, l'ensemble des architectes valident ou modifient leurs modèles selon les résultats de la simulation, et ainsi de suite jusqu'à obtenir une architecture satisfaisante pour l'ensemble des parties prenantes.

L'approche proposée s'inscrit en outre dans un cadre structurant qui reprend les concepts identifiés dans le métamodèle EAT-ME. Ce cadre est illustré par la figure 4.13. Ce cadre structure et organise les différents modèles créés par les architectes : métier, fonctionnel et

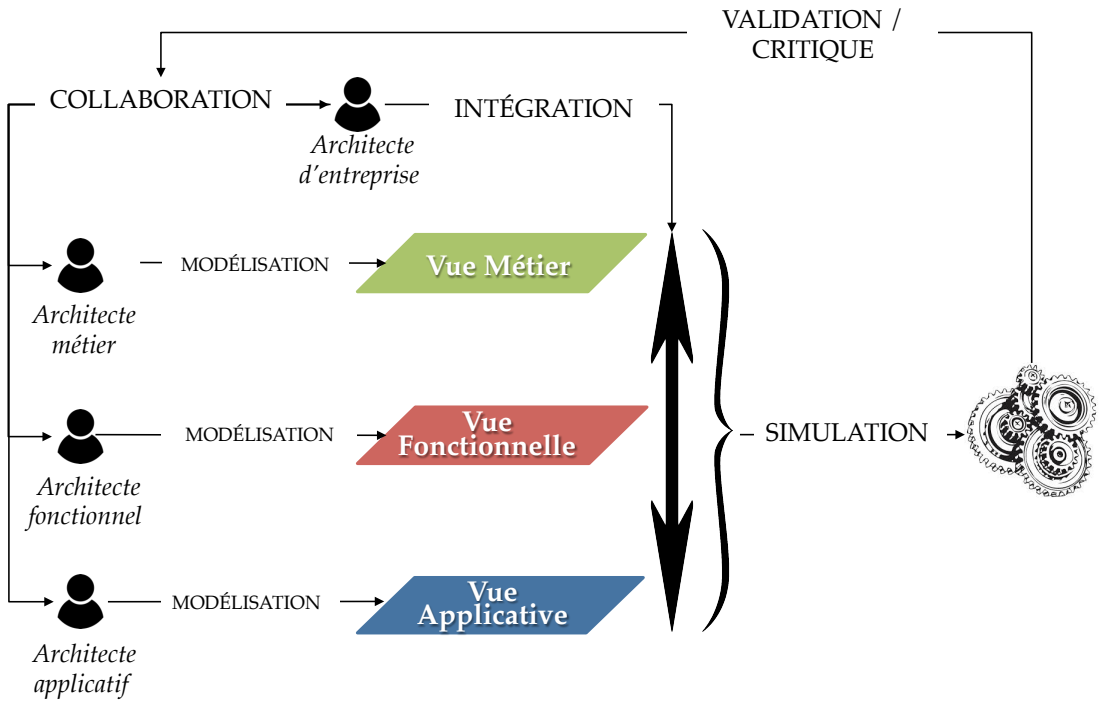

FIGURE 4.12 – Approche conceptuelle du *famework ExecuteEA*

applicatif. Chaque vue est en effet composée de trois aspects : information, processus et objectif. Notre contribution principale consiste à doter ce cadre d'une vue intégration transverse à toutes les autres vues. Cette vue reflète la perspective de l'architecte d'entreprise qui a pour mission de garantir la cohérence globale de l'architecture, et s'assure de l'alignement métier/IT. La vue intégration permet donc à l'architecte d'entreprise d'expliciter les liens de cohérence intra-vue (*via* les liens « remplit » et « utilise ») et inter-vues (via les liens de « raffinement »).

4.4.2 Analyse de l'architecture d'entreprise

Le cadre structurant et l'approche conceptuelle du *framework ExecuteEA* permettent de mieux articuler entre elles les activités d'analyse de la structure et du comportement. L'analyse automatisée de ces modèles facilite leur appréhension par l'acteur qui mène ces analyses (en l'occurrence l'architecte d'entreprise). Dans cette section, nous expliquons comment le recours aux modèles exécutables tel que préconisé par l'IDM permet de réaliser ces analyses.

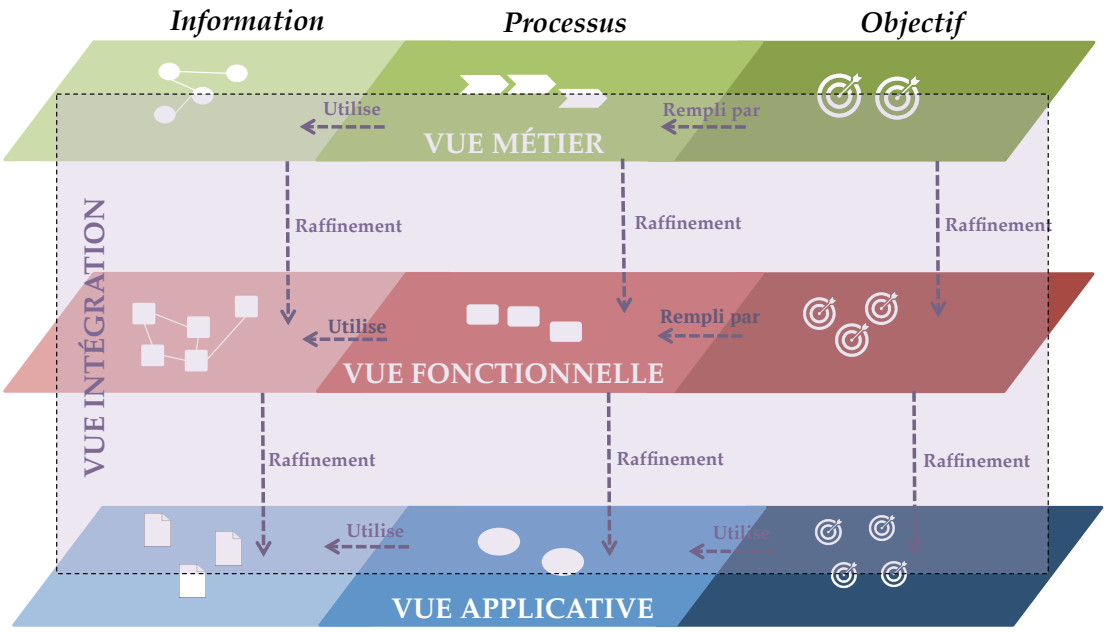

FIGURE 4.13 – Cadre structurant de *ExecuteEA*

4.4.2.1 Analyse de la structure

L'analyse structurelle d'une architecture d'entreprise a plusieurs visées. Il peut s'agir de (1) la vérification de la cohérence de l'ensemble de l'architecture (2) anticiper les changements en en mesurant par exemple l'impact [2005-110] sur la structure de l'architecture. Le métamodèle EAT-ME rend possible cette analyse en exploitant la relation de conformité qui lie un modèle à son métamodèle telle que présentée dans l'état de l'art concernant les relations de base de l'IDM au chapitre 3.

La formalisation d'un métamodèle pour l'EA permet de plus de contrôler la conformité des modèles d'architecture. En effet, le métamodèle contraint les modèles d'architecture et offre la possibilité de vérifier que l'ensemble des modèles respecte bien les contraintes exprimées par le métamodèle. Dans le cadre de EAT-ME, ces contraintes sont exprimées à travers les liens de cohérence. Par exemple, toutes les entités de base (action, objectif et information) doivent toujours appartenir à deux vues : la vue intégration en plus d'une autre vue parmi les vues métier, fonctionnelle, applicative et technique. Cette contrainte permet de vérifier que toutes les entités de l'architecture sont bien répertoriées dans la vue intégration.

Autre exemple de contraintes exprimées dans le métamodèle EAT-ME : tous les objectifs de la vue métier doivent avoir des liens de cohérence inter-vues avec les objectifs de la vue fonctionnelle, et de même pour ces derniers avec les objectifs de la vue applicative. Cette contrainte permet de vérifier que tous les objectifs exprimés par le métier sont bien

pris en compte par les applications de l'entreprise. Cette cohérence inter-vues est renforcée par une autre contrainte intra-vue exprimée dans le métamodèle EAT-ME. Chaque action doit disposer d'un lien de traçabilité vers l'objectif qu'elle remplit. Le croisement de ces deux contraintes, exprimées au niveau du métamodèle, permet de vérifier que les modèles disposent bien des liens de traçabilité nécessaire à la vérification de l'alignement métier/IT.

L'exploitation des contraintes exprimées sur les liens de traçabilité de la vue intégration permet de vérifier par exemple que toutes les tâches métier sont bien reliées aux fonctions qui les réalisent, et que ces fonctions sont bien reliées aux modules applicatifs qui les implémentent. L'analyse de cohérence permettra de vérifier que tous éléments de base de type action du modèle disposent bien de liens de traçabilité, et de les ajouter si nécessaire. Le même principe s'applique sur les liens de traçabilité entre informations appartenant à différentes vues, auquel cas l'analyse permettra de vérifier que tous les concepts métier disposent bien de liens vers leurs types dans la vue fonctionnelle, et vers leur format dans la vue applicative.

L'analyse de la cohérence permet de plus d'identifier les transformations de modèle utiles à l'intégration de l'architecture. Les transformations de modèle présentent en effet plusieurs atouts dans le contexte de l'IDM en général et de l'EA en particulier. Supposons par exemple qu'il s'avère que l'orchestration de deux modules est impossible à cause d'une incompatibilité de format à l'issue de l'analyse de la cohérence de la structure. Dans ce cas, il est possible de créer une transformation de modèle et de l'associer au lien de cohérence intra-vue dans la vue intégration. La figure 4.14 illustre ce principe dans le cas général d'une action et d'une information.

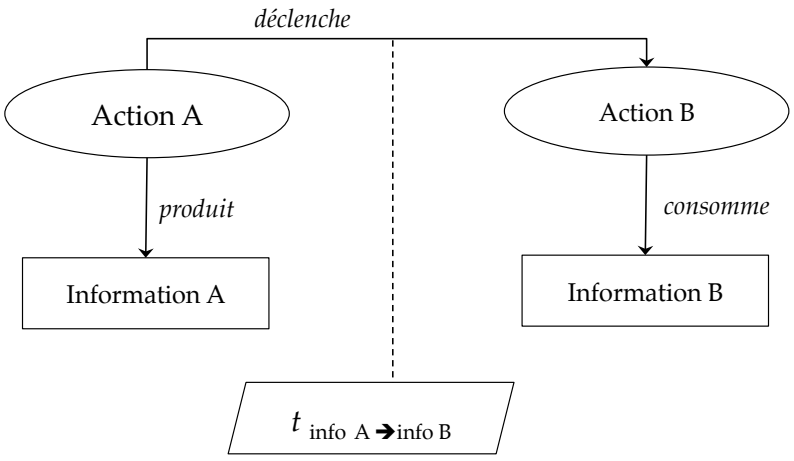

FIGURE 4.14 – Identification d'une transformation de modèles pour garantir une cohérence intra-vue

La transformation de modèle est un moyen de renforcer l'alignement métier/IT en automatisant le passage d'une vue à l'autre. Avec les technologies IDM actuelles, il est possible par

exemple de transformer une fonction en un code pour le module applicatif qui l'implémente. Le passage de la vue métier à la vue fonctionnelle demeure quant à lui essentiellement manuel. L'analyse de la cohérence vise à identifier, dans la vue intégration, ce type de transformation de modèle et les entités du modèle qu'il concerne. Les transformations de modèle constituent une solution pour capitaliser le savoir faire métier au niveau des modèles et le rendre plus indépendant de la plate-forme d'implémentation. Ainsi, il est possible de raisonner au niveau fonctionnel, en modifiant une fonction par exemple, et de réutiliser une transformation de modèle préalablement créée pour re-générer le code du module applicatif cible.

La vue d'intégration offre ainsi la possibilité d'analyser la cohérence de l'architecture d'entreprise à travers la méta-modélisation, la relation de conformité de l'IDM et les transformations de modèle.

Cette vue donne ainsi accès aux informations de traçabilité qui permettent de déterminer l'impact d'une modification ou d'une défaillance d'un module applicatif sur les processus métier. Elle permet aussi de vérifier que les formats applicatifs permettent d'encoder les types de données fonctionnels requis, qui eux-mêmes raffinent les concepts métiers. Cette vue détermine les éventuelles transformations de modèle nécessaires au déploiement en spécialisant l'association « raffine ». Le choix des modèles à transformer dépend fortement de leur nature (modèles graphiques, textuels, etc.) et mais aussi de leur niveau d'abstraction. Par exemple, la génération de code demande un modèle en entrée suffisamment détaillé pour exécuter une transformation pertinente. L'état de l'art actuel des langages de transformation de modèle privilégie l'usage des transformations de modèle entre la vue fonctionnelle et et la vue applicative.

Il est indispensable de s'assurer de la cohérence entres les modèles des différentes vues avant d'initier une analyse d'impact du changement qui soit pertinente pour les parties prenantes, en particulier pour l'architecte d'entreprise. L'analyse de l'impact du changement consiste à dévoiler les effets de bord d'un changement apporté à un élément de l'architecture.

Dans le contexte du « framework ExecuteEA », l'analyse d'impact exploite les liens de traçabilité pour identifier les éléments du modèle impacté par une modification apportée à une entité du modèle d'architecture. L'analyse d'impact évalue par exemple la conséquence de l'indisponibilité d'un module applicatif sur l'architecture globale : en mettant à profit les liens de cohérence de la vue intégration, il est alors possible de déterminer quels sont les processus métier ou fonctionnels touchés par cette défaillance applicative. De la même manière, il devient possible d'identifier les modules applicatifs existants pouvant participer à la réalisation d'un nouveau processus métier si ce processus fait intervenir des tâches métier déjà implémentées dans le SI.

Nous proposons donc d'analyser les impacts en exprimant des requêtes sur les modèles d'architecture et d'étudier le résultat de ces requêtes. Nous tirons profit des méthodes IDM telle la méta-modélisation en les associant à des langages capables d'exprimer et d'exécuter des requêtes sur les modèles d'architecture, tels que OCLinEcore ou QVT.

L'automatisation de l'analyse d'impact est particulièrement cruciale pour les grandes entreprises qui ont à gérer un patrimoine applicatif important et un nombre de processus métier conséquent. Il s'agit là d'une analyse statique, le raisonnement concerne uniquement l'aspect structurel. Nous abordons l'analyse du comportement de la structure dans la suite de ce chapitre.

4.4.2.2 Analyse du comportement : simulation dirigée par les processus métier

Comme relaté dans l'état de l'art, l'analyse fait partie des activités les moins courantes de l'EA quelle qu'en soit l'école de pensée [2008-109] [2013-55]. Et même lorsqu'une architecture d'entreprise est analysée, très peu d'approches recourent à la simulation pour analyser l'aspect comportemental [2011-60] [2015-58]. La simulation est pourtant une technique reconnue pour évaluer le comportement d'un système et/ou évaluer plusieurs stratégies concernant son fonctionnement [1975-57]. Notre approche préconise de simuler les modèles issus des activités de modélisation et d'intégration afin de les valider ou les critiquer par l'ensemble des acteurs impliqués dans l'EA.

Pour simuler le comportement d'une entreprise nous nous appuyons sur les modèles d'architecture d'entreprise préalablement définis par les différentes parties prenantes. L'EA permet de capturer l'essentiel des composants d'une entreprise sous forme d'abstractions. Une approche par points de vue guide la décomposition d'une entreprise en vues pertinentes pour les différents acteurs. Ces vues apportent une aide supplémentaire à la définition du périmètre des modèles de simulation. La vue intégration permet en particulier de définir les liens entres les différentes vues, donc entre les différents modèles de simulation et de garantir la cohérence et donc la pertinence de la simulation. Comme les modèles de simulation sont directement dérivés de l'architecture d'entreprise telle qu'elle est définie par les parties prenantes, elle est d'autant plus facile à appréhender. Ces derniers peuvent aussi aisément communiquer et échanger autour des résultats de la simulation.

Les modèles d'entreprise doivent offrir un niveau d'abstraction suffisant à la compréhension, l'analyse et la communication. Les modèles doivent donc permettre d'abstraire les détails techniques et les nombreuses interconnexions tout en garantissant la traçabilité et la cohérence de l'ensemble de l'architecture. Notre approche consiste donc à mettre en lumière les composants et les relations qui sont critiques pour le comportement de l'ensemble de l'architecture. En effet, modéliser l'architecture d'une entreprise revient à la modélisation de systèmes complexes. Herbert Simon [1990-111] dans ses travaux de modélisation de systèmes complexes affirme que « l'approximation judicieuse et non la puissance de calcul d'une machine » reste la manière la plus effective d'adresser des systèmes complexes.

La simulation des processus métier est souvent réduite à de la simple animation visuelle de diagrammes pour vérifier l'orchestration des tâches métier. Nous proposons de piloter la simulation du comportement de l'architecture par le processus métier. Dans ce cas, le calcul d'une valeur ne se fait pas au niveau de la tâche métier, mais au niveau du module applicatif qui l'implémente. Le processus métier est modélisé sous forme de diagrammes

d'activité fUML. Dans ce cas, la simulation du comportement de l'architecture d'entreprise est pilotée par les processus métier qui sont alors responsables d'orchestrer l'ensemble des modèles comme l'illustre la figure 4.15.

La simulation est lancée après l'intégration de l'architecture à travers la création de liens de cohérences intra-vue et inter-vues. Ces liens sont par la suite utilisés pour mettre en œuvre la simulation. D'une part, la *Tâche A* de la figure 4.15 appelle le *Module A* car le *Module A* raffine la *Fonction A* qui elle-même raffine la *Tâche A*. D'autre part, les liens de cohérences intra-vue garantissent une compatibilité entre les informations envoyées par la *Tâche A* et celles attendues par la *Tâche B*, de même entre la *Fonction A* et la *Fonction B* et entre le *Module A* et le *Module B*.

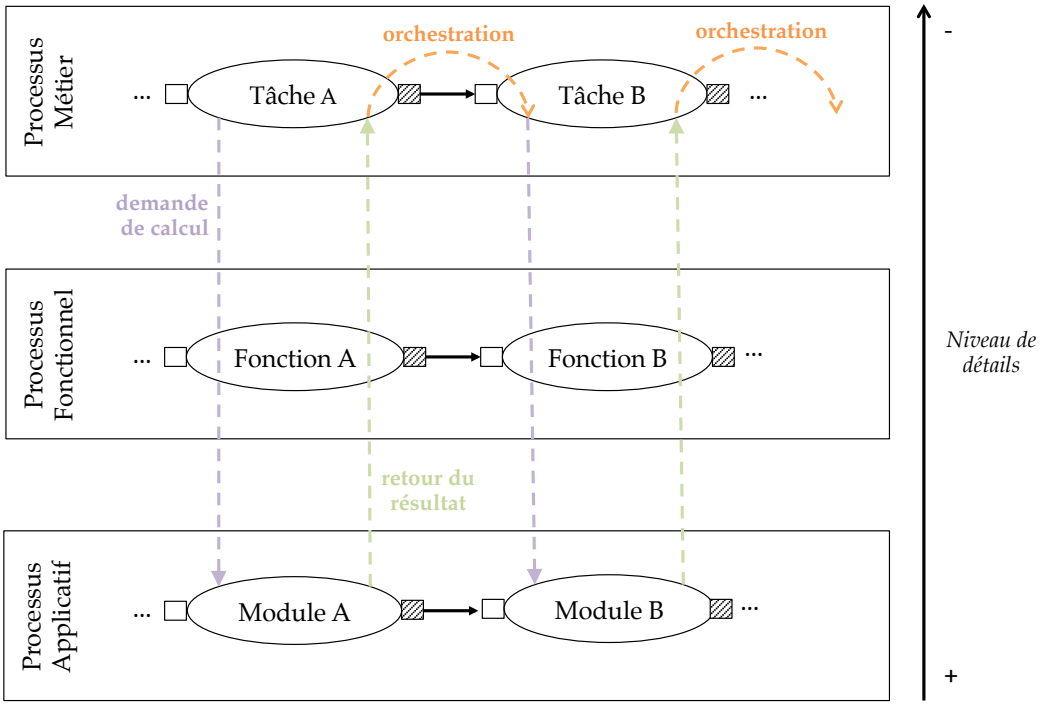

FIGURE 4.15 – Simulation de l'architecture dirigée par les processus

En plus de s'appuyer sur les processus métier pour piloter la simulation de l'ensemble du comportement de l'architecture d'entreprise, notre approche consiste à mettre à profit les techniques et les langages de l'IDM pour faciliter l'automatisation de l'activité d'analyse du comportement. Nous nous appuyons sur le manifeste de IBM [2006-93] concernant l'IDM dans la sélection de techniques et langages qui soient pertinents pour notre approche. Le manifeste de IBM recommande l'utilisation de langages (1) exécutables, (2) standardisés et (3) compréhensibles par les experts du domaine. C'est le cas de fUML, BPMN et OCL. La table 4.1 fait la correspondance entre les langages et la possibilité de les utiliser selon les

vues.

	Metier	Fonctionnel	Applicatif
BPMN	✓		
fUML	✓	✓	
OCL		✓	
MiniZinc			✓

TABLE 4.1 – Langages de l'IDM pour l'EA

Ces langages permettent une exécution directe des modèles créés, contrairement à d'autres méthodes de simulation de processus métier qui utilisent l'IDM pour isoler la définition du processus de son exécution. Ces méthodes font ensuite appel aux transformations de modèle pour automatiser la conversion entre les modèles de représentation et leur exécution.

4.5 Conclusion

Dans ce chapitre, nous avons présenté un *framework* offrant un cadre structurant et unificateur pour les différentes activités d'EA. Ce *framework*, appelé ExecuteEA, permet d'automatiser l'analyse structurelle et comportementale d'une architecture d'entreprise.

L'analyse structurelle s'appuie sur l'exploitation des liens de traçabilité du métamodèle EAT-ME. Ce métamodèle explicite en effet des liens de cohérence intra-vue et inter-vues reliant les différents composants d'une architecture d'entreprise.

Nous avons proposé, pour l'analyse comportementale, une approche de simulation d'architecture dirigée par les processus métier. La simulation est rendue possible par le recours aux transformations de modèle et aux langages de modélisation exécutable.

La définition du métamodèle EAT-ME a nécessité une analyse du domaine de l'EA en général, et dans le contexte des Smart Grids en particulier, conformément à la démarche préconisée par l'IDM.

Dans le chapitre suivant, nous décrivons une implémentation du *framework* ExecuteEA que nous éprouvons sur le cas métier d'une gestion de flotte de véhicules électriques.

Chapitre 5

Prototypage et validation du *framework ExecuteEA*

Sommaire

Dans le chapitre précédent, nous avons présenté en détails le *framework* ExecuteEA et son métamodèle EAT-ME dont l'objectif principal est de fournir un cadre cohérent pour l'analyse de la structure et du comportement d'une architecture d'entreprise. Le présent chapitre décrit l'implémentation qui concrétise les propositions que nous faisons à travers ce *framework*.

Cette concrétisation prend la forme d'un prototype. Nous expliquons d'abord les choix techniques relatif à la plate-forme de développement retenue (section 5.1). Nous détaillons ensuite la réalisation du prototype et nous exposons les difficultés rencontrées (section 5.2).

Grâce à ce prototype, nous éprouvons le *ExecuteEA* sur le cas métier de la gestion d'une flotte de véhicules électriques (section 5.3). Nous concluons ce chapitre en discutant les résultats de l'implémentation du *framework* ExecuteEA et en résumant les réalisations actuelles comme futurs.

5.1 Environnement retenu : la plate-forme Eclipse

Notre choix s'est orienté vers la plate-forme Eclipse et ce pour plusieurs raisons.

Tout d'abord, il s'agit d'une plate-forme open-source dont l'utilisation est particulièrement répandue dans la communauté IDM. En effet, la plate-forme Eclipse abrite le projet Eclipse Modeling Framework (EMF) [1] qui a pour objectif de doter Eclipse d'outils orientés IDM.

Ensuite, EMF s'appuit sur les standards du domaine. Par exemple, le méta-métamodèle Ecore, pilier central de EMF, se base sur le standard MOF [2]. Aure exemple, le langage de contraintes OCLinEcore se base sur le standard OCL. OCL et MOF sont des standards de l'OMG.

Puis, EMF se décompose en sous-projets orientés vers différents aspects de l'IDM tels que la méta-modélisation, la transformation de modèle, les éditeurs graphiques de modèles, les langages spécifiques au domaine. EMF offre ainsi un environnement personnalisable pour mettre en œuvre une approche IDM.

Enfin, il s'agit de la plate-forme retenu par le projet Pomme du département MIRE, dans lequel s'inscrivent nos travaux de thèse. Le projet Pomme vise à réaliser un outil pour la co-simulation des trois domaines qui composent un Smart Grid : SI, infrastructure électrique, infrastructure de télécommunication.

Pour ces travaux de thèse, nous avons donc retenu :

- Ecore pour la méta-modélisation ;
- OCLinEcore pour l'expression de contraintes et de requêtes sur le métamodèle ;
- Le langage Acceleo pour les transformations de modèle ;
- Le plugin Papyrus [3] pour la simulation des modèles d'architecture. Papyrus est compatible avec le standard fUML et permet donc d'exécuter les diagrammes de classes et d'activité.

1. http ://www.eclipse.org/modeling/emf/
2. http ://www.omg.org/mof/
3. https ://eclipse.org/papyrus/

5.2 Réalisation et difficultés rencontrées

5.2.1 Implémentation du métamodèle EAT-ME avec Eclipse Modeling Framework

Le métamodèle EAT-ME est le pilier central du *framework ExecuteEA*. Nous l'avons implémenté à l'aide de EMF. Il est donc conforme au méta-métamodèle Ecore. EMF génère automatiquement un éditeur de modèle à partir d'un métamodèle conforme à Ecore. Cet éditeur permet de créer des modèles d'architecture d'entreprise conformes au métamodèle EAT-ME ainsi implémenté. La figure 5.1 et la figure 5.2 sont une capture d'écran de l'éditeur de modèles généré automatiquement à partir du métamodèle EAT-ME. L'utilisateur a le choix de créer une vue métier, fonctionnelle, applicative ou intégration et pour chacune des vues il peut choisir de créer un aspect objectif, processus ou information.

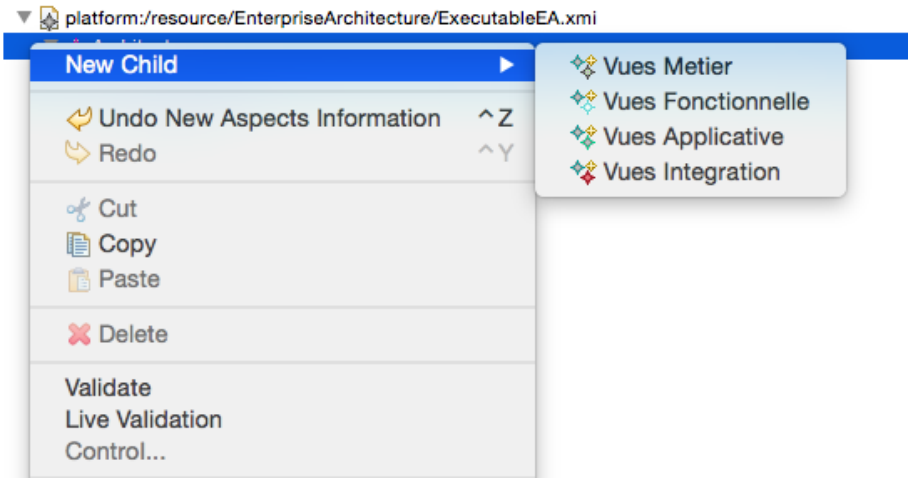

FIGURE 5.1 – Edition de modèles d'architecture d'entreprise avec ExecuteEA création de vues

La figure 5.3 représente une capture d'écran d'un modèle d'architecture d'entreprise créé avec l'éditeur de modèle ExecuteEA.

La difficulté majeure rencontrée en implémentant le métamodèle EAT-ME a été de trouver le bon équilibre entre deux impératifs : (1) implémenter le métamodèle en utilisant des concepts et des relations qui font sens d'un point de vue EA, en se gardant la possibilité de l'étendre facilement, notamment, par d'autres aspects ou d'autres vues, (2) tout en créant un éditeur suffisamment contraignant pour créer des modèles d'architecture corrects. Par exemple, utiliser uniquement le mécanisme de multiplicités entre la classe abstraite « Vue » et la classe « Architecture » ne contraint pas le modélisateur à ne pas créer plusieurs fois le même type de vue pour une seule architecture. Un modèle avec deux vues métier serait par conséquent conforme au métamodèle mais ne serait pas pertinent d'un point de vue EA.

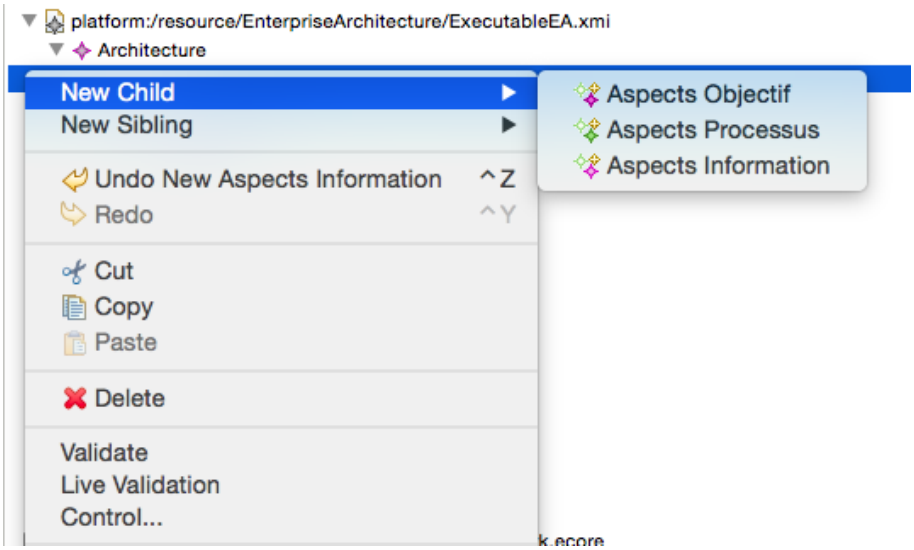

FIGURE 5.2 – Edition de modèles d'architecture d'entreprise avec ExecuteEA création d'aspects

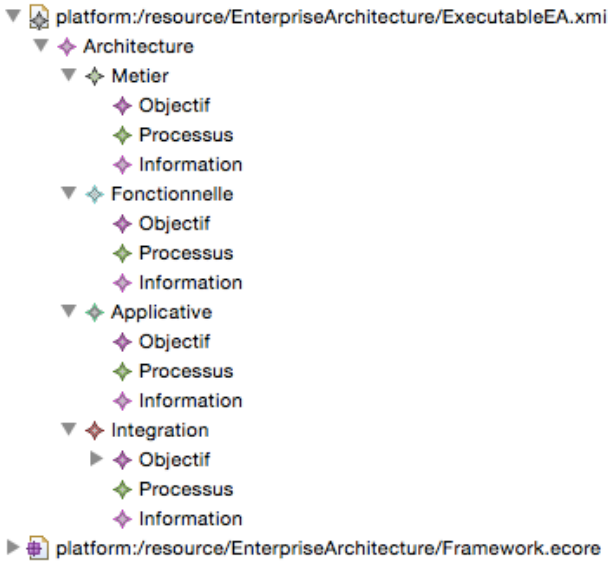

FIGURE 5.3 – Modèle d'architecture d'entreprise créé avec l'éditeur ExecuteEA

Nous avons donc exprimé des contraintes avec OCLinEcore pour ne créer que des modèles corrects tout en gardant un métamodèle pertinent d'un point de vue purement EA. Une contrainte est une expression à valeur booléenne qui sert à préciser ou restreindre n'importe quel élément du métamodèle.

La figure 5.4 illustre le métamodèle EAT-ME, implémenté à l'aide d'EMF, sous une forme textuelle [4]. Il s'agit donc de la classe « Architecture » à la quelle on a ajouté des contraintes de type « invariant » pour qu'elle ne contienne pas plusieurs vues du même type.

```
class Architecture
{
    property vues : Vue[2..4] { ordered composes };
    attribute name : String[?];
    invariant uneSeuleVueMetier:
    self.vues->select(obj | obj.oclIsKindOf(Vue_Metier))-> size()=1;
    invariant uneSeuleVueFonctionnelle:
    self.vues->select(obj | obj.oclIsKindOf(Vue_Fonctionnelle))-> size()=1;
    invariant uneSeuleVueApplicative:
    self.vues->select(obj | obj.oclIsKindOf(Vue_Applicative))-> size()=1;
    invariant uneSeuleVueIntegration:
    self.vues->select(obj | obj.oclIsKindOf(Vue_Integration))-> size()=1;
}
```

FIGURE 5.4 – Contraintes OCLinEcore attachées à la classe « Architecture »

5.2.2 Exécution des modèles d'architecture avec Papyrus

Papyrus est un atelier de modélisation orienté IDM doté d'un moteur d'exécution de diagrammes fUML conformément aux spécifications de l'OMG. Nous l'avons utilisé pour simuler l'architecture. Comme présenté dans le chapitre 4, nous proposons de simuler l'ensemble de l'architecture en la pilotant par les modèles. La difficulté majeure rencontrée à cette étape de l'implémentation a été de réussir à exécuter des comportements difficiles à modéliser avec un diagramme UML [5]. Le principe de simulation pilotée par les processus métier, tel que présenté dans la figure 4.15 (page 86), exige de plus de faire appel aux modules de la vue applicative.

Le standard fUML a prévu l'exécution de comportements spécifiques difficiles à modéliser avec des diagrammes d'activité. Il dédie à cet effet une action [6] spécialisée : l'*OpaqueAction*. Papyrus s'appuie sur le moteur d'exécution Moka pour simuler les diagrammes fUML. Moka est conforme à la sémantique d'exécution spécifiée par l'OMG pour fUML. Or la sémantique d'exécution de l'*OpaqueAction* n'a pas encore été spécifiée. Le standard fUML est en effet

4. EMF est doté d'un éditeur graphique qui permet de créer des métamodèles de manière graphique (par *drag and drop*) sous la forme d'un diagramme de classe et de générer le même métamodèle sous une forme textuelle.

5. Typiquement, l'optimisation d'une affectation de véhicules électriques à des tournées d'agents qui intervient dans le cas d'études présenté dans la suite de ce chapitre

6. Dans les spécifications d'UML et de fUML, une action est une étape unitaire à l'intérieur d'une activité. Une activité est donc composée de plusieurs actions.

relativement récent [7] et n'a pas encore été entièrement spécifié. Pour cette raison, le moteur d'exécution Moka ne prend pas en charge l'exécution de l'*OpaqueAction*

Cependant, Papyrus donne la possibilité d'étendre le moteur d'exécution Moka en attribuant le comportement souhaité à *OpaqueAction* et en l'exécutant comme les autres actions au cours d'une simulation de diagramme d'activité. Cette extension n'a toutefois pas été facile à implémenter. Les tutoriels mis à disposition par l'équipe de développement du module Moka sont utiles pour une utilisation basique mais ne répondent pas à des besoins d'utilisation pointue telle que l'extension du moteur d'exécution disponible. À cet effet, nous avons sollicité l'aide de contributeurs au développement du module Moka et nous nous sommes également appuyés sur un stagiaire en dernière école d'ingénieur. Le comportement de l'*OpaqueAction* mis en œuvre dans ces travaux de thèse consiste donc à appeler un module extérieur durant l'exécution d'un diagramme d'activité fUML et à retourner le résultat fourni par le module à l'action ou l'activité suivante.

Dans la suite de ce chapitre, nous éprouvons le *framework ExecuteEA* ainsi doté d'un environnement de modélisation et de simulation de modèles d'architecture. La validation est faite à travers le cas d'étude de la gestion d'une flotte de véhicules électriques. Nous avons créé à cet effet (1) les modèles d'architecture d'entreprise requis par les différentes vues, selon les aspects spécifiés par *ExecuteEA*, et en adoptant les langages prescrit par le *framework*, et (2) les transformations de modèle nécessaires. Nous avons ensuite analysé l'architecture obtenue.

5.3 Concrétisation de l'approche avec un cas d'étude

Les contraintes de temps et de confidentialité ne nous ont pas permis d'éprouver le framework ExecuteEA sur une architecture d'entreprise à taille réelle. Pour cette raison, le cas d'étude concerne plutôt un seul processus métier faisant intervenir plusieurs tâches et sa déclinaison sur les vues fonctionnelle et applicative. Il s'agit de la gestion d'une flotte de véhicules électriques. Ce cas métier nous permet néanmoins de concrétiser les propositions de ces travaux de thèse et de parer à la difficulté d'évaluer quantitativement ces propositions.

5.3.1 Présentation du cas d'étude : la gestion d'une flotte de véhicules électriques

Les raisons motivant le choix de ce cas d'étude pour éprouver le framework proposé dans ces travaux de thèse ont été discutées dans la section 4.2.1.3 (voir page 68). Dans cette partie, nous nous contentons donc simplement de le présenter.

Il s'agit du processus d'affectation de véhicules à des tournées d'agents (par exemple, une tournée d'un agent EDF qui relève les compteurs chez les clients, ou encore qui fait des

7. La première version de fUML a été publiée par l'OMG en févier 2011

réparations sur le réseau électrique). La mobilité électrique implique un changement de paradigme pour le gestionnaire de flotte de l'entreprise. D'une part, le véhicule électrique est limité par son autonomie et ne peut donc pas effectuer n'importe quelle tournée. D'autre part, la recharge d'un véhicule électrique implique des contraintes (temps de recharge, disponibilité des bornes) que ne présente pas le véhicule thermique qui se contente d'un plein de carburant.

Dès lors, le processus d'affectation de véhicules aux tournées des agents, la gestion de la flotte de véhicules dans son ensemble et donc le SI qui l'implante sont fortement impactés par l'arrivée massive des véhicules électriques. Nous proposons de modéliser de cas d'étude en mettant en œuvre le *framework ExecuteEA*. Nous adoptons donc l'approche conceptuelle préconisée par le framework (voir figure 4.12, 81). Nous commençons donc par modéliser les différentes vues d'architecture — métier, fonctionnelle, applicative et intégration — en respectant le cadre structurant du *framework ExecuteEA* (illustré par la figure 4.13, page 82) ce qui nous permettra ensuite d'analyser la structure et le comportement de l'architecture ainsi obtenue.

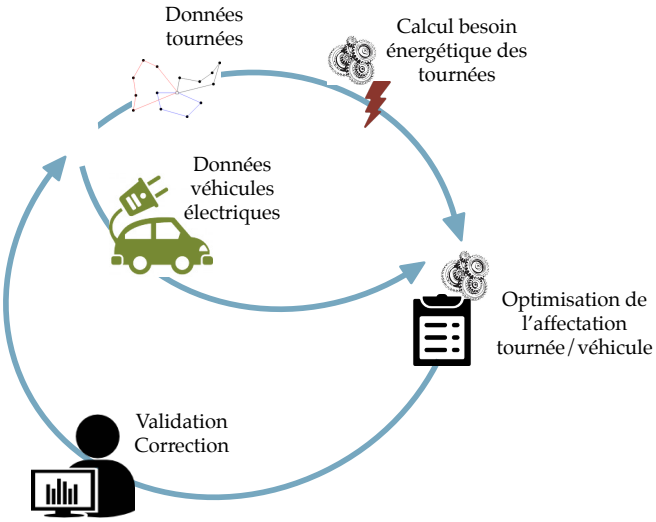

FIGURE 5.5 – Processus d'affectation de véhicules électriques à des tournées

Pour ce cas d'étude, l'analyse de la structure exploite les liens de la vue intégration et la conformité du modèle d'architecture au métamodèle EAT-ME. L'analyse du comportement consiste à simuler l'ensemble de l'architecture en pilotant la simulation par le processus métier. Cette simulation a pour objectif de valider et critiquer les choix de modélisation et d'anticiper l'éventuel dimensionnement de la flotte. Par exemple, si une forte proportion des tournées implique une distance effectuée supérieure à l'autonomie des véhicules électriques sans possibilité de recharge en cours de route (pas de borne à disposition au cours de la tournée), la simulation permet de trouver la proportion de véhicules thermiques à garder a minima dans une flotte. L'affectation doit aussi privilégier l'utilisation des véhicules

électriques car la rentabilité d'un parc de véhicules électriques est proportionnelle au nombre de kilomètres effectués par ces véhicules.

5.3.2 Mise en œuvre du *framework ExecuteEA* pour la modélisation des vues métier, fonctionnelle et applicative

La première étape consiste donc à créer les modèles des vues métier, fonctionnelle et applicative en recourant à des langages de modélisation exécutables. Selon les propositions de ces travaux de thèse, l'usage de ces langages automatise la manipulation des modèles et facilite leur analyse. La figure 5.6 présente l'architecture *in globo* du cas métier ainsi modélisé, selon le cadre structurant *ExecuteEA*.

5.3.2.1 Modélisation de la vue métier

Nous utilisons fUML comme langage exécutable pour modéliser cette vue. Le processus métier consiste à collecter les données relatives aux véhicules (électriques et thermiques) ainsi qu'aux tournées à effectuer, à calculer l'énergie nécessaire à chaque tournée et l'affectation véhicule/tournée avant de faire valider cette dernière par le manager de flotte. Nous modélisons ce processus métier sous forme de diagramme d'activité fUML en utilisant l'atelier de modélisation Papyrus. Selon notre cadre d'architecture, les modèles créés représentent donc l'aspect processus de la vue métier. La figure 5.7 présente le diagramme d'activité fUML obtenu à l'aide de Papyrus.

Pour l'aspect information, nous utilisons des diagrammes de classe UML pour représenter les concepts métier et leurs relations. Nous modélisons ainsi les concepts de Véhicule, Tournée et Affectation. La figure 5.8 présente le diagramme de classes fUML modélisé dans Papyrus.

Gérer une flotte de véhicules peut avoir plusieurs objectifs métier. Dans notre cas, il s'agit d'obtenir le meilleur retour sur investissement suite à l'intégration de véhicules électriques dans la flotte de véhicules. Nous modélisons l'objectif métier sous la forme d'une classe UML représentant l'aspect objectif de la vue métier comme l'illustre la figure 5.6.

5.3.2.2 Modélisation de la vue fonctionnelle

Nous modélisons l'aspect information de la vue fonctionnelle sous la forme d'un diagramme de classes fUML. Ce modèle raffine les concepts métier en spécifiant leurs types. Dans la vue fonctionnelle, le concept d'allocation prend la forme d'une association entre les véhicules et les tournées.

L'objectif fonctionnel est d'optimiser l'utilisation des véhicules électriques pour atteindre l'objectif métier qui est d'avoir un meilleur retour sur investissement. Nous modélisons cet objectif sous la forme d'une classe UML représentant l'aspect objectif de la vue fonctionnelle comme l'illustre la figure 5.6.

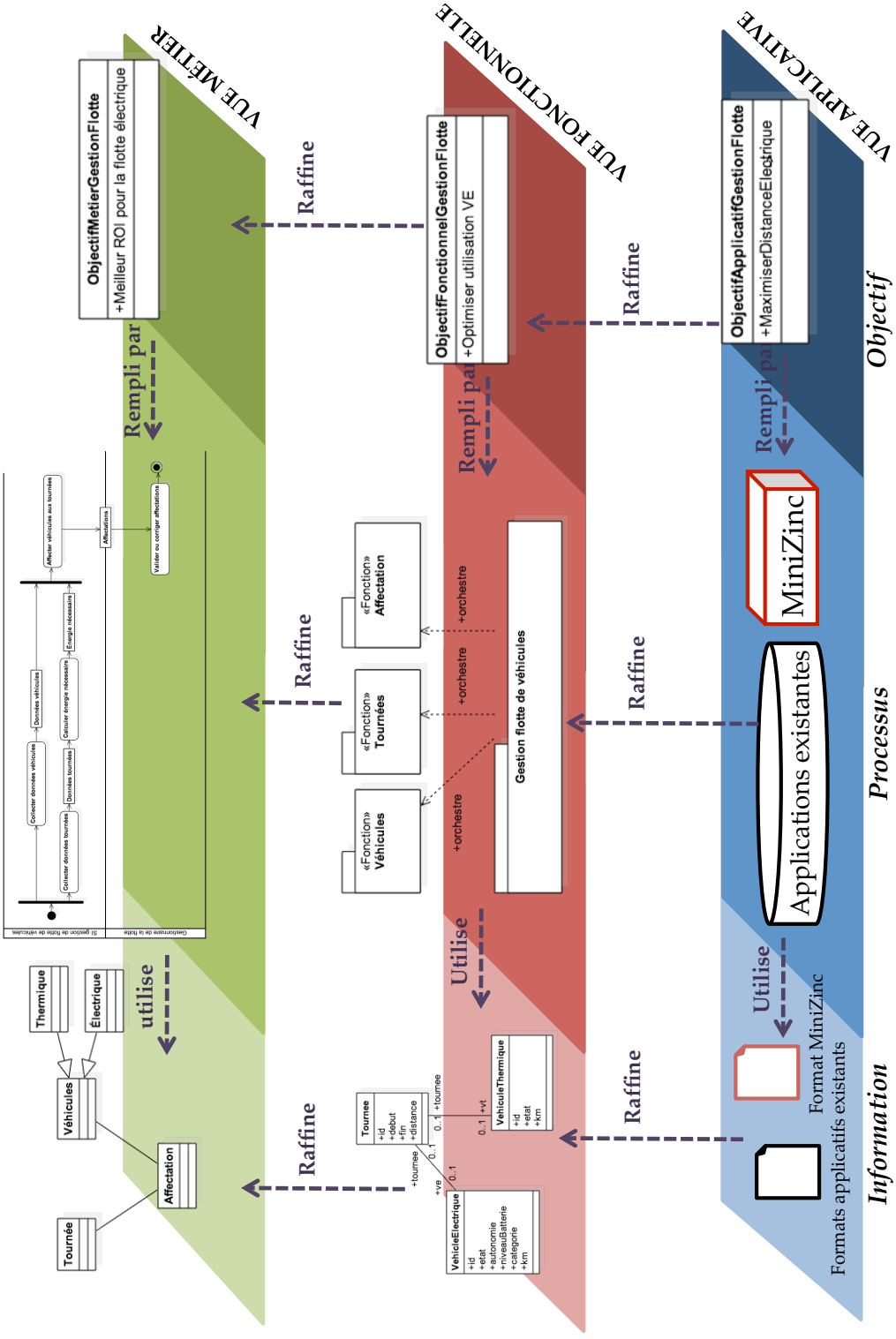

FIGURE 5.6 – Architecture globale du cas d'étude mettant en œuvre le *framework ExecuteEA*

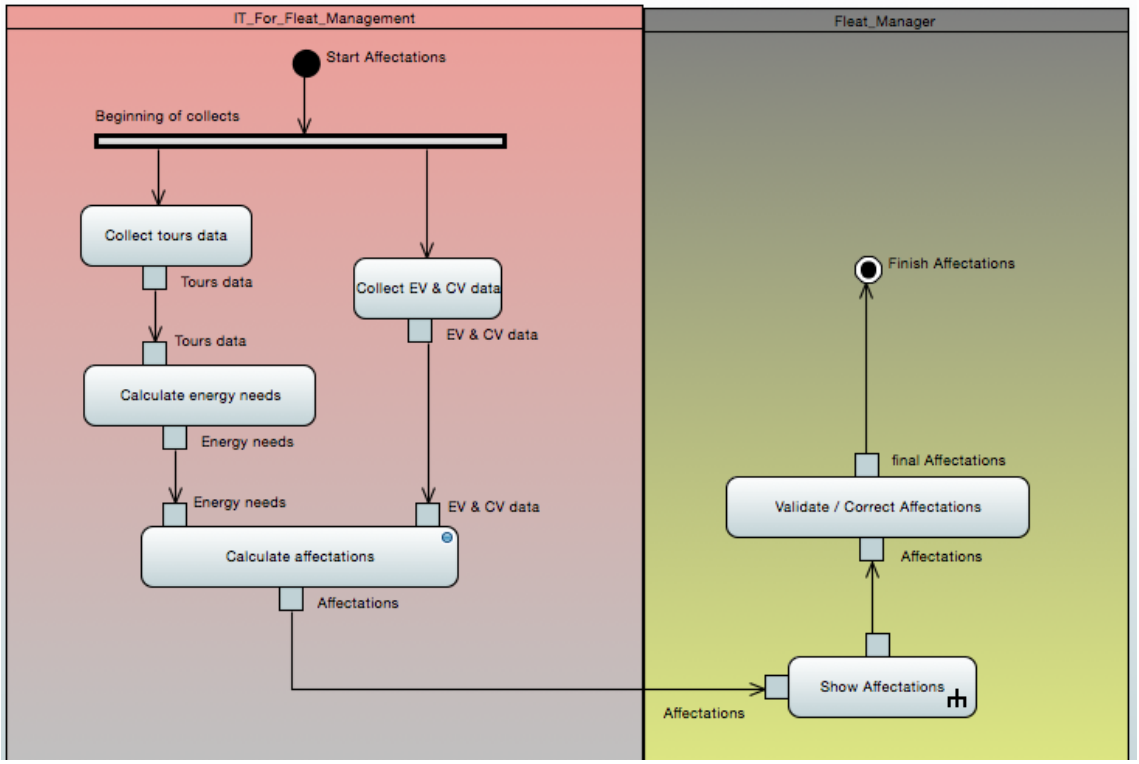

FIGURE 5.7 – Aspect processus de la vue métier modélisé sous la forme d'un diagramme d'activité fUML avec Papyrus

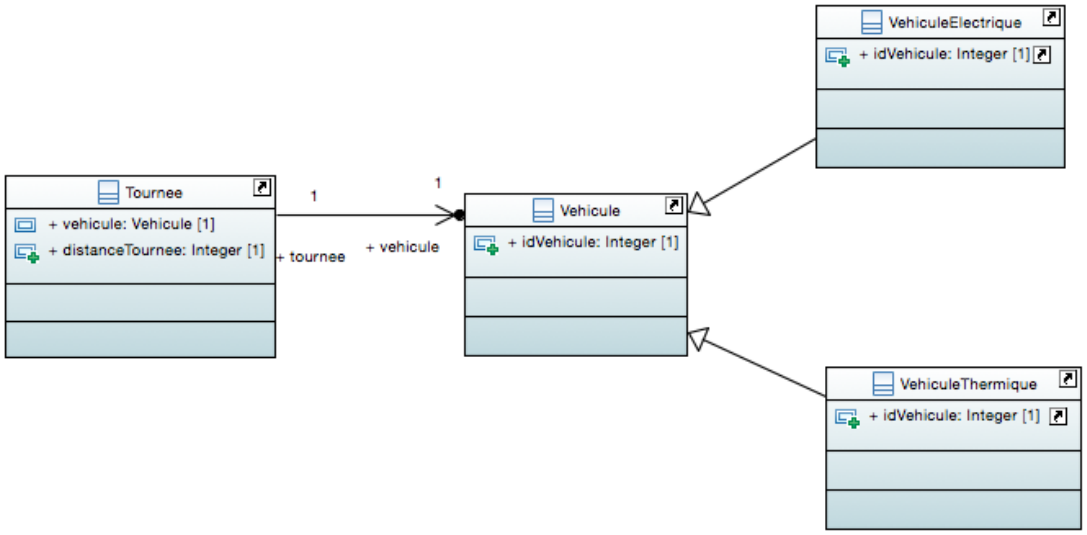

FIGURE 5.8 – Aspect information de la vue métier modélisé sous la forme d'un diagramme de classes fUML avec Papyrus

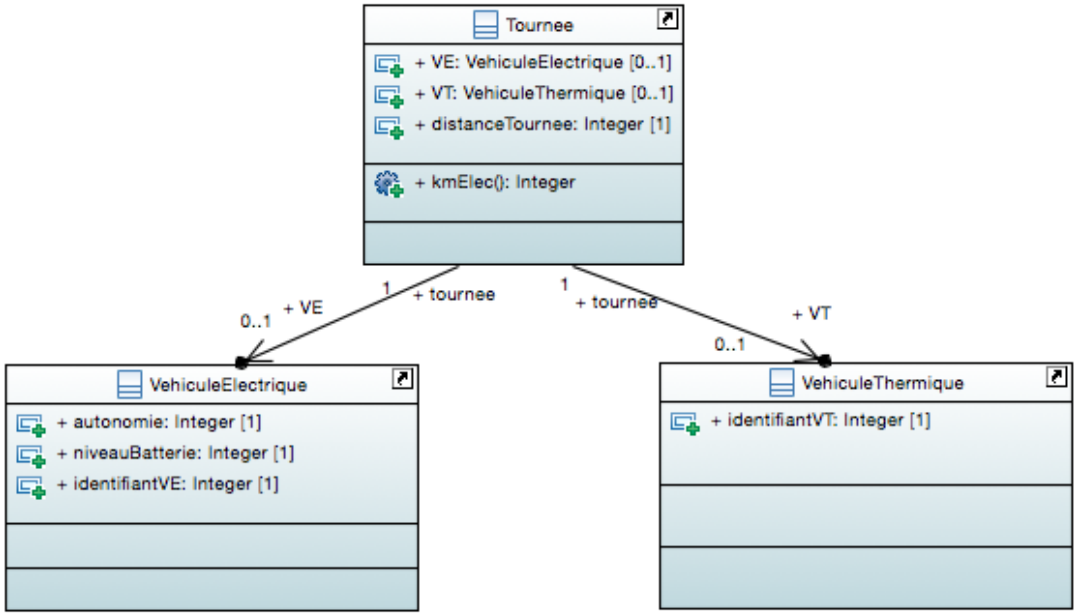

FIGURE 5.9 – Aspect information de la vue fonctionnelle modélisé sous la forme d'un diagramme de classes fUML avec Papyrus

Pour l'aspect processus de la vue fonctionnelle, nous commençons par identifier trois blocs fonctionnels : un bloc pour la gestion de la flotte de véhicules (électriques et thermiques), un bloc pour la gestion des tournées, un bloc pour la gestion de l'affectation (voir figure 5.6. Ces blocs contiennent les fonctions qui raffinent les tâches du processus métier. Comme expliqué plus tôt dans la démarche, le fait de les rassembler dans des blocs selon les concepts métier augmente la modularité et l'évoluvilité de l'architecture. De plus, nous consacrons un bloc à la gestion des processus fonctionnels. Ce bloc est responsable de l'orchestration des fonctions du processus fonctionnel.

Les blocs fonctionnels offrent une vue plus détaillée des tâches métier. Pour ce cas d'étude, nous détaillons uniquement la tâche métier qui consiste à affecter un véhicule à une tournée. Nous modélisons l'affectation des véhicules aux tournées sous la forme de contraintes OCL : pour affecter un véhicule à une tournée, il faut que l'énergie nécessaire à celle-ci soit inférieure à l'autonomie de la batterie. Dans notre cas d'application, nous considérons qu'il n'est pas possible de recharger le véhicule pendant la tournée de l'agent.

Nous modélisons la fonction d'affectation sous la forme de deux contraintes et d'une requête en utilisant OCL. La première contrainte OCL signifie que si un véhicule électrique est affecté à une tournée, alors l'énergie dont il dispose permet d'assurer la totalité de la tournée. La deuxième contrainte signifie que si aucun véhicule électrique n'est capable d'assurer une tournée donnée alors c'est un véhicule thermique qui lui est associé. Enfin, la requête calcule le nombre total de kilomètres électriques correspondant à la distance parcourue par les véhicules électriques après l'affectation. Cette requête permet d'évaluer l'utilisation des véhicules électriques dans l'optique d'atteindre l'objectif fonctionnel.

Il est possible de modéliser les autres algorithmes de traitement (calcul des tournées à partir de bons de travaux, calcul de l'énergie nécessaire à une tournée, etc.) à l'aide de diagrammes d'activité exécutables.

```
context Tournee
inv : self.ve.autonomie * self.ve.niveauBatterie > self.energieRequise

context Tournee
inv : self.ve <> undefined xor self.vt <> undefined

context Tournee :: elecKm() : int
body : (Tournee::allInstances() -> collect(t.ve <> undefined|t.distance)) -> sum()
```

Listing 5.1 – Contraintes OCL pour la fonction d'affectation

5.3.2.3 Modélisation de la vue applicative

Pour les processus applicatifs, nous commençons par identifier les applications nécessaires à l'implantation des blocs fonctionnels. Dans notre cas, le patrimoine applicatif de l'entreprise dispose déjà d'applications pour la gestion de tournées (calcul de tournées optimisé à partir de bons de travaux) et la gestion de véhicules (administration, maintenance, etc.). Pour la fonction d'allocation, nous faisons le choix d'utiliser MiniZinc pour modéliser les contraintes au niveau applicatif (voir figure 5.10).

```
constraint forall (i in Tournees) (
%affectation de vehicule electrique si l autonomie l autorise
constraint forall(i in  Tournees, j in VehiculesElec)
(tourneeVehicule[i]= identifiantVE[j] -> (autonomie[j]*niveauBatterie[j] > distanceTournee[i] /\
kmElec[i]=distanceTournee[i]));

% affectation d un vehicule thermique et dans ce cas kmElec est nul
constraint forall(i in  Tournees, k in VehiculesTherm)
(tourneeVehicule[i]= identifiantVT[k] -> kmElec[i]=0);

%maximiser le nombre de km de tounrnee fait par les vehicules electriques
solve maximize sum(i in Tournees)(kmElec[i]);
```

FIGURE 5.10 – Contraintes du module MiniZinc

MiniZinc est un langage de modélisation et de résolution de contraintes de niveau intermédiaire qui a pour vocation de devenir un langage de modélisation standard dans le domaine de la programmation par contraintes. L'aspect information contient les formats de données nécessaires aux différentes applications. La figure 5.11 représente le fichier de données (le format .dzn) nécessaire à l'application MiniZinc pour calculer l'affectation.

```
nbreVE = 4;
identifiantVE = [1, 2, 3, 4];
nbreVT = 4;
identifiantVT = [11, 22, 33, 44];
autonomie = [1500, 1200, 1500, 1200];
niveauBatterie = [8, 6, 5, 2];
nbreTournees = 4;
distanceTournee = [1000, 700, 800, 1600];
```

FIGURE 5.11 – Fichier de données pour le module MiniZinc

Nous modélisons l'objectif applicatif sous la forme d'une classe UML représentant l'aspect objectif de la vue applicative comme l'illustre la figure 5.6. Un véhicule électrique devient rentable par rapport à un véhicule thermique à partir d'un certain nombre de kilomètres parcourus. C'est pourquoi l'objectif fonctionnel qui est d'optimiser l'usage de la flotte électrique se traduit par la maximisation du nombre de kilomètres électriques, c'est à dire affecter aussi souvent que possible un véhicule électrique aux tournées. Ainsi, le module MiniZinc prend en compte cet objectif en résolvant les contraintes tout en maximisant la distance électrique.

5.3.3 Intégration et analyse de la structure

L'approche conceptuelle que nous proposons pour le *framework ExecuteEA*, illustrée par la figure 4.12 à la page 81, met l'accent sur le rôle intégrateur de l'architecte d'entreprise : il doit s'assurer de la cohérence intra-vue et inter-vues de l'architecture globale tout en collaborant avec les architectes métier, fonctionnel et applicatif.

L'intégration de l'architecture passe par la modélisation de la vue intégration, c'est-à-dire par la spécification des liens de cohérence intra-vue et et inter-vues conformément au

métamodèle EAT-ME. Nous avons donc modélisé ces liens pour l'ensemble des vues métier, fonctionnelle et applicative. Néanmoins, la figure 5.12 ne présente qu'une partie du modèle d'intégration pour des raisons de lisibilité. Nous y modélisons à titre d'illustration les liens de cohérence intra-vue entre la fonction d'affection, ses inputs et output en termes de types fonctionnels, ainsi que son objectif fonctionnel. Nous faisons de même pour le module d'optimisation MiniZinc, ses inputs et outputs ainsi que l'objectif applicatif qu'il remplit. Le même principe d'intégration intra-vue, c'est à dire entre les aspects d'une même vue, est applicable à tous les autres éléments des vues métier, fonctionnelle et applicative.

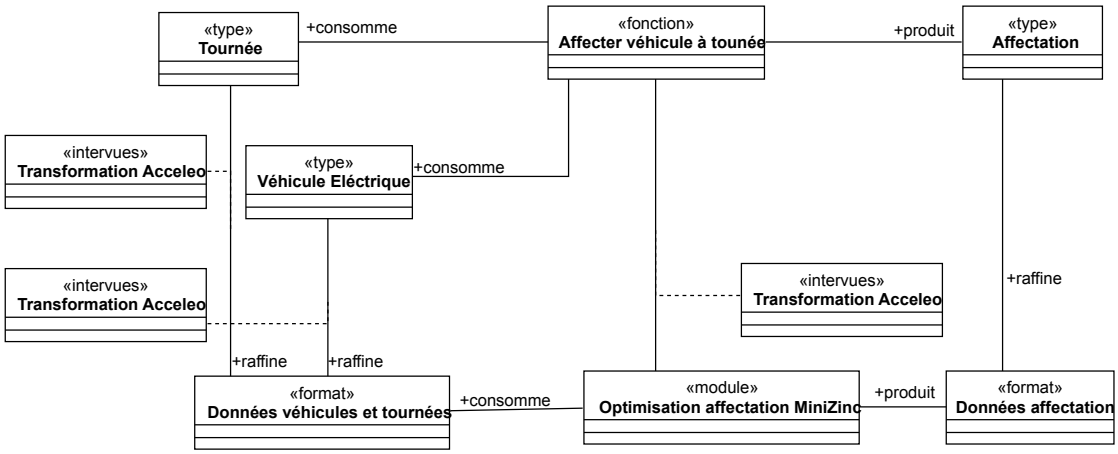

FIGURE 5.12 – Une vue partielle des modèles d'intégration de la vue fonctionnelle et de la vue applicative

De la même manière, nous modélisons les liens de cohérence inter-vues. Par exemple, le lien « consomme » exprime qu'un type de données est compatible avec la fonction qui l'utilise et qu'un format est compatible avec le module applicatif qui l'utilise en entrée. Pour garantir une bonne orchestration des processus, il faut que le « produit » d'une tâche (respectivement une fonction, un module applicatif) soit compatible avec ce que produit la tâche suivante (respectivement la fonction, le module applicatif).

L'intégration passe aussi par l'identification des éventuelles transformations de modèle. Rappelons ici que les transformations ont pour objectif d'améliorer l'alignement métier/IT en automatisant le passage d'une vue à une autre. Pour le cas métier de la gestion de flotte de véhicules, nous utilisons une transformation de modèle pour générer les contraintes pour le module de calcul d'affectation MiniZinc de la vue applicative à partir des contraintes OCL exprimées dans la fonction affectation de la vue fonctionnelle. La transformation est écrite dans le langage de transformation Acceleo. En plus de transformer les contraintes décrites dans l'aspect métier, cette transformation de modèle permet aussi de transformer les instances des types fonctionnels en instances dans le format « .dzn » utilisé par le module MiniZinc comme l'illustre la figure 5.12. La génération de code pour le module MiniZinc peut être lancée à partir de la vue fonctionnelle que nous avons précédemment modélisée dans Papyrus comme l'illustre la figure 5.13.

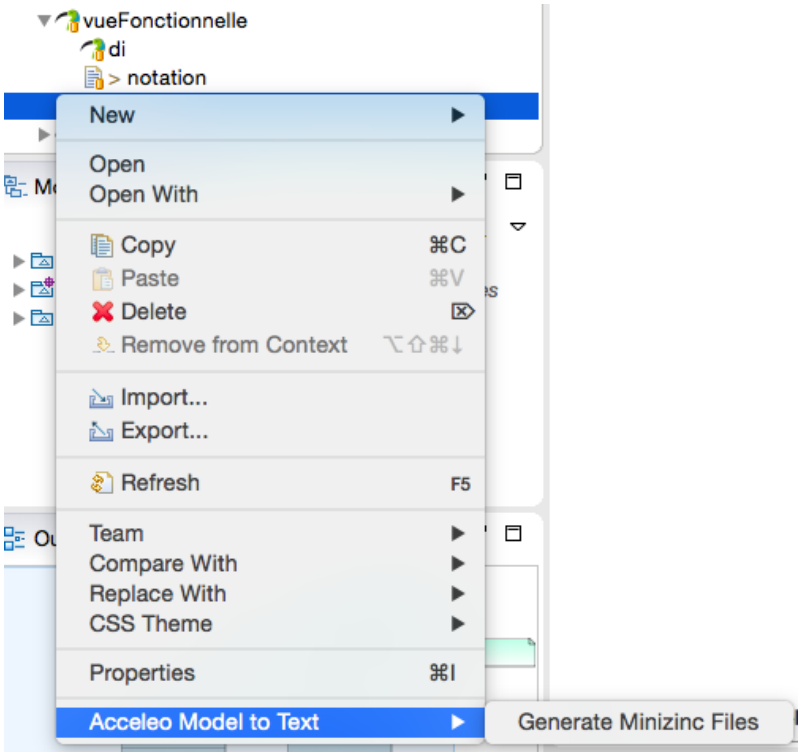

FIGURE 5.13 – Génération du code pour le module MiniZinc à partir de la vue fonctionnelle dans Papyrus

5.3.4 Simulation et analyse du comportement

Grâce aux langages exécutables, l'analyse du comportement des modèles d'architecture se fait directement dans les langages de modélisation utilisés pour les vues. Avec le *framework* ExecuteEA, nous proposons de simuler l'ensemble de l'architecture en la pilotant par l'exécution du processus métier. La simulation du processus métier se traduit par l'exécution du diagramme d'activité fUML à l'aide du moteur d'exécution Moka intégré à Papyrus. Un des avantages de l'outil Papyrus est de permettre la modélisation *et* la simulation de l'architecture dans un même environnement.

Notre approche préconise de modéliser les détails des tâches dans les vues inférieures afin de respecter le niveau d'abstraction requis par chaque point de vue et ainsi de ne pas altérer la compréhension des parties prenantes de la vue qui leur est destinée. Par exemple, l'analyste métier n'aura ainsi pas à discuter du détail des applications implémentant les tâches métier avec l'architecte applicatif.

Comme expliqué dans la section 5.2.2, Papyrus offre la possibilité d'étendre la sémantique d'exécution de fUML à travers les *Opaque Action*. Celles-ci permettent d'invoquer des modules d'applications extérieures au moment de l'exécution du diagramme d'activité fUML. Développée pour les besoins du cas d'étude, cette extension rend possible l'invocation directe du module MiniZinc pendant l'exécution du processus métier pour optimiser l'affectation des véhicules aux tournées. Rappelons que le module MiniZinc est obtenu par transformation de modèle à partir de la vue fonctionnelle.

La simulation prend la forme d'une animation de diagramme. La figure 5.14 est une capture d'écran montrant la simulation du processus métier en cours d'exécution. Papyrus offre la possibilité de mettre des *breakpoints* sur certaines activités et de paramétrer le pas de temps pour contrôler le déroulement du processus.

La simulation retourne comme résultat les affectations des véhicules aux tournées. Ce résultat est affiché dans la console fUML comme l'illustre la figure 5.15. La simulation a pour objectif de voir si l'utilisation des véhicules électriques est rentable en comparant la distance parcourue par le véhicule électrique et la distance minimale permettant de le rentabiliser. Dans ce cas, il est par exemple envisageable de reconfigurer les tournées de manière à ce que plus de véhicules électriques soient affectés. En effet, notre démarche a pour but l'analyse fonctionnelle. La validation s'appuie sur les indicateurs dérivés de l'aspect objectif et sur l'avis des experts. Des analyses statistiques peuvent être conduites mais elles ne rentrent pas dans le périmètre de nos travaux.

Comme exposé dans l'état de l'art, l'approche ExecuteEA s'inscrit dans l'école de pensée *Enterprise System Architecting* : par l'analyse du comportement de l'architecture, nous souhaitons non seulement vérifier l'alignement de l'IT à la stratégie de l'entreprise mais aussi donner la possibilité au métier d'évaluer sa propre stratégie en fonction de sa déclinaison au niveau de l'IT. Dans le contexte de la gestion d'une flotte de véhicules électriques, la simulation peut aboutir à la non adéquation du type de tournées avec l'impératif de rentabiliser l'investissement dans une flotte de véhicules électriques à cause de la distance

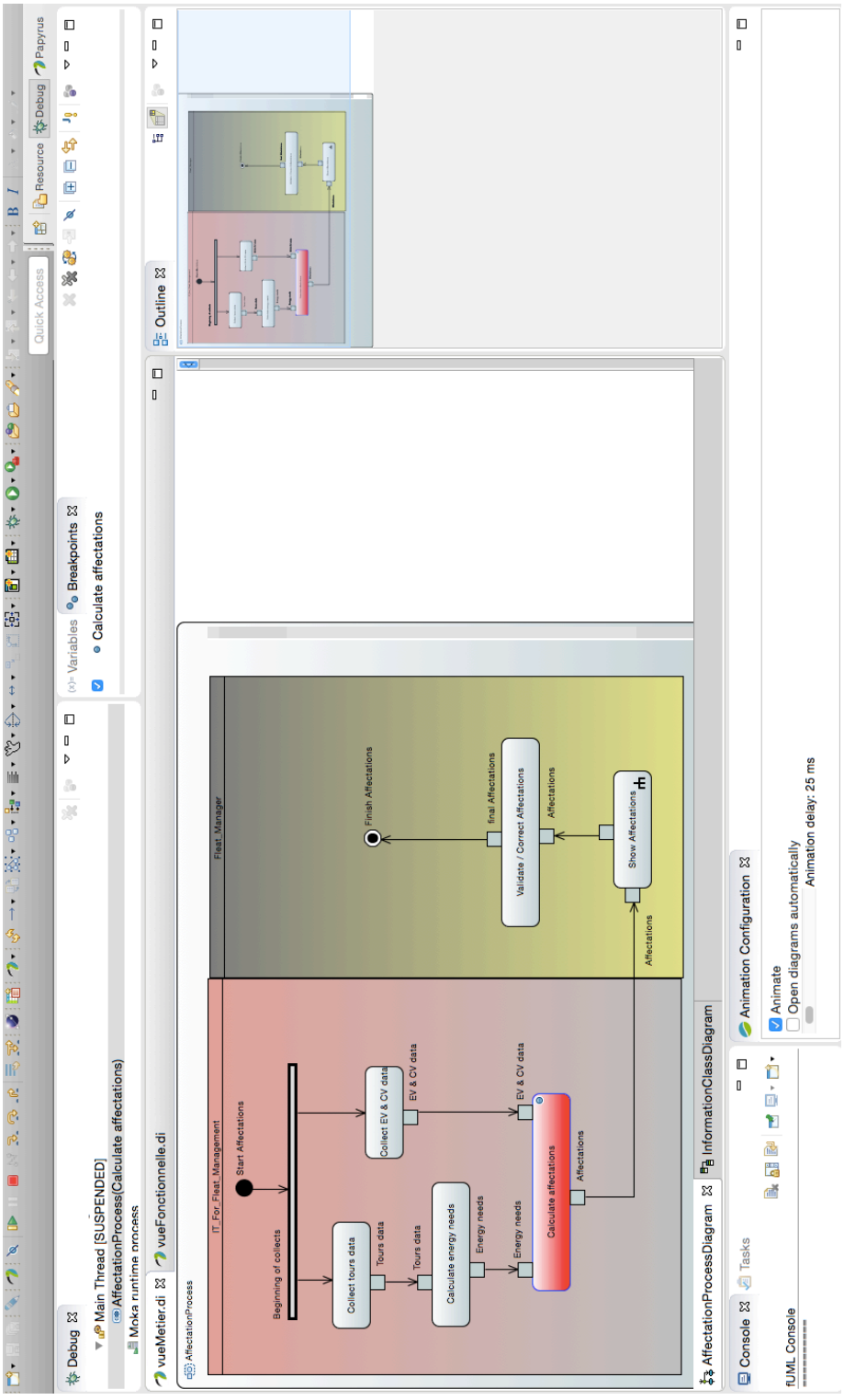

FIGURE 5.14 – Simulation de l'architecture sous Papyrus

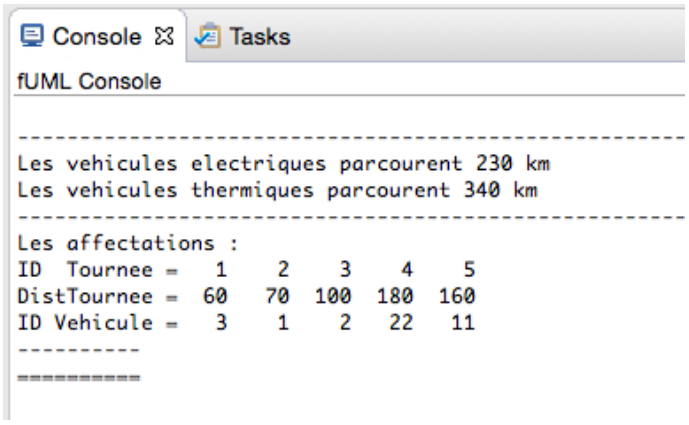

FIGURE 5.15 – Résultat retourné par la simulation du cas d'étude dans Papyrus

des tournées. En pareil cas, les architectes — métier, fonctionnel et applicatif — en étroite collaboration avec l'architecte d'entreprise peuvent envisager de faire évoluer l'application qui optimise les tournées quotidiennes créées à partir des bons de travaux pour en écourter la distance.

5.4 Discussion et perspectives pour le prototypage et la validation du *framework* ExecuteEA

Le développement du prototype et son application au cas métier de la gestion d'une flotte de véhicules électriques concrétise l'ensemble des propositions autour du *framework* ExecuteEA et du métamodèle EAT-ME et fournit un outil d'analyse de la structure et du comportement d'une architecture d'entreprise.

Bien que nous n'ayons modélisé et simulé qu'un seul processus métier, la mise en œuvre du *framework* ExecuteEA a permis de valider les apports de l'IDM en tant que cadre technologique et méthodologique aux différentes activités d'EA. La première limite de cet implémentation est donc le passage à l'échelle. C'est en effet dans le traitement d'une grande quantité d'artefacts utiles à l'EA, dont la gestion dépend souvent du savoir-faire de l'architecte d'entreprise uniquement sans assistance informatique particulière, que notre proche prouve sa valeur.

Une premier passage à l'échelle serait donc d'élargir le cas métier en incluant par exemple le processus de calcul de tournées journalières à partir de bons de travaux, le processus de recharge de la batterie et ses impacts sur le réseau électrique, le processus d'entretien des voitures, le processus de réservation de voitures par d'autres agents non impliqués dans les tournées, etc.

Lors de la simulation du processus métier, nous avons implémenté la tâche de l'affectation

de véhicules aux tournées. Cette implémentation a consisté à modéliser la fonction qui réalise cette tâche sous la forme de contraintes OCL puis à obtenir par transformation de modèle le code pour le module MiniZinc de la vue applicative. Ces développements valident le principe de simulation que nous proposons (cf. figure 4.15 page 86). Il serait donc intéressant de l'étende au reste des tâches métier du processus en utilisant cette fois des diagrammes d'activité fUML pour modéliser et simuler leur comportement.

De manière générale, bien qu'il ne s'agisse que d'un prototype, l'implémentation de la vue intégration avec EMF a permis de mettre en lumière les avantages que présente la méta-modélisation pour l'analyse structurelle des modèles d'architecture, notamment grâce à la relation de conformité qui lie un modèle à son métamodèle. La méta-modélisation avec Ecore est d'autant plus pertinente avec le recours aux contraintes OCLinEcore permettant de préciser d'avantage le métamodèle EAT-ME. Ces mêmes contraintes peuvent aussi servir à exprimer des requêtes sur le modèle. À titre d'exemple, nous sommes en train de développer des requêtes permettant de retrouver toutes les tâches (respectivement fonctions et modules) utilisant un certain concept (respectivement type et format). Une multitude de requêtes de ce type gagnent à être développées pour faciliter l'analyse structurelle de l'architecture.

Enfin, la question de l'adoption des langages de modélisation par les différentes parties-prenantes (architectes d'entreprise, métier, fonctionnel, applicatif) est cruciale. Bien que l'utilisation de UML comme langage de modélisation ne soit pas évident de prime abord par des personnes à profil non technique, les spécifications des *Use Cases* Smart Grids recourent de plus en plus à UML au sein de la CEI. Les entretiens menés avec les différents experts d'EA et du réseau électrique révèlent cependant la frilosité de certaines personnes à adopter UML pour décrire leurs spécifications dans leur intégralité. Néanmoins, ces personnes recourent de plus en plus à certains diagrammes comme les diagrammes d'activités pour décrire leurs processus. Dans ce contexte, le choix de fUML et de OCL nous semble être « raisonnable ».

Dans le chapitre suivant, nous présentons la dernière contribution de ces travaux à savoir une délimitation de l'objet d'étude. La simulation des architectures d'entreprise est une problématique à la fois large et très peu traitée dans la littérature. Nos travaux, de nature prospective, ont donc nécessité une recherche exploratoire pour appréhender notre objet d'étude, dans le contexte particulier des Smart Grids, et pour en définir les contours.

Chapitre 6

Délimitation de l'objet d'étude

Sommaire

Une démarche de recherche classique commence par la formulation d'une question de départ. La question qui a initié nos travaux est la suivante : « comment simuler, afin de les valider, les SI des Smart Grids ? ». Comme en témoigne notre état de l'art, cette question fait

l'objet de très peu de travaux (section 2.3.5.2). Une phase exploratoire préalable a donc été nécessaire pour mettre en évidence les caractéristiques de notre objet d'étude. À cet effet, une démarche de recherche inductive a été jugée pertinente.

Nous exposons cette démarche en la justifiant dans la section 6.1. Puis, nous reprenons les étapes de cette phase exploratoire par ordre chronologique : exploration de la vue métier dans la section 6.2, puis celle de la vue applicative dans la section 6.3 et enfin celle de la vue fonctionnelle dans la section 6.4. Notre plan de recherche a été construit au fur et à mesure de nos interactions avec le terrain d'étude.

6.1 Démarche engagée

Dans ces travaux de thèse, une attention particulière est apportée à la délimitation de l'objet d'étude. Cette délimitation est souvent le résultat de l'observation du terrain d'étude : l'entreprise et son SI d'une manière générale et l'entreprise et son SI dans le cas particulier des Smart Grids. L'entreprise et son SI forment un système complexe dont l'observation n'est pas triviale. La délimitation de l'objet d'étude a donc été, en soi, à l'origine d'une démarche de recherche.

Nous avons adopté pour cela une démarche de recherche inductive et ce pour plusieurs raisons. D'une part, une démarche inductive est pertinente pour formuler des hypothèses ou soulever des questions, et pour aborder un sujet qui a été peu étudié comme c'est le cas de la simulation du SI des Smart Grids. Cette démarche est donc adaptée pour :

1. délimiter l'objet de l'étude, c'est à dire identifier ce qui est dans le contexte de la simulation des SI et ce qui ne l'est pas ;
2. jeter les bases d'une étude théorique ultérieure.

D'autre part, l'induction est une démarche de recherche classique en sciences sociales. Elle correspond au raisonnement empiriste qui affirme que l'observation et l'expérience sont la source de la connaissance du monde réel et du concret [2001-112]. Nous cherchons en effet à comprendre notre objet d'étude empiriquement. Le recours à cette démarche est d'autant plus justifié par la nature socio-technique du SI. En définissant le SI, Robert Reix met en évidence sa composante sociale (voir chapitre 2).

La volonté de délimitation de notre objet d'étude est portée par la question suivante « Qu'est-ce que la simulation d'un SI d'entreprise ? ». Néanmoins, même empirique, une démarche de recherche doit nécessairement s'inscrire dans un cadre de cohérence. Nous avons donc veillé à construire un protocole d'investigation épistémologiquement valide et conforme aux critères de cohérence interne. Notre protocole d'investigation est axé sur l'observation et l'expérience. Il est constitué de trois grandes étapes :

1. observation du terrain d'étude, c'est à dire analyse des pratiques courantes des personnes impliquées par la simulation des SI des Smart Grids. Nous avons identifié deux catégories de personnes susceptibles d'être concernées : les personnes appartenant

au domaine du SI (ou de sa simulation), et les personnes susceptibles d'instrumenter la simulation des SI pour leurs travaux de recherche ou d'ingénierie (il s'agit là des utilisateurs finaux). Nous avons privilégié les ingénieurs-chercheurs d'EDF R&D car le contexte Convention Industrielle de Formation par la Recherche (CIFRE) de la thèse a facilité l'accès à ces personnes. L'observation aboutit à la formulation d'hypothèses « aprioristes ». Ce sont des hypothèses exploratoires visant à soulever des interrogations ;

2. développement d'un prototype de simulation tenant compte du résultat des observations de l'étape précédente. Le prototype n'a pas pour vocation de proposer une solution finale mais plutôt de tester rapidement les hypothèses formulées précédemment ;
3. validation ou mise à l'épreuve du prototype sur le terrain d'étude. Cette mise à l'épreuve commence par la définition d'un cas d'application pertinent permettant de vérifier les hypothèses formulées à l'étape d'observation. Elle se poursuit par la collecte et l'analyse du retour des personnes concernées. Le contexte CIFRE a là aussi facilité les échanges avec les ingénieurs-chercheurs de EDF R&D, et en particulier ceux du département MIRE. Notre intégration à l'équipe des ingénieurs-chercheurs du département MIRE, et en particulier à l'équipe du projet de simulation des Smart Grid, a contribué à la qualité des échanges avec les personnes interrogées.

Le raisonnement par induction aboutit à des propositions générales à partir de cas singuliers. Nous avons donc commencé par décomposer le terrain d'étude, c'est à dire le SI de l'entreprise. Les bases théoriques de la discipline des SI ont permis de procéder à cette décomposition afin de mettre en évidence ses singularités. Les approches par points de vue sont largement utilisées pour traiter la complexité des SI en le décomposant en plusieurs vues : la vue métier, la vue fonctionnelle, la vue applicative, la vue technique. Chaque vue correspond à la perspective d'un groupe de personnes aux profils différents mais complémentaires. Les investigations ont été menées sur les trois premières vues — métier, fonctionnelle, applicative. La vue technique n'a pas été traitée : le temps nécessaire aux expérimentations est incompatible avec les délais de cette thèse et son financement dans le cadre d'une CIFRE.

Le protocole d'investigation est alors appliqué à chacune des vues métier, fonctionnelle et applicative. L'objectif de la démarche engagée est de définir l'objet d'étude, en ayant comme question de départ « Qu'est-ce que la simulation d'un SI d'entreprise ? ». Cependant, nous avons veillé à garder une capacité d'ouverture aux idées nouvelles.

6.2 Investigations menées pour la vue métier

La première étape d'observation a débuté avec un stage de fin d'étude de six mois que nous avons effectué au sein du département MIRE. L'objectif du stage a consisté à explorer le

sujet « Simulation du SI des Smart Grids » afin de préparer un sujet de thèse. Il s'est donc accordé avec l'objectif des investigations menées pour la vue métier.

6.2.1 Observation

Pour cette première phase d'observation, des entretiens ont été menés avec des experts SI internes à l'entreprise, mais aussi externes à celle-ci lors d'un séminaire professionnel ayant pour thème la modélisation des SI [1]. Des entretiens ont aussi été menés avec des ingénieurs-chercheurs du département MIRE ayant participé à des démonstrateurs Smart Grids européens pour mettre en évidence les pratiques de spécification de la composante SI des Smart Grids.

Les hypothèses formulées à l'issue des ces observations sont les suivantes :

- la simulation de SI est une discipline peu étudiée ;
- la simulation des processus métier est pertinente pour les SI des Smart Grids dans la mesure où elle permet de valider les scénarios élaborés pour les démonstrateurs, mais aussi pour les SI tout court ;
- lors de cette simulation, il est nécessaire de maintenir une cohérence entre le processus et les données qu'il manipule.

6.2.2 Prototypage

Le prototypage a nécessité une étude des outils existants proposant de simuler des processus métier, ce qui a permis d'identifier les outils suivants : Enterprise Architect, Bonita, Amuse et Rhapsody. Il a ensuite fallu évaluer leur capacité de simulation selon une grille d'évaluation. Nous nous sommes servis, comme critère de sélection principal, de la capacité de l'outil à exécuter des diagrammes d'activité UML. En effet, c'est dans ce langage que sont représentés les processus métier dans les documents de spécification des démonstrateurs Smart Grid. Le deuxième critère d'évaluation retenu a été la possibilité d'ajout de nouvelles fonctionnalités à l'outil. À l'issue de cette étude comparative, l'outil Enterprise Architect dans sa version 9.2 a été retenu. En effet, à l'époque de l'étude, EA était le seul à pouvoir animer des diagrammes d'activité et à offrir la possibilité d'ajout de fonctionnalité par le mécanisme de plugin. C'était en outre l'outil de modélisation UML de référence de l'équipe d'ingénieurs-chercheurs au sein de laquelle nous avons effectué ce stage.

Cependant, dans sa version 9.2, l'outil n'assure pas la cohérence entre les objets métier (modélisés avec des diagrammes de classes) et le processus (modélisé avec un diagramme d'activité) au cours de la simulation. De plus, l'outil ne gère pas la persistance des résultats de la simulation. La mise en cohérence a donc nécessité le développement d'un plugin que nous avons baptisé DataSimu. DataSimu permet de (1) créer un jeu de données en entrée de la simulation à partir des concepts métier en instanciant un diagramme de classes (2) simuler

1. Model Driven Day, 21 novembre 2001, Paris Cœur Défense

le processus métier avec ce jeu de données (3) récupérer le jeu de données en sortie de la simulation et le stocker dans une base de données. Nous avons mené l'implémentation de DataSimu en binôme avec un étudiant en troisième année d'école d'ingénieur. L'interface graphique de DataSimu est illustrée par la figure 6.1.

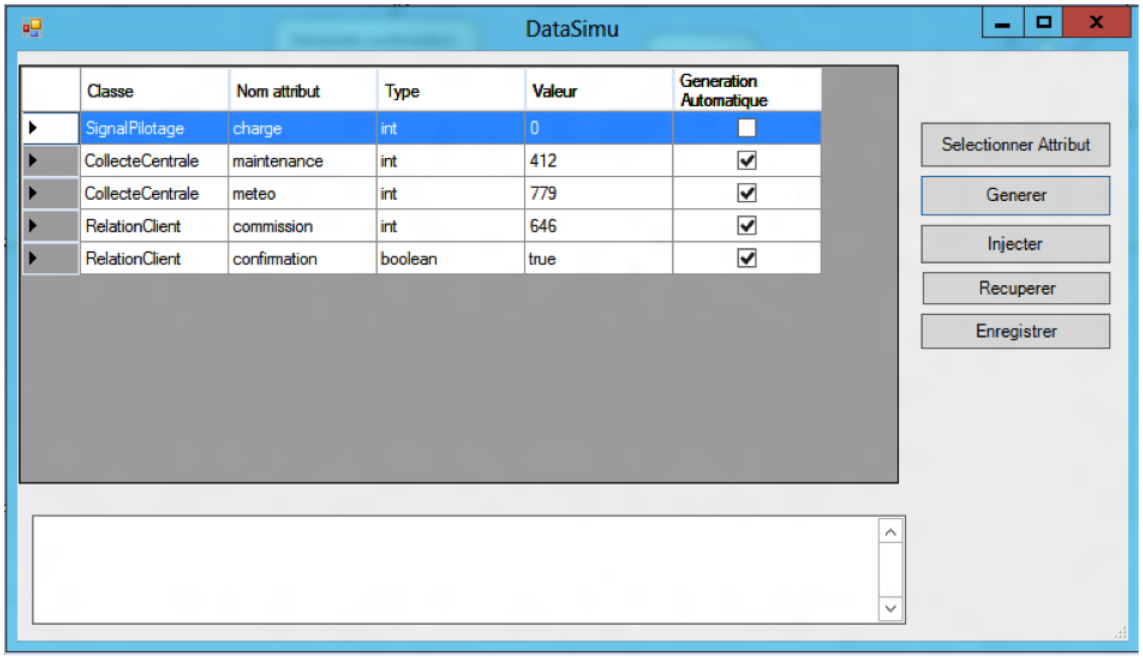

FIGURE 6.1 – Interface Graphique de DataSimu

6.2.3 Validation

Pour valider les hypothèses formulées à l'issue de l'observation, un cas d'application Smart Grid a été mis au point : le pilotage d'une charge domestique. Ce cas d'application est issu de l'étude des spécifications des démonstrateurs Smart Grid ADDRESS et PREMIO introduits dans la section 4.2.1.1. Il s'agit de piloter une batterie de stockage d'énergie installée chez un client (particulier ou industriel). En fonction de l'état du réseau, une centrale de pilotage contrôle cette batterie (stockage d'énergie pour une utilisation ultérieure), tout en tenant compte des consignes du client. Ce cas métier a été modélisé et simulé avec l'outil Enterprise Architect doté du plugin DataSimu.

Le prototype de simulation et sa mise en œuvre à travers le cas métier du pilotage d'une charge domestique ont été soumis aux experts SI du département MIRE et aux ingénieurs-chercheurs contribuant aux démonstrateurs Smart Grid PREMIO et ADDRESS. Les entretiens suivant la démonstration ont validé (1) la pertinence de la simulation dans le contexte des SI des Smart Grids (2) la séparation du processus métier et des objets métier tout en maintenant une cohérence lors de la simulation.

Ces entretiens, assortis à l'étude des outils de simulation des processus métier, ont permis de constater que la question de la simulation est peu abordée dans le contexte des SI. Ces travaux d'investigation ont de plus donné lieu à une publication [2012-98] et ont été poursuivis par les travaux présentés dans cette thèse.

6.2.4 Conclusion

Cette première application du protocole d'investigation a conforté nos hypothèses initiales à savoir que la simulation est peu abordée dans le contexte des SI mais qu'elle est pertinente pour valider/critiquer les scénarios de cas métier Smart Grid avant leur implémentation.

6.3 Investigations menées pour la vue applicative

La vue applicative tient une place de choix dans les SI des entreprises. Pour des SI fortement informatisés, il arrive même souvent que le SI soit réduit aux applications informatiques et à l'infrastructure qui les supporte. Cette constatation est encore plus avérée dans le cas des Smart Grids dont le principe est le déploiement de TIC sur le réseau électrique pour automatiser son pilotage. Le choix chronologique de poursuivre les investigations en abordant la vue métier découle de cette constatation.

6.3.1 Observation

Les investigations menées pour la vue applicative ont nécessité d'approfondir nos connaissances du fonctionnement du réseau électrique. Pour cette deuxième phase d'observation, nous avons conduit des entretiens avec deux profils de personnes : des experts et des ingénieurs-chercheurs spécialisés dans le réseau électrique de distribution appartement au département MIRE. En effet, plus que les réseaux de transport, ce sont les réseaux de distribution qui sont concernés par les Smart Grids. Les réseaux de transports français sont déjà fortement automatisés. L'objectif de ces entretiens est double : approfondir nos connaissances du réseau électrique et comprendre les pratiques des personnes interrogées en matière de simulation. Les experts sus-mentionnés sont responsables de la conception d'applications pour la conduite du réseau électrique. Le langage de conception le plus utilisé est les automates programmables. Les ingénieurs-chercheurs sont quand à eux responsables du développement des applications, le plus souvent en Matlab ou C++. Les hypothèses formulées à l'issue de cette observation sont les suivantes :

- la simulation du SI pour les Smart Grid est liée à la simulation des réseaux électriques ;
- la simulation du SI des Smart Grid nécessite de traiter la problématique de l'hétérogénéité des modèles.

6.3.2 Prototypage

Nous avons utilisé l'outil de modélisation hétérogène Ptolemy [2] pour développer un prototype de simulation illustré par la figure 6.2. L'observation a permis d'identifier les éléments à modéliser, à savoir le SI et le réseau électrique. Ici l'hétérogénéité des modèles provient de leur dépendance au temps. La modélisation du comportement du SI et celle du réseau électrique font intervenir deux domaines de calcul différents : à événements discrets pour le SI, et à flots de données périodiques pour le réseau électrique. Ainsi, dans le modèle Ptolemy illustré par la figure 6.2, le domaine de calcul adopté pour le SI est *Discrete Events* (DE), celui adopté pour le réseau de distribution électrique est *Synchronous Data Flow* (SDF). Ptolemy permet d'adapter ces deux domaines de calcul pendant la simulation d'un cas métier et d'adresser l'hétérogénéité des modèles du SI et du réseau électrique.

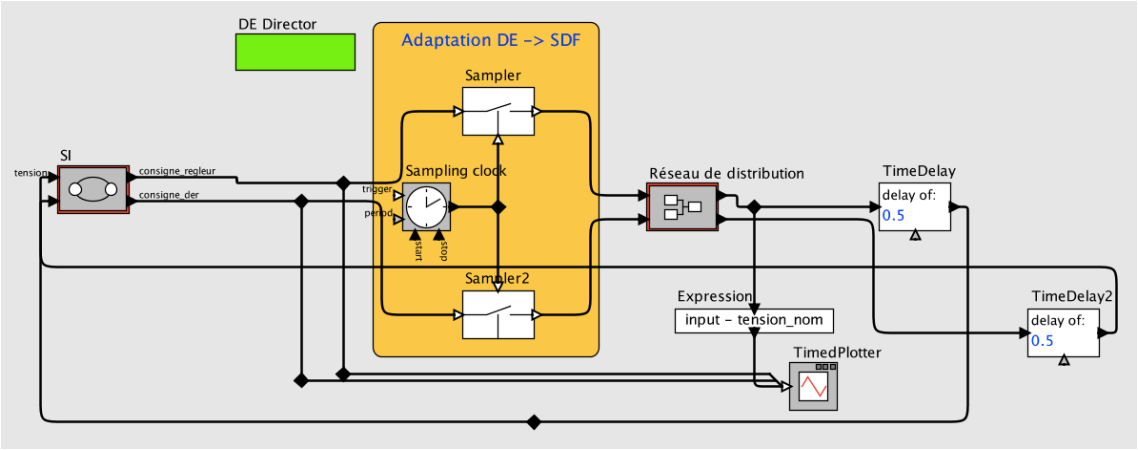

FIGURE 6.2 – Prototype Ptolemy pour une simulation hétérogène comprenant le SI (discret) et le réseau électrique (continu)

6.3.3 Validation

Le cas d'application du pilotage d'une charge domestique n'a pas été utilisé pour le prototypage de la vue applicative. Les modèles de la vue applicative nécessitent un niveau de détail plus élevé que les modèles de la vue métier. Or les spécifications des démonstrateurs ADDRESS et PREMIO n'offrent pas le niveau de détail nécessaire. Le *Use Case* normalisé par ENEL (voir section 4.2.1.2) pour la régulation de tension sur les réseaux de distribution a été adopté et affiné par les discutions avec les ingénieurs-chercheurs du département MIRE travaillant sur la même thématique.

Ils s'agit d'adapter la tension du réseau électrique en fonction de la charge et de la production

2. http ://ptolemy.eecs.berkeley.edu/

pour maintenir un niveau de tension acceptable. Deux leviers d'action sont utilisés, les régleurs en charge et les DER. Les régleurs en charges pilotés à distance agissent sur le niveau de tension au niveau des postes sources. Le pilotage des DER permet de contrôler la quantité d'énergie qu'ils injectent sur le réseau. Ce cas métier fait intervenir un SI qui calcule les consignes envoyées aux DER et au régleur en charge, et un réseau électrique qui réagit à ces consignes. Il est à noter que le SI est réduit à sa vue purement applicative en ne modélisant que l'application qui calcule les consignes.

Ce cas métier a été modélisé puis simulé avec l'outil Ptolemy de modélisation et de simulation hétérogènes. Le comportement du SI est modélisé avec une machine à états qui calcule des consignes pour le DER et le régleur en charge en fonction du niveau de tension du réseau électrique. Le DER et le régleur en charge régulent la tension du réseau électrique en appliquant ces consignes. Le réseau de distribution est composé d'un DER, d'un consommateur d'électricité et d'un régleur en charge dont les comportements font varier le niveau de tension.

La simulation de la régulation de tension avec Ptolemy a été soumise aux experts et ingénieurs chercheurs identifiés dans la phase d'observation. Bien que leurs retours aient permis de confirmer les hypothèses formulées concernant la problématique de l'hétérogénéité, cette problématique a été écartée de notre périmètre de recherche. En effet, les ingénieurs-chercheurs spécialistes du réseau électrique de distribution utilisent leur propres outils de simulation. Le prototype développé a permis de mettre en évidence la problématique de l'hétérogénéité des modèles mais pas de la résoudre. Un projet de co-simulation des domaines SI, réseau électrique et télécommunication a été lancé, suite aux résultats de différents projets de simulation dans le département MIRE, dont nos travaux de thèse pour le domaine SI.

6.3.4 Conclusion

Cette deuxième application du protocole d'investigation pour la vue applicative a permis de confirmer nos hypothèses concernant la problématique de l'hétérogénéité des modèles de simulation mais surtout de réduire notre périmètre de recherche. En effet, l'hétérogénéité est traitée par un projet du département MIRE, auquel nos travaux ont été intégrés. L'implémentation du cas métier de la régulation de tension à l'aide de Ptolemy, et le retour qu'en ont fait les personnes interrogées, nous a en outre permis d'affiner le cas métier de la régulation de tension. Sa réutilisation pour les investigations menées pour la vue fonctionnelle en a été d'autant plus facilitée.

6.4 Investigations menées pour la vue fonctionnelle

La vue fonctionnelle est une vue charnière entre le métier et les applications qui implémentent les processus métier. En effet, elle décompose chaque tâche métier en fonctions. Les

investigations menées sur la vue fonctionnelle sont essentielles dans la mesure où cette vue permet de maintenir le lien entre le métier et l'IT.

6.4.1 Observation

Pour cette dernière phase d'observation, nous avons mené des entretiens avec des personnes du domaine SI et des personnes du domaine du réseau électrique. Nos observations ont permis de constater que les ingénieurs-chercheurs du département MIRE qui conçoivent des applications pour les réseaux électriques, utilisent leurs propres systèmes de notations et ne sont que très peu familiers avec les concepts du domaine SI telle que la vue fonctionnelle. Il s'est avéré en effet, qu'une distinction entre SI de gestion et SI industriel est faite au sein du département. Les applications qui automatisent la conduite du réseau relèvent des SI industriels. Les SI de gestion, ou SI transverses, correspondent à des SI intégrant des actions humaines dans leurs processus ou gérant des activités liées aux personnes. Ainsi, les hypothèses formulées suite à ses observations sont les suivantes :

- la simulation des SI des Smart Grids est aussi pertinente pour les SI industriels que pour les SI de gestion ;
- le langage de modélisation du SI doit être compréhensible par les parties prenantes.

6.4.2 Prototypage

Les hypothèses formulées à l'issue de l'observation impliquent que le prototypage nécessite la mise au point préalable d'un cas métier. D'une part, les investigations pour la vue métier ont permis de valider la pertinence de la simulation d'un SI de gestion. En effet, le pilotage d'une batterie installée chez un client implique l'intervention systématique de ce dernier à travers son consentement/refus à stocker ou injecter l'énergie de sa batterie en fonction de la compensation tarifaire perçue. Il a été donc plus judicieux de s'orienter vers un SI industriel pour cette dernière application du protocole d'investigation. D'autre part, la deuxième hypothèse portant sur la compréhension du langage de modélisation du SI par les parties prenantes a orienté le prototypage vers la création d'un DSML. La création d'un DSML nécessite cependant de connaitre au préalable son domaine d'application, qui est ici le cas métier Smart Grid traité.

Le cas métier retenu est la régulation de tension d'un réseau électrique présentant une forte pénétration de DER. En effet, les ingénieurs-chercheurs du département MIRE considèrent qu'il relève du domaine des SI industriels. Son implémentation pour la vue applicative a permis en outre de l'affiner. Ainsi, le prototypage a consisté à développer un DSML pour la vue fonctionnelle selon une démarche IDM. Après l'analyse du processus fonctionnel de la régulation de tension, nous avons développé un métamodèle. Dans ce métamodèle, illustré par la figure 6.3, les concepts essentiels d'un processus fonctionnel de régulation de tension ont été définis en respectant les termes utilisés par les experts du réseau électrique :

– Événement (*Event*)
  Ce sont les événements qui peuvent apparaitre sur le réseau de distribution. Il s'agit de la contrainte haute (la tension sur le réseau dépasse la tension réglementaire U_{Max}), de la contrainte basse (la tension sur le réseau passe sous la tension réglementaire U_{Min}), et la contrainte à la fois haute et basse (en différents points du réseau) ;
– Action de régulation (*RegulationAction*)
  Ce sont les leviers à actionner pour adresser une contrainte : élever le niveau de tension via le régleur en charge (*IncreasePad*), baisser le niveau de tension via le régleur en charge (*DecreasePad*), ou en effaçant un DER (*DeleteDER*) ;
– Contrainte à respecter (*Constraint*)
  Le processus fonctionnel de régulation de tension est limité par des contraintes liées aux équipements du réseau. Le régleur en charge abaisse (respectivement élève) la tension sans dépasser une marge basse (*LowMargin*) (respectivement une marge haute(*HighMargin*)). Il n'est de plus pas possible de mettre le régleur en charge en butée haute ou basse (*LowerStop, UpperStop*). En effet, quand le régleur en charge abaisse ou élève la tension, il change de plot. Le nombre de plot étant limité, il est interdit d'utiliser les plots extrêmes par mesure de sécurité ;
– Structure de contrôle (*ControlStructure*)
  Pour ce prototype, nous avons uniquement implémenté le *If*.

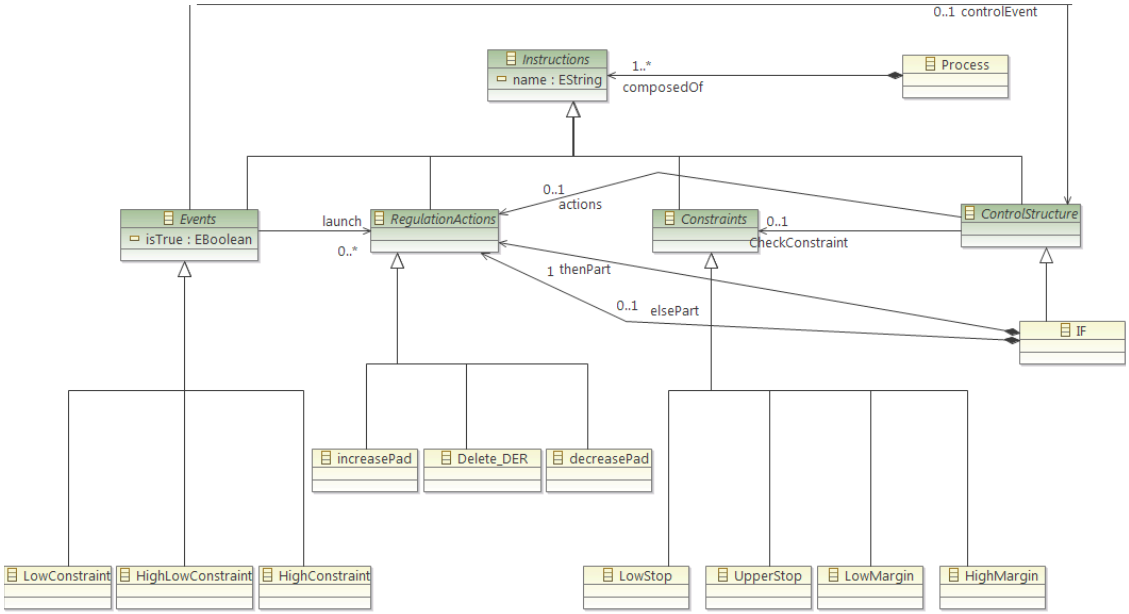

FIGURE 6.3 – Métamodèle d'un processus fonctionnel de régulation de tension sur un réseau de distribution électrique

Une syntaxe concrète et une sémantique d'exécution ont aussi été conçues pour ce DSML. Le DSML a ensuite été implémenté par un stagiaire dans l'environnement Eclipse. L'utilisation

d'Eclipse est largement répandue dans la communauté de l'IDM. La fondation Eclipse héberge le projet *Eclipse Modeling* qui propose des langages et outils dédiés au développement de DSML. La sémantique d'exécution a été implémentée avec Kermeta. Kermeta offre la possibilité de définir la sémantique d'exécution directement au niveau du métamodèle à l'aide un langage d'action.

6.4.3 Validation

Le prototype ainsi implémenté permet de créer et de simuler des processus fonctionnels pour la régulation de tension. La figure 6.4 est une capture d'écran représentant le prototype. La partie droite correspond à la palette de création de processus de régulation où se trouvent les concepts du métamodèle sous leur forme graphique. La partie gauche correspond à un exemple de processus modélisé avec cette palette.

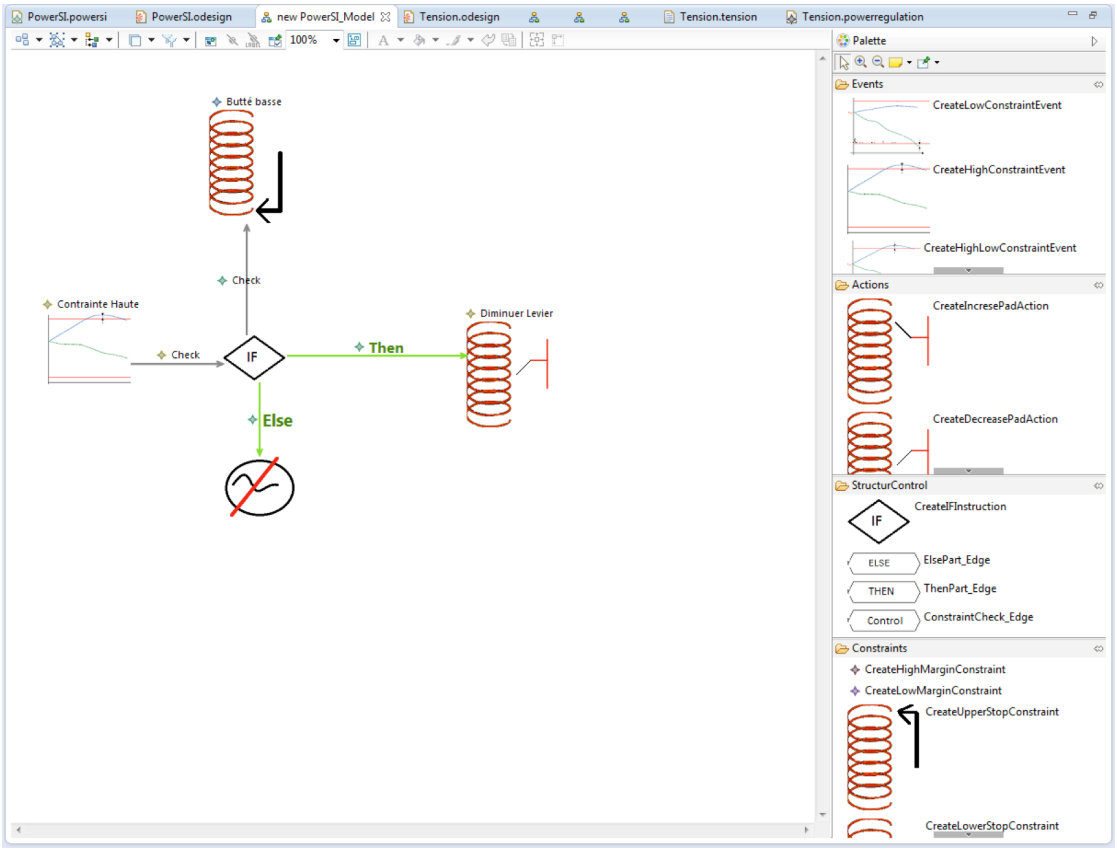

FIGURE 6.4 – Exemple de processus fonctionnel de régulation de tension modélisé avec le prototype de DSML

Le prototype a été soumis à des personnes appartenant aux profils identifiés dans l'observation, c'est-à-dire des personnes du domaine SI et des personnes du domaine du réseau

électrique. Les personnes du domaine SI sont principalement des architectes SI. Leur retour a été positif. Ils ont trouvé dans le DSML développé un moyen de modéliser des processus pour le métier et d'échanger avec les personnes du domaine électrique qui ne maitrisent pas toujours les langages traditionnellement utilisés dans le domaine SI comme UML. Le retour des personnes du domaine du réseau électrique n'a pas été aussi enthousiaste que celui des personnes du domaine SI. Il s'agit d'abord d'un problème de sémantique. En effet, il n'ont pas vu d'intérêt à modéliser des « fonctions » de conduite de réseau avec un nouveau DSML. Le terme « fonction » est associé au domaine purement électrique et non SI. Or ces fonctions sont habituellement modélisées avec les automates programmables ou encore le langage Matlab. Ces langages sont éprouvés pour le domaine du réseau électrique mais ne sont pas adaptés à la vue fonctionnelle du SI. Cette dernière ne met pas l'accent sur le détail de l'implémentation de la fonction, mais plutôt sur l'orchestration des différentes fonctions et leur structuration en blocs fonctionnels.

L'analyse du résultat des entretiens menés avec les personnes du domaine SI et les personnes du domaine des réseaux électriques a validé partiellement les hypothèses formulées à l'issue de l'observation. Ainsi, en montrant leur intérêt pour le DSML, les personnes du domaine SI ont affirmé l'intérêt de modéliser et de simuler le SI avec des langages compréhensibles par les personnes du métier, en l'occurrence celles du domaine du réseau électrique. En revanche, le retour des personnes du domaine du réseau électrique a, au regard de notre analyse, infirmé l'hypothèse selon laquelle la simulation des SI des Smart Grids est aussi pertinente pour les SI industriels que pour les SI de gestion. En effet, les SI industriels sont déjà l'objet de simulations (avec les automates programmables et Matlab par exemple).

6.4.4 Conclusion

Cette dernière application du protocole d'investigation à cette étude a permis en outre d'identifier deux types de SI pour les Smart Grid. L'étude a confirmé la pertinence de la simulation de SI pour la vue fonctionnelle, mais seulement pour un SI de gestion. Les SI purement informatiques, autrement dit ceux qui ne font pas intervenir de tâche humaine et qui pilotent directement les équipements électriques, ne sont pas concernés par notre recherche. En effet, il existe déjà des langages dédiés à leur modélisation et simulation (automates programmables et Matlab). Le terme « SI industriel » employé pour qualifier ces systèmes a été, à notre sens, trompeur dans le sens où il s'agit plutôt d'une restriction ou d'une spécialisation du terme SI. En effet, selon la définition de Reix (cf. section 2.1.1.1), toute ressource intervenant sur le cycle de vie d'une information au sein d'une organisation fait partie du SI, y compris le personnel.

6.5 Conclusion

Dans ce chapitre, nous avons abordé la question de la délimitation de l'objet d'étude. Cette délimitation a été, en soi, à l'origine d'une démarche de recherche à part en entière.

Nous avons adopté une démarche inductive en définissant un protocole d'investigation que nous avons appliqué à chacune des vues métier, fonctionnelle et applicative. Nous avons ainsi pu jeter les bases des travaux précédemment présentés dans ce mémoire et mettre en évidence certaines caractéristiques de notre objet d'étude à savoir : (1) la simulation du SI est pertinente pour valider/critiquer les scénarios des cas métier développés pour les Smart Grids avant leur implémentation (2) nos travaux seront plus pertinents pour les SI de gestion que pour les SI industriels pour lesquels il existe déjà des méthodes et langages de simulation dédiés (3) la simulation est d'autant plus cruciale pour le SI, en le traitant dans sa globalité et en explicitant les interactions entre ses macro-composants.

Troisième partie

Conclusion et perspectives

Chapitre 7

Bilan et perspectives

7.1 Bilan

Les entreprises d'aujourd'hui évoluent dans un contexte économique et technologique en constante et rapide mutation. Les Smart Grids en sont la parfaite illustration. Ces derniers sont en effet à l'origine d'un changement de paradigme sans précédent pour les énergéticiens et impliquent de ce fait l'adoption de nouveaux processus métier, de nouvelles technologies pour le réseau électrique, de nouveaux usages et services pour les consommateurs, etc.

Nous sommes convaincus que l'EA peut apporter, dans ce contexte, une aide précieuse pour anticiper ce changement de paradigme en dotant l'ensemble des parties prenantes (décideurs et experts métier compris) d'une vision holistique afin d'aligner les préoccupations des uns et des autres, qu'elles portent sur des aspects stratégiques ou foncièrement techniques. Dès lors, notre vision de l'EA ne se réduit pas au SI, et encore moins à l'IT de l'entreprise. L'EA reste certes un moyen indispensable à l'exécution de la stratégie d'une entreprise, mais nous croyons qu'elle peut aussi avantageusement orienter et éclairer cette stratégie.

Dans cette optique, l'architecture d'entreprise doit satisfaire aux impératifs d'évolutivité et de réactivité face aux mutations permanentes de l'environnement de l'entreprise. Cette ambition ne peut se concrétiser sans, d'une part, la garantie d'une grande cohérence entre les vues d'architecture et, d'autre part, la garantie d'une grande flexibilité, voire interchangeabilité, de ses composants. Notre approche se fait fort de répondre à ces besoins de cohérence et de flexibilité à travers l'intégration et la simulation d'une architecture d'entreprise.

En recourant à l'IDM comme cadre méthodologique et technologique, nous avons proposé à cet effet des modèles, outils et méthodes permettant 1) d'unifier les artefacts issus des différentes activités d'EA (documentation, analyse et conception) et 2) d'en faciliter la validation à travers la simulation.

En effet, ces activités sont aujourd'hui fastidieuses, sources d'erreurs, et débouchent sur

des artefacts hétérogènes prenant le plus souvent la forme de documents descriptifs. Ces modèles informels et purement contemplatifs ne permettent pas d'analyser efficacement la structure et le comportement de l'architecture d'entreprise obtenue.

Dans ce contexte, nous avons proposé un *framework* d'EA dirigé par les modèles exécutables : ExecuteEA. Ce framework fournit à la fois une approche conceptuelle pour l'analyse de la structure et du comportement d'une architecture d'entreprise, un cadre structurant dont nous avons formalisé le métamodèle (EAT-ME) ainsi qu'un ensemble de langages et de techniques issus de l'IDM et adaptés à la modélisation et à l'analyse d'une architecture d'entreprise.

L'analyse de la structure vise à s'assurer de la cohérence globale d'une architecture d'entreprise. Nous avons proposé pour cela de rajouter aux points de vue traditionnellement utilisés en EA (tels que les points de vue métier, fonctionnel et applicatif) un point de vue intégration. Ce point de vue permet de modéliser les liens de cohérence inter-vues (entre les autres vues) et les liens de cohérence intra-vue (entre les entités d'une même vue).

De plus, le développement d'un prototype et son application au cas métier de la gestion d'une flotte de véhicules électriques concrétise l'ensemble des propositions autour du *framework* ExecuteEA et du métamodèle EAT-ME et fournit un outil d'analyse de la structure et du comportement d'une architecture d'entreprise.

Enfin, la dernière contribution de ces travaux est relative à la délimitation de l'objet d'étude. Cette délimitation a été, en soi, à l'origine d'une démarche de recherche à part en entière. Nous avons ainsi pu jeter les bases des travaux ayant abouti à la défintion du *framework* ExecuteEA.

7.2 Perspectives

Les travaux de cette thèse, faisant partie du projet POMME d'EDF R&D, sont de nature prospective et visent à fournir des orientations claires sur la manière dont l'IDM, souvent réservée au génie logiciel, peut bénéficier à l'EA. Dès lors, de nombreuses perspectives restent envisageables.

Ces perspectives sont de trois types. Les unes concernent l'amélioration du *framework* ExecuteEA et et de son métamodèle en apportant des compléments à ce qui a déjà été réalisé, comme l'intégration du point de vue technique. Les autres correspondent à l'investigation pour l'EA de nouvelles méthodes et technologies IDM comme le *Model Typing* par exemple. D'autres, enfin, correspondent à des centres d'intérêt nouveaux ayant pour cadre l'EA, comme la modélisation du plan de migration d'une architecture d'entreprise actuelle à une architecture d'entreprise cible.

7.2.1 Model Typing

Une démarche IDM implique une multitude de modèles produits et manipulés par différentes transformations de modèles. Le maintien de la cohérence entre modèles et transformation de modèle fait l'objet d'une recherche active dans une discipline comme l'IDM. [2007-113] propose de typer les modèles en entrée des transformations en utilisant le Model Typing. Celui-ci permet de contrôler les modèles manipulés par les transformations. Autrement dit, une transformation ne prend en entrée qu'un certain type de modèle. Ainsi, tous les modèles conformes à ce type peuvent être manipulés par la transformation en question. Ceci a, de plus, l'avantage d'augmenter la réutilisabilité des transformations écrites en mettant en évidence les caractéristiques communes des modèles. Cette technique de typage est en outre outillée et intégrée dans le langage de méta-modélisation Kermeta [1] qui est basé sur EMOF [2] dans un environnement Eclipse. Il nous paraît donc intéressant de mettre en œuvre le Model Typing pour la vue intégration.

7.2.2 Vue technique

Dans ces travaux nous avons traité les vue métier, fonctionnelle et applicative. Nous souhaiterions également intégrer la vue technique au *framework* ExecuteEA afin d'obtenir une analyse de la totalité de l'architecture d'entreprise. Cette vue correspond à l'infrastructure technique du SI (matériel informatique et réseaux de télécommunication) et permettra donc de faire le lien direct avec les équipements du réseau électrique. En effet, les travaux présentés dans ce mémoire se poursuivent avec un nouveau sujet de thèse au département MIRE qui se porte entièrement sur la modélisation et la simulation de la vue technique, en continuité avec le *framework* ExectueEA.

7.2.3 Migration de l'architecture actuelle vers une architecture cible

Le *framework* ExecuteEA peut être utilisé pour créer des modèles d'architecture d'entreprise exécutables afin d'analyser les composants d'une entreprise dans son état courant ou cible de manière séparée. Cependant, pour retracer le cycle de vie d'une architecture d'entreprise et la localisation d'une architecture d'entreprise à travers ce cycle de vie, nous avons besoin d'un *framework* retraçant son évolution. Des recherches en ce sens sont en cours mais uniquement au niveau du SI d'une entreprise [2014-53]. Dès lors, il nous semble intéressant d'enrichir le *framework* ExecuteEA pour tenir compte de l'évolution d'une architecture d'entreprise à travers son cycle de vie.

Plus globalement, le recours systématique aux modèles exécutables à tous les niveaux de l'architecture reflète un parti pris fort de notre approche : celui de l'adéquation de ces langages aux profils des différentes parties prenantes. Même si l'usage du langage UML est

1. www.kermeta.org
2. Ecore Meta Object Facility

de plus en plus légitimé pour des vues non techniques telles que la vue métier (de nombreux démonstrateurs de Smart Grids font par exemple appel aux diagrammes d'activité et aux diagrammes de classes pendant la phase de spécification), il n'est cependant pas largement adopté par les experts métier et les architectes d'entreprise qui se tournent le plus souvent vers de la documentation. Nous réfléchissons, pour cette raison, à d'autres langages de modélisation exécutables plus spécifiques à ces profils d'utilisateurs et à leur cœur de métier en envisageant la création de DSML pour la vue métier et pour la vue intégration. Les DSML sont de plus promus par l'IDM.

Enfin, nous continuons à nous consacrer à (1) l'intégration du *framework* ExecuteEA à la plate-forme de co-simulation des domaines SI, électrotechnique et télécommunication développée dans le cadre du projet POMME, (2) l'intégration de l'approche conceptuelle proposée dans la démarche de spécification de Use Case pour les Smart Grids normalisée par la CEI.

Bibliographie

[2006-1] Jean-Marie FAVRE, Jacky ESTUBLIER, and Mireille BLAY-FORNARINO. L'ingénierie dirigée par les modèles. Au-delà du MDA (Traité IC2, série Informatique et Systèmes d'Information), 2006. xi, 6, 37, 38, 39

[2009-2] TOGAF Version 9. The Open Group Architecture Framework (TOGAF). *The Open Group*, 1, 2009, 2009. `http://www.opengroup.org/togaf`. xi, 23

[2013-3] Marc Lankhorst. {Enterprise Architecture at Work : Modelling, Communication and Analysis (The Enterprise Engineering Series)}. 2013. Springer, 2013. xi, 20, 22, 26, 27, 28, 30, 31, 33

[2009-4] Sabine Buckl, Florian Matthes, and Christian M Schweda. Classifying enterprise architecture analysis approaches. In *Enterprise Interoperability*, pages 66–79. Springer, 2009. xi, 27, 28, 29, 30, 33

[2014-5] Therese Clark, Vaishali Kulkarni, Balbir Barn, Robert France, Ulrich Frank, and Dan Turk. Towards the Model Driven Organization. pages 4817–4826. IEEE, 2014. xi, 26, 48, 49

[2012-6] James Lapalme. Three schools of thought on enterprise architecture. *IT professional*, (6) :37–43, 2012. IEEE, 2012. xiii, 17, 18, 19, 58

[1987-7] John Zachman et al. A framework for information systems architecture. *IBM systems journal*, 26(3) :276–292, 1987. IBM, 1987. xiii, 9, 17, 22, 56

[2003-8] David S Frankel, Paul Harmon, Jishnu Mukerji, James Odell, Martin Owen, Pete Rivitt, Mike Rosen, and Richard Mark Soley. The Zachman framework and the OMG's model driven architecture. *Business Process Trends*, 14, 2003, 2003. xiii, 48

[2013-9] Nicolas Le Dévédec and Fany Guis. L'humain augmenté, un enjeu social. *SociologieS*, 2013. Association internationales des sociologues de langue française (AISLF), 2013. 1

[10] European technology platform for the electricity networks of the future. Définition des Smart grids. `http://www.smartgrids.eu`. 2

[2014-11] L'énergie en question EDF. La panne d'électricité à Détroit, symbole de la vétusté du réseau électrique américain. 2014, 2014. `https://www.lenergieenquestions.fr/la-panne-delectricite-`

a-detroit-symbole-de-la-vetuste-du-reseau-electrique-americain/. 2

[2005-12] G Andersson, P Donalek, R Farmer, N Hatziargyriou, I Kamwa, P Kundur, N Martins, J Paserba, P Pourbeik, J Sanchez-Gasca, et al. Causes of the 2003 major grid blackouts in North America and Europe, and recommended means to improve system dynamic performance. *Power Systems, IEEE Transactions on*, 20(4) :1922–1928, 2005. IEEE, 2005. 2

[2014-13] JORDAN WIRFS-BROCK Inside Energy. Power Outages On The Rise Across The U.S. 2014, 2014. http://insideenergy.org/2014/08/18/power-outages-on-the-rise-across-the-u-s/. 2

[14] US department of enrergy. Introduction to Smart grids. energy.gov/oe/downloads/smart-grid-introduction-0. 2

[15] Smart Grids CRE. Définition des Smart grids. http://www.smartgrids-cre.fr/index.php?p=definition-smart-grids. 2, 5

[2014-16] Peter Palensky, Edmund Widl, and Atiyah Elsheikh. Simulating cyber-physical energy systems : challenges, tools and methods. *Systems, Man, and Cybernetics : Systems, IEEE Transactions on*, 44(3) :318–326, 2014. IEEE, 2014. 8

[2012-17] Jeremy Rifkin. *La troisième révolution industrielle : comment le pouvoir latéral va transformer l'énergie, l'économie et le monde.* Éditions Les liens qui libèrent, 2012. 8

[1997-18] John A Zachman. Enterprise architecture : The issue of the century. *Database Programming and Design*, 10(3) :44–53, 1997, 1997. 8, 9, 16, 17, 20, 31, 63

[2015-19] Rachida Seghiri, Frédéric Boulanger, Claire Lecocq, and Vincent Godefroy. Simulation des Systèmes d'Information des Smart Grids. 2015, 2015. 9, 58, 73

[2006-20] Jeanne W Ross, Peter Weill, and David Robertson. *Enterprise architecture as strategy : Creating a foundation for business execution.* Harvard Business Press, 2006. 9, 17

[2010-21] Tomaž Čater and Danijel Pučko. Factors of effective strategy implementation : Empirical evidence from Slovenian business practice. *Journal for east european Management Studies*, pages 207–236, 2010. JSTOR, 2010. 9

[2008-22] Leon Kappelman, Tom McGinnis, Alex Pettite, and Anna Sidorova. Enterprise architecture : Charting the territory for academic research. *AMCIS 2008 Proceedings*, page 162, 2008, 2008. 9, 17

[2010-23] Antonello Monti and Ferdinanda Ponci. Power grids of the future : Why smart means complex. pages 7–11. IEEE, 2010, ISBN : 1424459826. 10

[2004-24] Andrei Borshchev and Alexei Filippov. From system dynamics and discrete event to practical agent based modeling : reasons, techniques, tools. In *Proceedings of the 22nd international conference of the system dynamics society*, volume 22, 2004. 10

[2008-25] Sabine Buckl, Florian Matthes, Wolfgang Renz, Christian M Schweda, and Jan Sudeikat. Towards Simulation-supported Enterprise Architecture Management. In *MobIS*, pages 131–145. Citeseer, 2008. 10, 32

[2013-26] Vinay Kulkarni, Suman Roychoudhury, Sagar Sunkle, Tony Clark, and Balbir Barn. Modelling and Enterprises-The Past, the Present and the Future. In *MODELSWARD*, pages 95–100, 2013. 10, 27, 78

[1995-27] Robert Reix, Bernard Fallery, Michel Kalika, and Frantz Rowe. *Systèmes d'information et management des organisations*. Vuibert, 1995. 16

[2012-28] Scott A Bernard. *An introduction to enterprise architecture*. AuthorHouse, 2012. 17, 18

[2006-29] Robert Winter and Ronny Fischer. Essential layers, artifacts, and dependencies of enterprise architecture. In *Enterprise Distributed Object Computing Conference Workshops, 2006. EDOCW'06. 10th IEEE International*, pages 30–30. IEEE, 2006. 17, 25

[1993-30] Steven H Spewak and Steven C Hill. *Enterprise architecture planning : developing a blueprint for data, applications and technology*. QED Information Sciences, Inc., 1993. 17

[2005-31] Parthasarathy Ranganathan and Norman Jouppi. Enterprise IT trends and implications for architecture research. In *High-Performance Computer Architecture, 2005. HPCA-11. 11th International Symposium on*, pages 253–256. IEEE, 2005. 17, 18

[2012-32] Jan Mentz, Paula Kotzé, and Alta van der Merwe. A comparison of practitioner and researcher definitions of enterprise architecture using an interpretation method. *Advances in Enterprise Information Systems II*, pages 11–26, 2012. CRC Press, 2012. 18

[33] Jeanne Ross on Enterprise Architecture. Poscast from the MIT CISR. `https://www.youtube.com/watch?v=feI6_-v10Dk`. 20

[2007-34] Hanifa Shah and Mohamed El Kourdi. Frameworks for enterprise architecture. *It Professional*, 9(5) :36–41, 2007. IEEE, 2007. 20, 57, 60

[2004-35] Maarten WA Steen, David H Akehurst, HWLT Doest, and Marc M Lankhorst. Supporting viewpoint-oriented enterprise architecture. In *Enterprise Distributed Object Computing Conference, 2004. EDOC 2004. Proceedings. Eighth IEEE International*, pages 201–211. IEEE, 2004. 20, 21

[1999-36] Frank J Armour, Stephen H Kaisler, and Simon Y Liu. A big-picture look at enterprise architectures. *IT professional*, 1(1) :35–42, 1999, ISSN : 1520-9202, 1999. 21

[1979-37] Geoff P Mullery. CORE-a method for controlled requirement specification. In *Proceedings of the 4th international conference on Software engineering*, pages 126–135. IEEE Press, 1979. 21

[1992-38] Anthony Finkelstein, Jeff Kramer, Bashar Nuseibeh, Ludwik Finkelstein, and Michael Goedicke. Viewpoints : A framework for integrating multiple perspectives in system development. *International Journal of Software Engineering and Knowledge Engineering*, 2(01) :31–57, 1992. World Scientific, 1992. 21

[1996-39] Gerald Kotonya and Ian Sommerville. Requirements engineering with viewpoints. *Software Engineering Journal*, 11(1) :5–18, 1996. IET, 1996. 21

[1994-40] Bashar Ahmad Nuseibeh. *A multi-perspective framework for method integration.* PhD thesis, Imperial College, 1994. 21

[1993-41] Scott Douglas Meyers. *Representing software systems in multiple-view development environments.* PhD thesis, Brown University, 1993. 21

[2000-42] Rich Hilliard. Ieee-std-1471-2000 recommended practice for architectural description of software-intensive systems. *IEEE, http ://standards. ieee. org*, 12 :16–20, 2000, 2000. 21

[1995-43] Kerry Raymond. Reference model of open distributed processing (RM-ODP) : Introduction. In *Open distributed processing*, pages 3–14. Springer, 1995. 21

[2003-44] Anneke G Kleppe, Jos B Warmer, and Wim Bast. *MDA explained : the model driven architecture : practice and promise.* Addison-Wesley Professional, 2003. 21, 39, 40

[2008-45] Robert Winter and Joachim Schelp. Enterprise architecture governance : the need for a business-to-IT approach. In *Proceedings of the 2008 ACM symposium on Applied computing*, pages 548–552. ACM, 2008. 21, 22

[1992-46] David Nadler, Marc S Gerstein, and Robert B Shaw. *Organizational architecture : Designs for changing organizations*, volume 192. Jossey-Bass Inc Pub, 1992. 26

[2013-47] Vinay Kulkarni, Suman Roychoudhury, Sagar Sunkle, Tony Clark, and Balbir Barn. Modelling and Enterprises-The Past, the Present and the Future. pages 95–100, 2013. 26, 49

[2001-48] Jean Bézivin and Olivier Gerbé. Towards a precise definition of the OMG/MDA framework. pages 273–280. IEEE, 2001, ISBN : 076951426X. 26

[2013-49] Sagar Sunkle, Vinay Kulkarni, and Suman Roychoudhury. Analyzing enterprise models using enterprise architecture-based ontology. In *Model-Driven Engineering Languages and Systems*, pages 622–638. Springer, 2013, ISBN : 3642415326. 26, 31

[2011-50] Tony Clark, Balbir S Barn, and Samia Oussena. Leap : a precise lightweight framework for enterprise architecture. In *Proceedings of the 4th India Software Engineering Conference*, pages 85–94. ACM, 2011. 27, 49

[1974-51] Francisco G Varela, Humberto R Maturana, and Ricardo Uribe. Autopoiesis : the organization of living systems, its characterization and a model. *Biosystems*, 5(4) :187–196, 1974. Elsevier, 1974. 30

[2014-52] Roland Smook. Executable enterprise architecture models : enabling business analytics using the Monte Carlo method. 2014. University of Twente, 2014. 30

[2014-53] Alexandre Métrailler and Thibault Estier. EVOLIS Framework : A Method to Study Information Systems Evolution Records. pages 3798–3807. IEEE, 2014. 30, 127

[2005-54] Stephen H Kaisler, Frank Armour, and Michael Valivullah. Enterprise architecting : Critical problems. pages 224b–224b. IEEE, 2005, ISBN : 0769522688. 31, 58, 77

[2013-55] Balbir S Barn, Tony Clark, and Martin Loomes. Enterprise architecture coherence and the model driven enterprise : is simulation the answer or are we flying kites ? In *Proceedings of the 6th India Software Engineering Conference*, pages 97–102. ACM, 2013. 31, 56, 58, 79, 85

[2013-56] Hugo Brunelière, Jordi Cabot, Stéphane Drapeau, Flavien Somda, William Piers, Juan David Villa Calle, and Jean-Christophe Lafaurie. Un support IDM pour l'architecture d'entreprise dans un contexte industriel : l'exemple du framework TEAP. *Génie logiciel*, (107) :33–38, 2013, 2013. 32, 48

[1975-57] Robert E Shannon. Systems simulation. 1975. Prentice-Hall, 1975. 32, 85

[2015-58] Laura Manzur, Jorge Mario Ulloa, Mario Sánchez, and Jorge Villalobos. xArchiMate : Enterprise Architecture simulation, experimentation and analysis. *Simulation*, 91(3) :276–301, 2015. SAGE Publications, 2015. 33, 34, 50, 85

[2013-59] Kenneth C Hoffman, Christopher G Glazner, William J Bunting, Leonard A Wojcik, and Anne Cady. *Enterprise Dynamics Sourcebook*. CRC Press, 2013. 33

[2011-60] Christopher G Glazner. Enterprise transformation using a simulation of enterprise architecture. *Journal of Enterprise Transformation*, 1(3) :231–260, 2011. Taylor & Francis, 2011. 33, 85

[2011-61] Marie Ludwig, Nicolas Farcet, Jean-Philippe Babau, and Joël Champeau. Organizational configurations in executable Enterprise Architecture models. In *poster session of Complex Systems Design and Management 2011*, 2011. 33, 34

[2004-62] Jack Greenfield, Keith Short, Steve Cook, Stuart Kent, and John Crupi. Software factories : assembling applications with patterns, models, frameworks, and tools. 2004. Wiley Pub., 2004. 36

[2001-63] Jean Bézivin and Olivier Gerbé. Towards a precise definition of the OMG/MDA framework. In *Automated Software Engineering, 2001.(ASE 2001). Proceedings. 16th Annual International Conference on*, pages 273–280. IEEE, 2001. 36, 37

[2002-64] Stuart Kent. Model driven engineering. In *Integrated formal methods*, pages 286–298. Springer, 2002. 36

[2002-65] Juan De Lara and Hans Vangheluwe. Using Meta-Modelling and Graph Grammars to Process GPSS Models. In *ESM*, pages 100–107, 2002. 36

[2004-66] Jean Bézivin. In search of a basic principle for model driven engineering. *Novatica Journal, Special Issue*, 5(2) :21–24, 2004. Citeseer, 2004. 36, 37, 38

[2004-67] Grady Booch, Alan W Brown, Sridhar Iyengar, James Rumbaugh, and Bran Selic. An MDA manifesto. *Business Process Trends/MDA Journal*, 2004, 2004. 36

[2004-68] Jean Bézivin, Mireille Blay, Mokrane Bouzhegoub, Jacky Estublier, Jean-Marie Favre, Sébastien Gérard, and Jean Marc Jézéquel. Rapport de Synthèse de l ?AS CNRS sur le MDA (Model Driven Architecture). *CNRS, novembre*, 2004, 2004. 36, 38

[2005-69] Jean Bézivin. On the unification power of models. *Software & Systems Modeling*, 4(2) :171–188, 2005. Springer, 2005. 36, 44

[1967-70] Marvin L Minsky. *Computation : finite and infinite machines*. Prentice-Hall, Inc., 1967. 37

[2003-71] Ed Seidewitz. What models mean. *IEEE software*, (5) :26–32, 2003. IEEE, 2003. 37

[2003-72] Colin Atkinson and Thomas Kühne. Model-driven development : a metamodeling foundation. *Software, IEEE*, 20(5) :36–41, 2003. IEEE, 2003. 37

[2004-73] Jean-Marie Favre. Towards a basic theory to model model driven engineering. In *3rd Workshop in Software Model Engineering, WiSME*, pages 262–271. Citeseer, 2004. 38

[2011-74] QVT OMG. Meta Object Facility (MOF) 2.0 Query/View/Transformation, v1. 1, 2011. 39, 45

[2006-75] Tom Mens and Pieter Van Gorp. A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, 152 :125–142, 2006. Elsevier, 2006. 39

[2005-76] Laurence Tratt. Model transformations and tool integration. *Software & Systems Modeling*, 4(2) :112–122, 2005. Springer, 2005. 40, 41

[2000-77] Krzysztof Czarnecki and Ulrich W Eisenecker. Intentional programming. *Generative Programming. Methods, Tools, and Applications*, 2000, 2000. 41

[2006-78] Jean Bézivin, Salim Bouzitouna, Marcos Didonet Del Fabro, Marie-Pierre Gervais, Frédéric Jouault, Dimitrios Kolovos, Ivan Kurtev, and Richard F Paige. A canonical scheme for model composition. In *Model Driven Architecture–Foundations and Applications*, pages 346–360. Springer, 2006. 42

[2008-79] Franck Fleurey, Benoit Baudry, Robert France, and Sudipto Ghosh. A generic approach for automatic model composition. In *Models in Software Engineering*, pages 7–15. Springer, 2008. 42

[2007-80] Marcos Didonet Del Fabro and Patrick Valduriez. Semi-automatic model integration using matching transformations and weaving models. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 963–970. ACM, 2007. 42, 43

[2011-81] Eugene Syriani. *A multi-paradigm foundation for model transformation language engineering*. PhD thesis, McGill University, 2011. 42, 43

[2006-82] Frédéric Jouault and Ivan Kurtev. Transforming models with ATL. In *satellite events at the MoDELS 2005 Conference*, pages 128–138. Springer, 2006. 43, 45

[2010-83] Matthias Biehl, Chen DeJiu, and Martin Törngren. Integrating safety analysis into the model-based development toolchain of automotive embedded systems. In *ACM Sigplan Notices*, volume 45, pages 125–132. ACM, 2010. 43

[1985-84] Alfred V Aho, Ravi Sethi, and Jeffry D Ullman. Compilers Principles, Techniques, and Tools Addison-Wesley, 1986. *QA76*, 76 :C65A37, 1985, 1985. 43

[2006-85] Krzysztof Czarnecki and Simon Helsen. Feature-based survey of model transformation approaches. *IBM Systems Journal*, 45(3) :621–645, 2006. IBM, 2006. 43

[2011-86] Xavier Blanc and Olivier Salvatori. *MDA en action : Ingénierie logicielle guidée par les modèles.* Editions Eyrolles, 2011. 43

[2006-87] Jean Bézivin, Fabian Büttner, Martin Gogolla, Frédéric Jouault, Ivan Kurtev, and Arne Lindow. Model transformations ? transformation models ! In *Model driven engineering languages and systems*, pages 440–453. Springer, 2006. 44

[2008-88] Ivan Kurtev. State of the art of QVT : A model transformation language standard. In *Applications of graph transformations with industrial relevance*, pages 377–393. Springer, 2008. 45

[2007-89] Robert France and Bernhard Rumpe. Model-driven development of complex software : A research roadmap. In *2007 Future of Software Engineering*, pages 37–54. IEEE Computer Society, 2007. 48, 79

[2013-90] Hugo Bruneliere, Jordi Cabot, Stéphane Drapeau, Flavien Somda, William Piers, Juan David Villa Calle, and Jean-Christophe Lafaurie. MDE Support for Enterprise Architecture in an Industrial Context : the TEAP Framework Experience. In *TowArds the Model DrIveN Organization (AMINO 2013) workshop-a MODELS 2013 Satellite Event*, 2013. 49

[2012-91] Jean-Marc Jézéquel, Benoit Combemale, and Didier Vojtisek. *Ingénierie Dirigée par les Modèles : des concepts à la pratique...* Ellipses, 2012. 50, 63, 64

[2007-92] Anneke G Kleppe. A language description is more than a metamodel. 2007. megaplanet. org, 2007. 50

[2006-93] Henry Chesbrough and Jim Spohrer. A research manifesto for services science. *Communications of the ACM*, 49(7) :35–40, 2006. ACM, 2006. 50, 86

[2007-94] Nicholas Nethercote, Peter J Stuckey, Ralph Becket, Sebastian Brand, Gregory J Duck, and Guido Tack. Minizinc : Towards a standard CP modelling language. In *Principles and Practice of Constraint Programming–CP 2007*, pages 529–543. Springer, 2007. 51

[2007-95] Stephan Kurpjuweit and Robert Winter. Viewpoint-based Meta Model Engineering. In *EMISA*, volume 143, page 2007, 2007. 56, 78

[2013-96] Sascha Roth, Matheus Hauder, Matthias Farwick, Ruth Breu, and Florian Matthes. Enterprise Architecture Documentation : Current Practices and Future Directions. In *Wirtschaftsinformatik*, page 58, 2013. 57

[2012-97] Rachida Seghiri, Frédéric Boulanger, Claire Lecocq, and Vincent Godefroy. Simulation orientée utilisateur des Systèmes d'Information des Smart Grids. In *CIEL'14 : Conférence en IngénieriE du Logiciel*, pages 86–87, 2012. 58

[2012-98] Rachida Seghiri, Anne Picault, Claire Lecocq, and Bruno Traverson. Animation de modeles UML : application dans le contexte des smartgrids. In *CIEL'12 : Conférence en IngénieriE du Logiciel*, pages 1–6, 2012. 58, 114

[2004-99] Seth Bullock and Dave Cliff. Complexity and emergent behaviour in ICT systems. 2004. Hewlett-Packard Labs, 2004. 59

[1992-100] John S Gero. Creativity, emergence and evolution in design. *Preprints Computational Models of Creative Design*, pages 1–28, 1992, 1992. 59

[2006-101] Jean-Marc Jézéquel, Sébastien Gérard, and Benoit Baudry. Le génie logiciel et l'IDM : une approche unificatrice par les modèles. *L'ingénierie dirigée par les modèles*, 2006. Lavoisier, Hermes-science, 2006. 62, 63

[1981-102] Alvin Toffler, Wally Longul, and Harry Forbes. *The third wave*. Bantam books New York, 1981. 63

[2014-103] Frank Webster. *Theories of the information society*. Routledge, 2014. 63

[2013-104] Guillaume Barbier. bbb. 2013. Université Blaise Pascal, Clermont II, 2013. 64

[2005-105] Jean-Marie Favre. Foundations of meta-pyramids : Languages vs. metamodels–episode ii : Story of thotus the baboon1. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2005. 73

[2016-106] Rachida Seghiri, Frédéric Boulanger, Claire Lecocq, and Vincent Godefroy. An Executable Model Driven Framework for Enterprise Architecture Application to the Smart Grids Context. In *System Sciences (HICSS), 2016 49th Hawaii International Conference on*. IEEE, 2016. 73

[2004-107] Ali Arsanjani. Service-oriented modeling and architecture. *IBM developer works*, pages 1–15, 2004, 2004. 73

[2005-108] Jean Bézivin. On the unification power of models. *Software & Systems Modeling*, 4(2) :171–188, 2005. Springer, 2005. 78

[2008-109] David Chen, Guy Doumeingts, and François Vernadat. Architectures for enterprise integration and interoperability : Past, present and future. *Computers in industry*, 59(7) :647–659, 2008. Elsevier, 2008. 79, 85

[2005-110] Frank S de Boer, Marcello M Bonsangue, LPJ Groenewegen, AW Stam, L van der Torre, et al. Change impact analysis of enterprise architectures. In *Information Reuse and Integration, Conf, 2005. IRI-2005 IEEE International Conference on.*, pages 177–181. IEEE, 2005. 82

[1990-111] Herbert A Simon. Prediction and prescription in systems modeling. *Operations Research*, 38(1) :7–14, 1990. INFORMS, 1990. 85

[2001-112] GRAWITZ Madeleine. Méthodes des sciences sociales. *Paris, dalloz*, 2001, 2001. 110

[2007-113] Jim Steel and Jean-Marc Jézéquel. On model typing. *Software & Systems Modeling*, 6(4) :401–413, 2007. Springer, 2007. 127