Méthode, modèles et outil pour la méta-modélisation des processus d'ingénierie de systèmes d'information

Charlotte Hug

UNIVERSITÉ JOSEPH FOURIER - GRENOBLE I
École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique

Doctorat
Discipline : Informatique

Charlotte HUG

Méthode, modèles et outil pour la méta-modélisation des processus d'ingénierie de systèmes d'information

Thèse soutenue le 20 octobre 2009

Jury :
Mme Colette ROLLAND, Rapporteur
M. Khalid BENALI, Rapporteur
M. Jean-Pierre PEYRIN, Président
Mme Jolita RALYTÉ, Examinatrice
Mme Dominique RIEU, Directrice de thèse
Mme Agnès FRONT, Co-directrice de thèse

Thèse préparée au sein du Laboratoire d'Informatique de Grenoble
Equipe SIGMA

À mes grands-parents,

Je tiens à exprimer mon immense reconnaissance à mes directrices de thèse, Mesdames Dominique Rieu et Agnès Front. Elles m'ont guidée et soutenue durant ces trois années. Ce sont deux femmes exceptionnelles sans qui cette thèse n'aurait pas été aussi plaisante et enrichissante tant d'un point de vue professionnel que personnel.

Je souhaite également remercier Monsieur Jean-Pierre Giraudin, responsable de l'équipe SIGMA, de m'avoir accueillie au sein de son équipe et donné les moyens de mener ce travail à son terme. Il est pour moi un modèle de droiture et d'honnêteté professionnelle.

Je tiens à remercier les membres du jury :

Monsieur Jean-Pierre Peyrin, Professeur à l'Université Joseph Fourier de Grenoble, pour en avoir accepté la présidence,

Madame Colette Rolland, Professeure à l'Université Paris1-Panthéon-Sorbonne et rapporteur de cette thèse, que je remercie pour son intérêt pour mon travail et toute l'aide qu'elle a pu m'apporter durant la thèse,

Monsieur Khalid Benali, Maître de Conférences à l'Université Nancy 2, d'avoir accepté de rapporter cette thèse et que j'ai eu le plaisir de revoir depuis mon départ de la Miage de Nancy,

Madame Jolita Ralyté, Maître de Recherche et d'Enseignement de l'Université de Genève, pour avoir examiné ce travail.

Mon séjour en Australie a été rendu possible grâce à l'accueil de Monsieur Brian Henderson-Sellers, Professeur à l'University of Technology, Sydney, que je remercie pour cette opportunité unique. Je remercie également la section de Grenoble de l'Association Française des Femmes Diplômées d'Université qui a contribué à la réussite de ce séjour, ainsi que Monsieur Tom McBride, Senior Lecturer à l'UTS, pour m'avoir fait découvrir ses « favorite places to have lunch ».

Je remercie Madame Nadine Mandran, ingénieure à Marvelig, de m'avoir permis de réaliser les expérimentations ; il est rare de rencontrer des personnes aussi ouvertes et enthousiastes, sa présence en fin de thèse m'a réellement encouragée.

Merci aux « industriels » qui ont consacré un temps précieux aux entretiens : Mesdames Maria-Enrica Cannizzo et Cathy Descombes ainsi que Messieurs Didier Donsez, Laurent Iss, Emmanuel Jausseran, Julien Luc, Emmanuel Martin, Nicolas Pignier et Laurent Testard.

Je tiens également à remercier Mesdames Sophie Dupuy-Chessa, Claudia Roncancio et Christine Verdier, enseignantes-chercheuses dans l'équipe SIGMA pour leur gentillesse et leur disponibilité.

Merci aux docteurs et futurs docteurs de l'équipe SIGMA : Rajaa et Nicolas, que j'ai très souvent sollicités, ainsi que Luz-Maria, Marco et tous les autres pour leurs encouragements.

Cette thèse m'aura permis de rencontrer, au hasard des couloirs, d'autres doctorants avec qui j'ai partagé de très bons moments : merci à Christine, Sattisvar et Rami, mes compagnons de RU et surtout amis.

À Grenoble, mon chemin a croisé celui de Jaklyne et Viviana, que je remercie pour leurs conseils, leur optimisme et leur présence tout au long de cette thèse dans des ambiances jazzy et latino. Vous êtes géniales, surtout ne changez pas ☺

Spéciale dédicace à Michel, pour nos « prem's » et toutes les autres bêtises que nous avons échangées dès notre première rencontre !

Merci à la famille Iss-Péquignat pour les soirées en famille, et en particulier Laurent pour son dynamisme et son imagination sans limites quel que soit le sujet !

Je remercie enfin mes parents de m'avoir laissée faire mes propres choix et de m'avoir soutenue à leur manière durant ces longues années. Promis, je vais enfin me mettre au travail !

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

1. INTRODUCTION

Le travail de cette thèse se situe dans le domaine des méthodes pour l'ingénierie des systèmes d'information (SI) et plus particulièrement dans le domaine de la méta-modélisation des processus d'ingénierie de SI.

1.1. Contexte et problématique

Afin de produire de manière efficace des systèmes d'information répondant aux besoins des organisations, de nombreuses méthodes d'ingénierie de SI ont été introduites. Selon (Harmsen, 1997), *« une méthode pour l'ingénierie des systèmes d'information est une collection intégrée de procédures, techniques, descriptions de produits, et outils pour un support du processus d'ingénierie de SI performant, efficace et consistant. »*. Pour (Booch, 1991), une méthode est *« un processus rigoureux pour générer un ensemble de modèles décrivant différents aspects du logiciel en construction en utilisant une notation bien définie »*. Les premières méthodes introduites sont dites séquentielles comme la Cascade (Royce, 1970) ou le cycle de vie en V (Mc Dermid et Ripken, 1984), puis les méthodes itératives comme RUP (Kruchten, 2000), ou Symphony (Hassine, 2005) sont apparues et enfin les méthodes Agiles comme SCRUM (Schwaber et Beedle, 2001) et XP (Beck, 1999) ont émergé.

Malgré la diversité des méthodes d'ingénierie de SI, beaucoup d'organisations n'en utilisent pas à l'heure actuelle. Les chefs de projet suivent un processus défini « mentalement » et toute l'équipe de développement suit ce processus sans forcément savoir quel était le point de départ, quel est le point d'arrivée, qui intervient dans ce processus, quel est le rôle de chacun, etc. Ceci correspond au premier niveau du Capability Maturity Model Integration (CMMI, 2009) : la réussite des projets est imprévisible et elle est basée sur la capacité, la volonté des individus voire d'un seul individu. Aucune documentation n'est réalisée et la reproductibilité du projet n'est pas garantie.

A contrario, certaines organisations utilisent des méthodes, des processus pour l'ingénierie des systèmes d'information qui ne sont pas adaptés au contexte des projets : les cycles de vie sont trop longs, il y a trop de documentation à fournir, aucune marge de manœuvre n'est possible entre les spécifications réalisées en amont et le développement, etc. Le processus est alors lourd à gérer, les différents acteurs ne s'impliquent pas ou peu et plus personne ne sait qui fait quoi dans le projet. La durée des projets a donc tendance à s'allonger, les budgets augmentent, et les systèmes d'information réalisés sont livrés avec du retard et/ou ne correspondent pas à ce qu'attendaient les utilisateurs. La mise en place de méthodes dans ces situations est donc improductive.

Liées à une utilisation accrue des développements par composants, les méthodes Agiles ont donc réellement aujourd'hui leur place dans l'industrie du système d'information et du logiciel. Les projets sont divisés en cycles courts qui permettent un contact direct et régulier avec le client. Les systèmes d'information sont donc développés par assemblage de composants, chacun d'entre eux étant validés par le client

et mis à jour selon les nouveaux besoins. Cependant, sur de gros projets de type restructuration complète de systèmes d'information lourds qui peuvent durer plusieurs années, un cycle de vie plus global doit être défini, comme une couche supérieure aux processus agiles.

Il est donc aujourd'hui fondamental d'offrir aux organisations des méthodes d'ingénierie de SI utiles et utilisables. En particulier, le processus de développement doit être compréhensible par les différents acteurs d'un projet :

- La maitrise d'œuvre qui réalise le développement du système d'information, d'une part pour communiquer avec le client, l'informer de la progression du projet, d'autre part pour coordonner ses équipes en interne. Comme le préconise le CMMI, plus un processus est défini, moins la marge d'erreur est importante et plus le taux de réutilisation et de reproductibilité d'un projet est élevé. Ainsi, la durée et donc le coût des projets de développement de SI sont réduits.
- Le client : c'est à lui qu'est destiné le système d'information. Au cours d'un processus d'ingénierie de SI, il exprime ses besoins en amont du projet et valide les réalisations en aval. Dans les processus Agiles, il intervient régulièrement sur des périodes courtes (de quelques semaines à quelques mois). Le client doit donc comprendre à quels moments il doit intervenir dans le processus et ce qu'il doit faire. D'autre part, certains clients demandent expressément aux maitres d'œuvre de définir le cycle de vie des projets, ou imposent leurs propres cycles de vie. Les processus doivent donc être bien compris des deux côtés afin que le projet se réalise dans les meilleures conditions possible.

Il est donc nécessaire de définir des processus adaptés à la taille et à la complexité des projets, mais aussi à la culture de chaque organisation. Il faut donc permettre aux organisations d'adapter les processus existants aux contraintes et spécificités de leurs projets et/ou de définir de nouveaux processus « from scratch ».

Il existe des travaux dans le domaine de l'ingénierie des méthodes, notamment, l'ingénierie des méthodes situationnelles qui consiste à construire et à adapter des méthodes d'ingénierie de SI selon chaque projet (Kumar et Welke, 1992) (Harmsen, 1997) (Ralyté, 2001) (Ralyté et al., 2007). Ces approches se focalisent à la fois sur l'aspect produit de la méthode, c'est-à-dire la structure des artefacts réalisés, et l'aspect processus, les actions à réaliser pour produire ces artefacts. L'ingénierie des méthodes situationnelles permet de réutiliser des fragments de produit ou de processus, de les assembler à la volée pour guider la construction du système d'information selon le contexte. Notre approche est différente, car elle se focalise sur les processus d'ingénierie de SI et leur méta-modélisation, c'est-à-dire à un niveau d'abstraction plus élevé que l'approche de l'ingénierie des méthodes situationnelles.

1.2. Objectifs et contributions

Dans cette thèse, nous proposons une méthode pour construire des méta-modèles de processus pour l'ingénierie des systèmes d'information.

En effet, la définition de modèles de processus est à la fois guidée et contrainte par un méta-modèle en amont qui définit les concepts dont les différents acteurs d'un

processus d'ingénierie de SI ont besoin pour définir leurs processus. Ces méta-modèles sont la plupart du temps implicites (il s'agit d'une connaissance tacite), c'est-à-dire qu'ils ne sont ni décrits ni modélisés, mais les ingénieurs des méthodes qui définissent les modèles ont une structure en tête qu'il est fondamental de formaliser (il s'agit de la connaissance explicite). D'autre part, un méta-modèle permet de définir une trame générale des processus d'ingénierie de SI pouvant être réutilisable dans une organisation en définissant des modèles de processus propres à chaque projet à l'intérieur de cette même organisation. Les méta-modèles ont ainsi un rôle structurant très important.

Notre objectif est donc de permettre aux ingénieurs des méthodes de modéliser des méta-modèles de processus pour l'ingénierie des systèmes d'information en tenant compte des spécificités et des contraintes de leurs organisations. Ces méta-modèles doivent également prendre en considération les points de vue des différents acteurs du processus (maitre d'œuvre et client).

De plus, nous souhaitons guider les ingénieurs des méthodes dans la création de leurs méta-modèles. La méthode proposée doit permettre de créer des méta-modèles adaptés, tout en vérifiant leur cohérence, afin que les modèles instanciés par la suite soient eux aussi cohérents. Afin d'assurer une adaptation maximale, nous préconisons une approche par imitation de patrons pour l'enrichissement et la personnalisation des méta-modèles en cours de construction.

Nous proposons donc une méthode pour la construction de méta-modèles de processus d'ingénierie de SI. Ce guidage est basé sur un graphe conceptuel qui définit l'ensemble des concepts possibles pour définir un méta-modèle de processus tout en vérifiant la cohérence du méta-modèle réalisé. Le graphe conceptuel permet de guider l'utilisation des supports de la méthode qui sont le méta-modèle de domaine des processus d'ISI ainsi que des patrons génériques et de domaine pour la méta-modélisation des processus.

Le méta-modèle de domaine contient les concepts principaux des différents méta-modèles de processus existants. Il inclut les cinq points de vue définis dans la littérature : activité, produit, décision, contexte et stratégie. De plus, il prend en compte deux niveaux d'abstraction permettant de séparer l'aspect intentionnel et opérationnel des processus d'ingénierie de systèmes d'information.

Les patrons génériques et de domaine permettent, à partir des concepts principaux du méta-modèle de domaine, d'étendre, de compléter ou d'abstraire le nouveau méta-modèle de processus en cours de construction. Les patrons génériques sont des patrons spécifiques à la méta-modélisation des processus. Les patrons de domaine correspondent à des fragments de méta-modèles de processus existants.

La méthode proposée est implémentée sous la forme d'un prototype et expérimentée auprès d'experts en ingénierie de systèmes d'information issus du monde universitaire ainsi qu'auprès de professionnels du domaine.

1.3. Plan de la thèse

La première partie de cette thèse, consacrée à l'état de l'art, est composée de deux chapitres :

- le Chapitre 2 présente des travaux connexes à notre approche, en l'occurrence, l'ingénierie des méthodes situationnelles, puis détaille différents patrons utiles pour la méta-modélisation des processus ;
- le Chapitre 3 présente les différents méta-modèles de processus pour l'ingénierie de SI.

Dans la deuxième partie de cette thèse, nous présentons nos propositions :

- le Chapitre 4 présente les supports de la méthode qui sont : le méta-modèle de domaine, les patrons génériques et de domaine, et le graphe conceptuel ;
- le Chapitre 5 décrit la méthode pour la construction de méta-modèles de processus d'ingénierie des SI.

La troisième et dernière partie de la thèse concerne :

- l'implémentation d'un outil pour la méta-modélisation des processus d'ingénierie de SI, dans le Chapitre 6 ;
- la validation de la méthode par des experts du domaine, dans le Chapitre 7.

Enfin, nous concluons cette thèse en résumant les contributions de notre travail dans le domaine de la modélisation des processus d'ingénierie de SI. Nous abordons également les perspectives de notre travail.

Le tableau suivant résume l'organisation générale de cette thèse en trois parties.

État de l'art	Chapitre 2 : Modélisation des méthodes d'ingénierie de systèmes d'information	Méthodes situationnelles et patrons pour la méta-modélisation et la catégorisation.
	Chapitre 3 : Modèles et méta-modèles de processus	Modèles et méta-modèles de processus pour l'ingénierie des SI.
Proposition	Chapitre 4 : Les supports de la méthode	Méta-modèle de domaine, patrons génériques et de domaine, graphe conceptuel.
	Chapitre 5 : Une méthode pour la construction de méta-modèles de processus	Méthode proposée pour construire des méta-modèles de processus.
Implémentation et validation	Chapitre 6 : Implémentation de la méthode	Prototype proposé pour la méta-modélisation des processus.
	Chapitre 7 : Validation	Expérimentations utilisateurs validant notre proposition.

Tableau 1-1. Plan de la thèse.

ÉTAT DE L'ART

INTRODUCTION A L'ETAT DE L'ART

Le travail de cette thèse se situe dans le domaine de l'ingénierie des systèmes d'information et concerne plus particulièrement les processus d'ingénierie. En outre, notre travail se focalise sur la méta-modélisation de ces processus.

Une méthode est composée d'un modèle de processus et d'un ou plusieurs méta-modèles produit. La Figure 1 présente les quatre niveaux de modélisation pour les produits et les processus tels que définis par l'OMG (MOF, 2005). Les produits représentent le résultat à atteindre et les processus le chemin à parcourir pour atteindre le résultat.

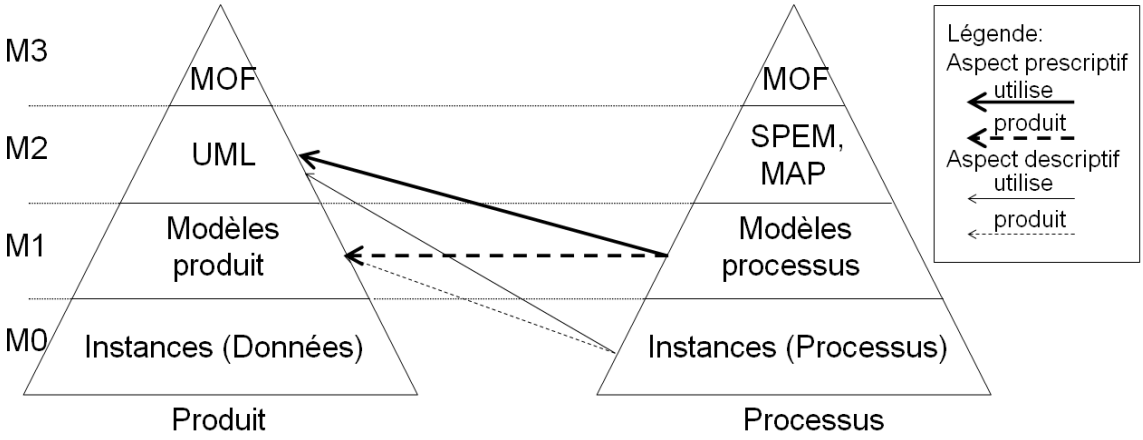


Figure 1. Les 4 niveaux de modélisation produit et processus.

Du côté des produits, le premier niveau M0 représente le niveau des instances, c'est-à-dire les objets du monde réel représentés dans le système. Le niveau M1 est le niveau des modèles produit, ce sont des diagrammes de classes par exemple. Le niveau M2 est le niveau du méta-modèle, le plus courant étant UML (OMG, 2007). UML définit les concepts qui pourront être instanciés pour créer des modèles composés de classes, d'associations et d'attributs, entre autres. Le niveau M3 est le niveau de la méta-méta-modélisation. Par exemple, l'OMG a défini le MOF (OMG, 2006) mais il en existe d'autres comme KM3 (Jouault et Bézivin, 2006) ou Ecore (Ecore, 2009).

Du côté des processus, le niveau M0 représente l'exécution des processus réels, dans notre cas, les processus d'ingénierie de SI. Le niveau M1 représente les modèles de processus à utiliser, comme le modèle de processus du RUP par exemple. Le niveau M2 est celui des méta-modèles de processus qui représentent les modèles qui permettront de définir des modèles de processus, comme le méta-modèle de l'OMG, SPEM (OMG, 2008) ou le méta-modèle de la MAP (Rolland et al., 1999). Enfin, le niveau M3 bénéficie du méta-méta-modèle du MOF par exemple.

Deux aspects sont à considérer. D'une part, les modèles de processus prescrivent l'utilisation de méta-modèles produit comme UML pour réaliser des modèles produit (en gras sur la figure), le modèle de processus représente donc la démarche à suivre a priori. D'autre part, un processus d'ingénierie de systèmes d'information réel (instance), utilise les méta-modèles produit pour produire des modèles produit : cet aspect du processus est descriptif, car on ne fait que modéliser la trace de ce qui se passe

réellement. Ainsi, RUP est composé du modèle de processus du RUP et le méta-modèle produit utilisé est UML.

Notre travail se focalise sur l'aspect processus des méthodes d'ingénierie de SI et tout particulièrement leur méta-modélisation.

La Figure 2 détaille les quatre niveaux de modélisation pour les processus d'ingénierie de SI de manière simplifiée. Le méta-méta-modèle (MOF par exemple) définit les éléments (par exemple la classe *Classifier*) pouvant être instanciés dans un méta-modèle. Le méta-modèle contient les classes permettant de décrire les modèles de processus. Le méta-modèle de la Figure 2 représente le fait qu'une *Activité* est réalisée par un ou plusieurs *Acteurs* et produit des *Ressources*. Les modèles de processus définissent des processus d'ingénierie de systèmes d'information dans une organisation en particulier. Le modèle de processus de la Figure 2 représente le fait qu'un analyste (instance d'*Acteur*) réalise l'analyse des processus métier (instance d'*Activité*) et produit un modèle de processus métier (instance de *Ressource*). Enfin, le modèle de processus est exécuté au niveau des instances, pour représenter le déroulement du processus d'ingénierie d'un SI en particulier.

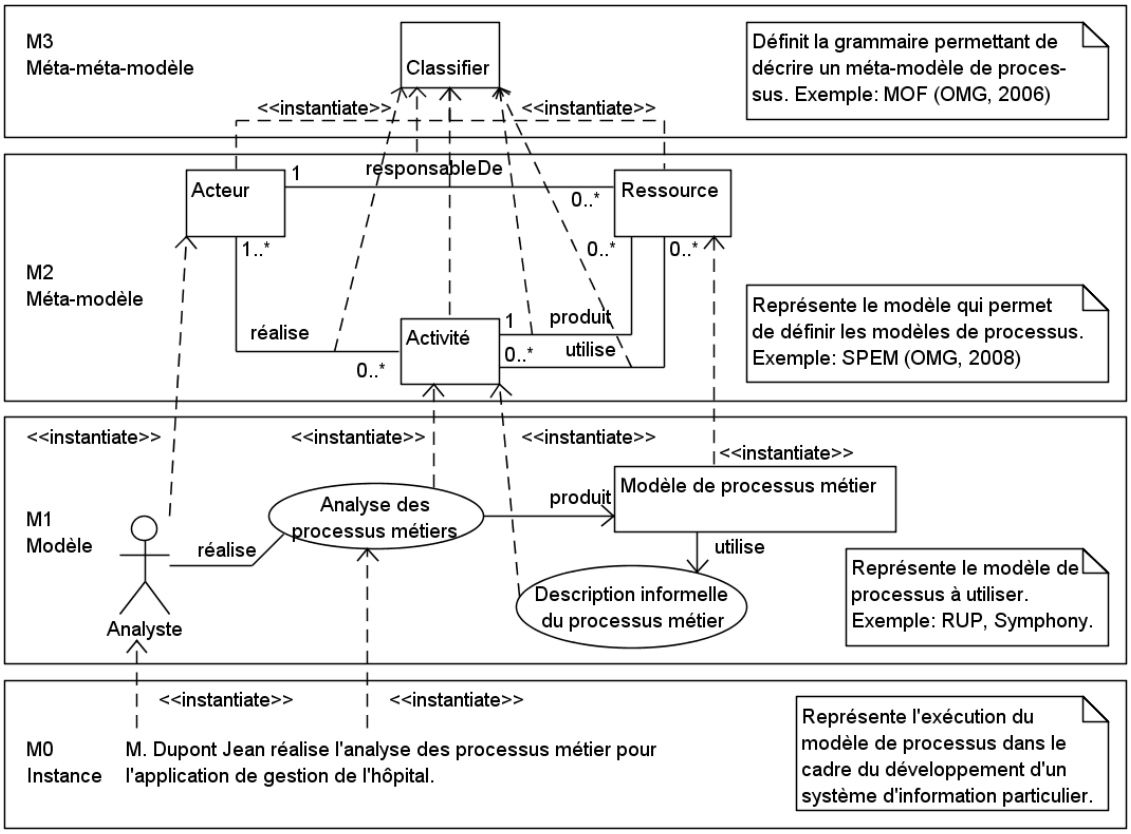


Figure 2. Les quatre niveaux de modélisation des processus.

Dans un premier chapitre, nous présentons l'ingénierie des méthodes situationnelles, qui est une approche similaire à la nôtre, mais concentrée au niveau de la modélisation (M1). Nous présentons ensuite une des techniques pour la méta-modélisation des processus d'ingénierie des SI qui est celle des patrons. Les patrons permettent de construire ou compléter des modèles ou méta-modèles. Nous présenterons en détail quelques patrons utiles pour la méta-modélisation des processus.

Dans le second chapitre, nous présentons les différents méta-modèles de processus existants pour l'ingénierie des systèmes d'information. Nous montrerons que les méta-modèles existants sont insuffisants en termes de représentation, de vocabulaire, d'adaptation et de complémentarité pour permettre aux ingénieurs des méthodes de modéliser les processus d'ingénierie de SI de leurs organisations.

2. MODELISATION DES METHODES D'INGENIERIE DE SYSTEMES D'INFORMATION

Une méthode d'ingénierie de systèmes d'information définit une démarche décrivant la construction d'un système d'information, de la définition des besoins à la livraison et à la maintenance du système.

Dans cette partie, nous abordons dans un premier temps l'approche d'ingénierie des méthodes situationnelles qui proposent certaines techniques pour la modélisation de méthode d'ingénierie des SI puis, nous exposons des techniques utiles pour la méta-modélisation des processus d'ingénierie de SI.

2.1. Approche de l'ingénierie des méthodes situationnelles

L'ingénierie des méthodes situationnelles (ou SME[1]) est « la discipline visant à construire et à adapter une méthode de développement de SI et les outils associés à chacun des projets spécifiques auxquels elle est appliquée » (Kumar et Welke, 1992). Les SME permettent la construction de nouvelles méthodes adaptées aux spécificités des organisations, des projets en se basant sur la réutilisation de fragments de méthodes et leur assemblage. De nombreux auteurs ont proposé des méthodes basées sur les SME comme (Harmsen, 1997) (Ralyté, 2001) et plus récemment une conférence a été organisée sur le sujet (Ralyté et al., 2007).

Les avantages des méthodes situationnelles sont leur grande adaptabilité et flexibilité aux contraintes et spécificités des différents projets et organisations. La construction des méthodes à la volée augmente leur adaptation au contexte. D'autre part, l'usage des fragments permet un gain de temps ainsi qu'une réutilisation des connaissances et bonnes pratiques définies et éprouvées dans d'autres méthodes.

Les SME sont basées sur des fragments. Il en existe deux types : des fragments produit et des fragments processus. Les fragments produit sont des spécifications de produits livrables ou requis dans une méthode (Harmsen, 1997), ce sont des sous-ensembles de méta-modèles produit comme UML (Henderson-Sellers et al., 2007). Les fragments processus représentent la description d'une activité qui doit être réalisée dans une méthode (Harmsen, 1997), ce sont des parties de modèles de processus comme des activités du RUP par exemple.

Les fragments de méthode représentent un fragment produit et un fragment processus. Plus précisément, un fragment de méthode, ou chunk, définit un produit (fragment produit) qui est réalisé par le fragment processus.

La Figure 2-1 présente un exemple de chunk composé d'un fragment processus (à gauche) représenté sous forme d'une MAP et d'un fragment produit (à droite) modélisé avec UML. Ce fragment est un extrait de la méthode CREWS – L'Écritoire (Ralyté, 2001b). Il permet en particulier de décrire et conceptualiser les scénarios afin de découvrir les besoins fonctionnels du futur système d'information. Le fragment processus modélise les étapes pour trouver et définir les scénarii, le fragment produit décrit la structure d'un scénario sous la forme d'un méta-modèle.

[1] Situational Method Engineering.

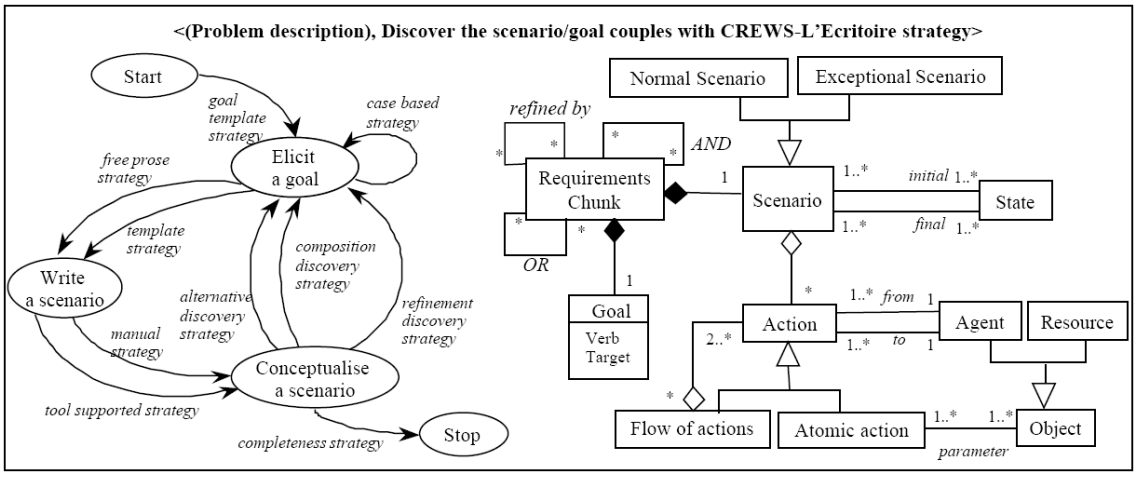


Figure 2-1. Exemple de chunk composé d'un fragment processus et produit.

(Brinkkemper et al., 1999) présente un exemple de fragment de méthode pour l'ingénierie des systèmes d'information, voir la figure ci-dessous. Ce fragment représente le méta-modèle de processus des Statecharts, vu comme un produit.

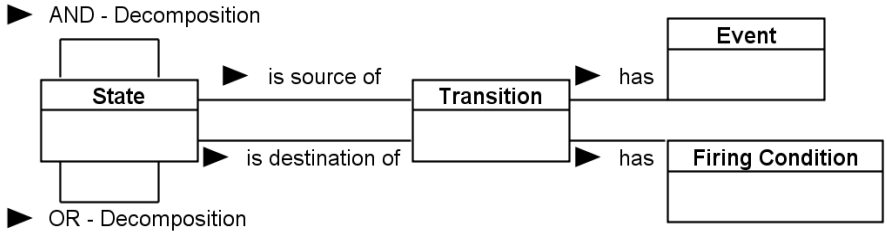


Figure 2-2. Fragment de méthode du Statechart.

Nous considérons que les fragments de méthode sont identiques à des patrons de domaine. D'après (Conte et al., 2001), les patrons de domaine apportent des solutions concernant des problèmes spécifiques à un domaine métier. Dans notre thèse, le domaine métier est celui de la méta-modélisation des processus d'ingénierie de SI. De plus, certains patrons génériques (dont les problèmes traités sont rencontrés dans différents domaines d'application (Conte et al., 2001)) initialement dédiés aux SI peuvent être utilisés pour la méta-modélisation des processus d'ingénierie. Ils sont présentés à partir de la section 2.2.

L'approche que nous proposons est différente de l'approche des SME. L'objectif des SME est de proposer des fragments produit et processus pour construire des systèmes d'information. Un ensemble de fragments correspond donc à une méthode pour l'ingénierie des SI. Notre travail concerne la méta-modélisation des processus d'ingénierie de systèmes d'information. Notre approche est focalisée sur les processus uniquement, à un niveau de modélisation supérieur à celui des SME. Cependant, la réutilisation de patrons de domaine (ou fragments) sera très utile pour la méta-modélisation des processus d'ISI comme nous le verrons dans le Chapitre 4.

2.2. Techniques pour la méta-modélisation des processus d'ISI

Deux principaux problèmes sont rencontrés au moment de réaliser des méta-modèles de processus d'ingénierie de SI :

- Il faut pouvoir catégoriser des éléments de processus. Par exemple, les méta-modèles de processus doivent permettre de distinguer les éléments de processus de type « Activité » et les éléments de processus de type « Phase ».
- Les attributs définis dans les méta-modèles de processus doivent pouvoir être instanciés au niveau de modélisation souhaité, c'est-à-dire au niveau des modèles de processus, mais également à l'exécution des processus. Ceci permettrait par exemple de spécifier des propriétés valables pour tous les modèles de processus d'une organisation (comme la durée maximale d'une phase ou le nombre maximum de personnes pouvant participer à une activité) et des propriétés spécifiques à une exécution de processus, c'est-à-dire pour un projet en particulier (la date de début et de fin d'une activité du projet, les personnes impliquées dans le projet).

Pour résoudre ces deux problèmes, il convient tout d'abord de s'intéresser à des solutions « génériques » (utilisables en modélisation ou méta-modélisation des produits) et de voir dans quelle mesure ces solutions pourraient être adaptées au cas de la méta-modélisation des processus.

Nous présentons dans un premier temps des patrons génériques pour la catégorisation, puis nous abordons des techniques qui permettent d'instancier des attributs à différents niveaux de modélisation. Enfin, nous présentons des patrons qui utilisent à la fois une technique de catégorisation et d'instanciation d'attributs sur plusieurs niveaux de modélisation.

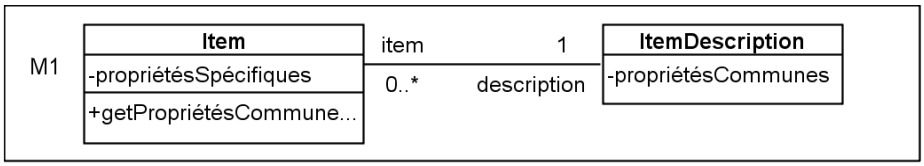
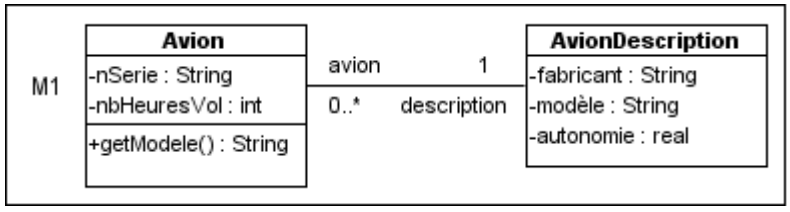
2.2.1. Catégorisation

La catégorisation permet de classer des concepts selon des propriétés communes. Le patron générique Item-Description proposé par (Coad, 1992) ainsi que le patron Type Object (Johnson et Woolf, 1996) détaillé en Annexe A permettent tous les deux de répondre à ce besoin.

a) Item-Description

Le patron Item-Description a été introduit par (Coad, 1992). Nous proposons ce patron tel que présenté par (Gzara, 2000) dans le Tableau 2-1, en utilisant le formalisme P-SIGMA (Conte et al., 2001).

Identifiant	Item-Description
Classification	Générique ^ Produit ^ Analyse
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Ce patron est utilisé lorsque certaines propriétés peuvent s'appliquer à plus d'un objet. Il permet de factoriser des propriétés communes à plusieurs objets.
Force	Ce patron permet de gérer des objets et leurs propriétés qui sont communes à d'autres objets.
Solution	Le patron Item-Description propose deux classes : Item et

Modèle	*ItemDescription*. La classe *ItemDescription* détient les valeurs des attributs qui peuvent s'appliquer à plus d'un objet de type *Item*. La classe *Item* détient les valeurs de ses propres attributs et diverses méthodes de consultation pour accéder aux valeurs des attributs de la classe *ItemDescription*. 
Cas d'Application	Un objet de type *Avion* connaît son propre nº de série (par exemple N123ABC) et son nombre d'heures de vol (par exemple 5000 heures). Il connaît également un unique objet de type *DescriptionAvion*. Un objet de type *DescriptionAvion* connaît son propre fabricant (par exemple Boeing), son modèle (par exemple 747-400) et l'altitude de croisière (par exemple 8333 miles). Un objet de type *DescriptionAvion* connaît également tous les objets de type *Avion* dont il est la description. Un objet de type *DescriptionAvion* détient toutes les propriétés communes aux objets *Avion*, telles que le fabricant, le modèle, etc. Ces propriétés partagées par tous les objets Avion sont décrites dans la classe *DescriptionAvion* et ne sont pas détenues dans les objets de type *Avion*. Par ailleurs, un objet *Avion* peut consulter certaines de ces propriétés. Les propriétés des objets de type *DescriptionAvion* restent visibles aux objets de type *Avion* en définissant des méthodes de consultation permettant de propager la valeur d'un attribut à travers les associations. Prenons l'exemple de la propriété modèle : quand on demande à un avion son modèle, il exécute la méthode *getModèle()*, définie dans sa classe *Avion*. Cette méthode consiste à retourner la valeur de l'attribut modèle de *DescriptionAvion*, à l'objet demandeur, c'est-à-dire l'avion (à travers le rôle description de l'association qui lie un avion à sa description). 
Conséquence d'application	L'avantage du patron Item-Description est la séparation des objets de leur description qui est commune à d'autres objets.

Alternative	Type Object (Johnson et Woolf, 1996) voir Annexe A.

Tableau 2-1. Patron Item-Description.

b) Exemple et synthèse pour la méta-modélisation des processus d'ISI

Les patrons Item-Description et Type Object sont définis pour être utilisés au niveau des modèles (M1).

La Figure 2-3 présente un exemple d'imitation du patron Item-Description pour la modélisation des processus d'ingénierie de SI. D'un côté, nous souhaitons présenter des unités de travail (imitation de la classe *Item*) et de l'autre, des catégories d'unité de travail (imitation de la classe *ItemDescription*).

Cette imitation pose deux problèmes :

– *Unité de travail* et *Catégorie d'unité de travail* étant des imitations des classes *Item* et *ItemDescription*, elles sont définies au niveau des modèles. Item-Description ne permet pas de définir les classes souhaitées au niveau des méta-modèles. Il n'est pas possible ici d'instancier *Analyse* et *Phase* pour décrire des objets de type Analyse et de type Phase spécifiques au déroulement de projets.

– Les attributs *statut*, *dateDebut* et *dateFin* ne concernent pas les phases d'analyse en général, mais les phases d'analyse dans des projets en particulier. Étant donné que le patron Item-Description est limité à deux niveaux de modélisation (M1 et M0), cela n'a pas de sens de valuer ces attributs dans l'objet *Analyse*.

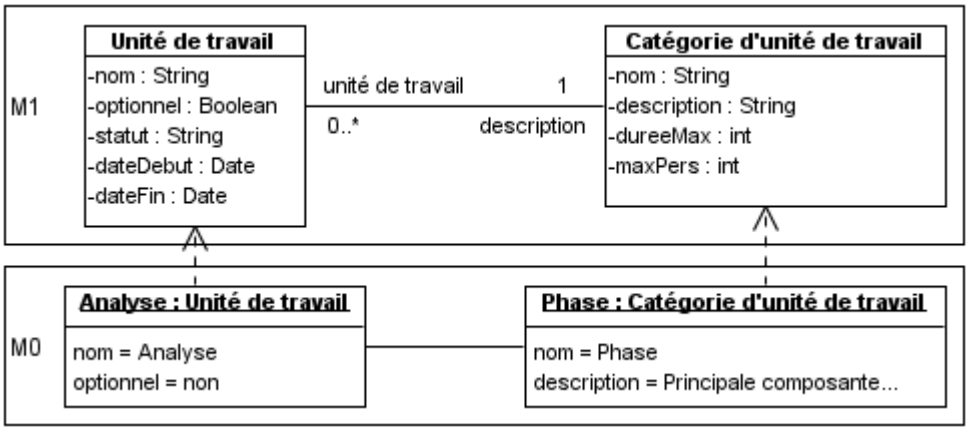


Figure 2-3. Exemple d'imitation du patron Item-Description.

2.2.2. Valuation d'attributs à différents niveaux de modélisation

La méta-modélisation, au sens d'UML, est dite stricte. (Atkinson and Kühne, 2002) donnent une définition de la méta-modélisation stricte :

Dans une architecture de N niveaux M_0, M_1, … M_{n-1},

(i) chaque élément de niveau M_m doit être une instance d'exactement un élément de niveau M_{m+1}, pour tout $0 \leq m < n-1$,

(ii) toute relation autre que « instance de » entre deux éléments X et Y implique que niveau(X)=niveau(Y).

Cette définition indique que seule l'instanciation permet de décrire des éléments d'un niveau de modélisation inférieur. Par exemple, les classes d'un modèle, leurs attributs et

associations sont instanciés pour représenter des diagrammes comprenant des objets, des propriétés valuées et des liens.

UML définit également que toute instance d'une classe est un objet dont les propriétés sont valuées et correspondent aux attributs de la classe (et ceux hérités des super-classes) dont il est instance (OMG, 2007b). Une classe ne peut donc contenir que des attributs et un objet, des propriétés valuées.

Cependant, il parait utile de définir des propriétés valuées dans des classes, afin que toutes les instances de ces classes partagent les mêmes propriétés. Par exemple, au sein d'une société, il faut pouvoir définir des propriétés dans les modèles de processus comme la durée maximale d'une phase, le nombre maximum d'acteurs pouvant participer à une activité. Cela permettrait d'éviter la réplication de données parmi tous les objets.

Dans cette section, nous présentons d'abord le principe du Clabject, repris dans des patrons tels que l'Instanciation Profonde présentée ensuite, puis dans les patrons Materialization et Powertype, présentés dans la partie suivante.

a) Le principe du Clabject

Le Clabject a été introduit pour représenter un élément ayant deux faces :

- une face classe qui contient des attributs typés, et
- une face objet qui comprend des propriétés valuées.

Ainsi, dans un seul et même élément, il est possible de représenter à la fois des attributs qui devront être instanciés par un objet et des propriétés valuées, qui généralisent des propriétés communes à tous les objets qui seront instanciés.

Un clabject (class+object) (Atkinson, 1997) peut être représenté de deux manières différentes :

- sous forme d'une seule classe (Odell, 1994), (Atkinson and Kühne, 2001) ;
- sous forme d'une classe et d'un objet couplés (Dahchour et al., 2002), (Henderson-Sellers et Gonzalez-Perez, 2005b).

La Figure 2-4 présente un exemple de clabject. Le clabject *Video* est à la fois une classe, car il contient des attributs typés (*isRented* et *renter*) et un objet puisqu'il a des propriétés valuées (*title* et *studio*).

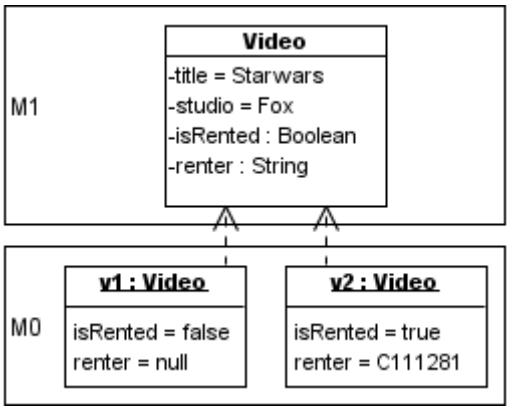


Figure 2-4. Exemple de Clabject.

La notion de Clabject ne doit pas être confondue avec la notion d'attribut de classe introduite dans les versions d'UML 1.3 à 1.5 (abandonnée depuis UML 2.0). Un attribut de classe permet de définir des propriétés partagées par tous les objets instances d'une même classe. Par défaut, il est possible d'affecter une valeur à ces attributs de classe dans le diagramme de classes (M1). Cependant, la valeur de l'attribut sera portée par l'objet instancié (en M0), alors qu'avec le Clabject, les propriétés partagées sont portées par le clabject (M1) et non ses instances.

Le concept de clabject peut être défini en s'appuyant sur la méta-modélisation stricte d'UML. La Figure 2-5 montre comment les attributs sont instanciés entre deux niveaux de modélisation et comment ils peuvent être propagés d'une classe mère à une classe fille. Le clabject apparait alors clairement comme étant l'union des classes de niveau M1, *ElementToRent* et *Video*. La notion de Clabject n'est donc pas un concept nouveau, mais une technique reposant sur des mécanismes objets de base permettant de simplifier un assemblage de deux classes liées par un lien d'héritage.

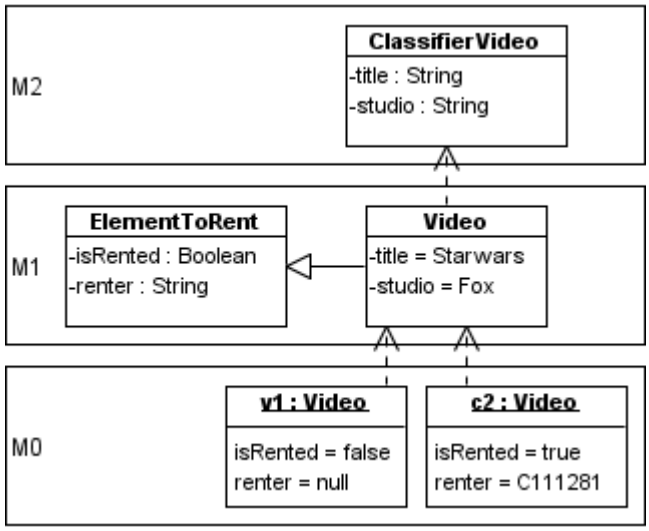
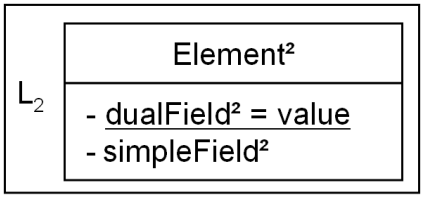


Figure 2-5. Clabject replacé dans les niveaux de méta-modélisation stricte d'UML.

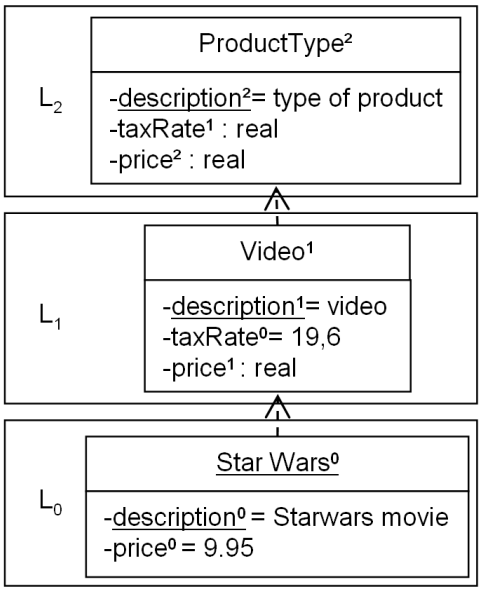
b) Instanciation Profonde

Le concept d'Instanciation profonde (Deep Instantiation) a été introduit par (Atkinson and Kühne, 2001), (Atkinson and Kühne, 2002). Nous le présentons ci-dessous sous la

forme d'un patron dans le formalisme P-Sigma pour des raisons d'uniformisation et de compréhension.

Identifiant	Instanciation Profonde
Classification	Générique ^ Produit
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Ce patron est utilisé lorsque l'on veut modéliser des attributs et des associations qui doivent être définis à travers plusieurs niveaux de modélisation.
Force	Ce patron évite les problèmes de : – classification ambiguë (lorsqu'une instance peut avoir plusieurs classifiers), – réplication des données.
Solution Modèle	La solution est définie dans l'espace logique L2[1]. Les deux principes clés de ce patron sont le concept d'exposant et celui de dual/simple field exposés ci-dessous. – L'exposant indique le nombre de niveaux de modélisation traversés par un élément. À chaque fois qu'un élément traverse un niveau, l'exposant est retranché de 1. Par exemple, la classe *Element* pourra être instanciée sur 2 niveaux inférieurs : L1 et L0. – Simple Field : un attribut simple est valué lorsque son exposant est égal à 0. Dans l'exemple, *simpleField* sera valué au niveau L0 (au niveau L1, son exposant aura la valeur 1). – Dual Field : un attribut dual peut être valué à tous les niveaux, tant que la valeur de son exposant le permet. Dans l'exemple, *dualField* aura une valeur aux niveaux L2, L1 et L0. La valeur d'un niveau à un autre peut changer. Une association peut également porter un exposant, retranché à chaque fois que l'association est instanciée au niveau inférieur de modélisation. 

[1] (Atkinson and Kühne, 2002) définissent les niveaux logiques L3, L2, L1 en comparaison avec les niveaux de l'OMG. Les niveaux logiques prennent en compte la classification des concepts dans un

Cas d'Application	Au niveau L2, une classe *ProductType* est définie avec : – un attribut simple *taxRate* avec un exposant égal à 1. – un attribut simple *price*, avec un exposant égal à 2. – un attribut dual *description* avec un exposant égal à 2. Au niveau L1, la classe *ProductType* est instanciée pour modéliser la classe *Video*. L'exposant de tous les éléments est retranché de 1. – l'exposant de *taxRate* vaut 0, il est donc valué. – l'exposant de *price* vaut 1. Rien ne se passe. – *Description* change de valeur. Au niveau logique L0, la classe Video est instanciée pour représenter le film Star Wars. – l'exposant de *price* est à 0, il est donc valué à 9.95. – *Description* change à nouveau de valeur. 

domaine. Pour (Atkinson and Kühne, 2002), les niveaux de modélisation de l'OMG, MX, mélangent deux dimensions : la dimension physique et la dimension logique. La dimension physique représente la structure du langage et la dimension logique représente la classification des concepts dans un domaine particulier. Pour simplifier la modélisation, nous utiliserons par la suite les niveaux de l'OMG.

Conséquence d'application	Le patron Instanciation Profonde permet de : – définir des propriétés valuées à tous les niveaux de modélisation (attribut dual). – définir au plus haut niveau de modélisation L2, à quel niveau seront valués les attributs (attribut simple). – décrire des mécanismes complexes avec un formalisme simple.

Tableau 2-2. Patron Instanciation Profonde.

c) Exemple et synthèse pour la méta-modélisation des processus d'ISI

La Figure 2-6 présente un exemple d'imitation du patron Instanciation Profonde pour la méta-modélisation des processus d'ingénierie de SI. Le patron permet de définir une méta-classe Unité de travail. Pour simplifier le formalisme et alléger la représentation graphique :

- les attributs ayant un exposant 1 sont représentés sans exposant ;
- les attributs ayant un exposant 0 étant valués, nous ne montrons pas l'exposant ;
- les classes n'étant définies que sur trois niveaux, nous ne précisons pas leur exposant.

Dans la méta-classe *Unité de travail*, cinq attributs simples sont définis : *nom* et *optionnel* sont des attributs qui seront valués au niveau des modèles de processus alors que *statut*, *dateDebut* et *dateFin* seront valués au niveau des instances. Ainsi, *Unité de travail* permet de définir des attributs qui seront définis dans les modèles de processus et à l'exécution de ces processus.

Dans la classe *Analyse*, les attributs *nom* et *optionnel* sont instanciés. Les exposants de *statut*, *dateDebut* et *dateFin* sont retranchés de 1. *Analyse* est un clabject.

Enfin, dans l'objet *A1*, instance d'*Analyse*, les attributs *statut*, *dateDebut* et *dateFin* peuvent être valués, pour représenter les caractéristiques de l'analyse dans un projet en particulier.

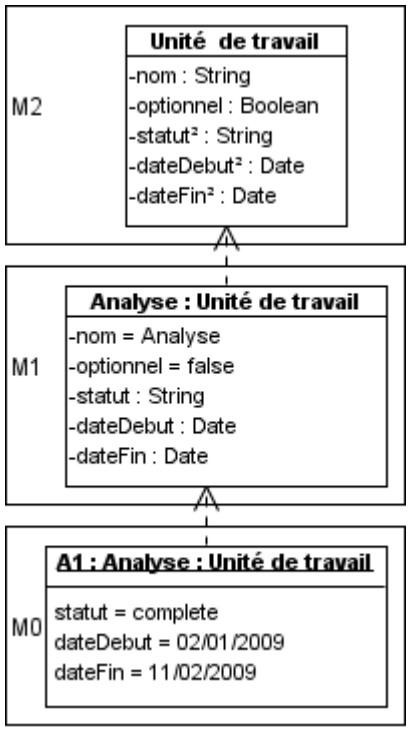


Figure 2-6. Exemple d'imitation du patron Instanciation Profonde.

Le concept de clabject associé à celui d'Instanciation Profonde permet de définir des attributs au niveau des méta-modèles et de les instancier au niveau de modélisation nécessaire. L'Instanciation Profonde utilise le concept de Clabject et le met en œuvre. Ces deux techniques permettent donc une plus grande flexibilité de modélisation. Dans le cadre de notre problématique, ces techniques répondent à un besoin important : pouvoir définir des propriétés valables pour des modèles de processus (M1) et des propriétés spécifiques à un projet en particulier (M0).

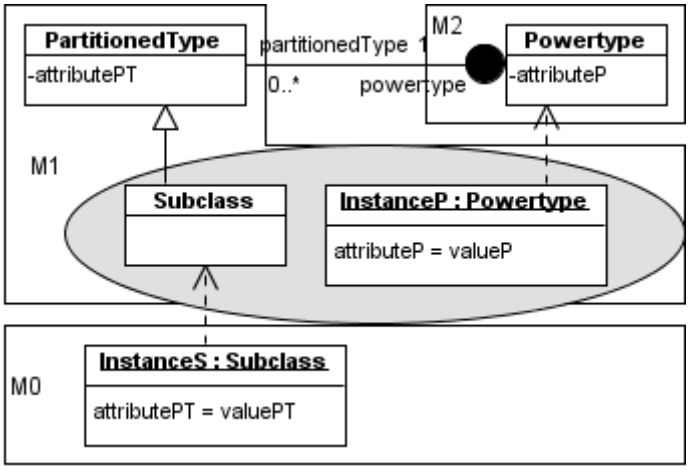
2.2.3. Catégorisation sur plusieurs niveaux de modélisation

Les deux concepts suivants permettent de catégoriser des éléments, sur plusieurs niveaux de modélisation. Nous les présentons dans la suite sous forme de patrons exprimés dans le formalisme P-Sigma.

a) Powertype

Le concept du Powertype a été introduit par (Odell, 1994) et repris par Brian Henderson-Sellers dans de nombreux articles sur la modélisation de processus, notamment (Henderson-Sellers et Gonzalez-Perez, 2005b).

Identifiant	Powertype
Classification	Générique ^ Produit
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Ce patron supporte la spécialisation dynamique en permettant aux sous-types d'une classe d'être définis comme des instances d'une autre classe par rapport à un discriminant donné.

Force	Il est possible de partitionner une classe en sous-classes en utilisant le sous-typage. Avec cette approche, chaque sous-type est implémenté comme une classe qui (i) doit être définie au moment de la conception et (ii) ne peut pas porter d'attributs valués. En définissant des sous-types comme des clabjects, les avantages des sous-types en tant que classes sont maintenus et combinés avec les avantages des sous-types en tant qu'objets, c'est-à-dire la capacité d'être créés à l'exécution et de porter des attributs valués.
Solution Modèle	La classe *Powertype* représente la classe qui permet de catégoriser les éléments instances de la classe *PartitionedType*. Le *Powertype* est représenté du côté du rond noir. Chaque sous-type introduit est implémenté comme un clabject, qui incorpore une face objet (instance du powertype) et une face classe (sous classe du type partitionné). Dans le clabject (oval gris), le *Powertype* est instancié et le *PartitionedType* est spécialisé. Seuls les attributs du *Powertype* sont valués. Au dernier niveau, la sous-classe *Subclass* est instanciée, les attributs hérités de *PartitionedType* sont valués.  Les niveaux de modélisation M2, M1 et M0 sont positionnés différemment par rapport aux exemples précédents : c'est ainsi que (Gonzalez-Perez et Henderson-Sellers, 2006) les placent dans l'architecture de l'OMG.
Cas d'Application	Il est possible de distinguer les arbres (*Tree*) de leur espèce (*TreeSpecies*). Le powertype *TreeSpecies* permet donc de partitionner l'ensemble des arbres. *TreeSpecies* a un attribut *name* et *Tree* a un attribut *height.* Le powertype *TreeSpecies* est instancié dans le clabject. L'attribut *name* est valué. Le type partitionné *Tree* est spécialisé dans le clabject pour spécifier

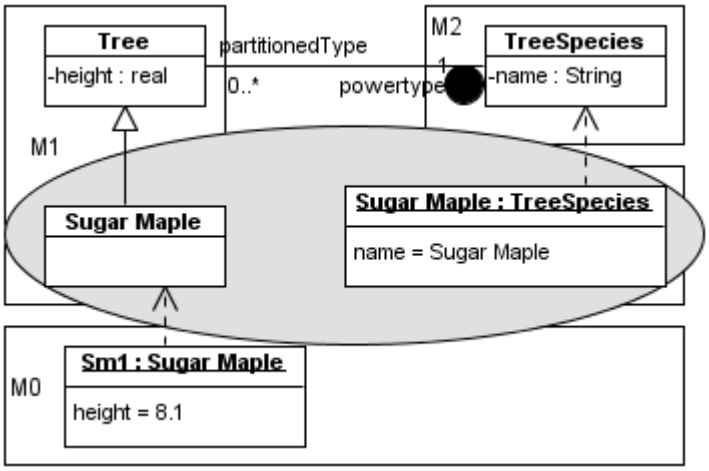
	la sous-classe *Sugar Maple* (érable). Cette sous-classe est ensuite instanciée pour représenter un érable *Sm1* en particulier, qui fait une hauteur de 8.1 pieds. 
Conséquence d'application	Avec ce patron, le sous-typage n'est plus limité au moment de la conception. Les sous-types des classes peuvent être introduits dynamiquement selon les besoins.
Alternative	Materialization (Dahchour et al., 2002)

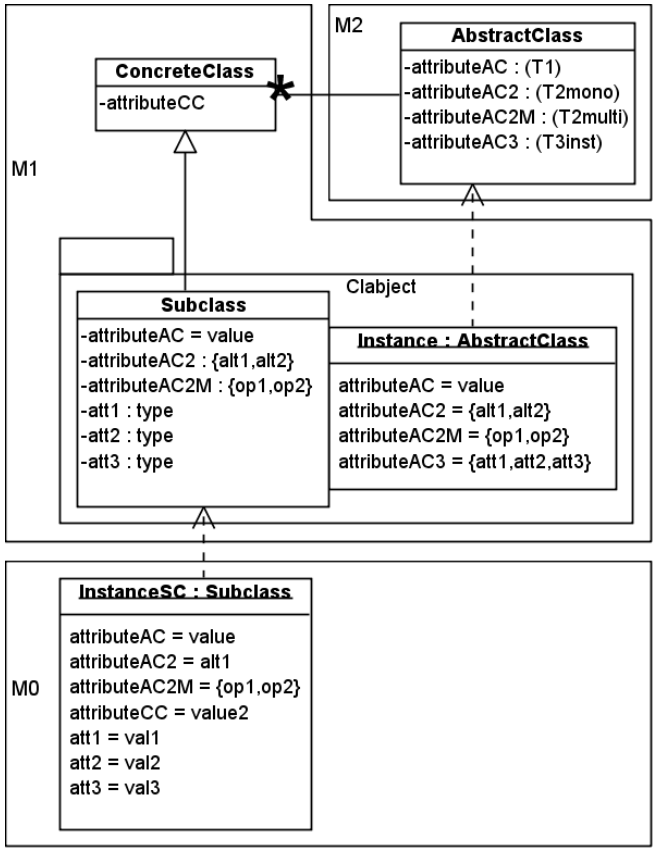
Tableau 2-3. Patron Powertype.

b) Materialization

Le patron Materialization a été introduit par (Dahchour et al., 2002).

Identifiant	Materialization
Classification	Générique ^ Produit
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Ce patron permet de lier des classes d'éléments abstraits avec des classes d'éléments plus concrets. Les éléments abstraits permettent de catégoriser les éléments concrets.
Force	Les avantages de ce patron sont : – la séparation des classes d'éléments abstraits et des classes d'éléments plus concrets. – la propagation des attributs de la classe d'éléments abstraits vers la classe d'éléments concrets.
Solution Modèle	Le patron est composé de deux classes principales : – *AbstractClass* qui correspond à la classe d'éléments les plus abstraits et qui catégorise les éléments de *ConcreteClass,* – *ConcreteClass* qui correspond à la classe d'éléments plus

	concrets. La classe la plus concrète est modélisée par l'étoile. Différents types d'attributs sont définis dans la classe *AbstractClass* : – *attributeAC* de type (T1) : cet attribut sera instancié et valué dans la classe instance de *Clabject*, *Instance*, et propagé à la sous-classe *SubClass* et à l'objet instance de *SubClass*. – *attributeAC2* (T2mono) est un attribut permettant de définir un domaine de valeurs possibles dans l'objet instance d'*AbstractClass*. L'attribut et son domaine sont propagés dans *Subclass*. L'objet instance de *Subclass* aura pour propriété une seule des valeurs définies dans le domaine (dans l'exemple ci dessous « alt1 »). – *attributeAC2M* de type (T2multi) a les mêmes caractéristiques que le type T2mono à l'exception du fait que plusieurs valeurs peuvent être choisies dans l'objet instance de *Subclass* (ici « op1 » et « op2 »). – *attributeAC3* (T3inst) permet de générer 3 attributs mono-valués dans la classe *Subclass* du *Clabject*. Ces attributs seront instanciés par l'objet instance de *Subclass*. La classe *AbstractClass* doit être systématiquement instanciée dans le clabject. Ses attributs doivent être valués. La classe *ConcreteClass* est systématiquement spécialisée dans le clabject. Les attributs valués de l'objet du clabject Instance sont propagés vers la sous-classe *Subclass*. Les attributs de type T3inst sont propagés sous forme d'attributs mono-valués. L'objet du dernier niveau *InstanceSC* est une instance de la sous-classe *Subclass*. Tous les attributs sont valués, dont les attributs hérités de *ConcreteClass*.

<table>
<tr><td></td><td> 
</td></tr>
<tr><td>Cas d'Application</td><td>La classe d'éléments abstraits CarModel permet de décrire n'importe quel modèle de voiture. La classe d'éléments concrets Car permet de décrire des exemplaires de voitures. Différents types d'attributs sont déclarés dans la classe CarModel :
– name et stickerPrice sont des attributs classiques, instanciés par l'objet FiatRetro.
– #doors et engineSize sont des attributs permettant de définir des domaines de valeurs, ils sont propagés à la sous-classe FiatRetro_Cars et sont instanciés par l'objet Nico's Fiat. Une seule des valeurs pourra être choisie pour valuer ces attributs.
– autoSound permet aussi de définir un domaine de valeurs, mais plusieurs valeurs pourront être choisies pour valuer l'attribut dans l'objet Nico's Fiat.
– specialEquip permet de définir trois attributs dans la classe FiatRetro_Cars qui seront instanciés dans l'objet Nico's Fiat.
La classe CarModel est instanciée dans le clabject, avec l'objet FiatRetro, qui représente toutes les propriétés possibles du modèle Fiat Retro. La classe Car est spécialisée dans le clabject par la sous-classe FiatRetro_Cars qui représente toutes les propriétés qu'aura chaque voiture du modèle Fiat Retro.</td></tr>
</table>

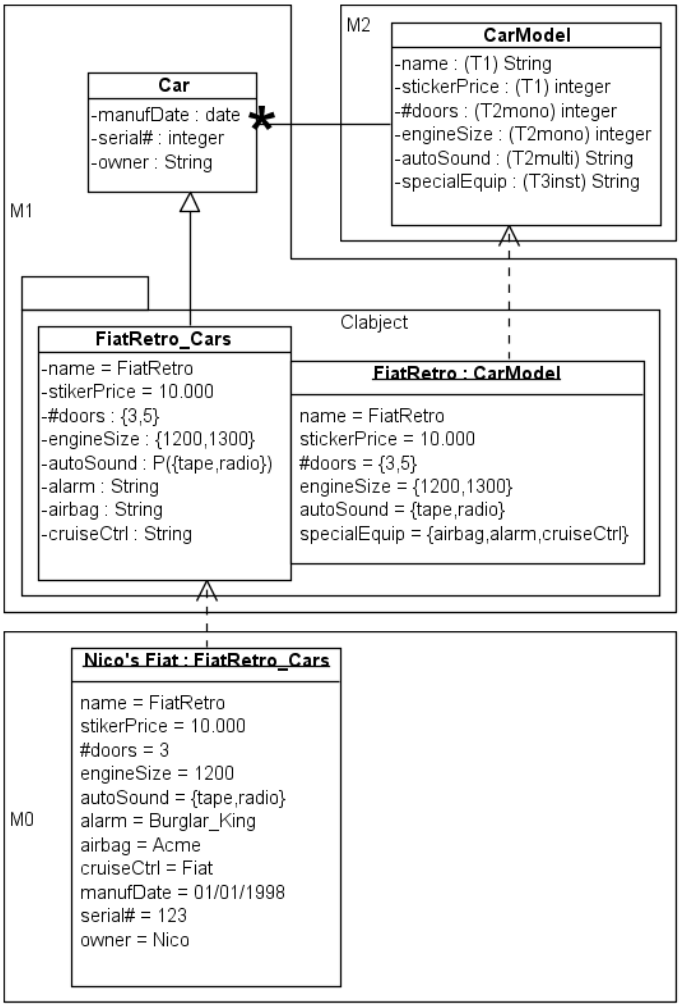
	Enfin, la classe *FiatRetro_Cars* est instanciée, pour créer l'objet *Nico's Fiat*, qui modélise la voiture Fiat Retro de Nico. Cette voiture a des propriétés spécifiques, comme la date de fabrication (*#manufDate*) ou le numéro de série (*#serial*), héritées de *Car*, mais également des propriétés générales à toutes les voitures du modèle Fiat Retro comme le prix (*stickerPrice*) ou le nombre de portes (*#doors*). 
Conséquence d'application	Les avantages du patron Materialization sont : – la relation entre des classes d'éléments abstraits avec des classes d'éléments plus concrets, – la propagation d'attributs des catégories abstraites vers les objets concrets.
Alternative	Powertype (Odell, 1994)

Tableau 2-4. Le patron Materialization.

Le patron Materialization a le même but et utilise la même technique de modélisation des clabjects que le Powertype. Il introduit cependant de nouveaux types d'attributs

(T2mono, T2multi, T3inst) qui rendent la modélisation beaucoup plus complexe, mais aussi plus précise.

c) Exemple et synthèse pour la méta-modélisation des processus d'ISI

La Figure 2-7 présente un exemple d'imitation du patron Powertype pour la méta-modélisation de processus d'ingénierie de SI. Le powertype *Type de phase* contient les attributs permettant de décrire n'importe quelle phase. Le type partitionné *Phase* permet de décrire des attributs spécifiques à une phase. Les attributs de *Type de phase* sont valués dans le clabject.

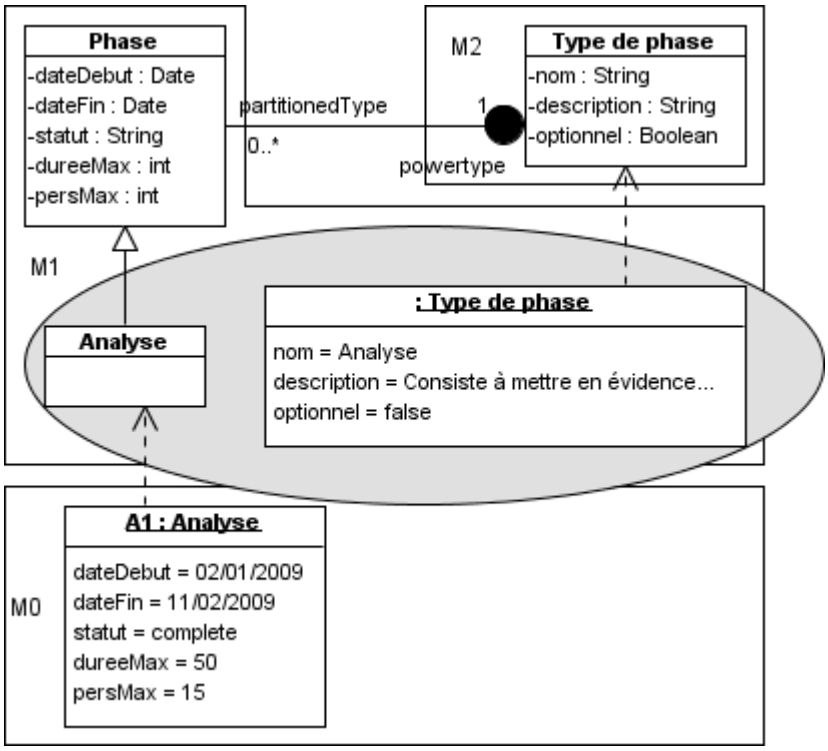


Figure 2-7. Exemple d'imitation du patron Powertype.

La Figure 2-8 présente un autre exemple d'imitation du patron Powertype pour les processus d'ingénierie de SI. Nous utilisons les mêmes attributs et classes que l'imitation du patron ItemDescription de la Figure 2-3. Comme le spécifie le Powertype, il faut instancier les attributs du powertype dans le clabject. Le premier problème qui se pose est l'instanciation des attributs *dureeMax* et *maxPers*. Nous souhaitons que ces attributs soient instanciés au niveau M0, c'est-à-dire au niveau des projets pour qu'à l'intérieur d'un projet il soit possible, par exemple, de définir que la durée maximale des phases sera de 50 jours et qu'au maximum 50 personnes seront affectées à une phase. Ceci n'est pas possible avec cette configuration, car il faudrait refaire un nouveau modèle de processus (M1) lorsque la durée maximale d'une phase change d'un projet à l'autre : un modèle de processus où la durée maximale d'une phase serait de 50 jours et un autre modèle de processus où cette durée serait fixée à 40 jours, par exemple.

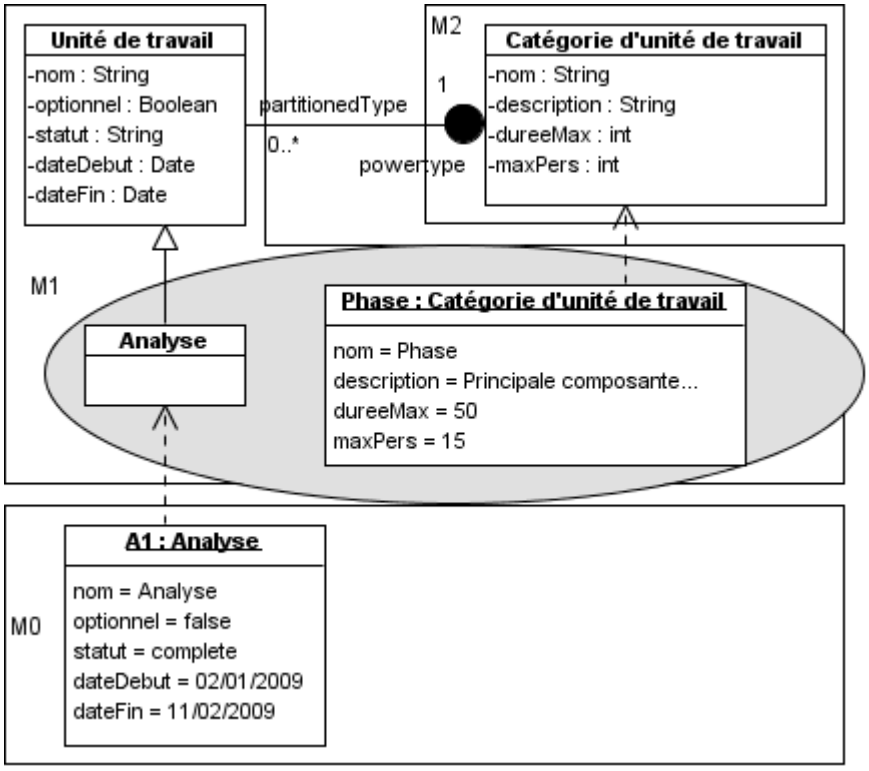


Figure 2-8. Exemple insatisfaisant d'imitation du patron Powertype.

Avec le patron Powertype, nous ne pouvons utiliser les concepts Unités de travail et Catégorie d'unité de travail. Il faut spécifier dès le méta-modèle que des objets pourront être des phases. Or, nous voulons laisser la liberté aux ingénieurs des méthodes de choisir les termes les plus adéquats dans leurs modèles de processus, car en particulier, *Phase* et *Type de phase* nous paraissent être des concepts trop précis pour être définis au niveau des méta-modèles.

D'autre part, l'instanciation des attributs n'est pas aussi flexible que ce que propose l'Instanciation Profonde. Les attributs sont instanciés au niveau de méta-modélisation strictement inférieur. Afin de respecter le principe de la méta-modélisation stricte d'UML, la structure de Powertype et Materialization en est complexifiée : séparation graphique du clabject en une classe et un objet et séparation du type partitionné et du powertype effacée au niveau M0. De plus, si l'on souhaite valuer des attributs au niveau M1, il faudra les placer dans le powertype (ou *AbstractClass*), et réciproquement, si l'on souhaite valuer des attributs au niveau M0, il faudra les placer dans le type partitionné (*ConcreteClass*). La catégorisation perd alors tout son sens puisque les attributs ne seront plus affectés à leur classe d'origine, mais en fonction du niveau où l'on souhaite les instancier.

Les concepts de Powertype et Materialization ne permettent donc pas de répondre avec satisfaction à nos besoins de catégorisation et de méta-modélisation. Ils permettent de catégoriser les unités de travail, mais de manière trop spécifique. Nous souhaitons définir Unités de travail et Catégorie d'unité de travail au niveau des méta-modèles (M2), ce qui n'est pas possible ici.

2.2.4. Synthèse

Dans cette section, nous avons étudié un certain nombre de solutions génériques pour les produits selon leur capacité à aider la méta-modélisation des processus d'ingénierie de SI. Cependant, nous constatons qu'aucune de ces solutions ne répond entièrement à nos besoins, comme présenté dans le Tableau 2-5.

Patrons / Besoins	Item-Description et Object Type	Instanciation Profonde	Powertype et Materialization
Catégorisation	✓		✓
Valuation d'attributs au niveau de modélisation souhaité		✓	

Tableau 2-5. Synthèse.

Les patrons Item-Description et Object Type permettent de séparer des éléments de leur catégorie. Les propriétés communes partagées par tous les éléments sont factorisées dans la catégorie correspondante. Ces patrons sont idéaux pour représenter des unités de travail et des catégories d'unités de travail, de même que des produits et des catégories de produits ou des rôles et des catégories de rôle. Cependant, les deux patrons ont été introduits pour la modélisation et non pour la méta-modélisation. Ils ne sont donc pas suffisants tels quels.

L'Instanciation Profonde permet de définir des méta-classes, classes et objets ainsi que leurs attributs qui peuvent être instanciés à n'importe quel niveau de modélisation (M2, M1, M0). Ce concept est un bon candidat pour la méta-modélisation des processus où il est essentiel de pouvoir définir des classes générales dans les méta-modèles, avec des attributs qui pourront être valués dans les modèles de processus pour définir des propriétés générales à tous les projets d'ingénierie d'une organisation, et à l'exécution des processus pour représenter des propriétés spécifiques à chaque projet. Cependant, l'Instanciation Profonde ne propose aucun mécanisme pour la catégorisation.

Enfin, Powertype et Materialization permettent de catégoriser des éléments. Cependant, ils ne permettent pas de valuer les attributs au niveau de modélisation souhaité : les attributs de la « catégorie » sont obligatoirement instanciés dans le clabject, c'est-à-dire au niveau des modèles de processus (M1), et les attributs des types partitionnés sont instanciés au niveau de l'exécution des processus (M0). Il n'est pas possible de choisir où les attributs seront instanciés. De plus, le formalisme des clabjects mêlant héritage et instanciation n'est pas évident à comprendre, alors que le formalisme utilisé dans l'Instanciation Profonde est beaucoup plus clair. Enfin, ces concepts ne permettent pas de nommer les méta-classes comme nous le souhaitons : il ne serait pas possible d'utiliser Unité de travail et Catégorie d'unité de travail par exemple. Nous devrions alors utiliser Phase et Type de phase, Activité et Type d'activité, etc., ce que nous voulons éviter, ces termes étant trop précis pour être définis au niveau des méta-modèles.

Suite à ces constatations, nous avons proposé le patron Concept – Catégorie de concept que nous présentons dans le Chapitre 4. Le patron Concept-Catégorie de concepts permet de catégoriser des concepts et de valuer leurs attributs au niveau de modélisation souhaité.

2.3. Conclusion

Dans ce chapitre, nous avons décrit les différents niveaux de modélisation de l'OMG et comment la modélisation des processus d'ingénierie des systèmes d'information s'y intégrait.

Nous avons ensuite présenté l'approche de l'ingénierie des méthodes situationnelles, basées sur des fragments de méthodes ou chunk. Les fragments sont similaires à des patrons de domaine, ils apportent des solutions produit ou processus prêtes à l'emploi pour des problèmes métier spécifiques. Les SME ne concernent pas le champ de notre recherche, néanmoins, l'utilisation des patrons de domaine est très utile pour la méta-modélisation des processus d'ISI. Dans cette thèse, les patrons de domaine produit les plus utilisés seront des fragments de méta-modèles de processus existant

Dans un deuxième temps, nous avons présenté des patrons génériques utiles pour la méta-modélisation des processus d'ingénierie de SI. Ces patrons génériques seront également utilisés dans notre méthode que nous présentons dans le Chapitre 5.

3. MODELES ET META-MODELES DE PROCESSUS

Dans ce chapitre, nous présentons les différents types de méta-modèles de processus existants. Chaque type représente un point de vue du processus d'ingénierie de SI. Pour chaque point de vue, nous présentons un exemple de modèle de processus dans un formalisme adéquat, puis nous décrivons l'un des méta-modèles existants sous la forme d'un diagramme de classes UML. Ensuite, nous montrons le même exemple sous forme d'un diagramme objets en UML, instance du méta-modèle présenté. Enfin, nous présentons une synthèse des concepts utilisés. La Figure 3-1 présente l'ordre chronologique dans lequel les différents types de modèles de processus ont été introduits. Les nouveaux modèles de processus ne remplacent pas les anciens ; les méthodes agiles par exemple, apparues dans les années 2000, sont basées sur des modèles de processus orientés activité, introduits dans les années 70.

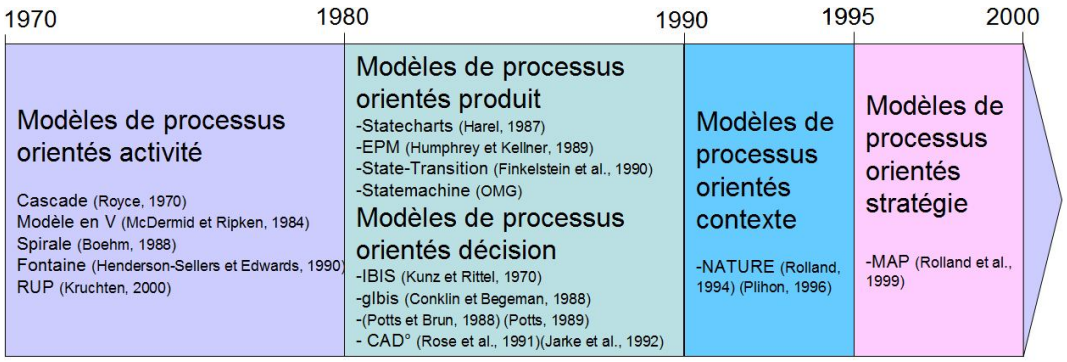


Figure 3-1. Les différents types de modèles de processus.

3.1. Les modèles et méta-modèles orientés activité

Les modèles de processus orientés activité représentent les activités et leur ordonnancement pour la réalisation d'un produit (Rolland, 2005).

3.1.1. Exemples de modèles de processus orientés activité

Les premiers modèles de processus orientés activité sont apparus dans les années 70 avec les modèles dits linéaires comme le modèle de la Cascade (Royce, 1970) et le modèle en V (Mc Dermid et Ripken, 1984), puis les modèles ont évolué vers l'itératif comme le modèle de la Spirale (Boehm, 1988) et le modèle de la Fontaine (Henderson-Sellers et Edwards, 1990). Par la suite, les modèles de processus itératifs et incrémentaux ont été introduits avec le Rapid Application Development (Martin, 1991) puis avec les processus unifiés tels que le Rational Unified Process (Kruchten, 2000), 2 Track Unified Process (Roques et Vallée, 2000) ou Symphony (Hassine, 2005). Les processus des nouvelles méthodes dites agiles sont également orientés activité : XP (Beck, 1999) et SCRUM (Schwaber et Beedle, 2001) par exemple.

La Figure 3-2 présente le modèle de processus orienté activité de la Cascade. Ce modèle est composé de sept phases qui s'enchainent, sans retour en arrière possible, d'où la dénomination de modèle linéaire.

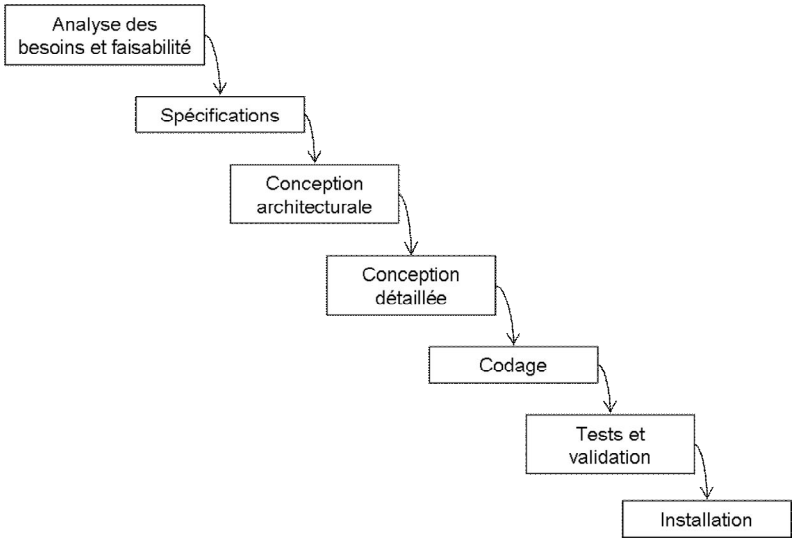


Figure 3-2. Modèle de processus orienté activité de la Cascade.

La Figure 3-3 représente le modèle de processus orienté activité de la méthode eXtreme Programming (XP, 2006). Ce modèle est composé d'étapes courtes. Le modèle est réitéré tant que le client délivre des scénarios utilisateurs (User Stories).

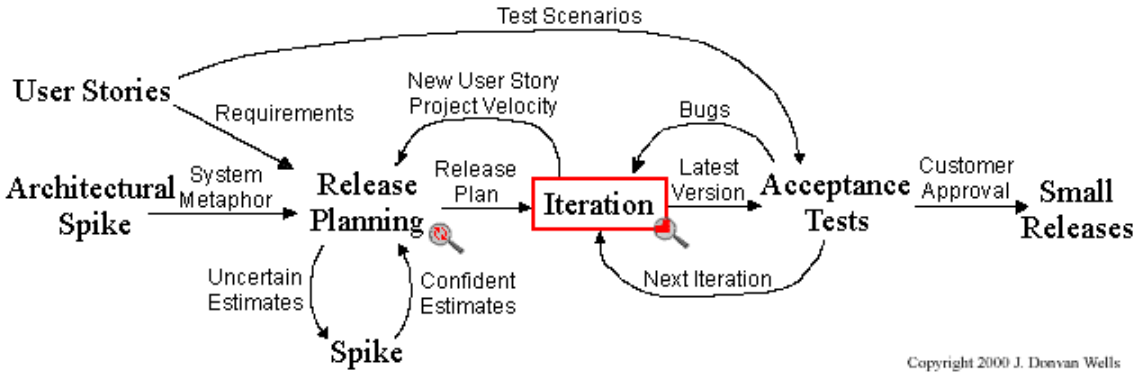


Figure 3-3. Modèle de processus orienté activité de la méthode XP.

3.1.2. Méta-modèles de processus orientés activité

Il existe de nombreux méta-modèles de processus orientés activité tels que OOSPICE[1] (OOSPICE, 2002), OPEN[2] Process Framework (OPF, 2005), la norme australienne Standard Metamodel for Software Development Methodologies (AS, 2004), la norme ISO 24744 (ISO/IEC, 2007) ainsi que la norme de l'OMG, SPEM 1.1 (OMG, 2005) et sa nouvelle version SPEM 2.0 (OMG, 2008). Nous détaillons dans cette section le méta-modèle de l'OPF, ainsi que les versions 1.1 et 2.0 de SPEM.

a) L'OPEN Process Framework

La Figure 3-4 présente le méta-modèle de processus de l'OPEN Process Framework (OPF, 2005). L'extrait présenté ne montre que les principales classes qui sont abstraites. Chacune de ces classes est spécialisée pour raffiner le concept. Une opération (*Endeavor*) représente une collaboration entre producteurs (*Producer*) qui réalisent des

[1] Software Process Improvement and Capability dEtermination for Object Oriented
[2] Object-oriented Process, Environment, and Notation

unités de travail (*Work Unit*) en un temps imparti que l'on peut traduire par « étape » (*Stage*). Les unités de travail sont de quatre types : activité (*Activity*), tâche (*Task*), *Work Flow* et *Technique*. La tâche est l'unité de travail de base qui peut être réalisée par un ou plusieurs producteurs. Une activité est composée de plusieurs tâches pour réaliser plusieurs produits, alors que le *Work Flow* est composé de plusieurs tâches pour ne produire qu'un seul produit. Une technique modélise la façon dont une unité de travail peut être réalisée. Les producteurs produisent des produits (*Work Product*) qui sont utilisés par les unités de travail. Un produit est documenté ou implémenté dans un langage (*Language*), il existe de nombreux types de produit comme des composants, diagrammes et patrons notamment. Deux types de producteurs sont définis : les producteurs directs qui sont des personnes ou des outils, et des producteurs indirects qui sont des rôles, des équipes ou des organisations. Enfin, le méta-modèle de l'OPF distingue les unités de travail, du temps qu'il faut pour les réaliser (les « étapes »). Il existe deux types d'« étape » : les étapes avec durée et les étapes sans durée. Les étapes avec durée peuvent être des cycles (*Cycle*), des phases (*Phase*) ou des itérations (*Build*). Les étapes sans durée représentent des bornes (*Milestone*) ou des micro-jalons (*Inch Pebble*) qui découpent les processus en très petits éléments, comme la méthode XP.

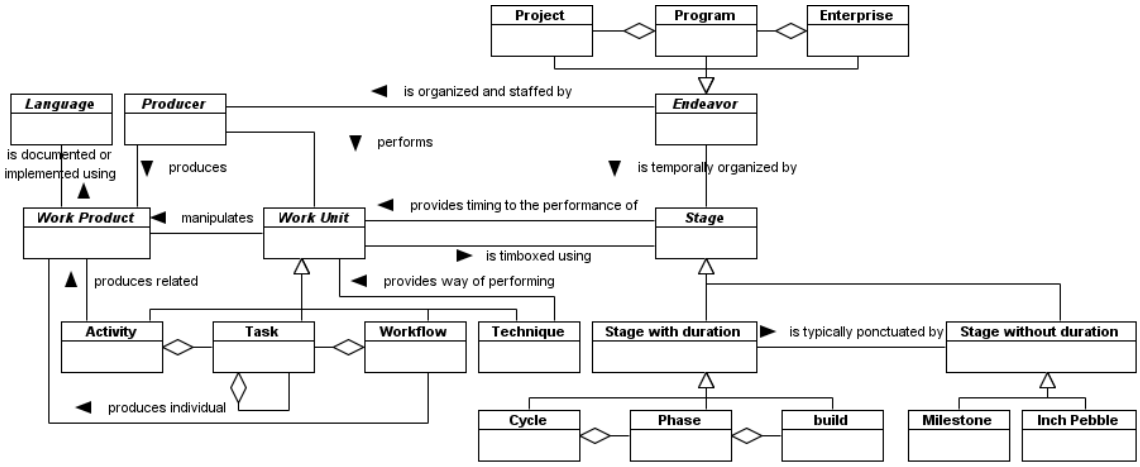


Figure 3-4. Méta-modèle de processus orienté activité de l'OPF.

Le méta-modèle de l'OPF peut être utilisé pour modéliser tous les modèles de processus orientés activité, à des niveaux plus ou moins détaillés. Par exemple, le modèle de processus de la Figure 3-2 représente une séquence de phases (sous-classe de *Stage*) organisées selon le cycle de vie de la Cascade (*Cycle* dans l'OPF). D'autre part, l'OPF ne préconise aucun formalisme pour la représentation des modèles, il est compatible avec des formalismes tels que UML ou OML[1] (OPF, 2005).

Le méta-modèle de l'OPF a pour avantage d'être facilement compréhensible. Grâce à une base de composants de processus et de méthodes (*Repositories*), il est possible d'enrichir le méta-modèle, voir Figure 3-5. L'héritage entre les classes du méta-modèle et les sous-classes utilise le mécanisme du Powertype, présenté dans le Chapitre 2. Le niveau des méthodes permet de réutiliser et de personnaliser des composants de méthode ou d'intégrer des composants de processus afin de spécifier les modèles de

[1] Open Modeling Language

processus. Le dernier niveau instancie le modèle de processus pour représenter l'exécution de ce processus.

Les inconvénients principaux de ce méta-modèle concernent d'une part le fait que la limite entre chaque niveau de modélisation (méta-modèle/modèle) n'est pas claire dans cette architecture. De plus, les sous-classes ajoutées au méta-modèle peuvent être très nombreuses, le méta-modèle simple à la base peut alors devenir très riche et donc moins lisible. Par exemple, l'OPF propose une dizaine de sous-classes pour *Endeavor* et *Language*, mais jusqu'à 300 sous-classes pour *Work Unit*. Le méta-modèle contiendrait alors des concepts qui ne nous semblent pas convenir au niveau des méta-modèles, mais qui devraient être spécifiés au niveau des modèles de processus. Par exemple, pour les rôles, l'OPF va jusqu'à définir tous les types de rôles internes comme architecte métier, ingénieur qualité, etc.

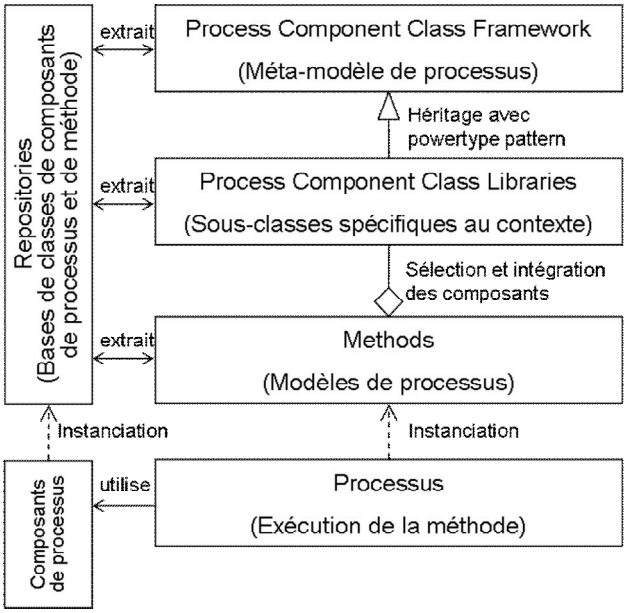


Figure 3-5. Structure globale de l'OPEN Process Framework.

b) SPEM 1.1

Le méta-modèle de processus SPEM de l'Object Management Group a été introduit en 2002 (OMG, 2002). Le méta-modèle de SPEM a été modifié en profondeur entre la version 1.1 (OMG, 2005) et la version 2.0 (OMG, 2008). À notre sens, ces deux versions représentent deux méta-modèles de processus différents. C'est pourquoi nous présentons d'abord la version SPEM 1.1 puis la version SPEM 2.0.

La Figure 3-6 représente le modèle conceptuel de SPEM 1.1 défini dans (OMG, 2005). Ce modèle comprend les trois principaux concepts du méta-modèle de SPEM 1.1 : un rôle (*Role*) est responsable d'un produit (*WorkProduct*) et réalise des activités (*Activity*) qui utilisent et produisent des produits. Ce modèle conceptuel permet de comprendre le principe général de SPEM 1.1, car le méta-modèle est très riche.

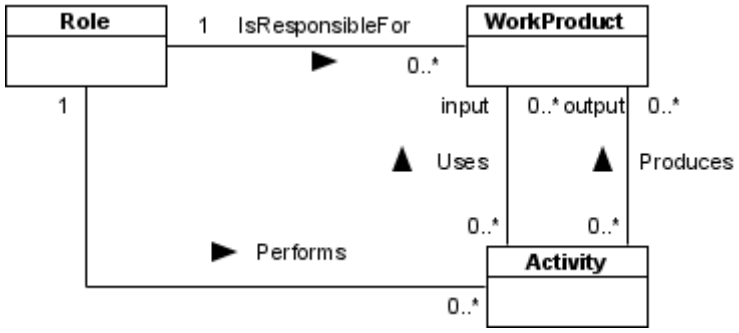


Figure 3-6. Modèle conceptuel de SPEM 1.1.

Le méta-modèle complet de SPEM 1.1 est montré à la Figure 3-7. La classe représentant les unités de travail est *WorkDefinition*, ses sous-classes sont activité (*Activity*), cycle de vie (*Lifecycle*), *Iteration* et *Phase*. Une unité de travail peut être décomposée en sous-unités de travail (association réflexive *parentwork/subwork*). Une unité de travail peut avoir des contraintes en entrée (*Precondition*) ou en sortie (*Goal*). Une activité est décomposée en éléments atomiques appelés étape (*Step*). Une unité de travail est réalisée par un exécutant (*ProcessPerformer*), alors qu'une activité est réalisée par un rôle (*ProcessRole*). La notion d'exécutant est plus abstraite que celle de rôle, elle sert lorsque des unités de travail n'ont pas d'exécutant précis. Un rôle peut être responsable de produits (*WorkProduct*), ce qu'un exécutant ne peut pas. Un type de produit (*WorkProductKind*) définit une catégorie de produit comme du texte, un diagramme ou un exécutable, par exemple. La classe *ActivityParameter* permet de préciser qu'un produit se trouve en entrée ou en sortie d'une unité de travail, il est cependant étonnant que cette classe n'ait pas été liée aux classes concernées dans le méta-modèle, comme le précise d'ailleurs le modèle conceptuel.

Enfin, un composant de processus (*ProcessComponent*) représente un élément de processus indépendant d'autres éléments qui pourra être assemblé avec d'autres composants de processus. Un composant de processus est couramment appelé « chunk » ou « fragment » dans le domaine de l'ingénierie des méthodes situationnelles. Un processus (*Process*) se distingue d'un composant de processus, car il n'a pas besoin d'être assemblé à d'autres processus pour avoir un sens, il représente le processus modélisé dans son ensemble, c'est la « racine » du modèle de processus. Il est associé à un cycle de vie (*Lifecycle*). Par exemple, le modèle de processus du RUP a un cycle de vie dit itératif et incrémental. *Discipline* est un élément qui catégorise des activités selon un thème. Dans le RUP, toutes les activités sont classées selon des disciplines : analyse et conception, déploiement… par exemple.

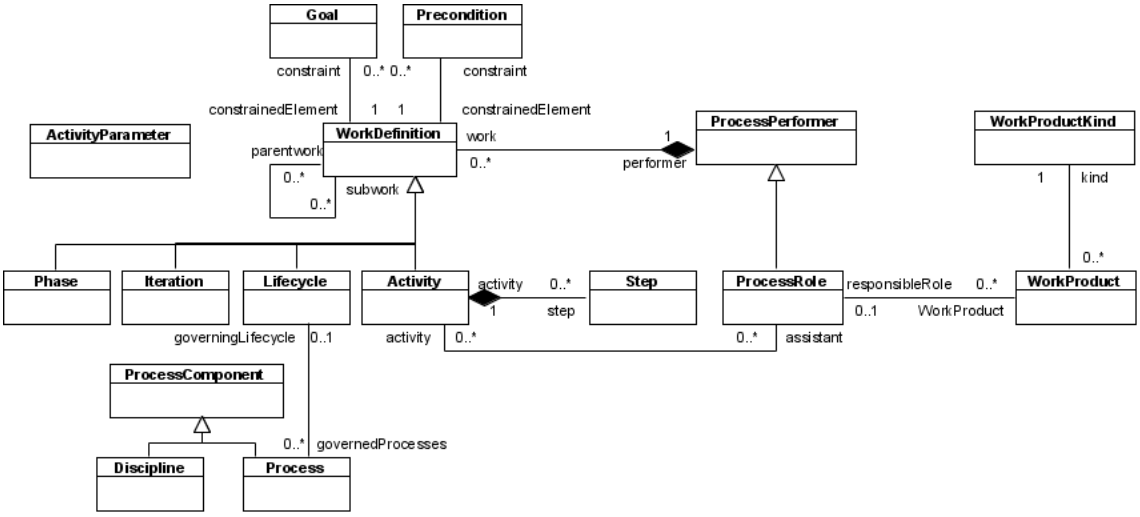


Figure 3-7. Méta-modèle de processus orienté activité de SPEM 1.1.

SPEM 1.1 et l'OPF manipulent les mêmes concepts généraux : unité de travail, produit et rôle. Les associations entre les concepts sont similaires d'un méta-modèle à l'autre, cependant, SPEM 1.1 distingue, via la classe *ActivityParameter*, les produits qui sont en entrée ou en sortie (par l'attribut *Kind* qui n'apparaît pas sur la figure). L'OPF ne distingue pas les produits en entrée/sortie, tout est modélisé par l'association *manipulates* (Henderson-Sellers et Gonzalez-Perez, 2005).

D'autre part, le point de différence important entre le méta-modèle SPEM 1.1 et l'OPF se situe au niveau des unités de travail. Pour l'OPF, une unité de travail spécifie l'opération à réaliser, et l'étape (*Stage*) précise le temps imparti pour réaliser cette opération. Dans SPEM 1.1, aucune différenciation n'est faite entre ces concepts : activité et étape (*Step*) sont bien des opérations à réaliser, mais phase, itération et cycle de vie correspondent à une notion temporelle. Ces concepts sont regroupés sous la même super-classe *WorkDefinition* alors qu'ils n'ont pas la même sémantique. Ce reproche adressé à SPEM 1.1 (Henderson-Sellers et Gonzalez-Perez, 2005) est corrigé dans la version 2.0 de SPEM.

c) SPEM 2.0

Selon l'OMG (OMG, 2008), la première version de SPEM étant difficile à comprendre, peu d'industriels ont adopté ce méta-modèle et très peu d'outils proposent l'implémentation de modèles de processus à partir de SPEM 1.1. La nouvelle version de SPEM permet a priori plus de flexibilité, pour une meilleure adaptation des modèles aux contextes et contraintes des organisations. SPEM 2.0 est également aligné à UML 2, en tant que profil. De plus, un outil permet de définir des modèles de processus selon le méta-modèle de SPEM 2.0 : ainsi, Eclipse Process Framework Composer (EPF, 2008) implémente une grande partie du méta-modèle. Cet outil permet de définir tout type de modèle de processus orienté activité. Des bibliothèques pour les méthodes XP et OpenUP sont disponibles. Ceci montre l'engouement pour SPEM 2.0.

Comme nous l'avons dit dans la section précédente, le méta-modèle SPEM a changé en profondeur entre la version 1.1 (OMG, 2005) et la version 2.0 (OMG, 2008). En effet, SPEM 2.0 différencie l'aspect opération de l'aspect temporel. Ces deux notions

forment deux méta-modèles différents présentés sous forme de packages : *Method Content* pour l'aspect opération et *Process Structure* pour l'aspect temporel.

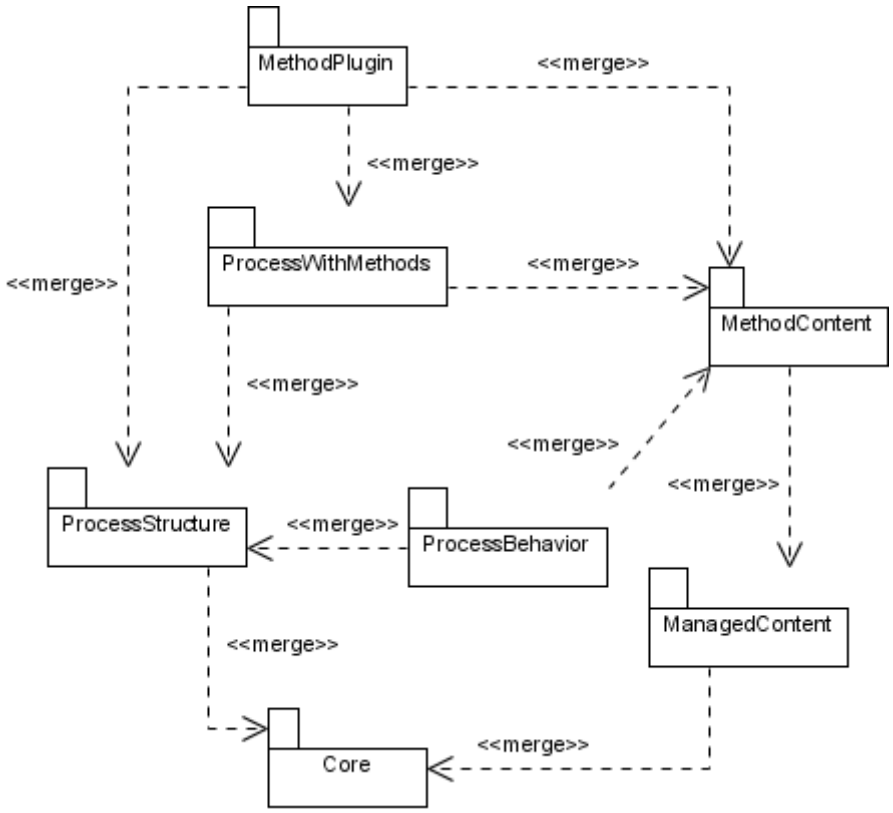


Figure 3-8. Structure du méta-modèle de SPEM 2.0.

Le méta-modèle de SPEM 2.0 est composé de plusieurs packages (cf. Figure 3-8) :

- *Method Content* décrit des rôles qui exécutent des tâches pour réaliser des produits, sans préciser l'enchainement des tâches et leur place précise dans un processus.
- *Process Structure* définit la structure des processus, c'est-à-dire l'ordonnancement des activités et la composition des structures des processus, pour définir par exemple qu'un élément de type cycle de vie est composé d'éléments de type phase, eux- mêmes composés d'activités.
- *Core* contient les classes communes définies dans les différents packages. Ce package contient plusieurs éléments importants :
  - *Kind* : permet aux utilisateurs de SPEM de préciser des termes spécifiques à leur environnement. Par exemple, *Phase* peut être définie comme un type (*Kind*) de *Breakdown Element* (classe définie dans le package *Process Structure*), de même que *Iteration* ou *Increment* pour spécifier différents types d'éléments de processus.
  - les trois concepts clés de SPEM : Unité de travail (*WorkDefinition*) a des produits en paramètres (*WorkDefinitionParameter*) et est réalisée par des rôles (*WorkDefinitionPerformer*). Ces concepts sont spécialisés dans les packages *Process Structure* et *Method Content*.

- *Managed Content* permet aux utilisateurs de SPEM de décrire leurs processus en langue naturelle, en gardant un lien entre le processus modélisé et la description.
- *Process Behavior* établit des liens avec le méta-modèle UML 2 pour définir les processus sous forme de diagrammes d'activités ou les produits sous forme de diagrammes états-transitions. Ce mécanisme est autorisé, car SPEM 2.0 est défini comme un profil d'UML 2.0. Ce package permet également d'utiliser le formalisme des diagrammes BPMN[1] proposés pour la modélisation des processus métier conformément au méta-modèle BPDM[2].
- *Process with Methods* spécifie les liens entre les éléments des packages *Method Content* et *Process Structure*, pour réutiliser les opérations à réaliser dans les structures temporelles spécifiques au processus défini.
- *Method Plugin* permet d'étendre et de personnaliser des instances de *Method Content* et *Process Structure* sans les modifier directement, mais grâce à un système d'extension, le plugin en lui-même. Ceci permet aux utilisateurs de réutiliser des plugins ou des configurations déjà définies, tout en pouvant les étendre et les personnaliser. Par exemple, dans l'outil EPF Composer, des librairies ont été définies pour les méthodes XP et OpenUP qui est un processus de développement agile proposé dans le projet Eclipse Process Framework (EPF, 2008).

La particularité de SPEM 2.0 est que le package *Method Content* définit les opérations à réaliser et le package *Process Structure* permet de définir la structure temporelle du processus. *Process with Methods* permet de réutiliser les opérations en les insérant dans le processus.

Dans les deux sous-parties qui suivent, nous détaillons les deux principaux packages *Method Content* et *Process Structure*.

Method Content

Le package *Method Content* décrit les opérations à réaliser, les produits créés durant ces opérations et les acteurs intervenants. Nous pouvons comparer les modèles créés avec *Method Content* à des composants de méthodes qui seront ensuite assemblés dans une structure temporelle via le package *Process Structure*.

Le package est représenté par la Figure 3-9. Les opérations à réaliser sont décrites dans la classe Tâche (*TaskDefinition*), elles peuvent être décomposées en étapes (*Step*) afin de décrire pas à pas comment atteindre l'objectif défini par la tâche. Aucune notion de temps n'est attachée à une tâche, ceci sera défini grâce au package *Process Structure*. Pour chaque tâche, on peut définir des produits (*WorkProductDefinition*) en paramètres, optionnels ou non. De la même manière, des rôles (*RoleDefinition*) sont associés à une tâche. Il est possible de décrire des qualifications nécessaires pour la réalisation d'une tâche ou les qualifications dont un rôle dispose, ceci dans le but de faire correspondre au mieux les rôles aux tâches à réaliser, en fonction des compétences requises. Il est également possible de définir les outils (*ToolDefinition*) qui peuvent être utilisés pour

[1] Business Process Modeling Notation
[2] Business Process Definition Metamodel

manipuler des produits. Des produits peuvent avoir des relations entre eux, par exemple la composition ou l'agrégation de produits, ce que modélise la classe *WorkProductDefinitionRelationship*. Enfin, un rôle peut avoir une responsabilité sur un produit (*Default_ResponsabilityAssignment*), ces responsabilités peuvent être par exemple : responsable, pour signature, pour consultation.

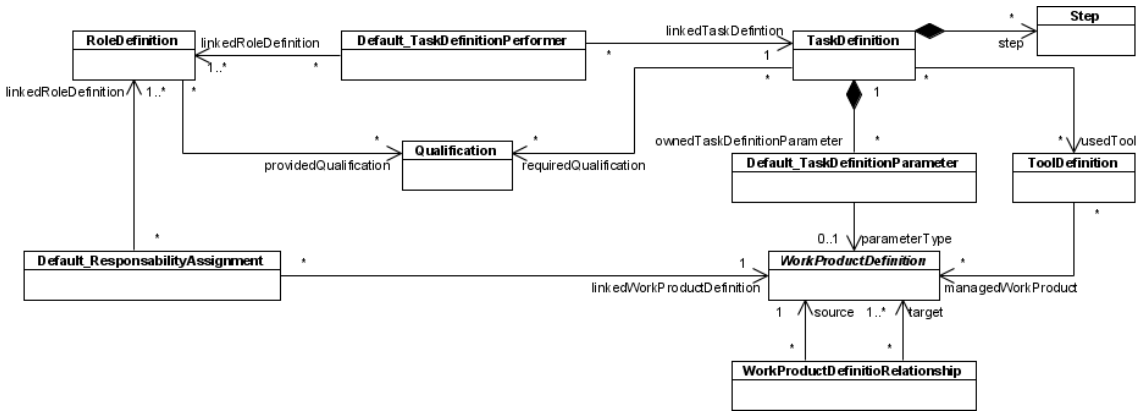


Figure 3-9. Méta-modèle de processus orienté activité de SPEM 2.0 – Method Content.

La Figure 3-10 présente un exemple de modèle de processus orienté activité instancié à partir du package *Method Content* du méta-modèle SPEM 2.0. Ce modèle est un extrait de la méthode OpenUP. Ce modèle présente les rôles et produits impliqués dans la tâche d'identification et de résumé des besoins. Cette tâche est exécutée par un rôle principal, *Analyst* et des rôles secondaires : *Architect*, *Tester*, *Developer* et *Stakeholder*. Tous les produits en entrée sont optionnels : *Glossary*, *System Wide Requirements*, *Use Case*, *Use Case Model* et *Vision*. Les produits en sortie sont *Use Case Model*, *Use Case* et *System Wide Requirements et Glossary*.

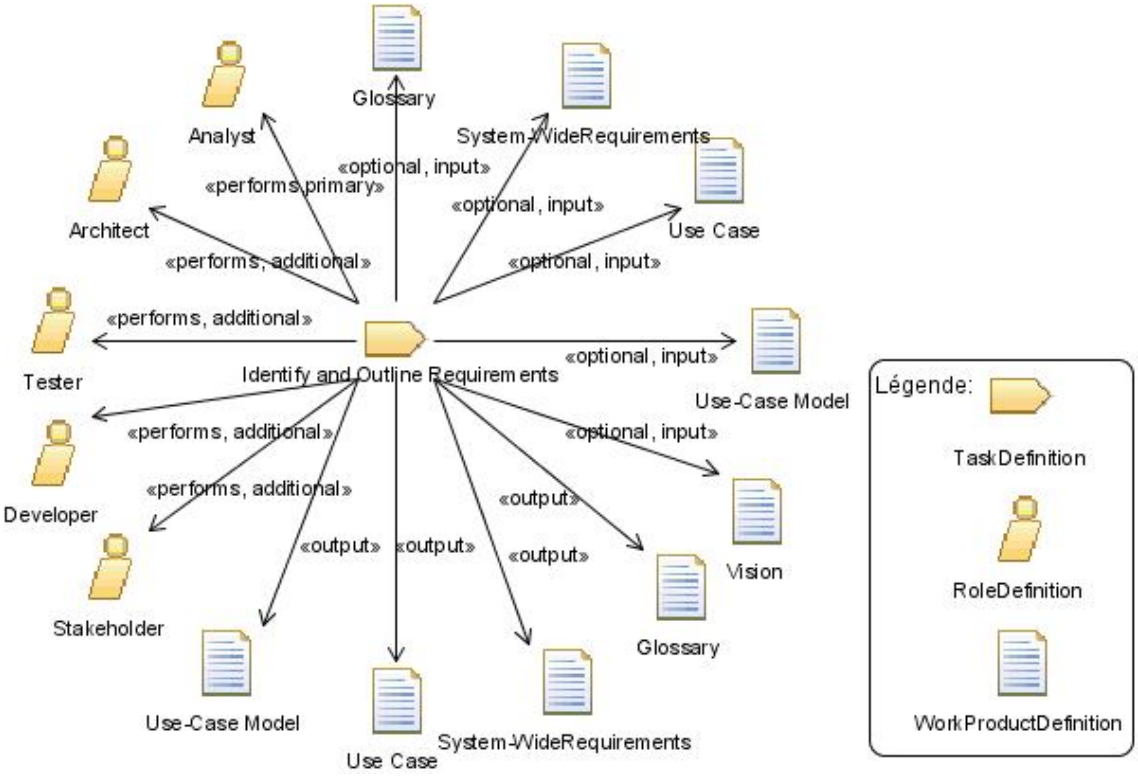


Figure 3-10. Exemple de modèle de processus orienté activité, instance du package Method Content de SPEM 2.0.

La Figure 3-11 montre le diagramme objets, instance du méta-modèle *Method Content* de SPEM 2.0, représentant un extrait du modèle de la Figure 3-10. *Analyst* est une instance de la classe *RoleDefinition*, *Identify and Outline Requirements* est une instance de la classe *TaskDefinition*, et *Use Case Model* est une instance de la classe *WorkProductDefinition*. *Performs* est instance de la classe *defaut_TaskDefinitionPerformer*, attaché à *Primary*, instance de *Kind* pour préciser que l'analyste est l'exécutant principal de la tâche. L'objet anonyme instance de la classe *Defaut_TaskDefinitionParameter* spécifie les propriétés du produit en paramètre *Use Case Model*, ici ce paramètre est optionnel et est en entrée de la tâche.

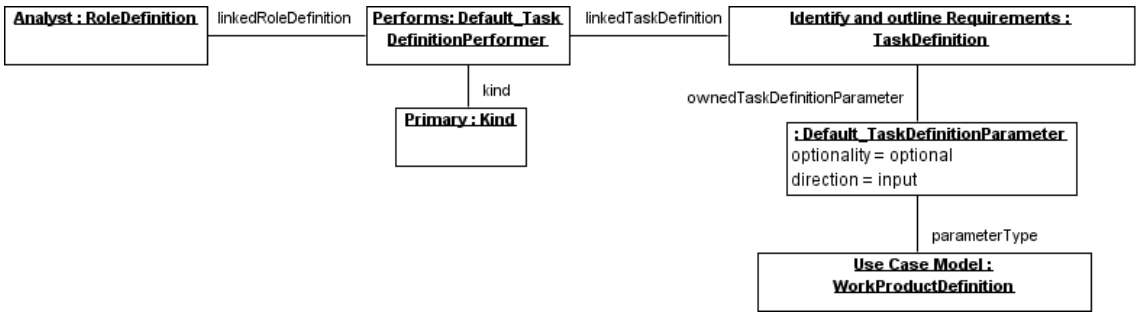


Figure 3-11. Diagramme objets instance du package Method Content du méta-modèle SPEM 2.0.

Process Structure

Le package *Process Structure* décrit la structure des processus, c'est-à-dire comment un processus est organisé : les éléments (*BreakdownElement*) sont décomposés

jusqu'aux activités (*Activity*) qui comprennent des produits (*WorkProductUse*) et des rôles (*RoleUse*). Les éléments faisant référence à des unités de travail (*WorkBreakdownElement*) peuvent être définis en séquence ou en parallèle (*WorkSequence*) (cf. Figure 3-12).

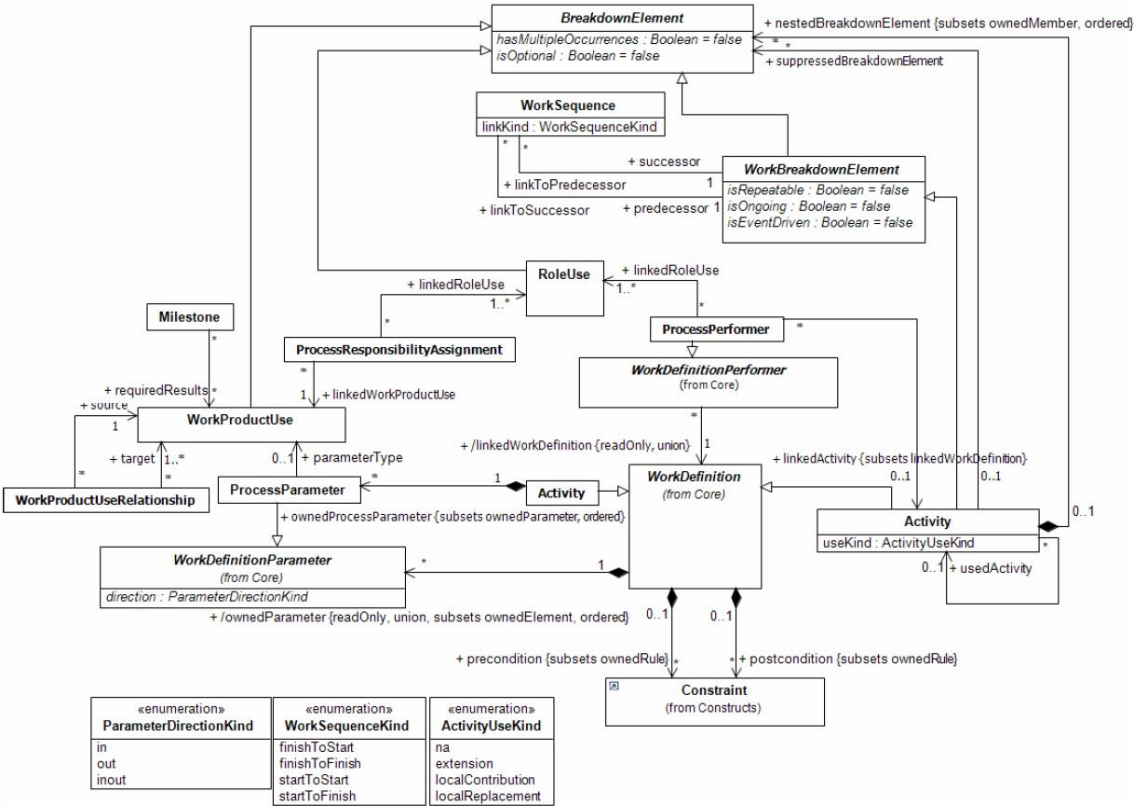


Figure 3-12. Méta-modèle de processus orienté activité de SPEM 2.0 – Process Structure.

Une unité de travail (*WorkDefinition*) est exécutée par un exécutant (*ProcessPerformer*). Une unité de travail peut avoir des pré-conditions et des post-conditions (*Constraint*). La classe *Activity* est une unité de travail concrète, elle admet des produits (*WorkProductUse*) en paramètres (*ProcessParameter*). L'attribut *direction* fait référence à la classe énumérée *ParameterDirectionKind* qui liste les types de direction des paramètres. *WorkProductUse* est la correspondance de la classe *WorkProductDefinition* dans *Method Content*. Comme dans le méta-modèle de *Method Content*, un rôle a des responsabilités sur un produit (*ProcessResponsabilityAssignment*), et un produit peut avoir des relations avec d'autres produits (*WorkProductUseRelationship*). Une borne (*Milestone*) modélise un évènement significatif dans le processus qui se rapporte souvent à un produit, par exemple, le rendu d'un livrable.

La classe abstraite centrale pour définir des structures est *BreakdownElement* qui peut être spécialisée par *WorkProductUse*, *RoleUse* ou *WorkBreakdownElement*. *WorkBreakdownElement* représente une unité de travail comme une activité (*Activity*) ou une borne (*Milestone*) (la spécialisation entre *WorkBreakdownElement* et *Milestone* n'est pas spécifiée sur la figure). Il est possible de définir des séquences entre

WorkBreakdownElement avec la classe *WorkSequence*, l'attribut *linkKind* fait référence à la classe énumérée *WorkSequenceKind* qui définit quatre types de séquence entre deux *WorkBreakdownElement*. Une activité peut contenir des *BreakdownElement*, des activités peuvent donc être elles-mêmes composées d'activités ou de bornes, ou uniquement de produits ou de rôles, ceci dans le but de permettre le maximum de flexibilité. Par exemple, certaines méthodes agiles ne définissent que les activités, les rôles participant à cette activité ainsi que les produits réalisés, sans détailler qui fait quoi. SPEM 2.0 permet de modéliser ces méthodes.

Une activité peut être réutilisée (association *usedActivity*) par une autre activité, ainsi la structure de l'activité source est recopiée dans la nouvelle activité. Cette copie peut être étendue et modifiée. L'attribut *useKind* fait référence au type de réutilisation de l'activité, type précisé dans la classe énumération *ActivityUseKind*.

La Figure 3-13 montre le cycle de vie de OpenUP, défini comme instance du package *Process Structure*. Le cycle de vie est composé de quatre phases en séquence : *Inception*, *Elaboration*, *Construction* et *Transition*.

Figure 3-13. Cycle de vie défini par OpenUP.

La Figure 3-14 présente le raffinement de la phase *Elaboration*. Cette phase est composée de plusieurs activités. Les activités sont composées d'éléments de type *TaskUse* qui réutilisent les éléments de type *TaskDefinition* définis dans la partie *Method Content*. *TaskUse* est une classe définie dans le package *Process With Methods*.

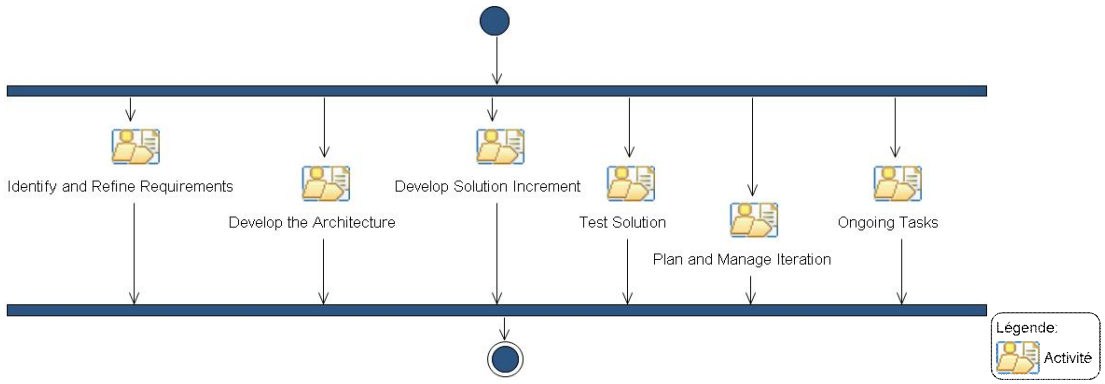


Figure 3-14. Exemple de modèle de processus orienté activité, instance du package Process Structure de SPEM 2.0.

La Figure 3-15 présente un extrait de l'activité *Identify and Refine Requirements* (à gauche sur la Figure 3-14) de la phase *Elaboration* où la tâche *Identify and Outline Requirements* (détaillée dans la Figure 3-10) a été réutilisée. Les propriétés de *Identify and Outline Requirements*, comme les produits ou les rôles, peuvent être identiques à celles définies dans le package *Method Content* ou adaptées. Ici, les produits en sortie (*Glossary*, *System Wide Requirements*, *Use Case*, *Use Case Model*) définis en option dans le package *Method Content* sont obligatoires et le seul rôle intervenant est *Analyst*.

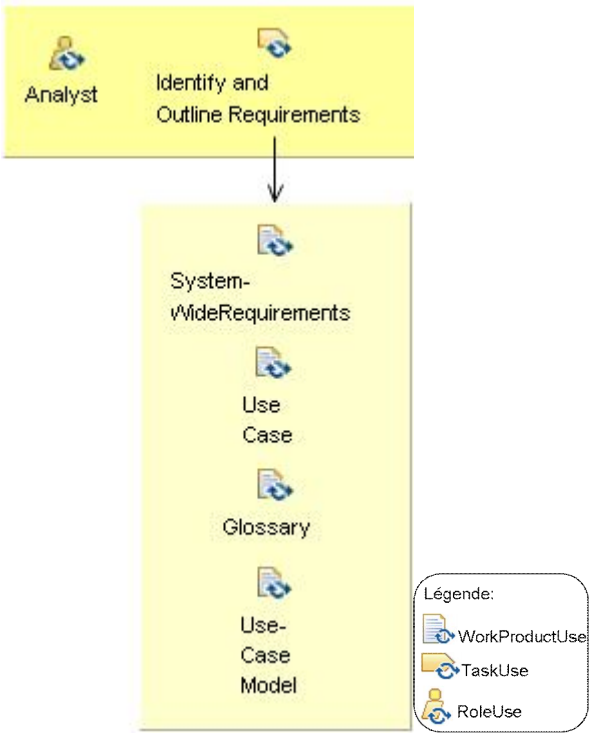


Figure 3-15. Extrait de l'activité Identify and Refine Requirements.

La Figure 3-16 présente le diagramme objets correspondants au modèle de processus présenté ci-dessus. *Analyst* est instance de la classe *RoleUse* qui exécute *Identify and Outline Requirements*, instance de *TaskUse*, ayant en paramètre de sortie *UseCase*, instance de la classe *WorkProductUse*.

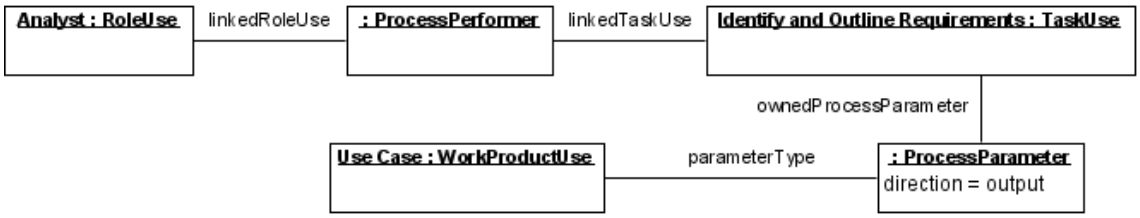


Figure 3-16. Diagramme objets instance du package Process Structure du méta-modèle SPEM 2.0.

3.1.3. Synthèse

Les méta-modèles de processus orientés activité sont beaucoup plus développés, documentés et détaillés que les autres méta-modèles. Ils ont fait l'objet de nombreuses normes.

La différence fondamentale entre l'OPF et SPEM 1.1 est la séparation des unités de travail et du temps affecté à ces unités de travail. Cette différence a disparu avec la version SPEM 2.0 puisque le méta-modèle présente deux packages différents : l'un (*Method Content*) pour la description des tâches, l'autre (*Process Structure*) pour l'ordonnancement, le séquencement des tâches à réaliser, ainsi que leur place dans une structure qui définit le cycle de vie du processus, ses différentes phases et les activités.

SPEM 2.0 et l'OPF ont donc la même vision des aspects activités et temps. Cependant, le méta-modèle de SPEM 2.0 est beaucoup plus complexe, car chaque

package est défini séparément et les concepts de ces packages sont mis en correspondance par d'autres packages. De plus SPEM 2.0 étant lié à UML 2, un package sert à effectuer la liaison entre les méta-modèles. La compréhension de SPEM 2.0 n'est donc pas aussi aisée que celle de l'OPF.

Le formalisme préconisé par SPEM 1.1 et SPEM 2.0 n'est pas le même. Celui de SPEM 2.0 est très proche du formalisme du RUP (Kruchten, 2000). L'OPF ne préconise aucun formalisme.

SPEM 1.1 et 2.0 peuvent être étendus, grâce au mécanisme des stéréotypes. SPEM étant défini comme un profil d'UML, les stéréotypes sont autorisés. Cependant, l'utilisation des stéréotypes doit être couplée avec la définition de contraintes OCL pour assurer la cohérence des modèles (OMG, 2008).

Le Tableau 3-1 présente une synthèse des concepts des méta-modèles de processus orientés activité présentés en détail dans cette section : OPF, SPEM 1.1, SPEM 2.0. Une correspondance avec les méta-modèles de processus orientés produit, décision, contexte et stratégie est également proposée. Ces méta-modèles seront détaillés par la suite.

	Orienté activité			Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
	OPF	SPEM 1.1	SPEM 2.0				
Unité de travail	Work Unit	WorkDefinition	WorkDefinition (Core), TaskDefinition, Step (Method Content)	Action	Step	Action	
Type d'unité de travail	Activity, Task, Work Flow, Technique	Activity, Step	Activity, Milestone (Process Structure)				
Unité de temps	Stage	WorkDefinition	WorkBreakdownElement (Process Structure)				
Type d'unité de temps	Phase, Build, Cycle, Milestone, Inch Pebble	Phase, Iteration, Lifecycle	Kind (Core)				
Produit	Work Product	WorkProduct	WorkProductUse (Process Structure), WorkProductDefinition (Method Content)	Entity	Artefact	Partie de produit	
Type de produit	Application, Architecture…	WorkProductKind	Kind (Core)				
Contrainte	Invariant (Precondition, Postcondition)	Constraint (Precondition, Goal)	Constraint	Cons-traint			
Rôle	Producer	ProcessPerformer, ProcessRole	WorkDefinitionPerformer (Core), ProcessPerformer (Process Structure), RoleDefinition (Method Content)				
Type de rôle	Organisation, Team, Role, Person, Tool		Kind (Core)				

Tableau 3-1. Synthèse des concepts des méta-modèles de processus orientés activité.

3.2. Les modèles et méta-modèles orientés produit

Les modèles de processus orientés produit couplent l'état du produit à l'activité qui génère cet état. Ils sont assimilables à des diagrammes états-transitions (Rolland, 2005). L'état d'un produit modélise la situation de celui-ci à un instant donné du processus, des transitions sont définies entre ces états, pour modéliser l'ordre dans lequel les états peuvent changer. Les transitions sont des relations entre un état source et un état cible, elles sont déclenchées par un évènement.

Il n'existe pas de méthode prédéfinie basée sur des modèles de processus orientés produit, contrairement aux méthodes basées sur des modèles de processus orientés activité. Les modèles de processus orientés produit sont créés selon les besoins, lorsque certains produits ont des états spécifiques complexes qui doivent être précisés.

3.2.1. Exemple de modèle de processus orienté produit

La Figure 3-17 présente le modèle de processus orienté produit d'un document lors d'un processus d'ingénierie de SI. Le modèle décrit les différents états d'un document, de sa création à sa destruction, en spécifiant les différents évènements qui déclenchent la transition d'un état vers un autre. Par exemple, lorsque le document est créé, il est dans un état « élaboré »où il reste tant qu'il subit des modifications. Lorsque le document est validé, il passe dans l'état « utilisé », c'est-à-dire qu'il sert de référence pour avancer dans le processus d'ingénierie. Lors de l'évènement « fin du projet », le document passe dans l'état « stocké ». Cinq ans après, il sera détruit.

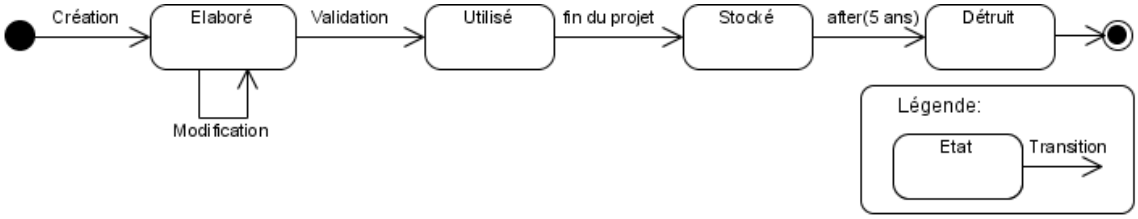


Figure 3-17. Modèle de processus orienté produit d'un document.

3.2.2. Méta-modèles de processus orientés produit

Le méta-modèle des Statecharts (Harel, 1987), des State Machines dans UML (OMG, 2007), ainsi que le méta-modèle du Entity Process Model (EPM) (Humphrey et Kellner, 1989) et le template State-Transition (Finkelstein et al., 1990) sont des exemples de méta-modèles de processus orientés produit. Le méta-modèle orienté activité de SPEM (1.1 et 2.0) étant défini comme un profil UML, il permet également de définir des modèles de processus orientés produit, instances du méta-modèle des State Machines.

La Figure 3-18 représente le méta-modèle simplifié du State Machines (OMG, 2007). Un produit est composé de plusieurs états. Les transitions permettent de définir le passage d'un état à un autre. Plusieurs propriétés sont définies sur les transitions : une transition est déclenchée par un évènement (*Trigger* dans le méta-modèle) ; pour qu'une transition soit déclenchée, une condition doit parfois être observée (*Guard*) ; une transition peut également générer une action (*Behavior*) qui doit être réalisée avant de passer à l'état cible. Enfin, les états peuvent être décomposés en sous-états.

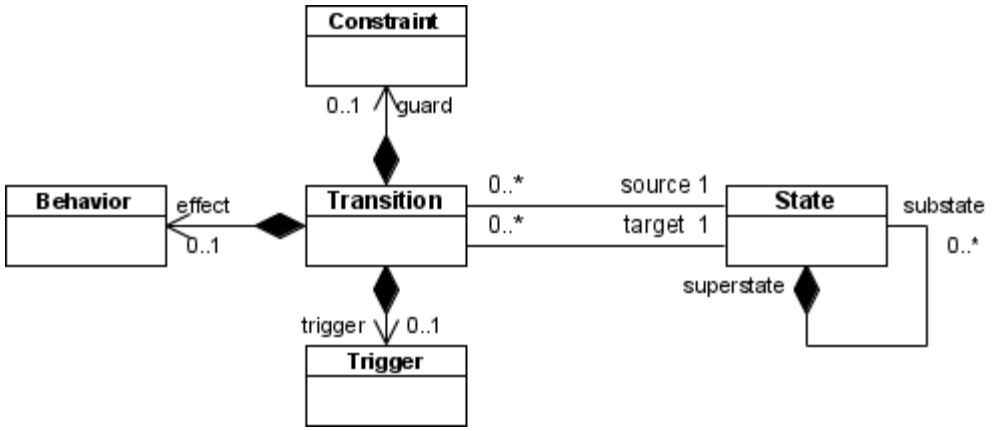


Figure 3-18. Méta-modèle de processus orienté produit simplifié du State Machines.

La Figure 3-19 présente une instance du méta-modèle de processus orienté produit du State Machine sous forme d'un diagramme objets. Ce diagramme objets correspond au modèle présenté dans la Figure 3-17. Chaque objet est instance d'une classe du méta-modèle : par exemple, « Élaboré » est une instance de la classe *State* et « Modification » est une instance de la classe *Transition*.

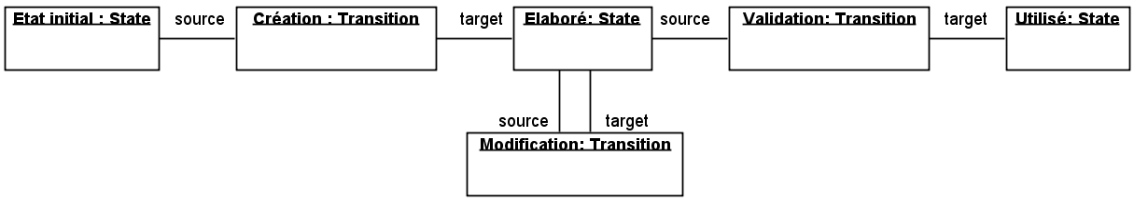


Figure 3-19. Diagramme objets instance du méta-modèle orienté produit du State Machine.

3.2.3. Synthèse

Le Tableau 3-2 présente une synthèse des concepts des méta-modèles de processus orientés produit et leur correspondance avec les autres types de méta-modèle. Le concept de *Produit* est présent dans quasiment tous les méta-modèles de processus, ainsi que le concept d'*Action* que nous assimilons à une unité de travail. Le concept de *Contrainte* existe aussi dans les modèles de processus orientés activité. Les concepts d'*Etat*, *Transition* et *Evènement* sont par contre propres aux méta-modèles de processus orientés produit.

Dans ce tableau de synthèse, nous ne faisons pas apparaître les concepts d'état et transition pour les méta-modèles de processus orientés activité, étant donné que seul SPEM (1.1 et 2.0) permet cela, en bénéficiant du méta-modèle des State Machines défini dans UML, par le mécanisme des profils.

	Orienté activité	Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
Unité de travail	Work Unit, Work definition	Action	Step	Action	
Produit	Work	Composition	Artefact	Partie de	

	Product	des états (implicite)		produit	
Contrainte	Constraint	Constraint			
Etat		State			
Transition		Transition			
Évènement		Trigger			

Tableau 3-2. Synthèse des concepts des méta-modèles de processus orientés produit.

L'utilisation seule de modèles de processus orientés produit pour décrire une méthode d'ingénierie de SI parait difficile, car ce sont les états du produit qui sont mis en avant. Les acteurs du processus s'intéressant principalement aux actions à réaliser, le seul objectif des produits est donc trop abstrait. La méthode du Work Product Pool Approach définie par (Gonzalez-Perez et Henderson-Sellers, 2008) permet de définir dans un premier temps les produits à réaliser, puis de spécifier les actions pour réaliser les produits désirés. Cette méthode montre bien que la seule définition des produits ne suffit pas, l'aspect activité est indispensable. Il est à noter que cette approche est basée sur un méta-modèle de processus orienté activité (ISO/IEC, 2007), et non pas sur un méta-modèle de processus orienté produit (la méthode ne s'occupe pas des états des produits, mais des activités pour réaliser les produits).

D'autre part, (Plihon, 1996) rappelle qu'il est très difficile, voire impossible, de créer des modèles de processus orientés produit suffisamment réalistes et décrivant de la manière la plus complète possible, les états et transitions des produits, ceci étant dû à la nature non déterministe des processus.

En outre, les modèles de processus orientés produit se couplent très bien avec les modèles de processus orientés activité. Il est intéressant de pouvoir suivre le processus tout en contrôlant l'état des produits concernés. SPEM 2.0 (OMG, 2008) y fait d'ailleurs référence en précisant dans le package *Process Behavior* quel doit être l'état d'un produit à l'entrée d'une unité de travail et son état à la sortie. Les visions orientées activité et produit sont donc complémentaires.

Les modèles de processus orientés activité et produit décrivent les étapes du processus d'ingénierie de SI, les transformations des produits, mais ne précisent pas pourquoi ces étapes ou transformations sont réalisées (Rolland et Grosz, 1994). Les modèles de processus orientés décision ont été introduits pour pallier ce manque.

3.3. Les modèles et méta-modèles orientés décision

Les modèles de processus orientés décision présentent les transformations ou les élicitations successives du produit dues à des décisions (Rolland, 2005). Dans les processus d'ingénierie de SI, il est essentiel de savoir ce qui a été fait, ou ce qui doit être fait, mais il faut aussi savoir pourquoi telle action a été faite plutôt qu'une autre. De plus, de nombreux débats et discussions ont lieu entre les différents acteurs d'un processus d'ingénierie de SI, comme les gestionnaires, les futurs utilisateurs, le chef de projet, et les équipes de maitrise d'œuvre. Il faut garder une trace de ces discussions et des décisions qui ont été prises : cela permet d'une part de mieux comprendre les raisons des choix effectués et d'autre part, de pouvoir surveiller si les décisions prises

durant le processus d'ingénierie de SI sont rationnelles (Kunz et Rittel, 1970), (Jarke et al., 1992).

Comme pour les modèles de processus orientés produit, il n'existe pas de méthode spécifiquement basée sur un ou des modèles de processus orientés décision. Dans chaque projet, différents modèles de processus orientés décision sont créés en fonction de la complexité de la situation et des problèmes posés.

3.3.1. Exemple de modèle de processus orienté décision

La Figure 3-20 présente un modèle de processus orienté décision utilisant un formalisme inspiré de (Potts et Bruns, 1988). Ce modèle représente le processus de décision à prendre à l'étape de modélisation des acteurs, concernant le fait de modéliser ou non le contexte et le métier de l'environnement du futur système d'information. Si le processus métier est conséquent et que l'équipe de Maitrise d'Œuvre (MOE) est peu habituée au domaine d'application du futur système d'information, il est conseillé de modéliser le contexte.

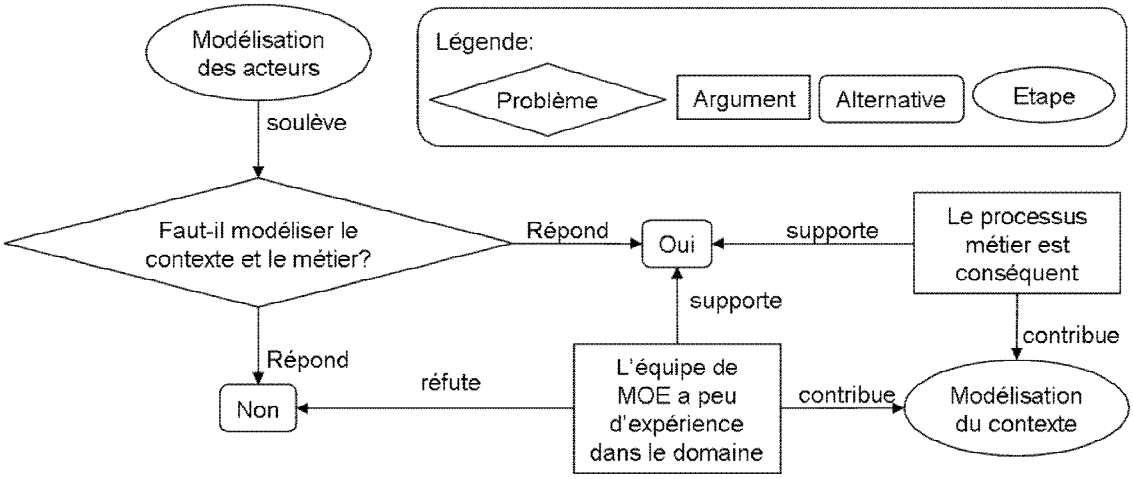


Figure 3-20. Exemple de modèle de processus orienté décision.

3.3.2. Méta-modèles de processus orientés décision

Le modèle Issue Based Information Systems (IBIS) (Kunz et Rittel, 1970) a introduit pour la première fois la notion de processus orienté décision. Ce modèle a été implémenté pour gérer les discussions, les problèmes rencontrés durant les phases amont de la conception de systèmes dans un outil appelé gIbis[8] (Conklin et Begeman, 1988). Il a ensuite été enrichi par (Potts et Bruns, 1988) et Potts (Potts, 1989). Le projet européen ESPRIT DAIDA a également mis au point un modèle de processus orienté décision, CAD°[9] (Rose et al., 1991), (Jarke et al., 1992).

La Figure 3-21 présente le méta-modèle de processus orienté décision de (Potts, 1988), initialisé dans (Potts et Bruns, 1988). Une étape (*Step*) dans le processus d'ingénierie de SI fait émerger des nouveaux problèmes/questions (*Issue*). Différentes alternatives (*Position*) sont habituellement proposées pour répondre à ces problèmes.

[8] graphical IBIS
[9] Conversation among Agents on Decisions over Objects

Chaque alternative doit être soutenue ou contestée par des arguments. Les arguments peuvent se baser sur des artefacts existants. Lorsqu'une alternative est choisie, elle contribuera à l'évolution de l'étape du processus d'ingénierie de SI.

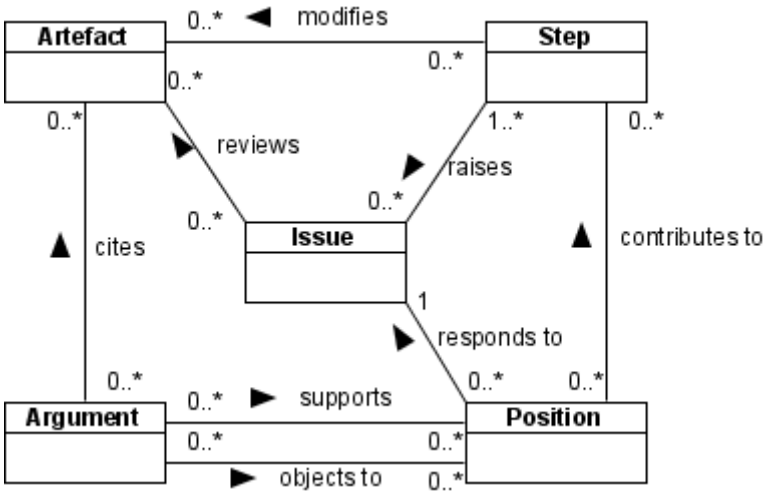


Figure 3-21. Méta-modèle de processus orienté décision de Potts.

La Figure 3-22 représente le diagramme objets, instance du méta-modèle de Potts, reprenant l'exemple du processus de décision de modélisation du contexte lors du processus d'ingénierie de SI. Chaque objet est instance d'une classe définie dans le méta-modèle. Par exemple, « Faut-il modéliser le contexte et le métier » est une instance de la classe *Issue* qui a deux alternatives « oui » et « non ».

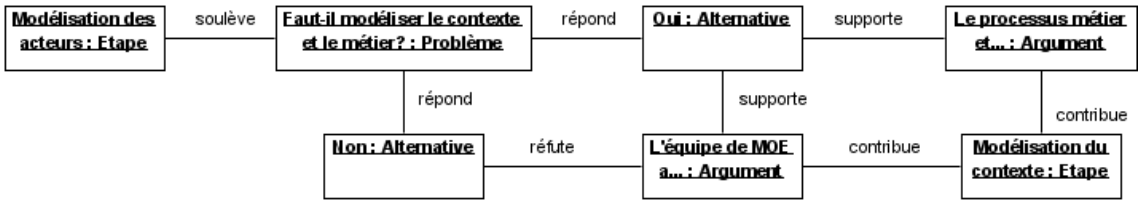


Figure 3-22. Diagramme objets instance du méta-modèle de processus orienté décision de Potts.

3.3.3. Synthèse

Le Tableau 3-3 présente la synthèse des concepts des méta-modèles de processus orientés décision. Les concepts *Step* et *Artefact* correspondent aux concepts d'unité de travail et de produit présents dans la majorité des autres types de méta-modèles de processus. Les concepts d'*Alternative* et d'*Argument* se retrouvent dans les méta-modèles de processus orientés contexte, notamment NATURE, car ces derniers sont issus des méta-modèles de processus orientés décision. Le concept de problème (*Issue*) n'est présent dans aucun autre type de méta-modèle de processus.

	Orienté activité	Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
Unité de travail	Work Unit, Work definition	Action	Step	Action	
Produit	Work Product	Composition des états	Artefact	Partie de produit	
Problème			Issue		
Alternative			Position	Alternative	
Argument			Argument	Argument	

Tableau 3-3. Synthèse des concepts des méta-modèles de processus orientés décision.

Les méta-modèles de processus orientés décision intègrent de nouveaux concepts par rapport aux méta-modèles de processus orientés activité et produit, car ils prennent en compte les raisons pour lesquelles les décisions sont prises et comment elles déclenchent les actions (Plihon, 1996). Les méta-modèles de processus orientés activité et produit ne donnent pas d'information sur l'aspect décisionnel du processus d'ingénierie de systèmes d'information. Ce changement d'abstraction est similaire aux niveaux intentionnels et organisationnels en systèmes d'information.

Cependant, un méta-modèle de processus orienté décision ne permet pas de décomposer le processus en étapes/sous-étapes, toutes les activités sont ici au même niveau.

D'autre part, le méta-modèle ne prend pas en compte les acteurs qui prennent les décisions ou proposent les alternatives. Ceci est pourtant un élément important du processus, d'après nous, le rôle de chaque acteur dans le processus de décision devrait être pris en compte.

Les modèles de processus orientés décision ont par la suite été complétés, pour notamment prendre en compte la situation dans laquelle une décision est prise. Ces nouveaux modèles de processus sont dits orientés contexte (Plihon, 1996) (Souveyet, 2006).

3.4. Les modèles et méta-modèles orientés contexte

Les modèles de processus orientés contexte prennent en compte l'intention et la situation d'un acteur (analyste, ingénieur des méthodes...) à un instant donné du projet (Rolland, 2005).

3.4.1. Exemple de modèle de processus orienté contexte

La Figure 3-23 présente le modèle de processus orienté contexte pour compléter une entité avec des attributs et une clé (Grosz et al., 1997). La racine de l'arbre commence par le contexte *<(Entité), Décrire entité>*. « Entité » représente la situation de l'ingénieur des méthodes, c'est-à-dire qu'il a déjà défini une entité, son intention est de la décrire. Ce contexte est découpé en trois parties, c'est un contexte plan, car les trois sous-contextes doivent être tous exécutés selon l'ordre dans lequel ils sont modélisés. Le premier sous-contexte *<(Entité, Problème identifié), Attacher attribut>* précise que

lorsque le problème est identifié, l'intention de l'ingénieur des méthodes est d'attacher un attribut à l'entité concernée. Ce sous-contexte est lui-même décomposé en deux sous-contextes. Le sous –contexte *<(Entité, Attribut), Discuter de l'attachement de l'attribut>* permet de valider ou non l'attribut pour l'entité donnée. C'est un contexte choix, l'ingénieur des méthodes doit choisir de confirmer ou d'annuler l'attachement de l'attribut à l'entité.

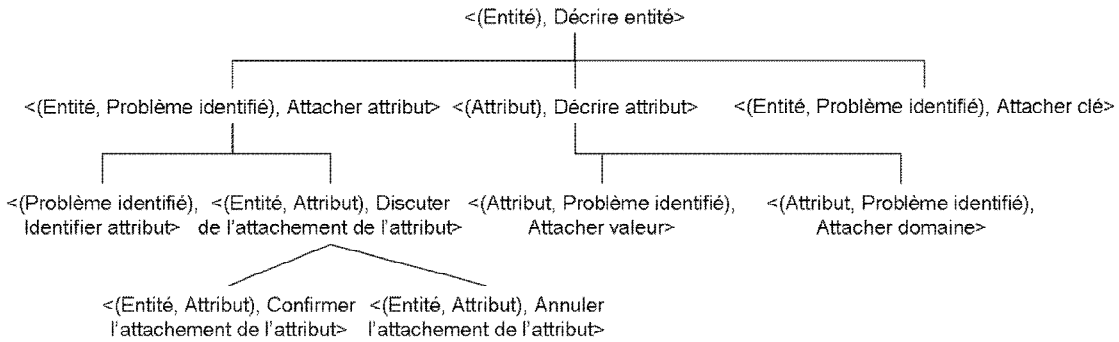


Figure 3-23. Exemple de modèle de processus orienté contexte.

3.4.2. Méta-modèle de processus orienté contexte

La Figure 3-24 présente le méta-modèle de processus orienté contexte NATURE initialisé dans (Rolland, 1994), puis repris par (Plihon, 1996) notamment et (Rolland et al., 2000). La classe principale *Contexte* est composée d'une *Situation* et d'une *Intention*. Une situation est toute partie du produit en cours de développement pouvant faire l'objet d'une décision (Plihon, 1996). L'intention représente l'objectif, le but que l'acteur souhaite atteindre, par rapport à la situation dans laquelle il se trouve (Plihon, 1996). Un contexte peut être de trois types différents :

- *Contexte Plan* : ce contexte est composé de sous-contextes. Ces contextes sont ordonnés et réalisés en séquence.
- *Contexte Exécutable* : lorsque l'ingénieur des méthodes est dans un contexte de ce type, l'intention va engendrer une *action* qui peut modifier une partie de produit.
- *Contexte Choix* : l'ingénieur des méthodes doit choisir entre deux contextes qui se différencient par leur intention. Pour aider l'ingénieur des méthodes dans son choix, des alternatives peuvent être précisées, ces alternatives sont soutenues ou réfutées par des arguments.

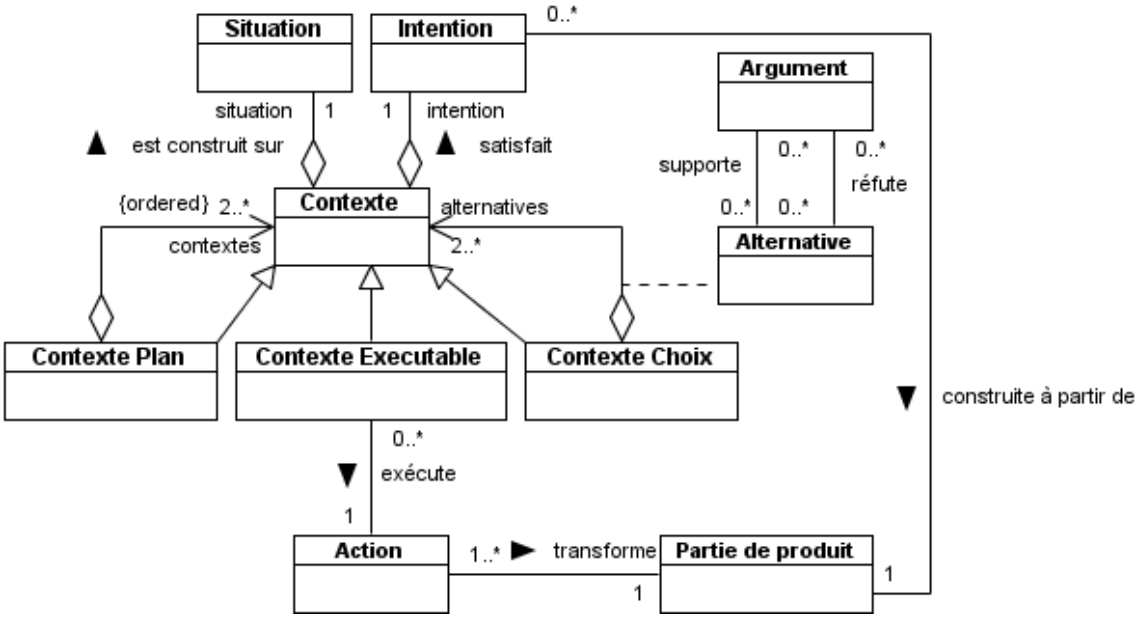


Figure 3-24. Méta-modèle de processus orienté contexte NATURE.

La Figure 3-25 présente le diagramme objets, instance du méta-modèle de processus orienté contexte NATURE correspondant au modèle de la Figure 3-23. Le contexte racine C1 est composé de la situation « Entité » et de l'intention « Décrire entité ». Le diagramme montre que le contexte plan C1 est composé de deux contextes plans C2 et C3.

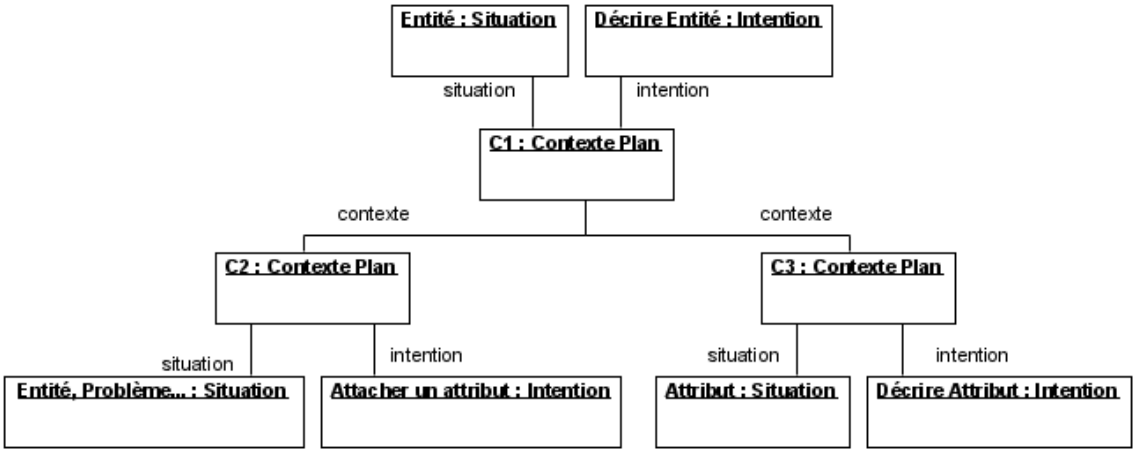


Figure 3-25.Diagramme objets d'un modèle de processus orienté contexte.

3.4.3. Synthèse

Le tableau ci-dessous présente une synthèse des concepts des méta-modèles de processus orientés contexte. Les concepts d'*Action* et de *Partie de produit* sont assimilables aux concepts d'unité de travail et de produit. Le concept d'*Intention* correspond au concept d'*Intention cible* et le concept de *Situation* correspond au concept d'*Intention source* dans les méta-modèles de processus orientés stratégie. Le concept de *Contexte* correspond donc à la composition d'une intention cible et d'une intention source dans les méta-modèles de processus orientés stratégie. Les concepts *Alternative* et *Argument* sont issus des méta-modèles de processus orientés décision.

	Orienté activité	Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
Unité de travail	Work Unit, Work definition	Action	Step	Action	
Produit	Work Product	Composition des états	Artefact	Partie de produit	
Contexte				Contexte	Intention (cible) + Intention (source)
Intention				Intention	Intention (cible)
Situation				Situation	Intention (source)
Alternative			Position	Alternative	
Argument			Argument	Argument	

Tableau 3-4. Synthèse des concepts des méta-modèles de processus orientés contexte.

Le méta-modèle de processus orienté contexte NATURE est le seul méta-modèle de ce type. Il a été introduit lors du projet NATURE (1992-1995) et a été implémenté dans l'outil MENTOR (Plihon, 1996). Ce type de méta-modèle n'est pas aussi usuel que les méta-modèles de processus orienté activité, produit et décision.

Le concept d'intention des méta-modèles de processus orientés contexte a été réutilisé pour modéliser des modèles de processus orientés stratégie, qui permettent de représenter différentes démarches pour atteindre des intentions.

3.5. Les modèles et méta-modèles orientés stratégie

Les modèles de processus orientés stratégie permettent de représenter les processus multi-démarches et prévoient plusieurs chemins possibles pour élaborer le produit en se basant sur les notions d'intention et de stratégie (Rolland, 2005).

3.5.1. Exemple de modèles de processus orientés stratégie

Il existe peu de modèles de processus orientés stratégie. La méthode d'ingénierie des besoins CREWS – L'Écritoire (Ralyté, 1999) définie en partie sur un modèle de processus orienté stratégie, permet de découvrir et d'identifier les besoins pour la définition de systèmes d'information. Les modèles de processus orientés stratégie ne sont pas restreints au domaine des processus d'ingénierie de SI, mais ont aussi été utilisés dans d'autres cas, comme pour modéliser une méthode d'analyse de similarité entre les besoins métier d'une organisation et les besoins fonctionnels d'un système d'information (Zoukar, 2005), ou encore pour modéliser l'alignement entre les processus d'entreprises et les systèmes d'information (Etien, 2006).

La Figure 3-26 représente le modèle de processus orienté stratégie de la méthode CREWS – L'Écritoire (Ralyté, 1999). Le formalisme utilisé pour ce modèle de processus est celui de la MAP (Rolland et al., 1999). Les nœuds représentent les

intentions de l'ingénieur des besoins à un moment donné du processus et les arcs représentent les stratégies pour atteindre ces intentions. Une carte commence toujours par l'intention « Start » et termine par l'intention « Stop ». Dans la méthode CREWS, si l'ingénieur des besoins veut décrire le scénario correspondant au but découvert, il peut le faire en utilisant la stratégie de prose libre qui consiste à écrire le scénario en langue naturelle. Ceci est spécifié par la section composée de l'intention source « Découvrir un but », de l'intention cible « Écrire un scénario » et de la stratégie « En prose libre ». L'ingénieur des méthodes, selon l'intention qu'il vient d'atteindre dans la carte et son intention cible, choisit la stratégie qui lui convient le mieux.

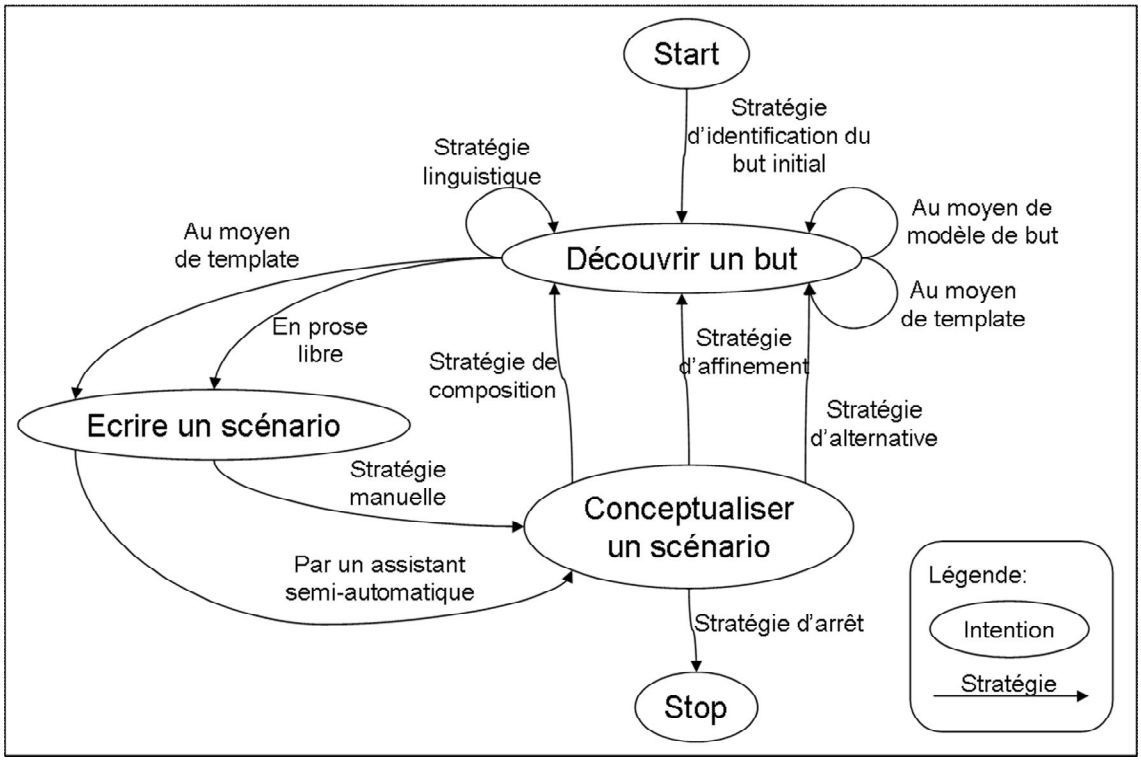


Figure 3-26. Modèle de processus orienté stratégie de la méthode CREWS – L'Écritoire.

3.5.2. Méta-modèle de processus orienté stratégie

La Figure 3-27 représente le méta-modèle de la MAP (Rolland et al., 1999). Une *MAP* ou carte est composée au minimum de deux *sections*. Une section est composée d'une *Intention source*, d'une *Intention cible* et d'une *stratégie* liant l'intention source à l'intention cible. Une intention représente un but, un objectif à atteindre. Une stratégie est une manière de réaliser une intention (Zoukar, 2005). Une section peut être affinée par une autre MAP : cela permet de décomposer une section pour la détailler. Il existe trois types de liens entre sections :

- multi-segment : deux sections sont multi-segment lorsqu'elles ont les mêmes intentions source et cible mais différentes stratégies complémentaires non exclusives ;
- multi-chemin : enchainement de plusieurs sections ayant la même intention cible et source mais définissant des chemins différents dans la carte ;

– cluster : deux sections forment un cluster si elles ont les mêmes intentions source et cible mais différentes stratégies exclusives.

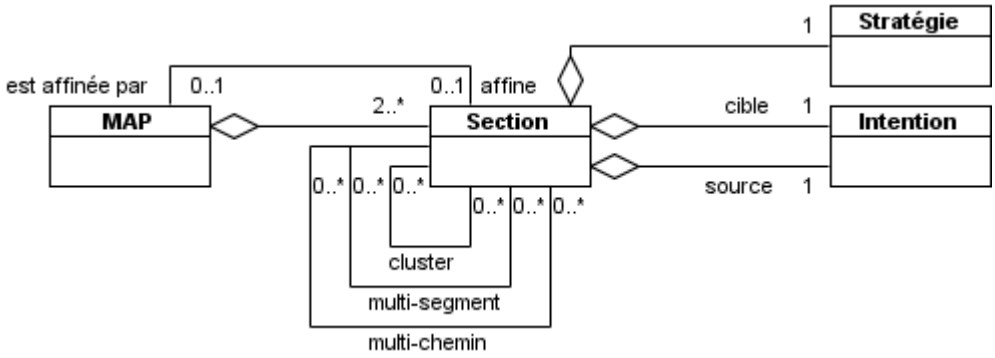


Figure 3-27. Méta-modèle de la MAP.

La Figure 3-28 présente un extrait du modèle de processus orienté stratégie présenté dans la Figure 3-26 sous forme d'un diagramme objets. Chaque objet est instance d'une classe définie dans le méta-modèle de la MAP. Par exemple, « Découvrir un but » est une instance de la classe *Intention* du méta-modèle.

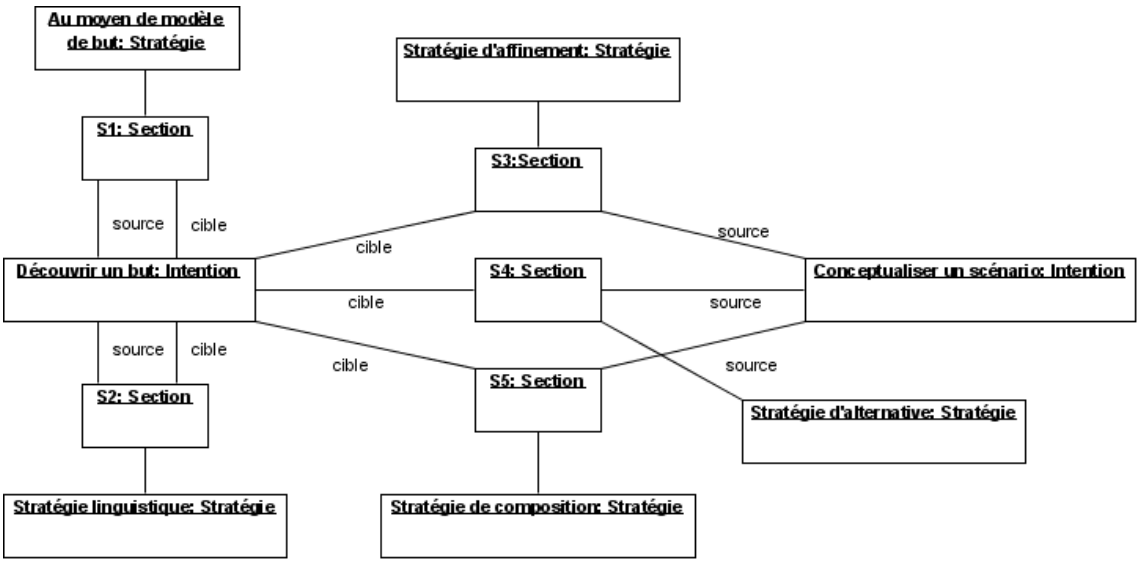


Figure 3-28. Diagramme objets de la méthode CREWS – L'Écritoire, instance du méta-modèle de processus orienté stratégie MAP.

3.5.3. Synthèse

Le Tableau 3-5 présente une synthèse des concepts des méta-modèles de processus orientés stratégie. Le concept de *Stratégie* est unique à ce type de méta-modèle. Le concept d'*Intention* se retrouve dans les méta-modèles de processus orienté contexte. Le concept de *Contexte* n'est pas équivalent à celui de *Section* : ils ont en commun une intention source (*Situation* dans les méta-modèles de processus orientés contexte) et une intention cible, mais les méta-modèles de processus orientés contexte ne prennent pas en compte les différentes stratégies possibles pour atteindre l'intention cible.

	Orienté activité	Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
Carte					MAP
Section					Section
Stratégie					Stratégie
Intention				Intention	Intention (cible)
Situation				Situation	Intention (source)
Contexte				Contexte	Intention (cible) + Intention (source)

Tableau 3-5. Synthèse des concepts des méta-modèles de processus orientés stratégie.

3.6. Synthèse

Dans ce chapitre, nous avons présenté un état de l'art des différents types de modèles et méta-modèles de processus. Nous avons pu établir des correspondances entre les méta-modèles que nous avons présentées dans des tableaux de synthèse.

Plusieurs conclusions peuvent en outre être tirées que nous résumons ci-dessous. Ces conclusions sont en termes de représentations des méta-modèles, de vocabulaire, d'organisation des méta-modèles, de complémentarité et de niveaux d'abstraction.

3.6.1. Représentation des méta-modèles

Nous avons présenté ici les méta-modèles sous forme de diagrammes de classes UML. Cependant, dans la littérature, les méta-modèles de processus sont représentés différemment : certains sont représentés en UML (SPEM, OPF, ISO24744), d'autres par des schémas O* (MAP, NATURE), certains encore ne sont décrits que textuellement ou à partir d'exemples. Il est donc difficile pour un ingénieur des méthodes de les comparer, de les évaluer et de sélectionner les méta-modèles les plus pertinents par rapport à ses besoins.

D'autre part, les méta-modèles de processus sont définis à différents degrés de détail. Par exemple, SPEM est très détaillé, alors que le méta-modèle orienté décision de Potts est plus succinct. Certains méta-modèles comme ISO24744 (ISO/IEC, 2007) préconisent quelques propriétés alors que d'autres n'en précisent pas du tout.

3.6.2. Vocabulaire

a) Vocabulaire divergent

Comme le montrent les tableaux de synthèse présentés pour chaque type de méta-modèle de processus, les termes utilisés sont différents d'un type de méta-modèle à un autre, mais peuvent être identiques d'un point de vue sémantique. Ainsi, les concepts d'unité de travail et de produit sont présents dans presque tous les types de méta-modèles, mais le même terme n'est pourtant jamais utilisé. D'autre part, entre méta-modèles de même type, des concepts de même sémantique n'ont pas le même nom, *Work Unit* et *Work Definition* dans OPF et SPEM, par exemple. Il est donc difficile de

repérer quels sont les concepts identiques d'un type de méta-modèle à l'autre ou d'un méta-modèle à un autre de même type.

b) Vocabulaire trop précis

Certains méta-modèles définissent des termes trop précis. Par exemple, SPEM 1.1 préconise voire oblige à l'utilisation de *Lifecycle*, *Phase*, *Iteration*, *Activity*. De même, l'OPF définit les termes *Technique*, *Workflow*, *Task*. Les ingénieurs des méthodes n'ont donc pas véritablement le choix des termes qu'ils veulent utiliser, même si sémantiquement ces termes sont identiques. D'après nous, les méta-modèles de processus ne devraient pas imposer des termes trop précis, mais laisser le choix aux ingénieurs de définir les termes qu'ils souhaitent utiliser, par rapport au vocabulaire utilisé dans les projets de leurs organisations. Ceci soulève en partie le problème d'adaptation des méta-modèles de processus aux spécificités des organisations.

3.6.3. Adaptation des méta-modèles

Les méta-modèles de processus existants ne peuvent pas être adaptés ou étendus, à l'exception de SPEM 1.1 et 2.0 via des stéréotypes.

Les méta-modèles fournis sont définis de manière monolithique, c'est-à-dire qu'ils sont utilisables tels quels, sans que les ingénieurs des méthodes puissent y apporter des extensions ou des modifications. L'adaptation au contexte des organisations est cependant nécessaire, car le processus d'ingénierie de systèmes d'information est différent selon le domaine, le produit à réaliser, la taille du projet, l'expérience des intervenants. Si le méta-modèle est figé, les modèles de processus instances de celui-ci devront respecter le cadre imposé. L'adaptation au contexte est donc limitée.

3.6.4. Complémentarité des méta-modèles

Les tableaux de synthèse montrent que les différents types de méta-modèles ont des concepts communs. Les différents types de méta-modèles ne sont donc pas isolés les uns des autres. Au contraire, ils ont des liens de correspondance via leurs concepts communs. Les lignes grisées du Tableau 3-6 montrent que certains concepts se retrouvent dans différents méta-modèles de processus.

	Orienté activité	Orienté produit	Orienté décision	Orienté contexte	Orienté stratégie
Unité de travail	✓	✓	✓	✓	
Type d'unité de travail	✓				
Unité de temps	✓				
Type d'unité de temps	✓				
Produit	✓	✓	✓	✓	

Type de produit	✓				
Contrainte	✓	✓			
Rôle	✓				
Type de rôle	✓				
État		✓			
Transition		✓			
Évènement		✓			
Problème			✓		
Alternative			✓	✓	
Argument			✓	✓	
Contexte				✓	(✓)
Situation				✓	
Intention				✓	✓
Carte					✓
Section					✓
Stratégie					✓

Tableau 3-6. Synthèse des concepts des méta-modèles de processus.

Aucun méta-modèle de processus ne permet de couvrir tous les points de vue à la fois, chacun étant positionné sur un point de vue particulier. Selon ce que les ingénieurs veulent modéliser, un méta-modèle est préconisé plutôt qu'un autre. Par exemple, si l'on veut modéliser comment le processus d'ingénierie de SI est organisé, les modèles de processus orientés activité sont les mieux adaptés pour modéliser ce point de vue. Les modèles de processus orientés produit sont les plus appropriés si l'on veut modéliser sur quoi porte le processus d'ingénierie. Les méta-modèles de processus orientés stratégie permettent également de définir comment le processus d'ingénierie de SI est organisé, mais à un niveau différent, ce que nous expliquons dans la section 0. Les méta-modèles de processus sont cependant complémentaires. Chacun présente un aspect du processus d'ingénierie de systèmes d'information. Toutefois, la complémentarité entre les différents méta-modèles de processus a été peu abordée et non définie de manière explicite. Nous pouvons néanmoins constater deux aspects principaux :

D'une part, le lien entre les méta-modèles de processus orientés activité et produit est évident. Les méta-modèles de processus orientés activité permettent de décrire les opérations qui sont réalisées sur des produits ; les méta-modèles de processus orientés produit décrivent les différents états des produits, modifiés par les opérations. Ces deux types de méta-modèles sont complémentaires, au même titre que les diagrammes d'activité et les diagrammes d'états-transitions en conception des systèmes orientés objet.

D'autre part, les méta-modèles de processus orientés décision permettent d'ajouter des informations supplémentaires aux processus en expliquant pourquoi les opérations

ont été réalisées. Les méta-modèles de processus orientés contexte prennent en compte la situation dans laquelle les décisions sont prises. Enfin, les méta-modèles de processus orientés stratégie permettent de décrire comment une intention (le quoi ?) peut être atteinte.

Le Tableau 3-7 résume chaque point de vue (type de méta-modèle de processus) et la question qui lui est associée (Souveyet, 2006) (Cauvet, 2006).

Point de vue	Question
Méta-modèles orientés activité	Comment ? (Qui ?)
Méta-modèles orienté produit	Quoi ?
Méta-modèles orienté décision	Pourquoi ?
Méta-modèles orienté contexte	Quand ?
Méta-modèles orienté stratégie	Comment ?

Tableau 3-7. Points de vue et apport.

3.6.5. Niveaux d'abstraction

Les différents points de vue des méta-modèles de processus permettent de mettre en évidence deux niveaux d'abstraction différents. Les méta-modèles de processus orientés contexte et stratégie se situent à un niveau d'abstraction intentionnel : ils permettent de modéliser les intentions, les objectifs du processus d'ingénierie de SI. Les méta-modèles de processus orientés activité, produit et décision sont situés à un niveau d'abstraction opérationnel : ils permettent de modéliser les opérations à réaliser pour concrétiser les objectifs définis au niveau intentionnel. Les concepts avec leurs niveaux d'abstraction et les points de vue sont représentés dans la Figure 3-29. Seuls les principaux concepts sont représentés sur cette figure.

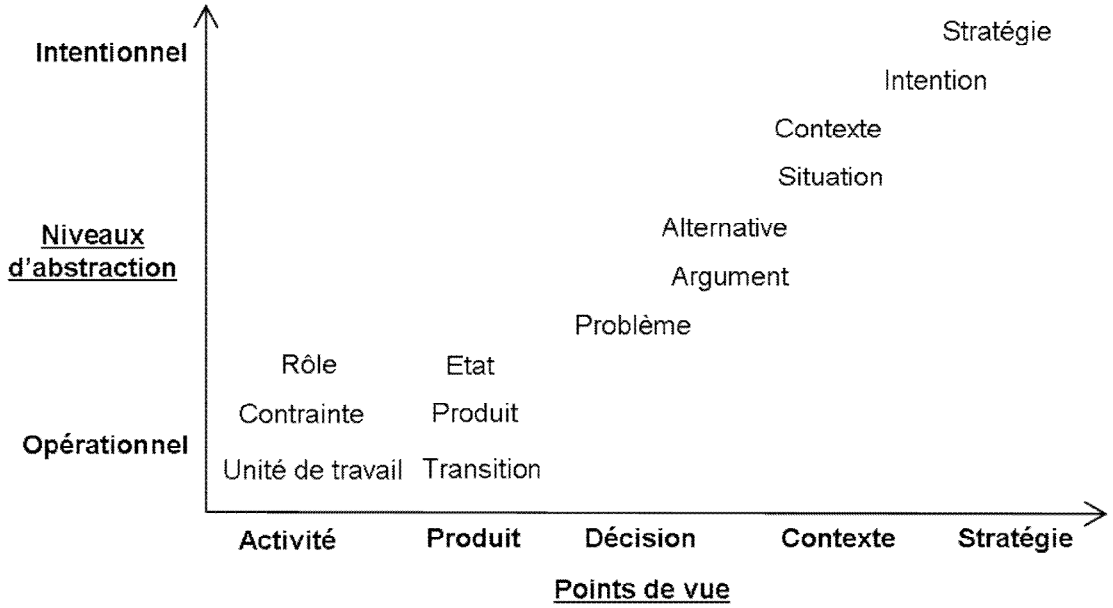


Figure 3-29. Points de vue et niveaux d'abstraction.

Nous avons montré dans ce chapitre qu'il existait de nombreux méta-modèles de processus pour l'ingénierie des systèmes d'information. Un certain nombre de problèmes ont été évoqués dans la synthèse : vocabulaire différent selon les méta-modèles, vocabulaire trop précis, méta-modèles figés... Notre objectif est de fournir aux ingénieurs des méthodes les ressources pour construire leurs propres méta-modèles de processus : adaptés aux contraintes et aux spécificités de leurs organisations, multi points de vue, et prenant en compte les différents niveaux d'abstraction.

CONCLUSION ET POSITIONNEMENT DE NOTRE APPROCHE

Dans cet état de l'art, nous avons dans un premier temps présenté quelques aspects de la modélisation des méthodes d'ingénierie de systèmes d'information comme les différents niveaux de modélisation de l'OMG, les patrons de domaine (ou fragments) et les patrons génériques. Dans un second chapitre, nous avons présenté les différents méta-modèles de processus existants pour l'ingénierie des systèmes d'information.

Nous résumons ici les principales limites des travaux présentés dans l'état de l'art et présentons ensuite le positionnement de notre approche pour répondre à ces limites.

1. Limites des travaux existants

1.1. Des méta-modèles de processus hétérogènes, immutables et implicitement complémentaires

Comme nous l'avons vu dans le chapitre précédent, les méta-modèles de processus existants comportent de nombreux problèmes :

- hétérogénéité des formalismes et du vocabulaire utilisé : les méta-modèles sont représentés dans des formalismes différents et utilisent un vocabulaire varié pour désigner des concepts identiques ;
- immutabilité des méta-modèles : les méta-modèles existants sont fixes et ne proposent pas de mécanismes d'extension ou d'adaptation afin d'être ajustés selon les spécificités et contraintes des organisations ;
- non prise en compte de la complémentarité des méta-modèles : chaque méta-modèle présente un point de vue différent du processus d'ingénierie de SI, les points de vue sont donc complémentaires, pourtant chaque méta-modèle est défini indépendamment des autres, sans préciser la correspondance de ses concepts par rapport aux concepts des autres méta-modèles.

D'une part, il est difficile pour un ingénieur des méthodes de choisir le méta-modèle de processus le plus adapté aux besoins de son organisation de par la diversité des formalismes et du vocabulaire. En outre, l'adaptabilité des méta-modèles de processus existants est limitée. Lors de la méta-modélisation des processus d'ISI d'une organisation, l'ingénieur des méthodes est donc confronté à l'instanciation du méta-modèle de processus qui sera contrainte par sa structure fixe. Enfin, lors de la manipulation de différents modèles de processus, aucune correspondance explicite ne pourra être établie entre les concepts manipulés, à cause du manque de définition de complémentarité entre les classes des méta-modèles correspondants.

1.2. Des patrons génériques inadaptés

Dans le Chapitre 2, nous avons montré que les patrons génériques existants ne répondaient pas aux deux besoins essentiels pour la méta-modélisation des processus d'ISI qui sont la catégorisation et la valuation d'attributs à différents niveaux de modélisation. La catégorisation permet de classifier des concepts selon des caractéristiques communes. Pour la méta-modélisation des processus d'ISI, il est utile,

voire nécessaire, de classifier toutes les unités de travail correspondant à des activités, ou des produits se rapportant à des diagrammes UML.

D'autre part, la valuation d'attributs à différents niveaux de modélisation permet de définir des propriétés globales à tous les méta-modèles de processus, des propriétés plus spécifiques aux modèles de processus d'une organisation et des propriétés propres à l'exécution d'un projet (instance du modèle de processus).

Aucun patron générique à l'heure actuelle ne permet de combiner ces deux besoins.

1.3. Des patrons de domaines spécifiques

Dans le Chapitre 2, nous avons également présenté des patrons de domaine (ou fragments). Les patrons de domaine existants, notamment ceux pour l'ingénierie des méthodes situationnelles, ne répondent pas aux mêmes nécessités que celles requises par la méta-modélisation des processus d'ingénierie de SI. Les patrons de domaine dont nous avons particulièrement besoin correspondent à des parties de méta-modèles de processus existants qui doivent être définis de manière exhaustive.

2. Positionnement de notre approche

Nous présentons brièvement le positionnement de notre approche, en réponse aux problèmes relevés dans l'état de l'art.

2.1. Adaptabilité, flexibilité

Pour augmenter l'adaptabilité et la flexibilité des méta-modèles de processus pour l'ingénierie des SI, nous utilisons le principe des patrons. Similairement aux méthodes d'ingénierie situationnelles qui permettent de construire des méthodes d'ingénierie de SI par assemblage de fragments, notre méthode permet de construire des méta-modèles de processus par imitation de patrons génériques et de domaine. Ainsi, chaque méta-modèle de processus peut être construit selon le contexte, les contraintes et les spécificités des organisations ou des projets.

2.2. Capitalisation et réutilisation des connaissances

Les patrons de domaine permettent de capitaliser les connaissances concernant les méta-modèles de processus existant afin d'être réutilisés pour la construction de nouveaux méta-modèles. Les patrons génériques permettent quant à eux de capitaliser les techniques utiles pour la méta-modélisation des processus. Il s'agit bien ici de réutiliser les connaissances acquises par les communautés de l'ingénierie des SI et des patrons, de les adapter pour les rentre utilisables dans le cadre de la méta-modélisation des processus d'ingénierie de systèmes d'information.

2.3. Spécificités de la méta-modélisation des processus d'ingénierie de SI

Le domaine de la méta-modélisation des processus d'ISI impose des adaptations particulières. Les patrons génériques utilisés sont spécifiques à la méta-modélisation des processus d'ingénierie de SI pour pouvoir prendre en compte à la fois la catégorisation

et la valuation des attributs à différents niveaux de modélisation. D'autres patrons génériques sont proposés pour aider à la méta-modélisation des processus.

De plus, les patrons de domaine proposés correspondent tous à des fragments de méta-modèles de processus existants.

Enfin, les concepts des méta-modèles de processus d'ingénierie sont spécifiques, nous proposons un ensemble de concepts permettant de modéliser tous les types de méta-modèles.

PROPOSITION

INTRODUCTION A LA PROPOSITION

Afin de permettre aux ingénieurs des méthodes de ne pas être totalement dépendants des méta-modèles de processus et modèles de processus existants, nous proposons une méthode leur permettant de concevoir leurs propres méta-modèles de processus d'ingénierie de systèmes d'information.

1. Les forces

Nous présentons ici les forces de notre proposition : notre méthode permet de construire des méta-modèles de processus multi-points de vue, adaptables et fédérateurs.

1.1. Les points de vue

Comme nous l'avons déjà indiqué dans le Chapitre 3, nous considérons que les différents types de méta-modèles de processus (activité, produit, décision, contexte, stratégie) correspondent à différents points de vue du processus d'ingénierie de systèmes d'information. (Rolland, 1998) et (Cauvet, 2006) définissent ces points de vue par le terme de paradigme.

Ces points de vue sont :

- Complémentaires : chaque point de vue présente une perspective différente d'un même processus d'ingénierie de SI,
- Intentionnels ou opérationnels : un point de vue se situe à un niveau d'abstraction intentionnel s'il représente les objectifs du processus d'ingénierie de SI ou opérationnel s'il représente les actions à réaliser pour concrétiser ces objectifs.

Notre proposition permet de construire des méta-modèles de processus comportant les différents points de vue et leur complémentarité de manière explicite, tout en faisant apparaitre le niveau d'abstraction de chaque concept.

1.2. L'adaptabilité

Notre méthode permet de construire des méta-modèles de processus adaptés aux contraintes et aux spécificités des organisations. Les ingénieurs des méthodes peuvent ajouter les concepts utiles pour la modélisation des processus d'ingénierie de SI de leurs organisations, sans prendre en compte des concepts inutiles qui surchargent les méta-modèles. Chaque méta-modèle créé est donc spécifique à une organisation ou un type de projet. Les ingénieurs des méthodes ne sont plus contraints de choisir entre différents méta-modèles existants pour modéliser les processus d'ingénierie des SI de leurs organisations.

L'adaptabilité est facilitée grâce à l'utilisation de patrons génériques et de domaine qui permettent d'adapter le méta-modèle de processus en cours de construction selon ce que souhaitent modéliser les ingénieurs des méthodes.

1.3. La fédération

Jusqu'à présent, les méta-modèles de processus étaient définis séparément, indépendamment les uns des autres. Notre méthode propose de fédérer les points de vue et donc les concepts, afin de ne manipuler qu'un seul méta-modèle de processus présentant tous les points de vue nécessaires et les différents niveaux d'abstraction. De plus, les concepts sont formalisés uniformément dans un seul diagramme de classes et à un même niveau de détail.

Ainsi, les ingénieurs des méthodes ne travailleront qu'avec un seul méta-modèle de processus multi-points de vue, adaptable et fédérateur.

2. Cadre général

La méthode que nous proposons est bornée par un cube à trois dimensions emprunté à Merise 2 (Panet et Letouche, 1994) :

- la dimension de complétude permet aux ingénieurs de compléter, d'étendre leurs méta-modèles de processus. Elle représente la couverture des différents points de vue : activité, produit, décision, contexte et stratégie ;
- la dimension de précision permet aux ingénieurs de raffiner les concepts de leurs méta-modèles de processus en fonction des exigences et contraintes des organisations en faisant appel aux patrons génériques et de domaine ;
- la dimension d'abstraction permet d'abstraire ou de concrétiser les concepts des méta-modèles de processus selon les niveaux d'abstraction intentionnel ou opérationnel.

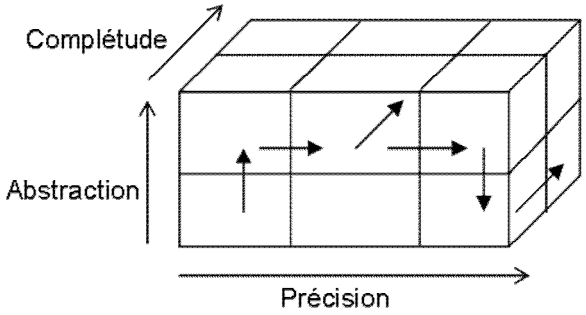


Figure 3-30. Le cube à trois dimensions complétude/précision/abstraction.

Les ingénieurs des méthodes peuvent construire leurs méta-modèles de processus selon les trois dimensions du cube, le processus de construction d'un méta-modèle de processus correspondant à un chemin dans le cube.

3. La méthode

La Figure 3-31 présente une vue globale de la méthode que nous proposons aux ingénieurs des méthodes pour construire les méta-modèles de processus correspondant aux contraintes et spécificités de leurs organisations.

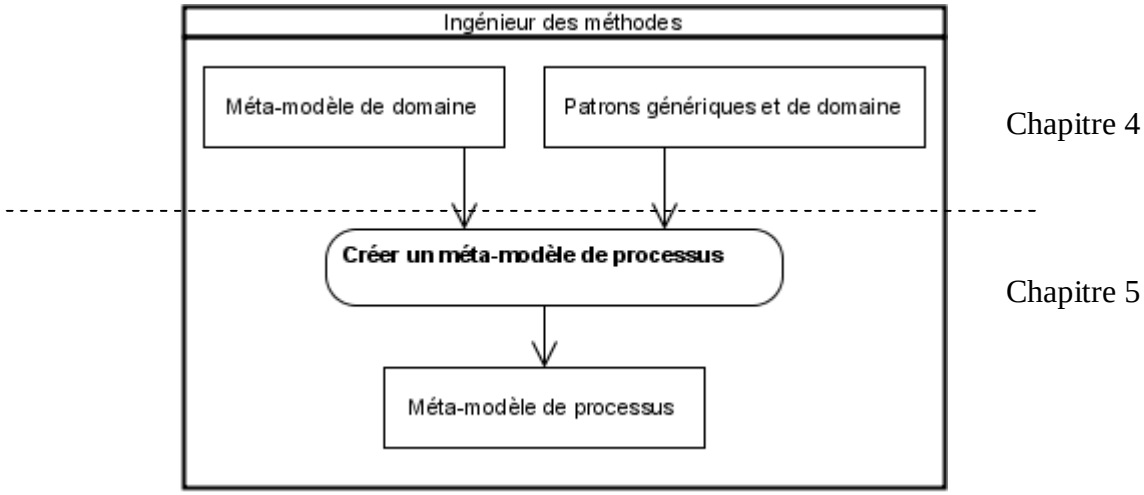


Figure 3-31. Vue globale de la méthode.

Deux ressources principales sont utilisées pour construire les méta-modèles de processus :

- le méta-modèle de domaine, proposé dans le Chapitre 4, modélise les principaux concepts de la méta-modélisation des processus d'ingénierie de systèmes d'information ainsi que les associations entre ces concepts et les relations d'abstraction. Ces concepts proviennent des méta-modèles de processus existants décrits dans le Chapitre 3 ;
- des patrons génériques et de domaine qui servent à raffiner les méta-modèles de processus en cours de construction. De tels patrons sont proposés dans le Chapitre 4.

La méthode est détaillée dans le Chapitre 5 et illustrée par un cas d'étude.

Dans un dernier temps, le méta-modèle de processus obtenu est instancié pour modéliser des modèles de processus pour l'ingénierie des systèmes d'information. Cette partie est également présentée dans le Chapitre 5.

4. LES SUPPORTS DE LA METHODE

Dans ce chapitre, nous présentons les différents supports utilisés dans notre méthode, présentée dans le chapitre suivant. Ces supports sont : le méta-modèle de domaine des processus d'ingénierie de SI, des patrons génériques et des patrons de domaine pour la méta-modélisation des processus d'ingénierie de SI. Enfin, nous proposons un graphe conceptuel basé sur le méta-modèle de domaine et les patrons.

4.1. Méta-modèle de domaine des processus d'ingénierie de SI

Dans cette partie, nous décrivons le méta-modèle de domaine des processus d'ingénierie de SI (Hug et al., 2008b) (Hug et al., 2008c) (Hug et al., 2009). Il a été construit à partir des différentes classes des méta-modèles de processus existants, en prenant en compte les différents points de vue et les niveaux d'abstraction. Il ne permet pas de représenter de manière complète tous les méta-modèles de processus, il sert uniquement de base commune à la conception de nouveaux méta-modèles de processus, qui pourront être enrichis et affinés grâce à des patrons génériques et/ou de domaine pour modéliser les spécificités des processus d'ingénierie de SI de chaque organisation.

4.1.1. Construction du méta-modèle de domaine

Dans un premier temps, nous expliquons comment nous avons construit le méta-modèle de domaine, en partant d'une table d'équivalence de toutes les classes des méta-modèles de processus existants puis en affectant des coefficients à chaque classe afin de ne retenir que les classes principales.

a) Table d'équivalence des classes

L'objectif de la table d'équivalence est d'établir une correspondance entre tous les concepts des différents méta-modèles de processus existants quelque soit leur point de vue (cf. Tableau 4-1) :

- méta-modèles de processus orientés activité : OPF (OPF, 2005), SPEM 1.1 (OMG, 2005) et SPEM 2.0 (OMG, 2008) ;
- méta-modèles de processus orientés produit : State Machines (OMG, 2007), Entity Process Model (Humphrey et Kellner, 1989) et Statecharts (Harel, 1987) ;
- méta-modèle de processus orienté décision : Potts (Potts, 1989) ;
- méta-modèle de processus orienté contexte : NATURE (Rolland, 1994) (Plihon, 1996) ;
- méta-modèle de processus orienté stratégie : MAP (Rolland et al., 1999).

Nous avons pu établir cette table d'équivalence grâce à une étude approfondie de chaque méta-modèle et en utilisant une partie des travaux réalisés par (Henderson-Sellers et Gonzalez-Perez, 2005) qui présentent une table équivalente contenant uniquement les classes de méta-modèles de processus orientés activité.

Pour chaque ligne de la table, représentant chacune un concept, c'est-à-dire des classes ayant le même sens dans les méta-modèles, nous donnons un terme générique que nous utiliserons pour la construction de notre méta-modèle de domaine (colonne « Classe potentielle » dans le Tableau 4-1). Le vocabulaire est ainsi unifié.

b) Élimination des sous-classes

Nous ne souhaitons garder dans le méta-modèle de domaine que les classes fondamentales, c'est-à-dire significatives, de chaque type de méta-modèle de processus. Ainsi, nous procédons à une élimination des sous-classes dans la table d'équivalence. Dans le cas où une classe d'un méta-modèle correspond à une sous-classe d'un autre méta-modèle, nous reportons cette classe au niveau de la super-classe. Par exemple, la classe *Action* de NATURE correspond à la sous-classe *Tâche* dans les autres méta-modèles. Nous supprimons le concept de Tâche, mais afin de ne pas le perdre dans le méta-modèle NATURE, nous le remontons au niveau de la super-classe *Unité de travail*. Le Tableau 4-2 présente la table des équivalences après élimination des sous-classes.

Les sous-classes supprimées pourront être ajoutées selon les besoins des ingénieurs des méthodes dans leurs modèles de processus, comme présenté dans la méthode au Chapitre 5.

OPF	SPEM 1.1	SPEM 2.0	State Machines	EPM	Statecharts	Potts	NATURE	MAP	Classe potentielle
Work unit	Workdefinition	Workdefinition							Unité de travail
	Activityparameter	Workdefinitionparameter							Paramètre
Activity	Discipline	Activity							Activité
	Activity								Activité autre
Task	Step	Taskdefinition, step	Behavior	Action		Step	Action		Tâche
Workflow		Workbreakdownelement							Workflow
Technique									Technique
Stage	workdefinition	Breakdownelement kind							Unité de temps
Stage with duration									Unité de temps avec durée
Phase	Phase	Breakdownelement kind							Phase
Build	Iteration	Breakdownelement kind							Itération
Cycle									Cycle
Life cycle	Lifecycle								Cycle de vie
	Processcomponent								Composant de processus
Development cycle	Process								Cycle de développement
Stage without Duration									Unité de temps sans durée
Milestone									Borne
Inch pebble									Micro-jalon
Producer	Processperformer	Workdefinitionperformer/ Processperformer							Producteur
Indirect producer									Producteur indirect
Organization									Organisation
Team									Équipe
Role	Processrole	Roleuse, Roledefinition							Rôle
Direct producer									Producteur direct
Person									Personne
Tool									Outil
Endeavor									Affaire
Enterprise									Entreprise

Program									Programme
Project									Projet
Convention									Convention
Language									Langage
Work product	Workproduct	Workproductuse/ Workproductdefinition		Entity		Artefact	Partie de produit		Produit
Work product set									Ensemble de produits
Work product version									Version de produit
	Workproductkind								Catégorie de produits
	Constraint	Constraint							Condition
	Precondition		Constraint	Guard	Firing condition				Pré-condition
	goal								Post-condition
			State	State	State				État
			Transition	Transition	Transition				Transition
			Trigger	Event	Event				Évènement
						Position	Alternative		Alternative
						Argument	Argument		Argument
						Issue			Problème
							Situation	Intention (source)	Situation
							Intention	Intention (cible)	Intention
							Contexte		Contexte
							Contexte plan		Contexte plan
							Contexte choix		Contexte choix
							Contexte exécutable		Contexte exécutable
								Stratégie	Stratégie
								Section	Section
								Map	Map

Tableau 4-1. Table des équivalences des classes des différents méta-modèles de processus existants.

OPF	SPEM 1.1	SPEM 2.0	State Machines	EPM	Statecharts	Potts	NATURE	MAP	Classe potentielle
Work unit	Workdefinition	Workdefinition	Behavior	Action		Step	Action		Unité de travail
	Activityparameter	Workdefinitionparameter							Paramètre
Stage	workdefinition	Breakdownelement kind							Unité de temps
Producer	Processperformer	Workdefinitionperformer/ Processperformer							Producteur
Endeavor									Affaire
Language									Langage
Work product	Workproduct	Workproductuse/ Workproductdefinition		Entity		Artefact	Partie de produit		Produit
Work product set									Ensemble de produits
Work product version									Version de produit
	Workproductkind								Catégorie de produits
	Constraint	Constraint	Constraint	Guard	Firing condition				Condition
			State	State	State				État
			Transition	Transition	Transition				Transition
			Trigger	Event	Event				Évènement
						Position	Alternative		Alternative
						Argument	Argument		Argument
						Issue			Problème
							Situation	Intention (source)	Situation
							Intention	Intention (cible)	Intention
							Contexte		Contexte
								Stratégie	Stratégie
								Section	Section
								Map	Map

Tableau 4-2. Table des équivalences après élimination des sous-classes.

c) Pondération et sélection des classes

L'objectif ici est de sélectionner les classes restantes selon leur fréquence : les plus fréquentes seront gardées pour constituer le méta-modèle de domaine, les autres seront intégrées sous forme de patrons.

Pour ce faire, nous avons affecté un coefficient de 1 à chacune des classes de la table d'équivalence simplifiée.

Nous avons ensuite réalisé une moyenne pondérée par ligne (c'est-à-dire par classe équivalente) par type de méta-modèle de processus (activité, produit, contexte, etc.). En effet, il y a trois méta-modèles de processus différents pour les points de vue activité et produit, il faut donc prendre en compte ce paramètre dans le calcul. $M\mu$ est donc calculé pour chaque classe potentielle de la table. Le Tableau 4-3 montre la moyenne pondérée de chaque classe potentielle.

$$M\mu = \frac{\frac{coef(opf) + coef(spem1) + coef(spem2)}{3} + \frac{coef(sm) + coef(epm) + coef(sc)}{3} + coef(potts) + coef(nature) + coef(map)}{5}$$

Équation 4-1. *Moyenne pondérée de chaque ligne de la table d'équivalence.*

Nous avons ensuite fixé un seuil qui est la somme de la moyenne globale des moyennes pondérées $M\mu$ et de l'écart type de cette moyenne globale.

$$seuil = \sum_{n}^{1} M\mu_i + \sigma$$

Équation 4-2. Calcul du seuil.

Le premier seuil est donc fixé à 0,43 (0,25 + 0,18), cf. Tableau 4-3. Deux classes ont une moyenne pondérée supérieure à ce seuil : *Unité de travail* et *Produit*. Ces classes sont donc gardées pour être représentées dans le méta-modèle de domaine.

Le second seuil que nous fixons est égal à la moyenne globale de 0,25. Cinq classes ont une moyenne pondérée comprise entre 0,25 et 0,43 : *Condition, Alternative, Argument, Situation* et *Intention*. Toutes ces classes seront également intégrées au méta-modèle de domaine.

Enfin, nous fixons un troisième seuil à 0,20. Les classes dont la moyenne pondérée se situe entre 0,20 et 0,25 sont traitées individuellement afin de déterminer si elles seront intégrées telles quelles dans le méta-modèle de domaine ou si elles seront utilisables sous forme de patrons de domaine :

- les classes *État, Transition* et *Évènement* ne sont présentes que dans les méta-modèles de processus orientés produit. Elles seront donc représentées sous forme de patrons de domaine, accessibles à partir de la classe *Produit* ;
- la classe *Unité de temps* est uniquement présente dans les méta-modèles de processus orientés activité. Elle pourra être représentée sous la forme d'un patron de domaine, utilisable depuis la classe *Unité de travail* ;
- les classes *Section* et *Map* sont présentes dans un seul méta-modèle de processus, elles seront donc utilisées sous la forme d'un patron de domaine ;
- la classe *Producteur* est uniquement présente dans les méta-modèles de processus orientés activité, cependant, sa couverture nous semble plus large. Elle pourrait être

notamment associée à des classes des méta-modèles de processus orientés décision, pour prendre en compte les rôles intervenant dans les processus de décision. Cette classe est donc gardée pour construire le méta-modèle de domaine. Nous utiliserons le terme *Rôle* qui nous semble plus approprié ;

- les classes *Alternative* et *Argument* se rapportent de manière dépendante à la classe *Problème*. En effet, une alternative est soutenue par des arguments et elle répond à un problème, la classe *Problème* est donc gardée pour construire le méta-modèle de domaine ;
- les classes *Contexte* et *Stratégie* sont des classes clés des méta-modèles de processus orientés contexte et stratégie. Elles sont donc gardées pour construire le méta-modèle de domaine.

Les classes dont la moyenne pondérée est inférieure à 0,20 sont également traitées individuellement :

- la classe *Paramètre* issue du méta-modèle SPEM ne nous parait pas cohérente, car elle représente les produits en paramètres d'une unité de travail, les paramètres peuvent donc être modélisés directement par des associations entre les classes *Produit* et *Unité de travail*. Cette classe est donc supprimée ;
- une *Affaire* représente une collaboration entre *Producteur* dans un temps donné (*Unité de temps*). Cette classe pourra être ajoutée via un patron de domaine ;
- *Langage* permet de définir dans quel langage est défini un produit. Nous permettrons via le patron générique Concept-Catégorie de concepts de préciser la catégorie d'un produit comme diagramme UML, texte, etc. ;
- *Ensemble de produits* peut être créé en utilisant une composition sur la classe *Produit*, afin de préciser qu'un produit en particulier est composé d'un ensemble de produits. Cette classe est donc supprimée et pourra être recréée via un patron générique ;
- *Version de produit* peut être remplacée par un attribut dans la classe Produit, comme nous le montrerons dans la suite de ce chapitre. Cette classe est donc supprimée ;
- *Catégorie de produit* est similaire à *Langage*, puisqu'il décrit le formalisme ou le type d'un produit. Le patron générique Concept-Catégorie de concepts permet de définir cela.

Le Tableau 4-3 présente le résultat des actions réalisées sur la table d'équivalence des classes. Nous utilisons les sigles MMD pour méta-modèle de domaine, P. Dom pour patron de domaine et P. Gen pour patron générique. Les classes dont la destination est MMD sont présentes dans le méta-modèle de domaine. Les patrons génériques et de domaine pour les autres classes sont présentés plus loin dans ce chapitre.

Méta-modèles / Classes	OPF	SPEM1.1	SPEM 2.0	State Machines	EPM	Statecharts	Potts	NATURE	MAP	Moyenne pondérée	Destination
Unité de travail	1	1	1	1	1		1	1		**0,73**	**MMD**
Paramètre		1	1							0,13	
Unité de temps	1	1	1							**0,20**	**P. Dom.**
Producteur (Rôle)	1	1	1							**0,20**	**MMD**
Affaire	1									0,07	**P. Dom**
Langage	1									0,07	**P. Gen**
Produit	1	1	1		1		1	1		**0,67**	**MMD**
Ensemble de produits	1									0,07	**P. Gen**
Version de produit	1									0,07	
Catégorie de produits		1								0,07	**P. Gen.**
Condition		1	1	1	1	1				**0,33**	**MMD**
État				1	1	1				**0,20**	**P. Dom**
Transition				1	1	1				**0,20**	**P. Dom**
Évènement				1	1	1				**0,20**	**P. Dom**
Alternative							1	1		**0,40**	**MMD**
Argument							1	1		**0,40**	**MMD**
Problème							1			**0,20**	**MMD**
Situation								1	1	**0,40**	**MMD**
Intention								1	1	**0,40**	**MMD**
Contexte								1		**0,20**	**MMD**
Stratégie									1	**0,20**	**MMD**
Section									1	**0,20**	**P. Dom**
Map									1	**0,20**	**P. Dom**
							Moyenne globale		0,25		
							Écart Type		0,18		

Tableau 4-3. Pondération des classes et leur destination.

d) Matrice des classes et associations

Pour déterminer les associations entre les classes sélectionnées, nous réalisons une matrice d'adjacence, permettant de représenter les associations entre les classes des différents méta-modèles de processus existants (cf. Tableau 4-5). Ainsi, nous retenons les associations suivantes :

- *Source* et *Cible* entre les classes *Stratégie* et *Intention* ;
- *Est construit sur* (composition) entre *Contexte* et *Situation* ;
- *Satisfait* (composition) entre *Contexte* et *Intention* ;
- *Réfute* et *Supporte* entre *Argument* et *Alternative* ;
- *Répond à* entre *Alternative* et *Problème* ;
- *Fait Emerger* entre *Unité de travail* et *Problème* ;
- *Contribue à* entre *Alternative* et *Unité de travail* ;
- *Réalise* entre *Rôle* et *Unité de travail* ;
- *Est responsable de* entre *Rôle* et *Produit* ;
- *Concerne* entre *Condition* et *Unité de travail* ;
- *Cite* entre *Argument* et *Produit* ;
- *Entrée* et *Sortie* entre *Unité de travail* et *Produit*.

e) Introduction des niveaux d'abstraction

Parmi les classes sélectionnées, nous définissons la notion de niveau d'abstraction. Nous introduisons deux niveaux d'abstraction : intentionnel et opérationnel. Le niveau d'abstraction intentionnel permet de définir les objectifs des processus d'ingénierie de SI. Le niveau opérationnel permet de modéliser les opérations à réaliser pour concrétiser les objectifs définis au niveau intentionnel. L'abstraction est l'opération qui consiste à passer d'une classe de niveau opérationnel à une classe de niveau intentionnel. La concrétisation est l'opération inverse. Le Tableau 4-4 présente la correspondance entre les classes du niveau intentionnel et les classes du niveau opérationnel.

Intentionnel	Opérationnel
Stratégie	Unité de travail
Intention	Produit
Situation	Condition

Tableau 4-4. Correspondance entre les classes des niveaux intentionnel et opérationnel.

Nous proposons donc qu'une stratégie dans un modèle de processus corresponde à une unité de travail dans ce même modèle de processus. En effet, ces deux notions représentent bien le « comment », mais sont décrites à deux niveaux différents. De même pour *Intention* et *Produit* : une intention est un objectif à atteindre dans un modèle de processus orienté stratégie ou contexte, c'est le « quoi », le produit concrétise ce quoi, c'est l'objectif à réaliser concrètement. Enfin, une situation correspond à l'état du processus d'ingénierie de SI à un instant donné, une condition représente cet état sous forme de contraintes tangibles.

Classe source \ Classe cible	Unité de travail	Rôle	Produit	Condition	Alternative	Argument	Problème	Situation	Intention	Contexte	Stratégie
Unité de travail			manipulates, produces				raises				
Rôle	performs, performer		produces, responsible								
Produit											
Condition	constraints										
Alternative	contributes to						responds to				
Argument			cites		objects to, supports						
Problème											
Situation											
Intention											
Contexte								est construit sur (agrégation)	satisfait (agrégation)		
Stratégie									a pour source, a pour cible		

Tableau 4-5. Matrice des classes du MMD et des associations définies dans les méta-modèles de processus existants.

4.1.2. Le méta-modèle de domaine

Nous décrivons le méta-modèle de domaine obtenu à partir de l'analyse présentée précédemment.

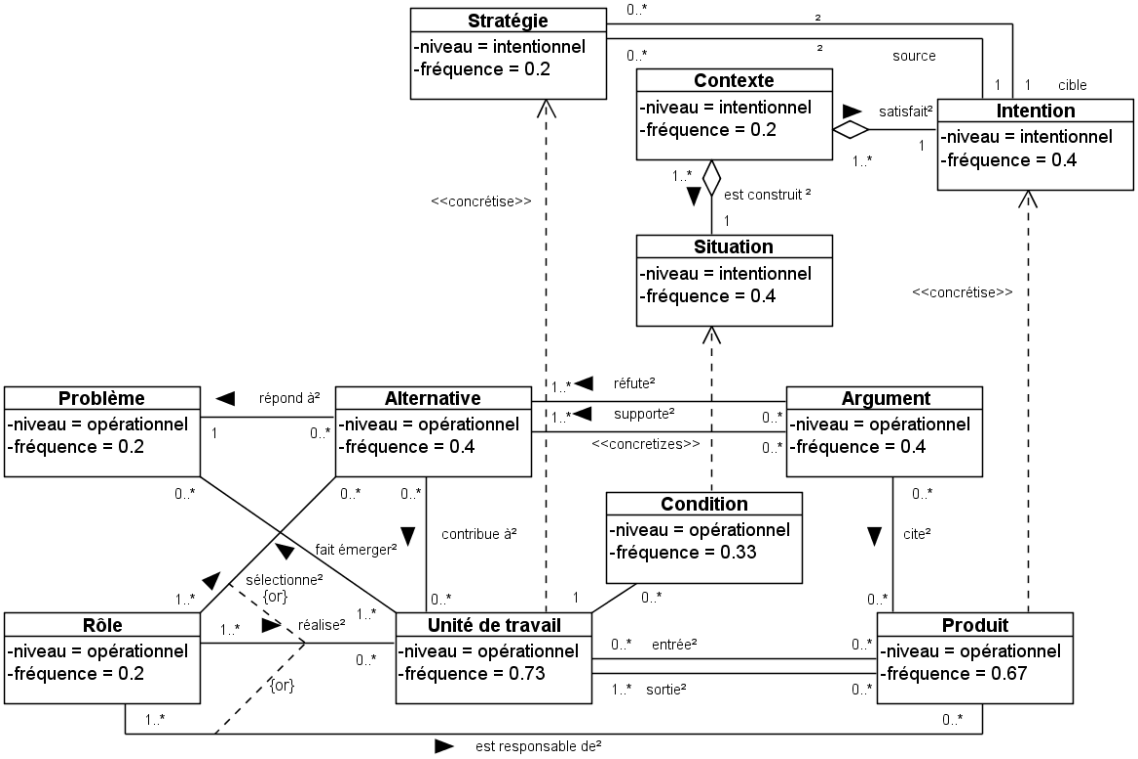


Figure 4-1. Le méta-modèle de domaine.

a) Les classes

Les classes sont issues des concepts qui résultent de l'analyse réalisée en section 0. Au niveau intentionnel, représenté par la propriété valuée *niveau* (=intentionnel) de chaque classe, nous avons retenu les quatre classes *Contexte, Situation, Intention* et *Stratégie*. Ces classes sont les concepts clés des méta-modèles de processus orientés contexte et stratégie.

Au niveau opérationnel, nous avons retenu les concepts de *Problème*, *Alternative* et *Argument* pour modéliser les processus orientés décision. Pour représenter les méta-modèles de processus orientés activité, nous avons gardé les concepts d'*Unité de travail* et de *Condition*. La classe Producteur a été introduite sous le nom *Rôle*, car la classe modélisée réalise aussi des unités de travail et prend des décisions. La notion de rôle est plus générale que celle de producteur. Enfin, le concept *Produit* permet de représenter en partie les méta-modèles de processus orientés produit.

Chaque classe a un attribut *fréquence* qui représente la moyenne pondérée obtenue dans l'analyse présentée précédemment. Cet attribut peut indiquer aux ingénieurs des méthodes quels sont les concepts les plus fréquents dans les méta-modèles de processus.

b) Les associations

Les associations entre les classes du méta-modèle de domaine ont été sélectionnées à partir des méta-modèles de processus existants grâce à la matrice présentée précédemment.

La seule association que nous avons ajoutée est S*électionne* entre *Rôle* et *Alternative*, pour prendre en compte le rôle des acteurs d'un projet dans l'aspect décisionnel d'un processus d'ingénierie de SI.

Les associations portent toutes un exposant 2, afin de respecter le principe de l'Instanciation Profonde présenté dans l'état de l'art. Cela signifie que les associations du méta-modèle peuvent être définies dans les modèles de processus (M1) et instanciées à l'exécution du processus (M0).

Certaines classes du méta-modèle de domaine sont indissociables d'autres classes. Par exemple, la classe *Stratégie* n'a pas de sens si elle n'est pas associée à la classe *Intention*, la classe *Stratégie* est donc dépendante de la classe *Intention*. Le Tableau 4-6 présente les classes dépendantes et les classes dependees auxquelles elles se rapportent.

Classe dépendantes	Classe (s) dependee (s)
Stratégie	Intention
Contexte	{Situation ∧ Intention}
Argument	Alternative
Alternative	Problème
Condition	Unité de travail
Rôle	{Alternative ∨ Produit ∨ Unité de travail}

Tableau 4-6. Concepts dépendants.

Un contexte ne peut exister sans une situation et une intention, en effet lorsque la classe *Contexte* est dans un méta-modèle de processus, les classes *Situation* et *Intention* sont également présentes. La classe *Argument* est dépendante de la classe *Alternative*, elle-même dépendante de la classe *Problème*. Si *Argument* est présent dans le méta-modèle, *Alternative* devra aussi y être ainsi que *Problème*. Une condition ne peut exister sans être associée à une unité de travail. Enfin, un rôle doit être associé à une alternative ou à un produit ou à une unité de travail.

Ces contraintes sont modélisées dans le méta-modèle de domaine sous forme de cardinalités et de contraintes sur les associations.

c) Les liens de concrétisation

Nous définissons entre les concepts du niveau intentionnel et du niveau opérationnel un lien de concrétisation, comme décrit à la section e), sous la forme de dépendances stéréotypées « concrétise ».

d) Les attributs

Nous proposons un certain nombre d'attributs qui nous semblent pertinents et utiles pour la modélisation des processus d'ingénierie de SI. Grâce à l'instanciation profonde,

nous pouvons définir les attributs directement dans le méta-modèle, ceux-ci seront instanciés soit dans les modèles de processus (M1), soit à l'exécution (M0).

La Figure 4-2 présente quatre classes du méta-modèle de domaine avec les attributs que nous suggérons, inspirés de différents méta-modèles de processus existants dont (ISO/IEC, 2007).

Un objet de type *Rôle* peut avoir un nom, une description, et au niveau de l'exécution, un acteur réel est affecté au rôle. Cet acteur peut être une personne ou une machine par exemple. L'attribut description est un attribut dual (il bénéficie de l'instanciation profonde et il est souligné), il peut être valué à tous les niveaux de modélisation : M2, M1 et M0.

Une *Unité de travail* a un nom, une description, elle peut être optionnelle ou non dans le processus d'ingénierie de SI, a un statut (non commencée, en cours, terminée, par exemple), une date de début et une date de fin. Ces trois derniers attributs seront valués lors de l'exécution du processus.

Un *Produit* a également un nom et une description, il peut correspondre à un livrable du processus. De plus, chaque produit, lors de l'exécution du processus, a une date de création, un numéro de version, des dates de modification, une date de validité et est rédigé dans une langue (en anglais ou français par exemple).

Une *Condition* a une expression qui peut être exprimée en langage naturel ou dans un langage interprétable par une machine comme le langage OCL (OMG, 2007b). Cet attribut est valué à l'exécution des processus (M0).

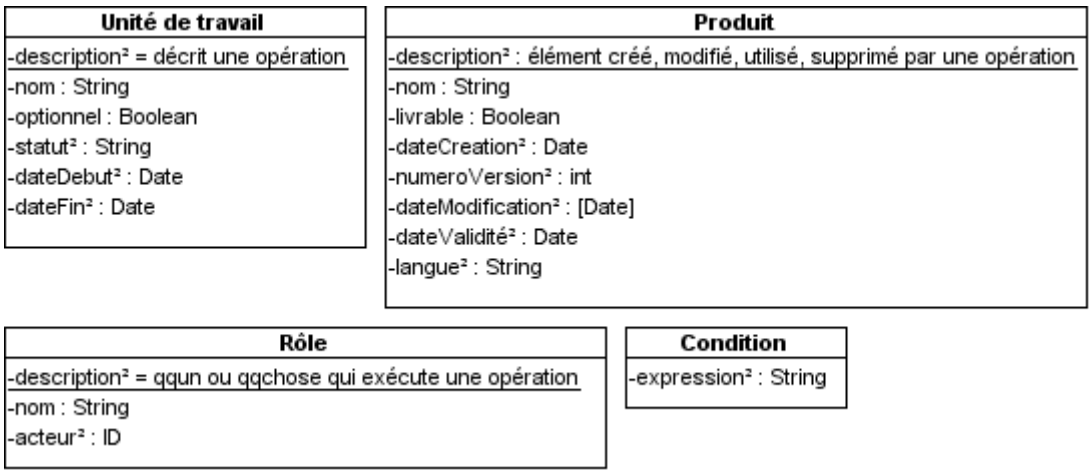


Figure 4-2. Description des attributs des classes Rôle, Unité de travail, Produit et Condition.

La Figure 4-3 présente les différents attributs des classes du méta-modèle de domaine *Problème*, *Alternative* et *Argument*. Un *Problème* a une description, un nom et une priorité qui peut aller de 1 à 5 par exemple (1 étant la priorité la plus importante). Une *Alternative* a également une description, un nom et à l'exécution du processus, on précise si elle a été sélectionnée ou non pour répondre au problème. Un *Argument* a également une description et un nom, un poids peut lui être attribué pour faciliter le choix d'une alternative lors du processus d'ingénierie de SI : plus le poids est important, plus l'argument compte dans le refus ou le soutien d'une alternative.

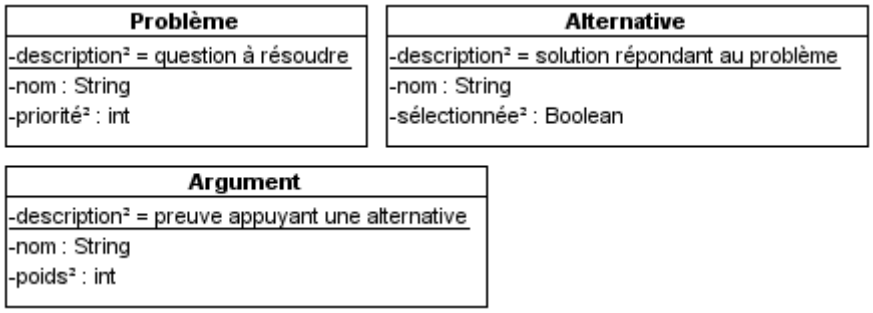


Figure 4-3. Description des attributs des classes Problème, Alternative et Argument.

Les classes *Stratégie*, *Intention*, *Contexte* et *Situation* ont les mêmes attributs description et nom.

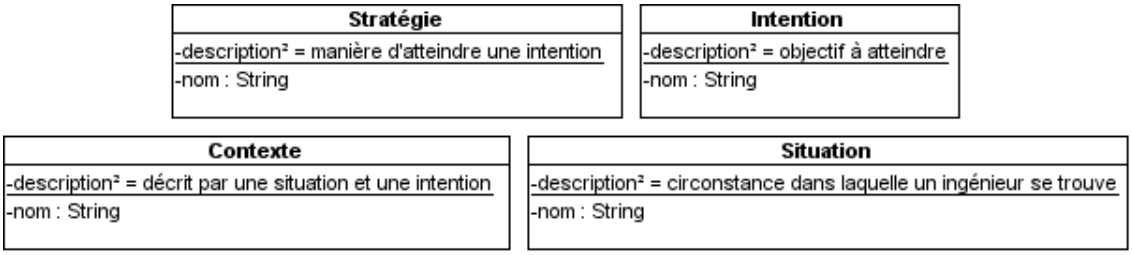


Figure 4-4. Description des attributs des classes Stratégie, Intention, Contexte et Situation.

Dans cette partie du chapitre, nous avons présenté le méta-modèle de domaine qui sert de base à la construction de nouveaux méta-modèles de processus pour l'ingénierie de SI. Nous présentons maintenant des patrons génériques et des patrons de domaine qui permettent d'enrichir les méta-modèles de processus que les ingénieurs des méthodes conçoivent.

4.2. Des patrons génériques

Nous présentons ici quelques patrons génériques permettant d'enrichir les méta-modèles de processus en construction.

4.2.1. Concept-Catégorie de Concepts

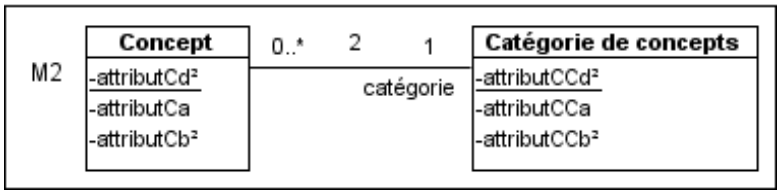
Nous proposons le patron Concept-Catégorie de Concepts (Hug et al., 2007) (Hug et al., 2008) pour répondre à deux besoins :

- la catégorisation des concepts des méta-modèles de processus pour distinguer une unité de travail de son type, par exemple : « Analyse des risques» est une unité de travail de type « Activité » ;
- la définition de propriétés à différents niveaux de modélisation : au niveau des méta-modèles de processus, au niveau des modèles de processus et au niveau de l'exécution des processus.

Le patron Concept-Catégorie de Concepts est basé sur deux patrons existants (cf. Chapitre 2) :

- le patron ItemDescription (Coad, 1992) pour permettre la catégorisation de concepts ;

– l'Instanciation Profonde (Atkinson and Kühne, 2001) pour l'instanciation des propriétés à différents niveaux de modélisation.

Nom	Concept-Catégorie de Concepts
Classification	Générique ^ Produit ^ Méta-modélisation ^ Catégorisation
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Permet de catégoriser des concepts dotés de propriétés spécifiques et partageant des propriétés communes sur trois niveaux de modélisation.
Force	Ce patron permet de partitionner les concepts d'un méta-modèle de processus en deux classes : les concepts et les catégories de concepts. Les imitations de concept et de catégorie de concepts font partie du méta-modèle de processus. Leurs instances définissent les éléments faisant partie des modèles de processus et des catégories d'éléments auxquels ils se rattachent. Ce patron permet également d'instancier les propriétés des concepts à deux niveaux de modélisation : modèle (M1) et exécution (M0) et de valuer des propriétés dans le méta-modèle (niveau M2). Tous les attributs sont définis au niveau M2, dans le méta-modèle de processus. De plus, les associations sont définies au niveau des méta-modèles (M2) et sont instanciées dans les modèles (M1) et à l'exécution des processus (M0)
Solution-modèle	 Les attributs *attributCd²* et *attributCCd²* sont valués dès le niveau M2, et peuvent être valués à tous les niveaux de modélisation suivants. Ce sont des attributs duaux, on les distingue des autres attributs en les soulignant. Les attributs *attributCa* et *attributCCa* sont classiquement instanciés au niveau M1. Les attributs *attributCb²* et *attributCCb²* entre *Concept* et *Catégorie de Concepts* bénéficient de l'instanciation profonde et sont instanciés au niveau M0. L'association entre Concept et Catégorie de concepts porte un exposant 2 et est instanciée aux niveaux M1 et M0.
Cas d'application	Nous souhaitons définir des unités de travail et leur catégorie dans le méta-modèle de processus : – *Catégorie d'unité de travail* définit les propriétés instanciables par les catégories d'unité de travail (par exemple *Activité*) et les instances des catégories d'unité de travail (*Act1* dans la figure). – *Unité de travail* définit les propriétés instanciables par les

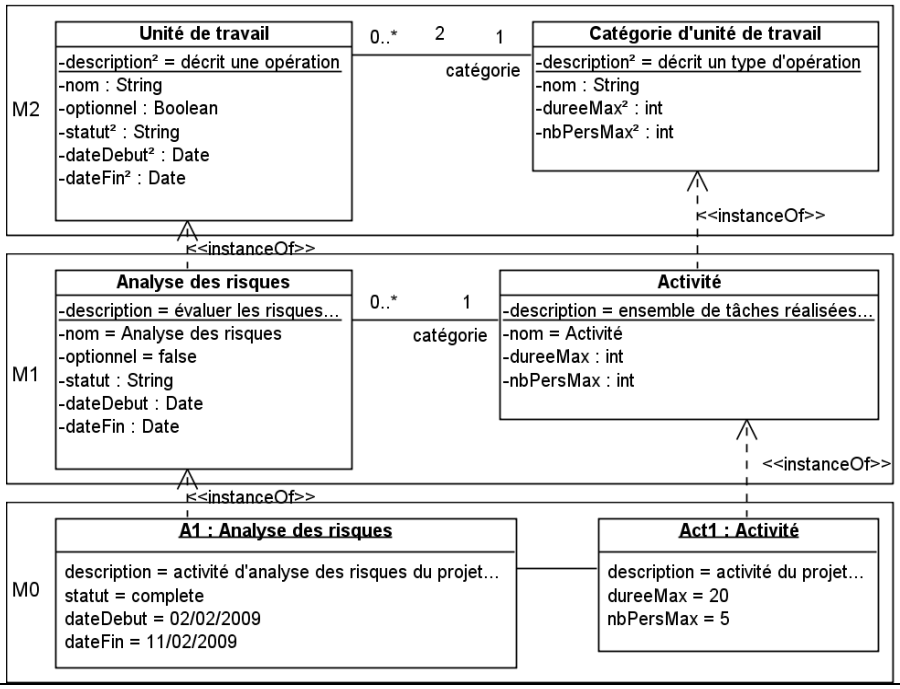
	unités de travail (par exemple *Analyse des risques*) et les instances des unités de travail (*A1* dans la figure). 
Conséquence d'application	Ce patron permet de : – catégoriser des concepts, – valuer des attributs à différents niveaux de modélisation.

Tableau 4-7. Patron Concept-Catégorie de Concepts.

4.2.2. Association réflexive

Le patron ci-dessous permet d'ajouter une association réflexive sur une classe du méta-modèle.

Nom	Association réflexive
Classification	Générique ^ Produit ^ Méta-modélisation
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Permet d'enrichir une classe d'un méta-modèle par une association réflexive afin de définir au niveau des modèles des liens entre objets instances de la même classe.
Force	Ce patron permet d'ajouter une association réflexive sur une classe d'un méta-modèle de processus.
Solution-démarche	1. Créer l'association réflexive sur la classe désirée. 2. Nommer l'association et ses rôles. 3. Modifier les cardinalités si nécessaire. 4. Ajouter une contrainte sur l'association comme {sequence} si une séquence doit être définie.
Solution-modèle	L'association porte sur une classe en particulier du méta-modèle de processus. Les cardinalités par défaut sont « 0..* », mais elles peuvent

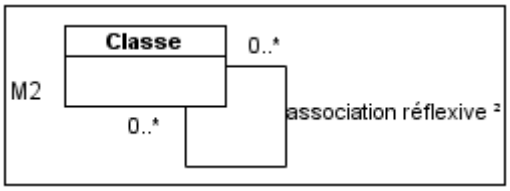
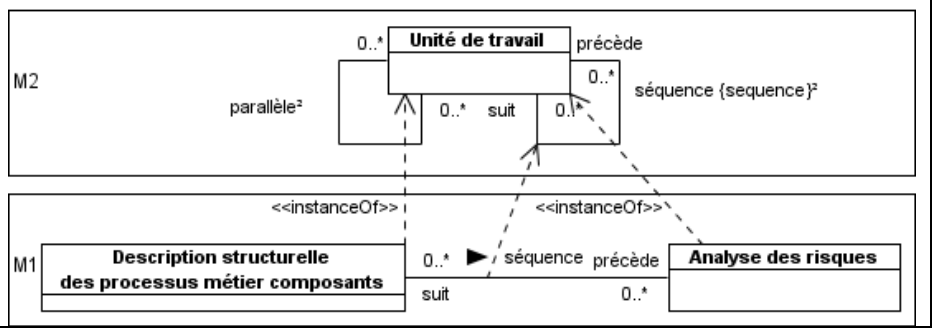
	être adaptées selon le contexte. Le nom de l'association et des rôles sont également modifiables. 
Cas d'application	Afin de préciser que deux unités de travail peuvent s'exécuter en parallèle ou en séquence, il faut imiter deux fois le patron Association réflexive sur la classe Unité de travail du méta-modèle de processus, en nommant la première association parallèle et la seconde séquence, avec les rôles suit et précède et la contrainte {sequence}. Dans le modèle de processus instancié, il est donc possible de définir que deux unités de travail comme Description structurelle des processus métier composants et Analyse des risques doivent être exécutées en séquence. 

Tableau 4-8. Patron Association réflexive.

4.2.3. Composition – Agrégation réflexive

Le patron ci-dessous permet d'ajouter une composition ou une agrégation réflexive à une classe du méta-modèle.

Nom	Composition-Agrégation réflexive
Classification	Générique ^ Produit ^ Méta-modélisation
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Permet de définir une composition ou une agrégation réflexive sur une classe d'un méta-modèle afin de préciser au niveau du modèle qu'un objet peut être composé (ou agrégé) d'un autre objet, tous deux étant instances de la même classe.
Force	La solution proposée permet d'ajouter une composition réflexive ou une agrégation réflexive sur une classe du méta-modèle de processus, selon le contexte de l'imitation.
Solution-démarche	1. Choisir d'appliquer la composition ou l'agrégation réflexive sur la classe désirée. 2. Nommer l'association et ses rôles.

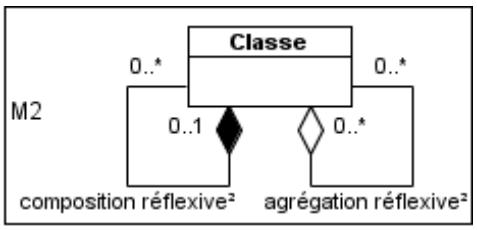
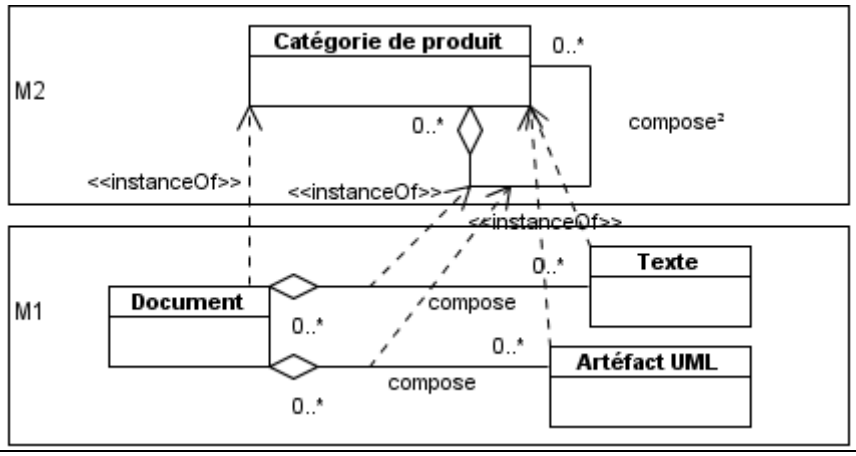
	3. Modifier les cardinalités si nécessaire. 4. Ajouter des contraintes sur l'association comme {ordered} si les composants sont ordonnés.
Solution-modèle	La composition ou agrégation porte sur une classe en particulier du méta-modèle de processus. Les cardinalités par défaut sont « 0..* » mais elles peuvent être adaptées selon le contexte. Le nom de la composition/agrégation est également modifiable.  Il est possible de modifier les cardinalités du côté opposé à l'agrégation /composition. La cardinalité du côté de l'agrégation peut être modifiée à « 1..1 ». La cardinalité du côté de la composition est non modifiable.
Cas d'application	Pour spécifier qu'un document comprend à la fois du texte et des artéfacts UML, on imite le patron Composition-Agrégation réflexive, en choisissant l'agrégation. Ainsi, lors de l'instanciation de la classe Catégorie de Produit dans le modèle de processus, il est possible de spécifier les catégories de produits (Texte et Artéfact UML) qui composent d'autres catégories de produits (Document). 

Tableau 4-9. Patron Composition-Agrégation réflexive.

4.3. Des patrons de domaine

Nous présentons ici des patrons de domaine issus des méta-modèles de processus existants qui peuvent être imités pour spécifier les méta-modèles de processus en construction. Afin de correspondre au mieux aux classes définies dans le méta-modèle de domaine, certaines classes ont été renommées. Par exemple, la classe *Action* dans le méta-modèle État transition a été renommée *Unité de travail* dans le patron de domaine État-Transition. Nous ne présentons pas d'exemple pour les patrons État-Transition,

NATURE et MAP, des exemples ayant déjà été présentés dans l'état de l'art au Chapitre 3.

4.3.1. Patron État-Transition

Ce patron permet de prendre en compte les différents états d'un produit et les transitions entre ces états.

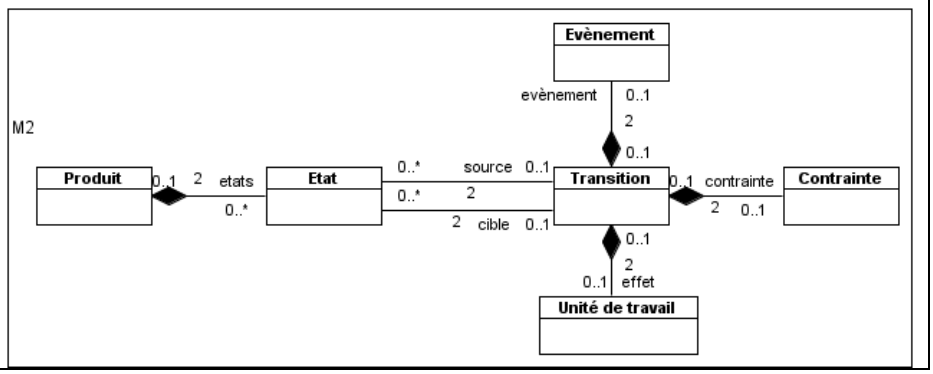
Nom	État-Transition
Classification	Domaine ^ Produit ^ Méta-modélisation ^ Produit
Contexte	La classe Produit doit être définie dans le méta-modèle de processus
Problème	Permet de décrire les différents états d'un produit et les transitions entre ces états.
Force	Ce patron permet de raffiner la classe Produit en décrivant : – les différents états dans lesquels le produit peut se trouver ; – les transitions entre les différents états avec des propriétés comme l'évènement déclenchant la transition, la contrainte portant sur la transition et l'unité de travail déclenchée par la transition.
Solution-démarche	L'application de ce patron peut se faire en deux étapes : 1. Imitation de la classe État et de l'association entre Produit et État. 2. Imitation de Transition et des classes composantes.
Solution-modèle	Un produit est composé de plusieurs états. Des transitions peuvent être définies entre ces états. Une transition peut être déclenchée par un évènement, respecter une contrainte et entrainer l'exécution d'une unité de travail. 

Tableau 4-10. Patron État-Transition.

4.3.2. Patron NATURE

Ce patron est basé sur le méta-modèle de processus orienté contexte NATURE présenté au Chapitre 3. Il permet de représenter différents types de contexte sous forme d'une hiérarchie.

Nom	NATURE
Classification	Domaine ^ Produit ^ Méta-modélisation ^ Contexte

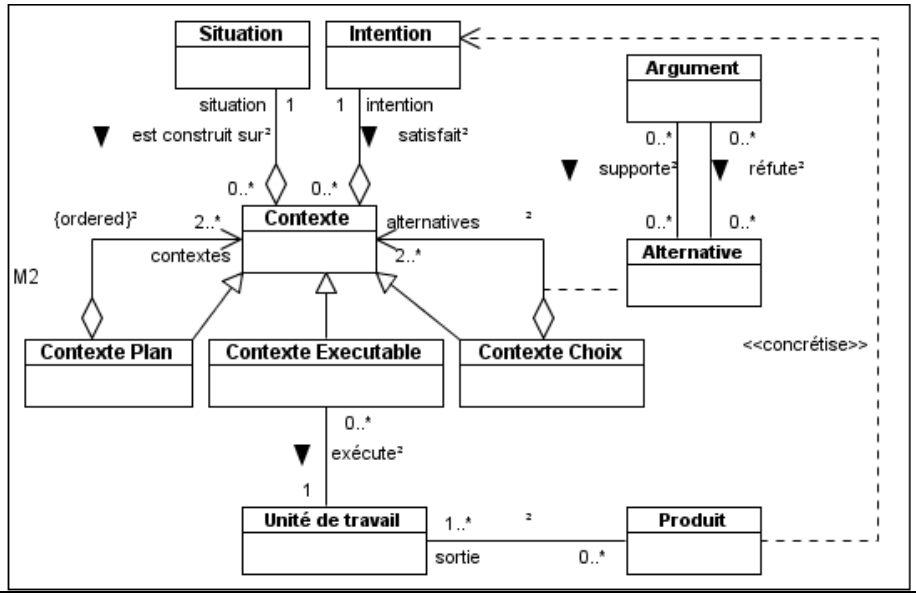
Contexte	La classe *Contexte* doit être définie dans le méta-modèle de processus.
Problème	Permet de représenter des méta-modèles de processus d'ingénierie de SI orientés contexte.
Force	Ce patron permet de spécifier trois types de contexte : – contexte plan qui précise de manière ordonnée quels sont les plans qui le composent ; – contexte choix qui permet de choisir un contexte parmi plusieurs en se basant sur des alternatives et des arguments ; – contexte exécutable qui engendre l'exécution d'une unité de travail qui modifie ou crée un produit.
Solution-modèle	Une structure de contextes est composée de contextes plans qui doivent être tous réalisés selon un ordre, de contextes choix pouvant être réalisés ou non et de contextes exécutables qui lancent l'exécution d'unités de travail produisant des produits, qui concrétisent des intentions. 

Tableau 4-11. Patron NATURE.

4.3.3. Patron MAP

Le patron MAP a été extrait du méta-modèle de processus du même nom. Il permet de représenter des structures complexes de stratégies et d'intentions.

Nom	MAP
Classification	Domaine ^ Produit ^ Méta-modélisation ^ Stratégie
Contexte	La classe Stratégie doit être définie dans le méta-modèle de processus.
Problème	Permet de représenter des méta-modèles de processus d'ingénierie de SI orientés stratégie.
Force	Ce patron permet de modéliser plusieurs chemins possibles pour atteindre une intention.

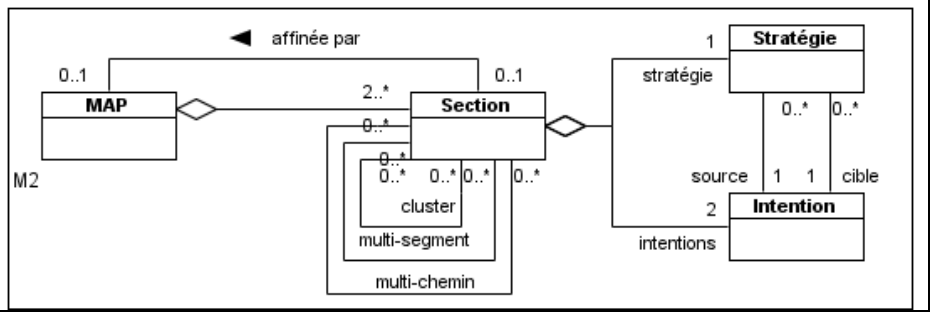
Solution-modèle	– Une section est composée d'une stratégie et de deux intentions. Deux intentions sont liées par une stratégie. – Une MAP est composée de plusieurs sections. – Plusieurs types de section peuvent être définis : – multi-segment : deux stratégies complémentaires ou plus permettent d'atteindre la même intention à partir de la même intention source ; – multi-chemin : il existe plusieurs chemins dans une MAP pour atteindre la même intention à partir de la même intention source ; – cluster : deux stratégies mutuellement exclusives ou plus permettent d'atteindre la même intention à partir de la même intention source ; – une section peut être affinée par une MAP (une section peut donc être décomposée en plusieurs sous-sections). 

Tableau 4-12. Patron MAP.

4.3.4. Unité de temps

Ce patron de domaine permet de compléter la classe Unité de travail pour prendre en compte l'aspect temporel d'une unité de travail.

Nom	Unité de temps
Classification	Domaine ^ Produit ^ Méta-modélisation ^ Activité
Contexte	La classe Unité de travail doit être définie dans le méta-modèle de processus.
Problème	Permet de représenter le concept d'unité de temps dans les méta-modèles de processus.
Force	Ce patron permet de séparer les unités de travail du temps qui les organisent. Phase et Cycle de vie sont considérés comme des unités de temps par exemple.
Solution-modèle	Une unité de travail peut être exécutée selon une ou plusieurs unités de temps.

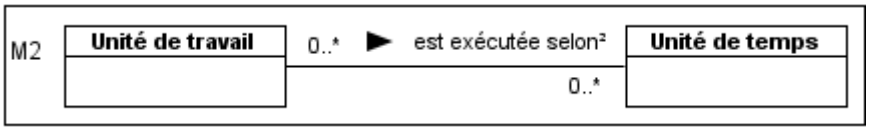
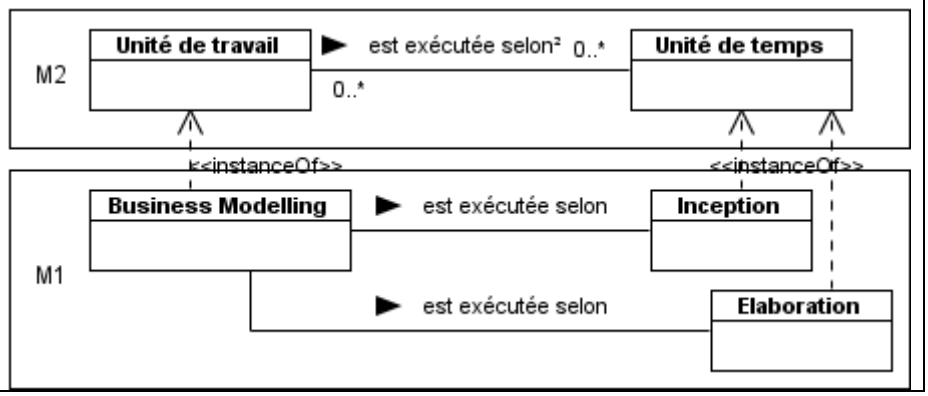
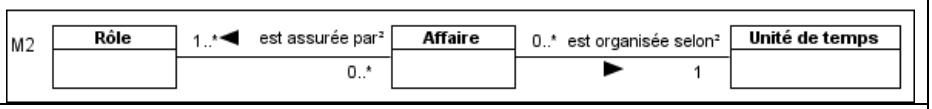
	
Exemple	Dans le Rational Unified Process, l'unité de travail Business Modelling est exécutée selon les unités de temps Inception, Elaboration et Construction (non représentée dans l'exemple). 

Tableau 4-13. Patron Unité de temps.

4.3.5. Affaire

Ce patron de domaine est issu du méta-modèle de l'OPF (OPF, 2005). Il permet de définir une affaire conclue entre des rôles sur un temps donné.

Nom	Affaire
Classification	Domaine ^ Produit ^ Méta-modélisation ^ Activité
Contexte	La classe Rôle ou la classe Unité de temps doit être définie dans le méta-modèle de processus.
Problème	Permet de représenter le concept d'affaire dans les méta-modèles de processus.
Force	Ce patron permet de représenter des affaires, c'est-à-dire des collaborations entre des rôles durant un certain temps.
Solution-modèle	Une affaire est assurée par un ou plusieurs rôles et est organisée selon une unité de temps. 
Exemple	Dans le cadre d'un projet, plusieurs équipes peuvent être impliquées durant un cycle de vie correspondant à celui du modèle de la cascade. Lorsque le cycle est terminé, le projet se termine également.

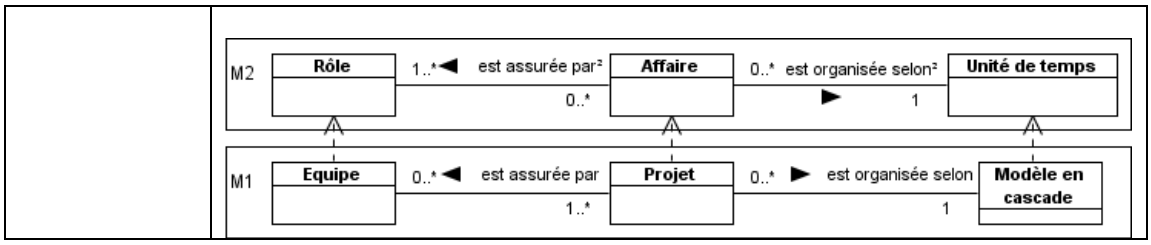


Tableau 4-14. Patron Affaire.

4.4. Exemple d'imitations de patrons

Nous présentons ici un exemple de méta-modèle de processus que l'on peut construire, en utilisant le méta-modèle de domaine et les patrons génériques et de domaine.

Par exemple, le patron générique « Concept-Catégorie de concepts » a été imité sur les classes *Produit* et *Unité de travail* du méta-modèle. Le patron générique « Composition-Agrégation réflexive » a été imité sur les classes *Intention*, *Produit*, *Catégorie de Produit*, *Unité de travail* et *Catégorie d'unité de travail*. Enfin, le patron générique « Association réflexive » a été imité deux fois sur la classe *Unité de travail*.

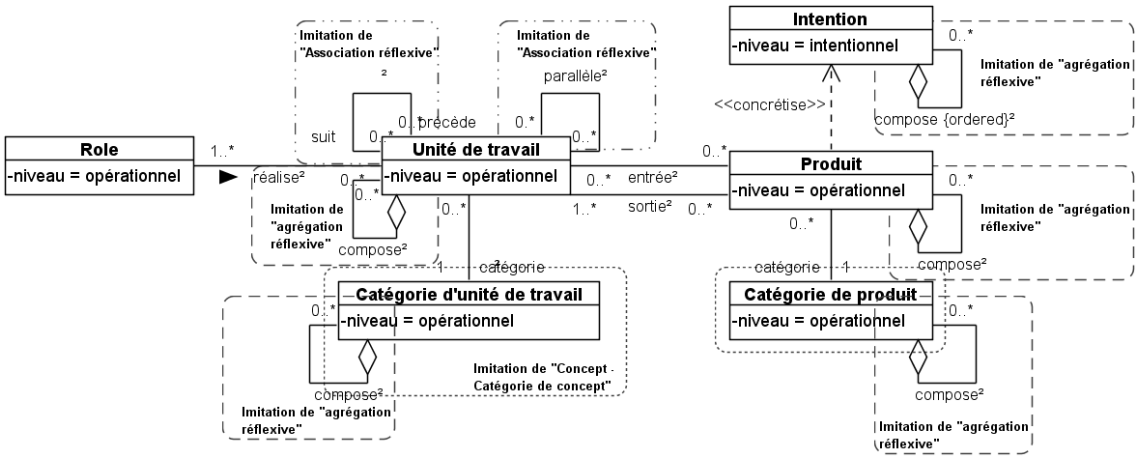


Figure 4-5. Exemple de méta-modèle de processus avec imitations de patrons.

4.5. Graphe conceptuel

À partir du cube complétude – précision – abstraction et des ressources présentées dans ce chapitre, nous proposons un graphe conceptuel représentant l'ensemble des concepts de la méta-modélisation des processus d'ingénierie de SI (cf. Figure 4-7). Chaque nœud du graphe conceptuel correspond à un concept de la méta-modélisation des processus et peut être intégré sous forme de classe ou d'association dans les méta-modèles de processus construits par les ingénieurs des méthodes.

Les concepts principaux (en gras dans la Figure 4-7) correspondent aux classes du méta-modèle de domaine présenté dans la section 0, les concepts secondaires correspondent à l'imitation de patrons génériques ou de domaine sur ces classes.

Les relations définies dans le graphe conceptuel correspondent aux trois dimensions complétude – précision – abstraction définies dans le cube (cf. section 2 de l'Introduction à la proposition). La complétude et l'abstraction permettent d'atteindre des concepts correspondant à des classes du méta-modèle de domaine. La relation de

précision permet d'atteindre des concepts correspondant à l'imitation de patrons génériques et de domaine sur ces classes. Nous présentons en détail ci-après les concepts et les relations.

4.5.1. Les concepts

Deux types de concepts sont présents dans le graphe conceptuel :

- les concepts principaux qui correspondent aux classes définies dans le méta-modèle de domaine (en gras dans la Figure 4-7). Ce sont les concepts principaux de la méta-modélisation des processus d'ingénierie de SI. Ces concepts sont : Intention, Stratégie, Contexte, Situation, Produit, Problème, Alternative, Argument, Rôle, Condition et Unité de travail. Ces concepts sont atteints par les relations de complétude et d'abstraction.
- les concepts secondaires qui correspondent à l'imitation des patrons génériques ou de domaine sur les concepts principaux. Ces concepts sont atteints via la relation de précision.

Chaque point de vue de la modélisation des processus d'ingénierie de SI peut être retrouvé dans le graphe conceptuel, comme le montre la Figure 4-6.

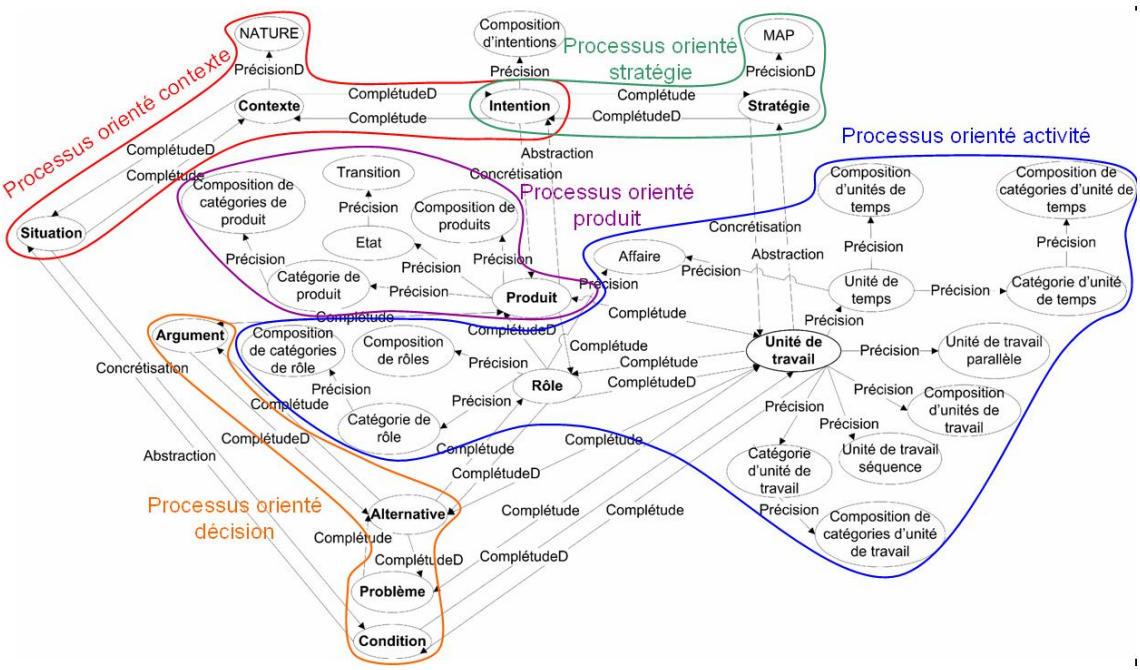


Figure 4-6. Les points de vue dans le graphe conceptuel.

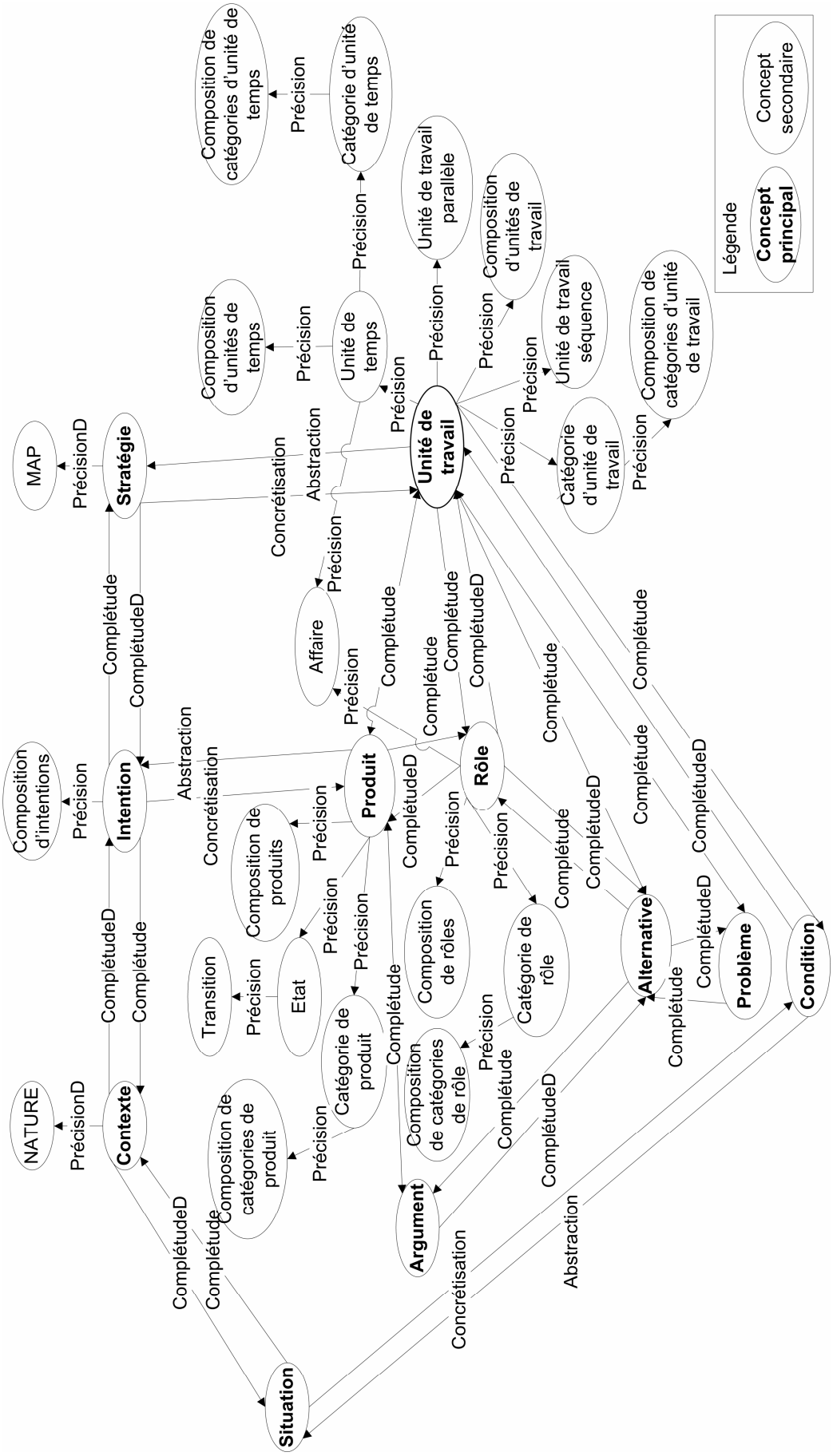


Figure 4-7. Graphe conceptuel.

4.5.2. Les relations

Les relations sont issues de l'espace en trois dimensions inspiré de Merise 2 (Panet et Letouche, 1994) : complétude, précision et abstraction.

a) Complétude

La relation de complétude permet d'étendre la couverture des méta-modèles de processus en cours de construction. La complétude est basée sur le méta-modèle de domaine défini dans la section 0. Cette relation permet de lier uniquement des concepts principaux. La Figure 4-8 montre que le concept Unité de travail étend le concept Produit et inversement, le concept Produit étend le concept Unité de travail. La relation de complétude est :

- symétrique : tout concept A qui étend un autre concept B est lui-même étendu par cet autre concept B. Cette relation est présentée de manière bidirectionnelle dans la Figure 4-8 entre le concept Unité de travail et le concept Produit ;
- non-transitive : un concept A étendant un concept B, lui-même étendant un concept C, n'étend pas ce concept C ;
- non-réflexive : un concept ne peut s'étendre lui-même.

Figure 4-8. Exemple de la relation de complétude.

Nous introduisons un second type de relation de complétude qui est complétudeD. Cette relation permet de spécifier qu'un concept est dépendant d'un autre concept, comme nous l'avons présenté dans la section 1.1.1.a). Par exemple, le concept Rôle ne peut exister tout seul, il doit être obligatoirement lié à Unité de Travail ou à Produit, la relation complétudeD est donc définie entre ces concepts. La relation complétudeD est non-symétrique, non-transitive et non-réflexive.

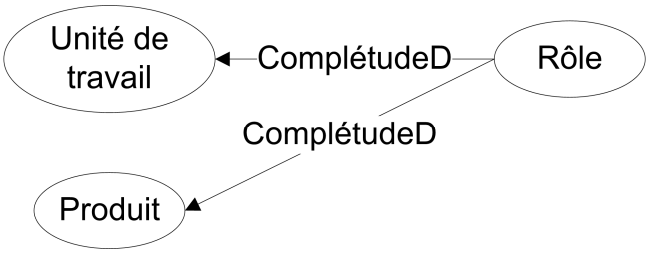


Figure 4-9. Exemple de la relation de complétudeD.

b) Abstraction

Les niveaux d'abstraction permettent de représenter les classes des méta-modèles de processus en cours de construction sur les deux niveaux d'abstraction intentionnel et opérationnel. La relation d'abstraction permet d'abstraire un concept. Elle ne lie que des concepts principaux. Par exemple, le concept Unité de travail est abstrait par le concept Stratégie, cf. Figure 4-10. La relation d'abstraction est :

- non-symétrique : un concept A qui abstrait un autre concept B n'est pas abstrait par cet autre concept B ;

- non-transitive : un concept A abstrayant un concept B, lui-même abstrayant un concept C, n'abstrait pas ce concept C ;
- non-réflexive : un concept ne peut s'abstraire lui-même.

La relation inverse d'abstraction est concrétisation qui a les mêmes propriétés que la relation d'abstraction. Par exemple, le concept Stratégie est concrétisé par le concept Unité de travail.

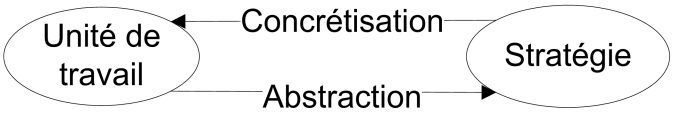


Figure 4-10. Exemple de la relation d'abstraction et de concrétisation.

c) Précision

La dimension de précision permet de raffiner, de détailler une classe du méta-modèle de processus en cours de construction en utilisant des patrons génériques ou de domaine. Par exemple, le concept Unité de travail peut être raffiné par les concepts Catégorie d'unité de travail et Composition d'unités de travail, cf. Figure 4-11.

La relation de précision est :

- non-symétrique : un concept A qui affine un autre concept B n'est pas affiné par cet autre concept B ;
- non-transitive : un concept A affinant un concept B, lui-même affinant un concept C, n'affine pas ce concept C ;
- non-réflexive : un concept ne peut s'affiner lui-même.

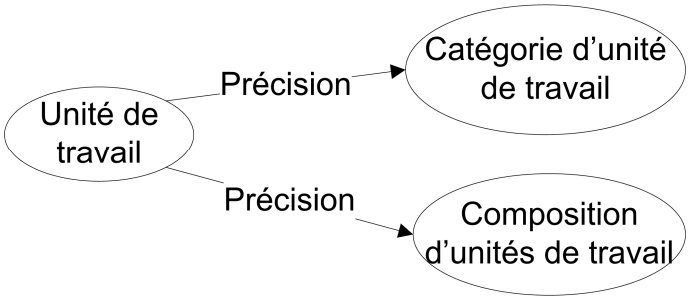


Figure 4-11. Exemple de la relation de précision.

Nous avons également introduit une relation de précision plus forte appelée précisionD. Ce type de relation permet de spécifier qu'un concept est obligatoirement précisé par un autre concept. L'exemple de la Figure 4-12 montre que le concept Stratégie est obligatoirement précisé par le concept MAP. De même, tel que défini dans le graphe conceptuel, le concept Contexte est obligatoirement précisé par le concept NATURE.

Figure 4-12. Exemple de la relation de précisionD.

4.5.3. Évolution du graphe conceptuel

Le graphe conceptuel constitue la ressource essentielle de notre méthode. Il permet aux ingénieurs des méthodes de naviguer entre tous les concepts de la méta-modélisation des processus d'ingénierie de SI.

Le graphe est défini de manière statique. Il permet ainsi de cadrer l'ensemble des concepts pouvant être utilisés dans la méta-modélisation des processus d'ingénierie de SI à un instant donné. L'aspect statique du graphe permet une utilisation plus rapide qu'un graphe qui serait généré à la volée. Cependant, le graphe que nous proposons est moins souple qu'un graphe dynamique puisque les concepts sont prédéfinis.

L'aspect évolutif de notre graphe conceptuel est donc primordial, afin de contrer ce défaut de souplesse. Lorsqu'une nouvelle classe est intégrée au méta-modèle de domaine ou lorsqu'un nouveau patron de méta-modélisation peut être imité sur des classes du méta-modèle de domaine, un expert peut mettre à jour le graphe conceptuel pour définir :

- des nouveaux concepts principaux dans le cas de nouvelles classes dans le méta-modèle de domaine ;
- des concepts secondaires dans le cas d'un nouveau patron.

4.6. Conclusion

Dans ce chapitre, nous avons présenté le méta-modèle de domaine qui regroupe les principales classes et associations des méta-modèles de processus existants pour l'ingénierie des systèmes d'information.

Dans une seconde partie, nous avons présenté différents patrons génériques pour la méta-modélisation des processus d'ingénierie de SI. Nous avons également exposé les différents patrons de domaine, issus des méta-modèles de processus existants. Nous avons ensuite présenté un bref exemple de méta-modèle de processus et les différents patrons génériques et de domaine utilisés pour le créer.

Nous avons enfin proposé un graphe conceptuel composé de l'ensemble des concepts pouvant être pris en compte dans un méta-modèle de processus. Les concepts proviennent du méta-modèle de domaine et de l'imitation des patrons génériques et de domaine sur ses classes. Les relations définies entre les concepts sont issues du cube à trois dimensions : complétude, précision, abstraction.

Le graphe conceptuel constitue le point de départ de la méthode que nous présentons dans le chapitre suivant.

5. UNE METHODE POUR LA CONSTRUCTION DE META-MODELES DE PROCESSUS D'INGENIERIE DE SYSTEMES D'INFORMATION

Dans ce chapitre, nous présentons notre principale contribution à l'ingénierie des processus qui consiste en une méthode pour construire des méta-modèles de processus multi-points de vues, adaptables et fédérateurs pour l'ingénierie des systèmes d'information. Nous proposons également un guide pour instancier ces méta-modèles afin de représenter des modèles de processus pour l'ingénierie des SI.

Dans un premier temps, nous détaillons la méthode pour créer un méta-modèle de processus, puis nous montrons comment ce méta-modèle peut être instancié. Une étude de cas illustre la méthode.

5.1. Création du méta-modèle de processus

Nous présentons ici la méthode qui utilise le graphe conceptuel ainsi que le méta-modèle de domaine et les patrons présentés dans le Chapitre 4. La méthode est composée de deux activités principales :

- la sélection d'un concept ;
- l'intégration du concept.

Ces deux activités sont réitérées jusqu'à obtenir un Méta-modèle de Processus en Cours de Construction (MPCC) complet.

La dernière activité, Ajout des attributs, permet de compléter le méta-modèle de processus obtenu avec des attributs.

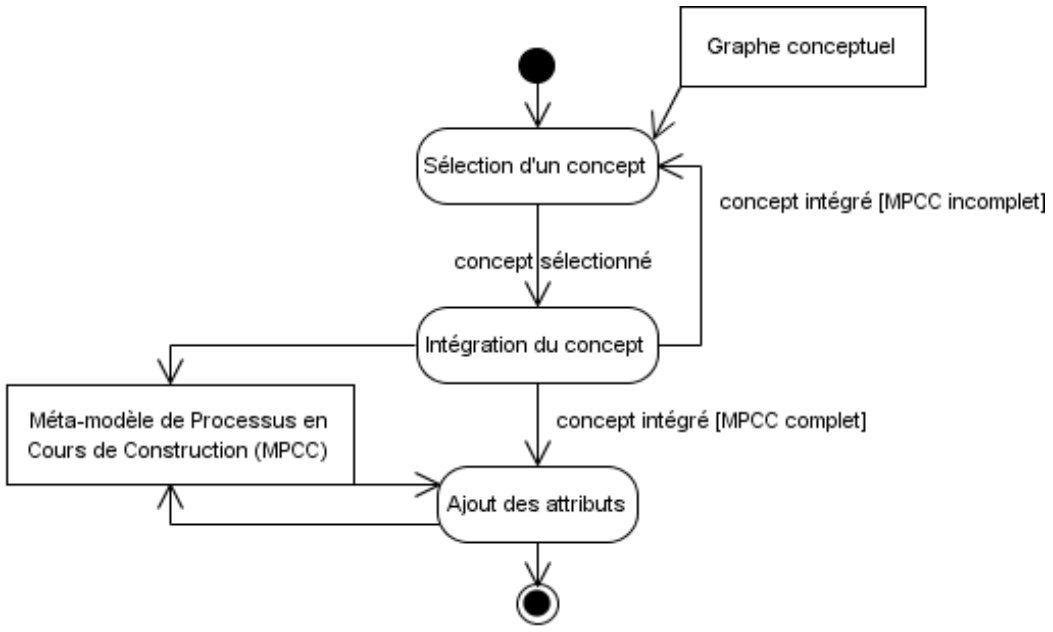


Figure 5-1. Méthode pour la construction du méta-modèle de processus.

L'ingénieur des méthodes peut sélectionner autant de concepts que nécessaire. Autant d'ajouts de classes ou d'imitations de patrons seront réalisés dans le méta-modèle de processus.

Lorsque tous les concepts nécessaires ont été sélectionnés et donc intégrés au méta-modèle de processus, les itérations peuvent être arrêtées. Des attributs peuvent être

ajoutés et le méta-modèle de processus obtenu peut être instancié pour définir un ou des modèles de processus pour l'ingénierie des SI.

5.1.1. Le Méta-modèle de Processus en Cours de Construction

L'objectif de notre méthode est de créer un méta-modèle de processus d'ingénierie de SI répondant aux contraintes et spécificités de chaque organisation.

Le MPCC est construit au fur et à mesure de l'exécution de la méthode, basée sur le graphe conceptuel. Le MPCC correspond donc à un sous-ensemble connexe du graphe conceptuel.

Le MPCC doit respecter un certain nombre de contraintes statiques qui découlent directement de la structure du graphe conceptuel décrit dans le Chapitre 4. De plus, la méthode elle-même est guidée par des contraintes dynamiques qui garantissent le respect des contraintes statiques lors de la construction du méta-modèle de processus. Les contraintes dynamiques sont présentées au fur et à mesure de la présentation de la méthode. Les contraintes statiques sont présentées ci-dessous.

Afin de vérifier que le Méta-modèle de Processus en Cours de Construction est cohérent, nous avons défini quelques règles.

> R1 : chaque classe du Méta-modèle de Processus en Cours de Construction doit être associée à au moins une autre classe du méta-modèle.

Aucune classe du Méta-modèle de Processus en Cours de Construction ne peut être isolée des autres classes, chaque classe doit donc être associée à au moins une autre classe du méta-modèle de processus. Cette incohérence est a priori impossible étant donnée la méthode qui restreint l'ajout des classes dans le méta-modèle de processus.

> R2 : toutes les classes dépendantes du méta-modèle en cours de construction doivent être associées avec la ou les classes dependees.

Si le Méta-modèle de Processus en Cours de Construction contient des classes dépendantes, les classes dependees doivent être dans le méta-modèle et être associées aux classes dépendantes. De plus, tous les concepts ayant pour source la relation précisionD doivent également être associés aux concepts cibles de la relation, dans le méta-modèle de processus correspondant. Cette incohérence est également évitée au travers du cadre de la méthode qui fait les ajouts des classes dependees tout au long du processus, grâce aux relations définies dans le graphe conceptuel.

> R3 : les associations et dépendances intégrées au Méta-modèle de Processus en Cours de Construction ne peuvent être supprimées.

Afin de respecter la cohérence du Méta-modèle de Processus en Cours de Construction et par rapport au méta-modèle de domaine défini, aucune association ni dépendance stéréotypée ajoutée au méta-modèle ne peut être supprimée.

5.1.2. Sélection d'un concept

La première étape de la méthode consiste à sélectionner un concept. Lors de la première itération, le concept devra être choisi via un dictionnaire de concepts présentant la définition, des synonymes et des exemples de chaque concept du graphe conceptuel afin de faciliter la sélection. Dans les itérations suivantes, le concept pourra être choisi de deux manières : via le dictionnaire des concepts ou via les relations partant de ce concept dans le graphe conceptuel : par précision, complétude, abstraction ou concrétisation.

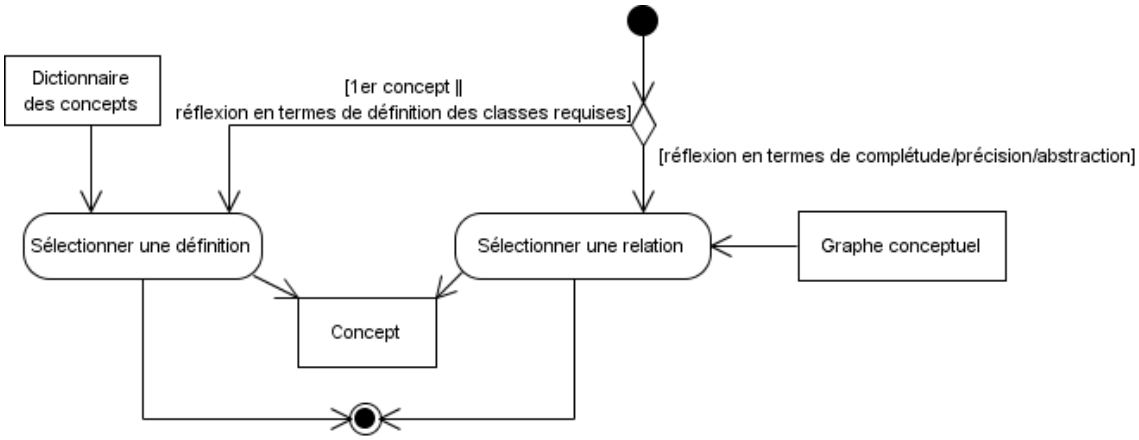


Figure 5-2. Choisir un concept.

a) Sélectionner une définition

Lors de la première itération, seules les définitions des concepts principaux peuvent être sélectionnées. Le Tableau 5-1 présente la définition, les synonymes et exemples pour chacun des principaux concepts du graphe conceptuel qui correspondent aux classes du méta-modèle de domaine.

Concepts	Définitions	Synonymes	Exemples
OPERATIONNEL			
Unité de travail	Tâche exécutée durant le processus d'Ingénierie de Systèmes d'Information (ISI).	Opération, exécution, action	Définir le système, sélectionner les besoins des utilisateurs, implémenter les composants, analyse structurelle, analyse dynamique
Condition	Contrainte sur une unité de travail.		Pré-condition, post-condition
Produit	Quelque chose qui est produit, utilisé ou modifié par une unité de travail durant le processus d'ISI.	Résultat, production	Modèle de conception, cahier des charges, modèle d'analyse,

			user stories
Rôle	Quelqu'un ou quelque chose qui réalise une unité de travail durant le processus d'ISI ou qui est responsable d'un produit.	Acteur	Développeur, analyste, système informatique
Problème	Problème rencontré durant le processus d'ISI.	Question, difficulté, dilemme	Faut-il modéliser le contexte métier ?
Alternative	Réponse à un problème rencontré durant le processus d'ISI.	Choix, possibilité, éventualité, option	Oui le contexte métier doit être modélisé, Non le contexte métier n'a pas besoin d'être modélisé
Argument	Preuve pour supporter ou rejeter une alternative.	Raison, motif	Le processus métier est conséquent
INTENTIONNEL			
Intention	Objectif du processus d'ISI.	But	Décrire entité, Décrire attribut
Stratégie	Manière dont une intention est atteinte à partir d'une intention source.	Tactique, approche	Stratégie d'alternative, stratégie d'affinement
Situation	Situation à un instant donné du processus d'ISI.	Position	Entité décrite, attribut décrit
Contexte	Couple formé d'une intention et d'une situation à un instant donné du processus d'ISI.		(Entité décrite ; Décrire attribut)

Tableau 5-1. Dictionnaire des concepts principaux.

Lors des itérations suivantes, il sera possible de sélectionner d'autres définitions : celles des concepts principaux qui n'ont pas encore été intégrés dans le méta-modèle en cours de construction et celles correspondant aux autres concepts représentants l'imitation des patrons génériques ou de domaine (cf. Tableau 5-2).

Concepts	Définitions	Synonymes	Exemples
OPERATIONNEL			
Unité de travail parallèle	Des unités de travail peuvent s'exécuter en parallèle.		Analyse est exécutée en parallèle d' Architecture technique
Unité de travail séquence	Des unités de travail peuvent s'exécuter en séquence.		Analyse structurelle précède Analyse dynamique

Composition d'unités de travail	Sous-ensemble d'unités de travail.		Analyse est composée de l'analyse structurelle et de l'analyse dynamique
Catégorie d'unité de travail	Ensemble d'unités de travail ayant des caractéristiques communes.		Activité, tâche
Composition de catégories d'unité de travail	Sous-ensemble d'unités de travail ayant des caractéristiques communes.		Une activité est composée de tâches.
État	Différents états d'un produit.		Validé, en cours de construction
Transition	Transitions entre les différents états d'un produit.		Fin du projet, modification
Composition de produits	Sous-ensemble de produits.		Le modèle d'analyse est composé de la description des objets métier et de la cartographie des objets métier.
Catégorie de produit	Ensemble de produits ayant des caractéristiques communes.		Document, texte, modèle, diagramme, logiciel
Composition de catégories de produit	Sous-ensemble de produits ayant des caractéristiques communes.		Un document est composé de modèles et de textes.
Unité de temps	Temps durant lequel une unité de travail est exécutée.		Inception, Elaboration, cycle de vie du RUP
Composition d'unités de temps	Sous-ensemble d'unités de temps		Le cycle de vie du RUP est composé d'Inception et Elaboration
Catégorie d'unité de temps	Ensemble d'unités de temps ayant des caractéristiques communes		Phase, cycle de vie
Composition de catégories d'unité de temps	Sous-ensemble d'unités de temps ayant des caractéristiques communes.		Un cycle de vie est composé de phases.
Affaire	Collaboration entre des		Un projet peut

	rôles dans une unité de temps définie.		faire collaborer plusieurs entreprises durant un certain cycle de vie.
Composition de rôles	Sous-ensemble de rôles		L'équipe de développement est composée d'un chef de projet et d'un développeur.
Catégorie de rôle	Ensemble de rôles ayant des caractéristiques communes.		Équipe, personne, projet
Composition de catégories de rôle	Sous-ensemble de rôles ayant des caractéristiques communes.		Une équipe est composée de personnes.
INTENTIONNEL			
Composition d'intentions	Sous-objectifs d'un processus d'ISI.	Sous-but	Décrire entité est composée d'Attacher attribut et Décrire attribut.
MAP	Graphe composé de stratégies reliant des intentions.		
NATURE	Arbre de contextes et de sous-contextes.		

Tableau 5-2. Dictionnaire des autres concepts.

Cependant, afin de garantir la cohérence du Méta-modèle de Processus en Cours de Construction, seules les définitions des concepts cibles des relations de complétude – abstraction - précision des concepts déjà intégrés seront présentées. Ainsi, sur l'exemple de la Figure 5-3, il ne sera pas possible de sélectionner la définition du concept Composition de catégories d'unité de travail tant que le concept Catégorie d'unité de travail ne sera pas intégré au méta-modèle en cours de construction. Si Unité de travail est déjà intégrée, seules les définitions des concepts Catégorie d'unité de travail et Composition d'unités de travail pourront être sélectionnées.

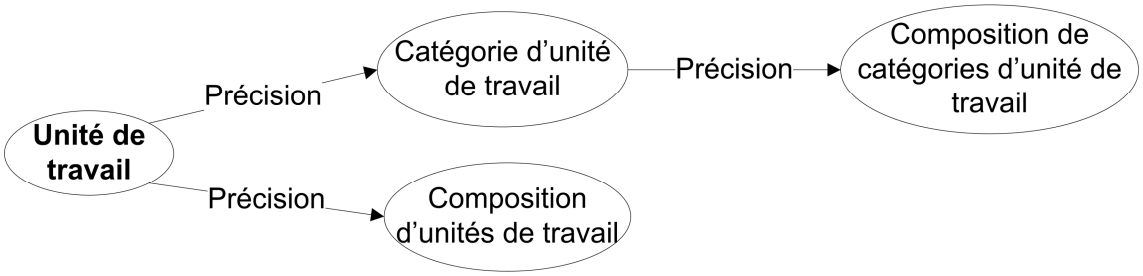


Figure 5-3. Exemple pour la sélection de définitions.

Nous fixons les deux règles suivantes permettant d'assurer la cohérence du Méta-modèle de Processus en Cours de Construction lors de la sélection des définitions.

R4 : à la 1ère itération, seules les définitions des concepts principaux peuvent être sélectionnées.

R5 : à partir de la seconde itération, seules les définitions des concepts principaux et des concepts cibles des relations des concepts déjà intégrés dans le MPCC peuvent être sélectionnées.

b) Sélectionner une relation

À partir de la seconde itération, il est possible d'utiliser les relations du graphe conceptuel pour sélectionner les concepts. Lors de la première itération, une définition est sélectionnée, le point de départ pour l'itération suivante sera le concept correspondant dans le graphe conceptuel. Il est alors possible à partir du concept, de découvrir les relations qui partent de ce concept pour l'étendre (complétude), le raffiner (précision), l'abstraire ou le concrétiser (abstraction/concrétisation).

Par exemple, comme le montre la Figure 5-4, si le concept Unité de travail a été intégré, tous les concepts dont les relations proviennent d'Unité de travail peuvent être sélectionnés.

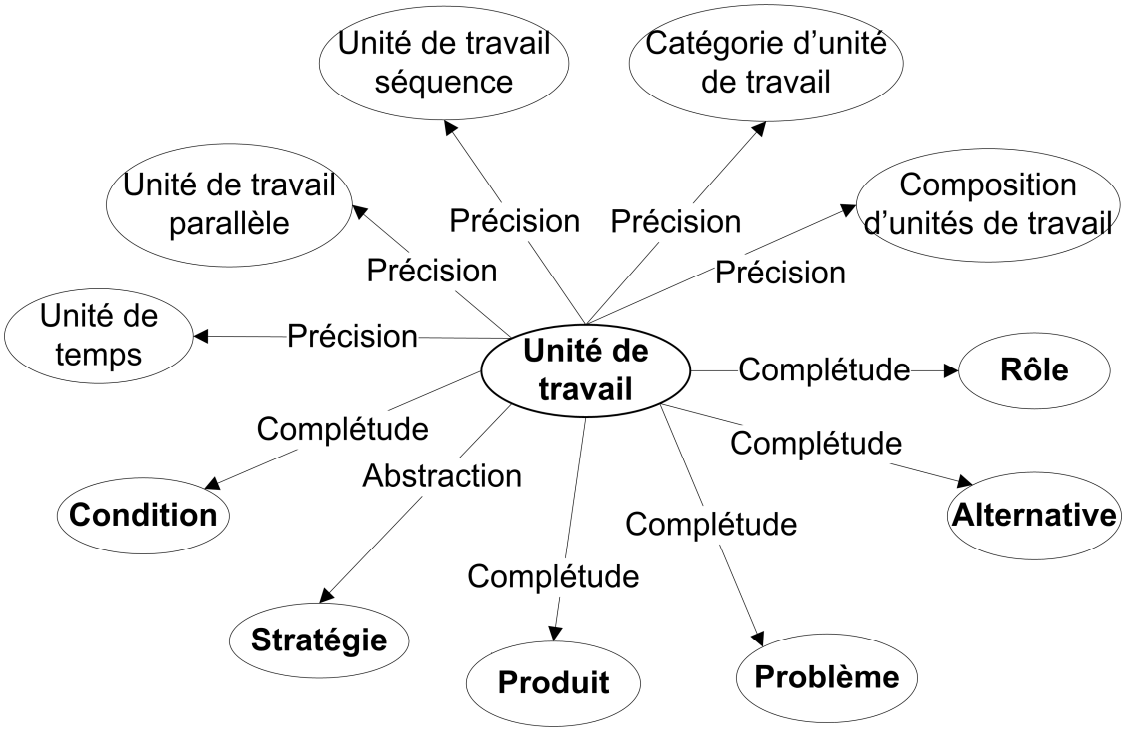


Figure 5-4. Exemple pour la sélection de relation.

Nous fixons la règle suivante, pour assurer la cohérence du Méta-modèle de Processus en Cours de Construction lors de la sélection des relations du graphe conceptuel.

R6 : seules les relations des concepts déjà intégrés dans le MPCC peuvent être sélectionnées.

5.1.3. Intégration d'un concept

L'intégration du concept est différente selon le type de concept sélectionné. Si le concept est :

- primaire, il s'agira d'intégrer la classe correspondante dans le Méta-modèle de Processus en Cours de Construction (MPCC) ;
- secondaire, le concept correspond à un patron générique ou à un patron de domaine qu'il faut imiter sur la classe correspondant au concept source dans le graphe conceptuel.

Dans le cas où une classe du méta-modèle de domaine est intégrée dans le méta-modèle en cours de construction et que cette classe est dépendante d'une autre classe, celle-ci est également intégrée, ainsi que toutes les associations définies dans le méta-modèle de domaine entre les classes déjà intégrées dans le Méta-modèle de Processus en Cours de Construction.

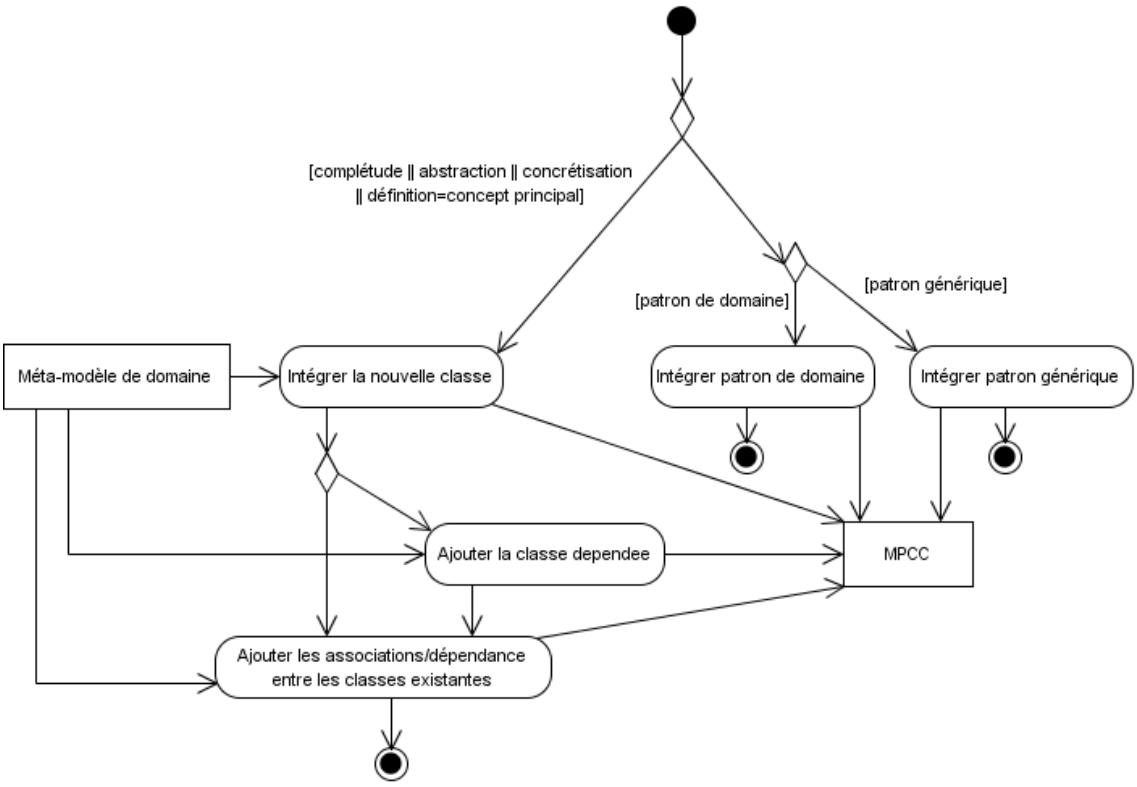


Figure 5-5. Intégrer un concept.

a) Intégrer la nouvelle classe

Lorsqu'un concept est choisi et qu'il correspond à une classe du méta-modèle de domaine, celle-ci doit être intégrée dans le Méta-modèle de Processus en Cours de Construction. De plus, si cette classe est dépendante d'autres classes, la ou les classes dependees doivent être également intégrées au MPCC. Enfin, il faut intégrer au MPCC les associations et dépendances définies dans le méta-modèle de domaine afin d'associer les classes venant d'être intégrées.

b) Intégrer un patron générique

Le patron générique est imité sur la classe concernée du Méta-modèle de Processus en Cours de Construction. Par exemple, le patron générique Concept-Catégorie de concepts sera imité sur la classe Unité de travail si le concept Catégorie d'unité de travail a été sélectionné dans le graphe conceptuel.

L'imitation consiste à reproduire les classes, associations et autres propriétés définies dans le patron, sur la classe concernée du méta-modèle de processus. Certains patrons génériques ne nécessitent pas d'adaptation supplémentaire, comme le patron Concept-Catégorie de concepts. Par contre, des patrons génériques comme Association réflexive ou Composition réflexive nécessitent des adaptations : renommage des associations, éventuellement des rôles et modifications des cardinalités.

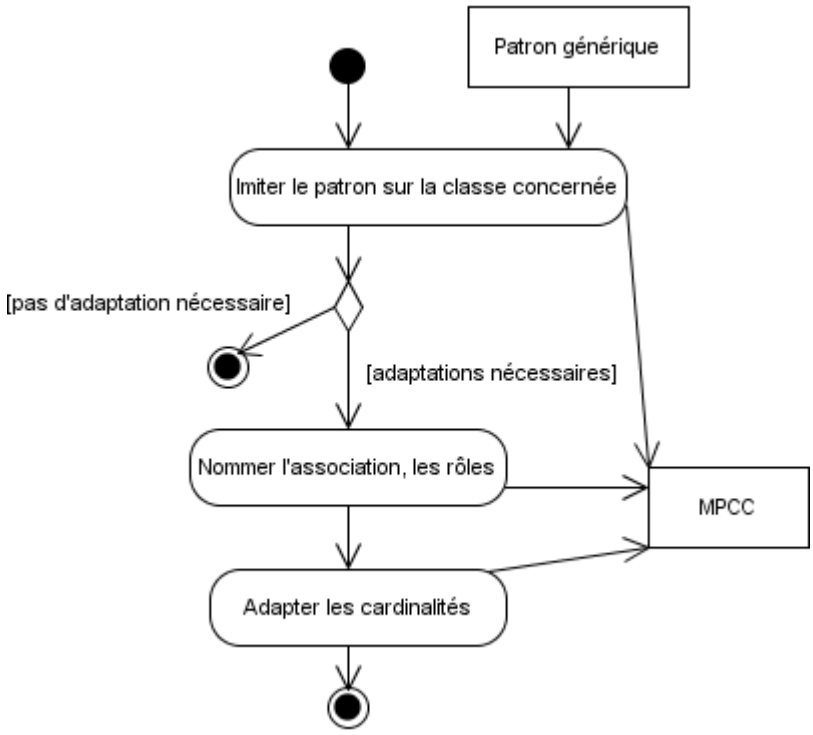


Figure 5-6. Intégrer un patron générique.

Par exemple le Tableau 5-3 présente les trois étapes d'imitation du patron générique Composition - Agrégation réflexive sur la classe Intention du MPCC.

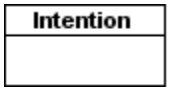
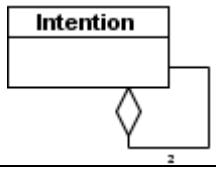
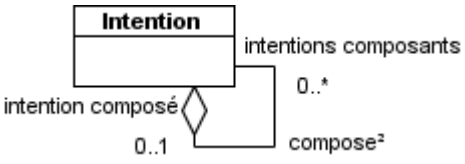
Classe du MPCC	Imitation du patron Composition - Agrégation réflexive	Nommage de l'association, des rôles et adaptations des cardinalités
		

Tableau 5-3. Exemple d'imitation du patron Composition - Agrégation réflexive.

c) Intégrer un patron de domaine

Pour l'intégration d'un patron de domaine, la première étape consiste en une analyse de similarité des classes entre le Méta-modèle de Processus en Cours de Construction (MPCC) et le patron de domaine, correspondant au concept choisi dans le graphe conceptuel.

Lorsque les classes communes sont trouvées, il faut ajouter les associations définies entre ces classes dans le patron de domaine au Méta-modèle de Processus en Cours de Construction.

Enfin, il faut ajouter les nouvelles classes du patron de domaine et les associations entre ces classes dans le Méta-modèle de Processus en Cours de Construction.

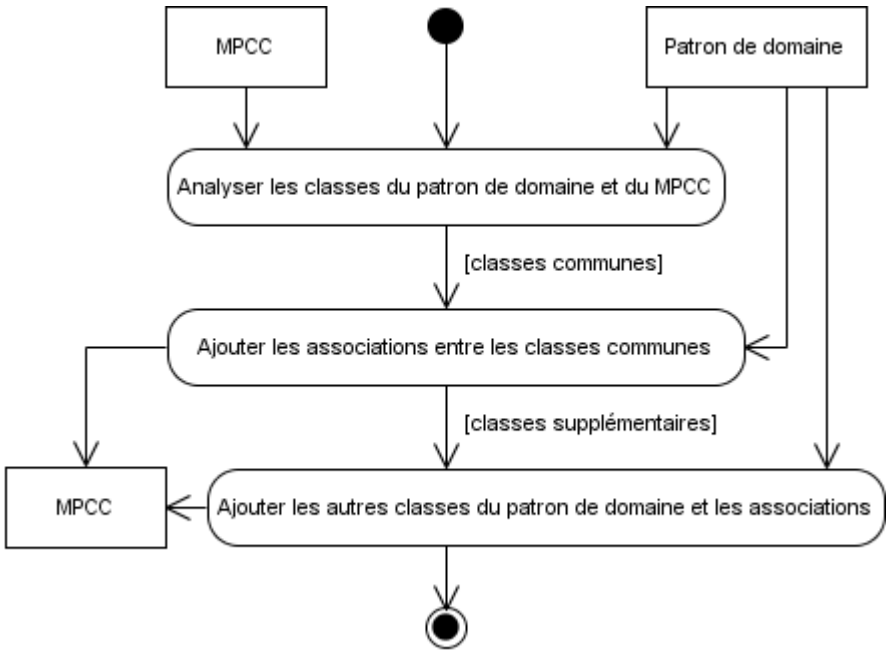


Figure 5-7. Intégrer un patron de domaine.

Le Tableau 5-4 présente un exemple d'imitation du patron de domaine Affaire à partir de la classe Rôle déjà présente dans le PMUC.

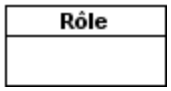
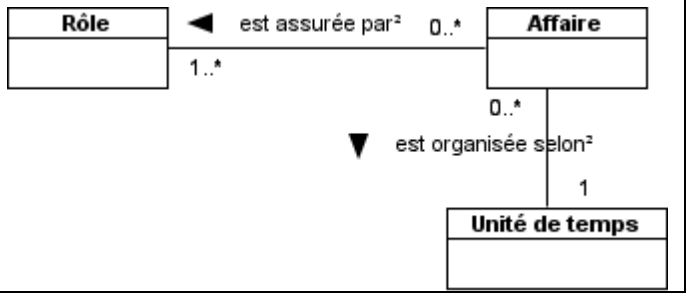
Analyse des classes communes	Ajout des associations entre classes communes	Ajout des autres classes et associations du patron de domaine
	Aucune association à ajouter	

Tableau 5-4. Exemple d'imitation du patron de domaine Affaire.

5.1.4. Ajout des attributs

Dans le Chapitre 4, nous avons présenté des attributs pertinents pour la modélisation des processus d'ingénierie de SI. Nous présentons ici comment la méthode permet de compléter le Méta-modèle de Processus en Cours de Construction en ajoutant des attributs aux classes.

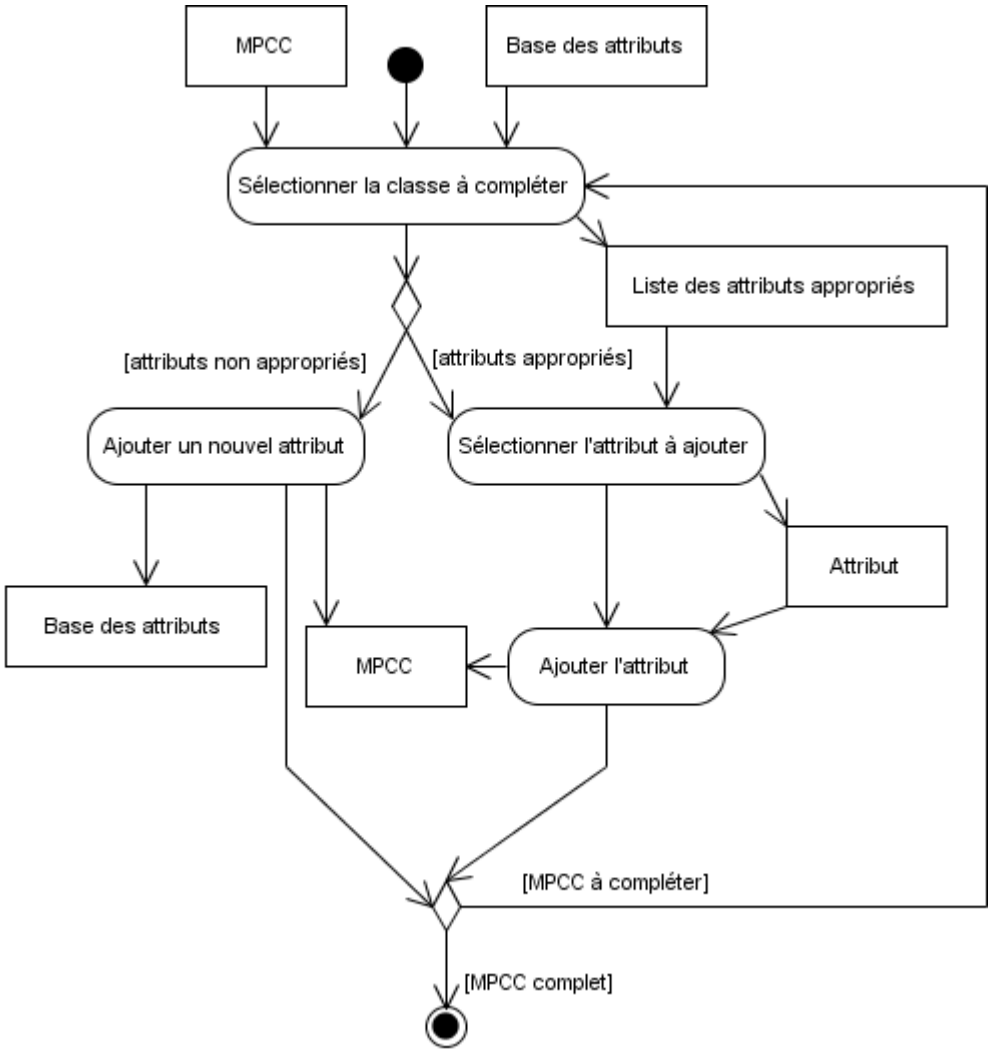


Figure 5-8. Ajouter un attribut à une classe du méta-modèle de processus.

L'ingénieur des méthodes doit d'abord choisir la classe du Méta-modèle de Processus en Cours de Construction à laquelle il veut ajouter des attributs. Tous les attributs pouvant être ajoutés à la classe sélectionnée sont proposés à l'ingénieur. Si un attribut l'intéresse, il le sélectionne et celui-ci est ajouté à la classe du Méta-modèle de Processus en Cours de Construction.

Les attributs pouvant être ajoutés aux classes correspondant aux concepts secondaires du graphe conceptuel sont en Annexe B.

Si aucun attribut proposé ne correspond à la propriété qu'il veut ajouter à la classe, l'ingénieur des méthodes a la possibilité d'ajouter un nouvel attribut.

Toutes les classes du méta-modèle doivent ainsi être complétées par des attributs.

Tous les attributs sont stockés dans une base qui sera enrichie au fur et à mesure de la création de nouveaux attributs.

5.1.5. Étude de cas

a) Présentation de l'étude de cas

Nous présentons ici l'extrait d'un cas d'étude du service des systèmes d'information de l'hôpital de Grenoble[10]. Ce service gère un système d'information composé d'une quarantaine d'applications destinées aux services de l'hôpital comme les services de soins (maternité, urgences, neurologie…) et les services supports (comptabilité, ressources humaines...). Ces applications sont utilisées par des assistants médicaux, des médecins et du personnel administratif. Le service SI s'occupe également de la mise à jour régulière de ces applications en fonction des nouveaux besoins des utilisateurs.

Le responsable du service SI souhaite modéliser les processus d'ingénierie de SI utilisés dans le service pour pouvoir ensuite améliorer leur cohérence et leur performance. Ainsi, un processus d'ingénierie de SI commun pourra être défini pour tous les chefs de projet SI et leurs équipes de développement.

D'autre part, la modélisation des processus d'ingénierie de SI permettra de collecter et réutiliser les connaissances en processus et gestion de projet des chefs de projet et développeurs du service, afin de produire les fonctionnalités du SI plus efficacement, en termes de temps et de ressources, et donc de coûts. La modélisation des processus est donc une étape importante pour une meilleure réalisation des projets.

Un des chefs de projet du service est chargé de réaliser l'analyse des processus d'ingénierie de SI existants et jouera donc le rôle de l'ingénieur des méthodes. Ces processus sont analysés à deux niveaux :

- auprès des chefs de services soins et supports ;
- auprès du personnel du service SI.

Les chefs de services soins et supports s'intéressent aux objectifs d'un processus d'ingénierie de systèmes d'information. Par exemple, dans le cas de la mise en place de la gestion centralisée du dossier patient, les chefs de service veulent connaître l'impact de la nouvelle fonctionnalité du SI sur leur service. Cet objectif peut être détaillé en plusieurs points dont, par exemple :

- connaître l'impact du changement sur l'organisation de leur service, c'est-à-dire les processus métier ;
- connaître les personnes de leur service impactées par le changement ;
- connaître l'impact sur l'informatisation, la mise à niveau informatique des bureaux et salles.

Dans le service SI, les chefs de projets et développeurs s'intéressent plus à l'aspect opérationnel des processus d'ingénierie de SI, c'est-à-dire aux différentes étapes à suivre pour réaliser les fonctionnalités, aux produits réalisés et aux acteurs intervenant durant les étapes.

Par exemple, l'activité de pré-étude fonctionnelle qui consiste à définir de manière globale les besoins des utilisateurs, est composée de plusieurs tâches dont :

- la rédaction du cahier des charges simplifié par l'analyste, qui permet de produire un cahier des charges simplifié qui est un document textuel. Ce

[10] http://www.chu-grenoble.fr/

document décrit la fonctionnalité étudiée, ainsi que les services concernés et les nouveautés introduites ;

- la constitution du glossaire des termes métiers pour produire un glossaire des termes métier qui est également un document textuel. Ce document permet de fixer un vocabulaire commun entre maîtrise d'œuvre et maîtrise d'ouvrage ;
- la modélisation des acteurs qui produit un diagramme d'acteurs qui est un modèle UML. Ce modèle décrit les acteurs concernés par la fonctionnalité du SI étudiée.

L'ensemble des documents produits par l'activité de pré-étude fonctionnelle constitue un document appelé dossier fonctionnel.

Nous allons présenter ici comment nous pouvons construire le méta-modèle de processus d'ingénierie de SI qui permet de définir les processus décrits ci-dessus. Ce méta-modèle sera ensuite instancié pour modéliser les processus d'ingénierie de SI tels que définis dans le cas d'étude.

b) Exécution de la méthode

La première étape de la méthode consiste à sélectionner un concept via le dictionnaire de concepts présenté dans la section 0. La première définition, « Représente un but ou objectif du processus d'ISI. » correspondant au concept d'Intention, permettrait de représenter les objectifs des chefs de services soins et supports. Ce concept est donc sélectionné et intégré au Méta-modèle de Processus en Cours de Construction sous la forme de la classe *Intention*, le concept Intention du graphe conceptuel correspondant à la classe *Intention* définie dans le méta-modèle de domaine. Aucune classe dependee n'est définie pour *Intention*, la première itération est donc terminée.

À partir de la deuxième itération, il est possible de sélectionner un nouveau concept via les définitions ou via les relations partant du concept Intention dans le graphe conceptuel. Les objectifs des chefs de services soins et supports pouvant être détaillés, il faut préciser le concept d'Intention, d'où le choix de sélectionner un nouveau concept par une réflexion en termes de relation de précision. Le seul concept sélectionnable à partir d'Intention avec la relation précision est le concept Composition d'intentions. Ce concept correspond à l'imitation du patron « Composition-Agrégation réflexive » sur la classe *Intention* du Méta-modèle de Processus en Cours de Construction. Une agrégation réflexive est donc ajoutée sur la classe *Intention*, on nomme l'agrégation « compose ». Les cardinalités ne sont pas modifiées, mais on ajoute la contrainte {ordered} afin de spécifier un ordre dans les sous-intentions.

La Figure 5-9 présente le chemin parcouru dans le graphe conceptuel durant les deux premières itérations de la méthode.

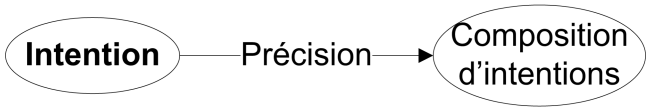


Figure 5-9. Chemin parcouru dans le graphe lors des deux premières itérations.

La Figure 5-10 présente le Méta-modèle de Processus en Cours de Construction correspondant aux deux itérations : intégration de la classe *Intention* et imitation du patron « Composition-Agrégation réflexive ».

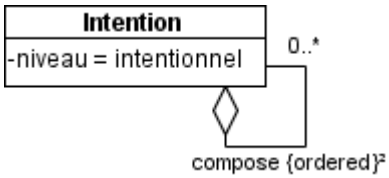


Figure 5-10. Méta-modèle en cours de construction suite aux deux premières itérations.

Les processus d'ingénierie de SI tels qu'ils sont perçus par les chefs de services soins et supports peuvent être entièrement définis par le Méta-modèle de Processus en Cours de Construction. Il faut maintenant enrichir le méta-modèle pour pouvoir définir les processus d'ingénierie de SI tels que vus par les chefs de projets et développeurs du service SI.

L'ingénieur des méthodes décide de sélectionner un nouveau concept via les définitions, car il ne sait pas exactement quel concept choisir à partir des relations définies dans le graphe conceptuel. Cependant, seules les définitions des concepts liés à Intention et Composition d'intentions dans le graphe conceptuel seront affichées, conformément à la règle R5. La définition « Représente quelque chose qui est produit, utilisé ou modifié durant le processus d'ISI. » correspondant au concept Produit, permettra de modéliser les documents et modèles produits durant le processus d'ingénierie de SI. Le concept Produit du graphe conceptuel correspond à la classe *Produit* du méta-modèle de domaine. Ce concept est donc sélectionné et intégré au Méta-modèle de Processus en Cours de Construction sous la forme d'une classe Produit. Les associations définies entre Produit et les classes existantes du Méta-modèle de Processus en Cours de Construction sont intégrées : il s'agit de la dépendance stéréotype « concrétise » entre les classes *Produit* et *Intention*. Aucune classe dependee n'est définie pour *Produit*. La troisième itération est donc terminée.

L'ingénieur des méthodes doit préciser le concept de Produit, afin de définir dans un premier temps qu'un produit peut être de type document textuel ou modèle UML. Pour cela, grâce à la relation de précision, il peut sélectionner le concept Catégorie de produit. Ce concept correspond à l'imitation du patron « Concept-Catégorie de concept » sur la classe *Produit* du méta-modèle en cours de construction. La classe *Catégorie de produit* est donc ajoutée au méta-modèle de processus ainsi que l'association entre celle-ci et *Produit*. Aucune autre adaptation n'est à faire lors de l'imitation de ce patron.

Dans un deuxième temps, l'ingénieur des méthodes doit spécifier qu'un produit comme « Dossier fonctionnel » est composé d'un produit comme « Diagramme

d'acteurs ». Pour cela, il doit préciser le concept Produit grâce au concept Produit composition. Le patron « Composition-Agrégation réflexive » est imité sur la classe *Produit* du Méta-modèle de Processus en Cours de Construction. Une agrégation réflexive est donc ajoutée, elle est nommée « Compose ».

Enfin, l'ingénieur des méthodes doit spécifier qu'une catégorie de produit comme document est composée de modèles, textes et figures. Le concept Catégorie de produit doit être précisé avec le concept Composition de catégories de produit. Le patron « Composition-Agrégation réflexive » est imité sur la classe *Catégorie de produit* du Méta-modèle de Processus en Cours de Construction : une agrégation réflexive est ajoutée.

La Figure 5-11 présente le chemin parcouru depuis le démarrage de l'exécution de la méthode. Le concept de Produit a été atteint par définition, mais il était aussi lié à Intention par le lien de concrétisation qui n'a pas été utilisé pour choisir Produit. Produit a ensuite été précisé par les concepts Composition de produits et Catégorie de produit.

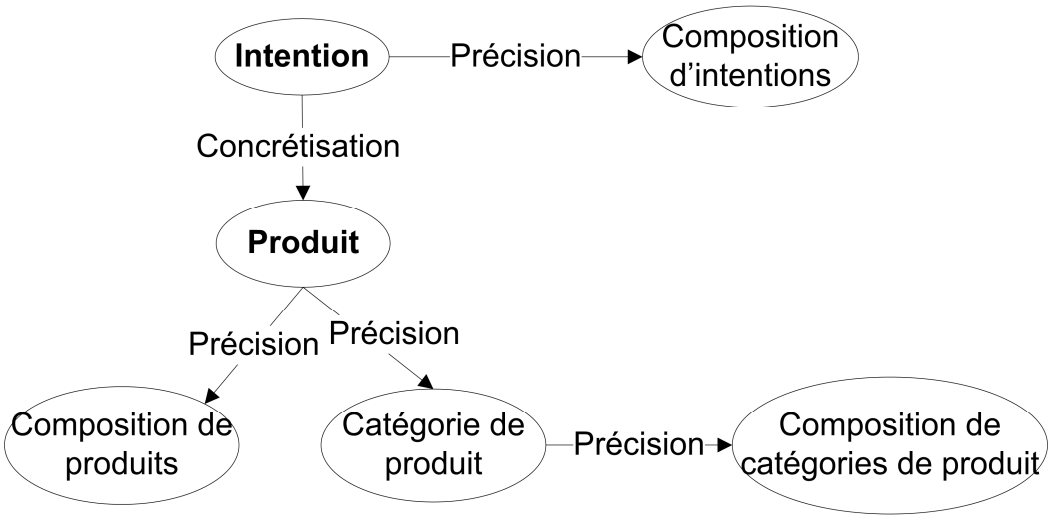


Figure 5-11. Chemin parcouru dans le graphe.

La Figure 5-12 présente le Méta-modèle de Processus en Cours de Construction correspondant aux différentes opérations réalisées. Les agrégations réflexives correspondent à l'imitation du patron « Composition-Agrégation réflexive », la classe *Produit* a été intégrée à partir du méta-modèle de domaine et la classe *Catégorie de produit* résulte de l'imitation du patron « Concept-Catégorie de concept » sur la classe *Produit*.

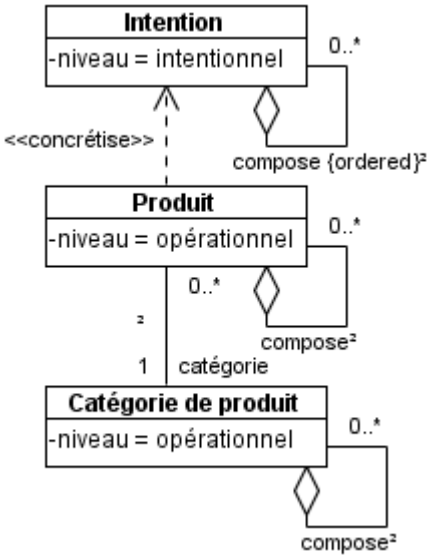


Figure 5-12. Méta-modèle de Processus en Cours de Construction.

Le méta-modèle en cours de construction permet donc de représenter les objectifs des chefs de services soins et supports ainsi que les produits réalisés par les chefs de projets et développeurs du service SI. Il manque cependant l'aspect du processus d'ingénierie correspondant aux activités et tâches qui les composent.

Il s'agit donc de compléter le méta-modèle de processus via la relation de complétude. Cette relation permet d'accéder à différents concepts du graphe conceptuel dont Rôle, Argument, Contexte, Stratégie et Unité de travail. Le concept permettant de représenter les activités est celui d'Unité de travail, il est donc sélectionné. Il correspond à une classe du méta-modèle de domaine, la classe *Unité de travail* est ajoutée au Méta-modèle de Processus en Cours de Construction. Les relations définies dans le méta-modèle de domaine entre la classe *Unité de travail* et les classes déjà définies dans le Méta-modèle de Processus en Cours de Construction sont ajoutées : il s'agit des relations entrée et sortie entre *Unité de travail* et *Produit*.

Il faut maintenant préciser le concept d'Unité de travail pour modéliser le fait que deux unités de travail peuvent s'exécuter en séquence et en parallèle. Les deux concepts Unité de travail parallèle et Unité de travail séquence permettent cela. Ils correspondent à l'imitation du patron « Association réflexive » sur la classe *Unité de travail* du Méta-modèle de Processus en Cours de Construction. Une association réflexive nommée « parallèle » est ajoutée à la classe Unité de travail, ainsi qu'une autre association réflexive dont les rôles sont nommés « précède, suit ».

De plus, il faut pouvoir distinguer des unités de travail de type « activité » et de type « tâche ». Cela est possible via la relation de précision d'Unité de travail vers Catégorie d'unité de travail. Le patron « Concept-Catégorie de concept » est imité sur la classe *Unité de travail*, ce qui permet de créer la classe *Catégorie d'unité de travail* ainsi qu'une association entre ces deux dernières classes.

Afin de spécifier que l'activité « Pré-étude fonctionnelle » est composée des tâches « Rédaction du cahier des charges simplifié » et « Constitution du glossaire des termes métier », l'ingénieur des méthodes doit préciser le concept Unité de travail par le concept Composition d'unités de travail qui permet d'imiter le patron « Composition-

Agrégation réflexive » sur la classe *Unité de travail* du méta-modèle en cours de construction. D'autre part, pour modéliser le fait qu'une activité est composée de tâches, il faut préciser le concept de Catégorie d'unité de travail avec le concept de Composition de catégories d'unité de travail. Le patron « Composition-Agrégation réflexive » est imité sur la classe *Catégorie d'unité de travail* dans le méta-modèle.

Enfin, afin de spécifier quel acteur réalise telle activité, il faut compléter le Méta-modèle de Processus en Cours de Construction en utilisant la relation complétude. Différents concepts permettent de compléter le méta-modèle, mais seul le concept de Rôle répond au besoin énoncé. Ce concept correspond à la classe *Rôle* dans le méta-modèle de domaine, la classe est donc intégrée au méta-modèle de processus. Cette classe est dépendante d'au moins une des classes suivantes : *Unité de travail*, *Produit* et *Alternative*. Les classes *Unité de travail* et *Produit* sont déjà présentes dans le méta-modèle de processus, la règle R5 est donc respectée en ajoutant les associations définies dans le méta-modèle de domaine.

La Figure 5-13 présente le chemin global parcouru dans le graphe conceptuel grâce aux définitions et aux relations définies entre les concepts.

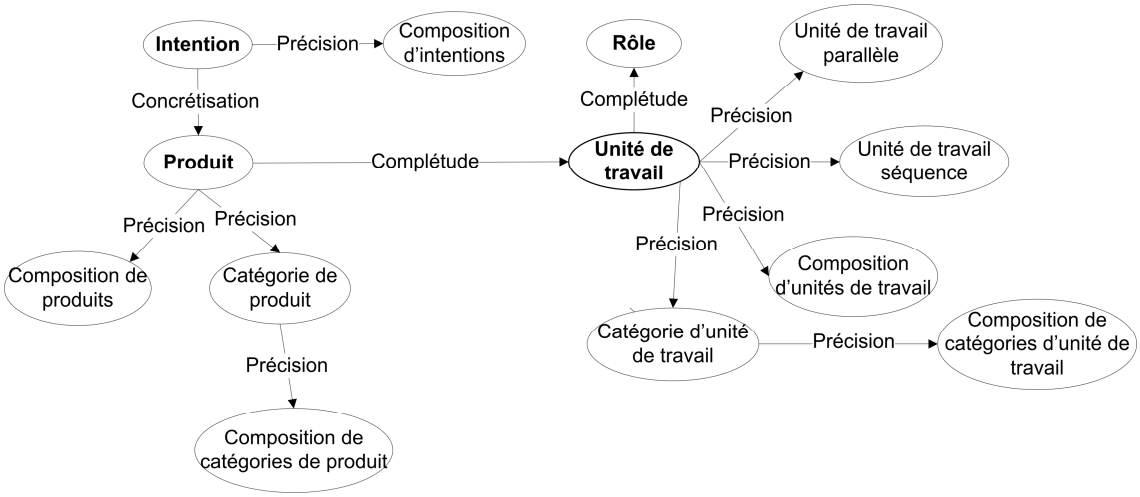


Figure 5-13. Ensemble du chemin parcouru dans le graphe conceptuel.

Le méta-modèle de processus final obtenu, présenté dans la Figure 5-14, permet de répondre aux besoins en modélisation des processus d'ingénierie de SI des chefs de services soins et supports ainsi que des chefs de projets et développeurs du service SI. Ce méta-modèle est défini sur deux niveaux :

- le niveau intentionnel qui modélise les intentions du processus d'ingénierie de SI du point de vue des chefs de services soins et supports ;
- le niveau opérationnel qui modélise les activités et produits réalisés durant le processus d'ingénierie de SI par les chefs de projets et développeurs du service SI.

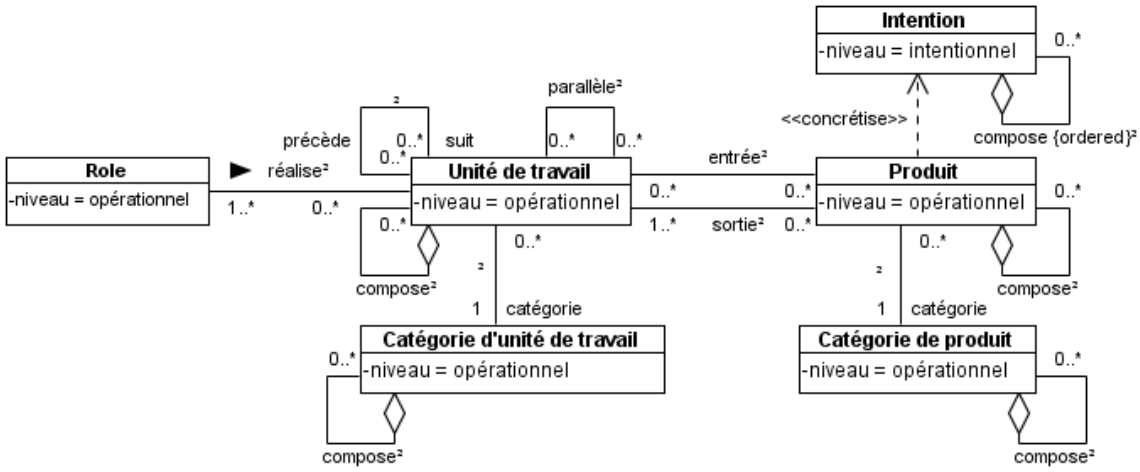


Figure 5-14. Méta-modèle de processus final.

Les classes du méta-modèle peuvent maintenant être enrichies par des attributs. Nous avons choisi ici d'ajouter tous les attributs proposés pour les classes du méta-modèle de processus. La majorité des attributs a été présentée dans le Chapitre 4 sauf les attributs des classes *Catégorie d'unité de travail* et *Catégorie de produit.*

Ces deux classes ont un attribut *description* qui permet de décrire la classe ou l'objet aux niveaux du méta-modèle, du modèle et de l'exécution. L'attribut *nom* permet de nommer la catégorie de produit ou d'unité de travail. Les attributs *dureeMax* et *nbPersMax* de la classe *Catégorie d'unité de travail* permettent de définir la durée maximale de toutes les catégories d'unité de travail d'un projet ainsi que le nombre maximal de personnes travaillant sur une même catégorie d'unité de travail. Par exemple, une activité ne peut durer plus de 30 jours et le nombre maximal de personnes participant à une activité ne peut être supérieur à 10.

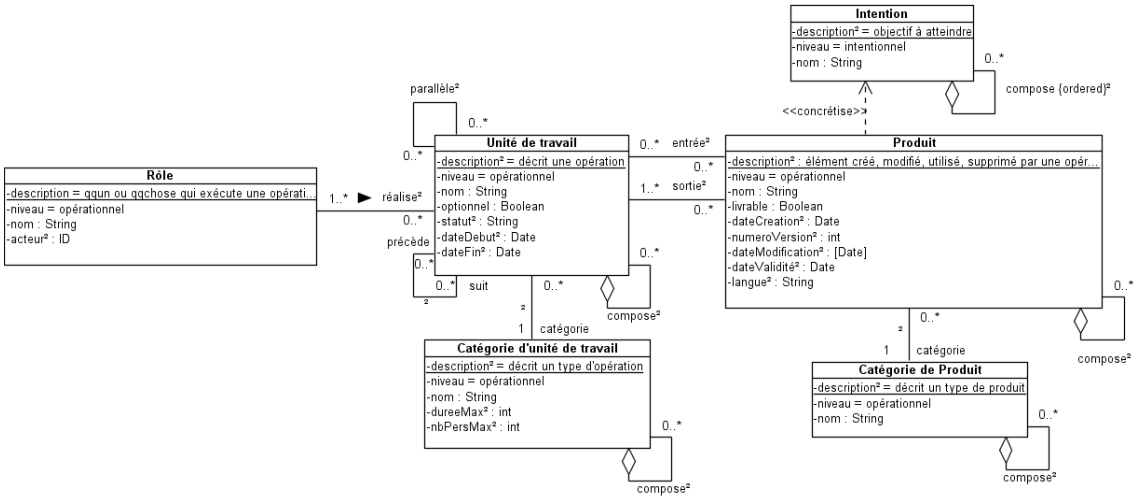


Figure 5-15. Méta-modèle de processus final complété avec les attributs.

5.2. Création du modèle de processus

Lorsque le méta-modèle de processus est final, il peut être instancié afin de modéliser des processus d'ingénierie de SI. L'objectif est ici d'instancier les modèles de processus pour que les chefs de projet puissent gérer et suivre l'exécution des projets d'ingénierie de SI, comme le montre la Figure 5-16.

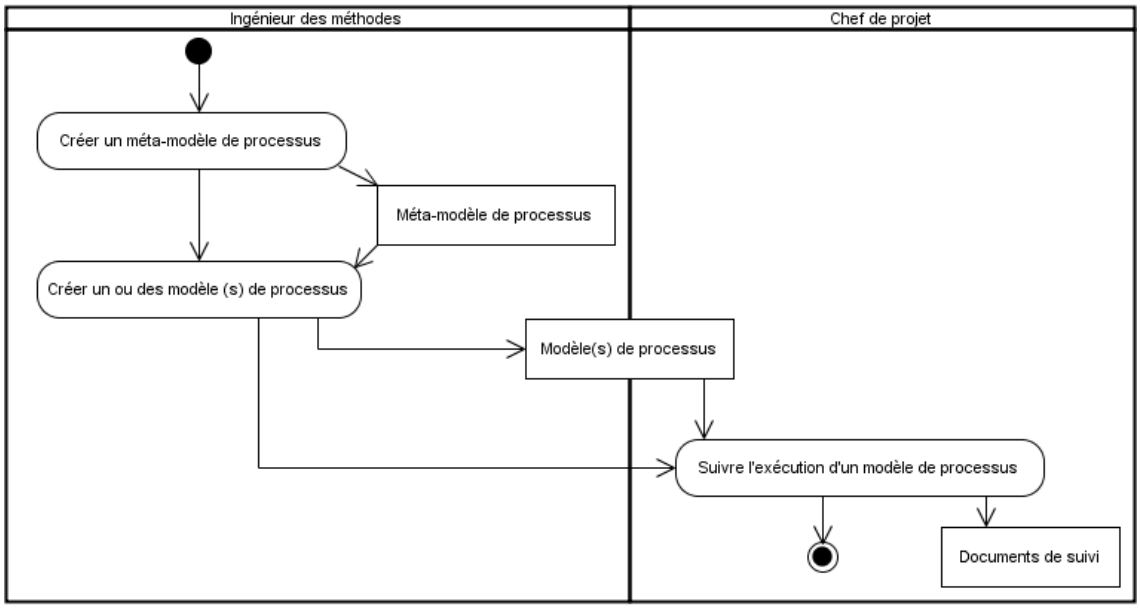


Figure 5-16.Méta-modèle, modèle et exécution des processus.

5.2.1. Instanciation du méta-modèle

L'instanciation du méta-modèle permet de créer des modèles de processus. Ces modèles décrivent ou prescrivent les processus d'ingénierie de SI dans une organisation ou un projet.

Le modèle de processus instancié à partir du méta-modèle peut être représenté sous forme d'un diagramme d'objets UML. Chaque objet du diagramme est instance d'une classe du méta-modèle.

Par exemple, dans le cadre de notre étude de cas, nous définissons les intentions suivantes :

- Définir l'impact du changement sur le service, décomposé en deux intentions :
  - Définir l'impact du changement sur l'organisation du service ;
  - Définir les personnes impactées.

Ces deux sous-intentions sont concrétisées par des produits, comme défini dans le méta-modèle de processus décrit précédemment. Ces deux produits sont :

- Cahier des charges simplifié, de catégorie « Texte » ;
- Diagramme d'acteurs, de catégorie « Artéfact UML ».

Ces produits sont le résultat de deux unités de travail, respectivement :

- Rédaction du cahier des charges simplifié, de catégorie « Tâche » ;
- Modélisation des acteurs, de catégorie « Tâche ».

Ces unités de travail font partie de Pré-étude fonctionnelle qui est de catégorie « Activité ».

Enfin, les produits Cahier des charges simplifié, Diagramme d'acteur et Glossaire des termes métiers composent le Dossier fonctionnel qui est de catégorie « Document ».

Le modèle de processus correspondant est présenté sous forme de diagramme d'objets UML à la Figure 5-17.

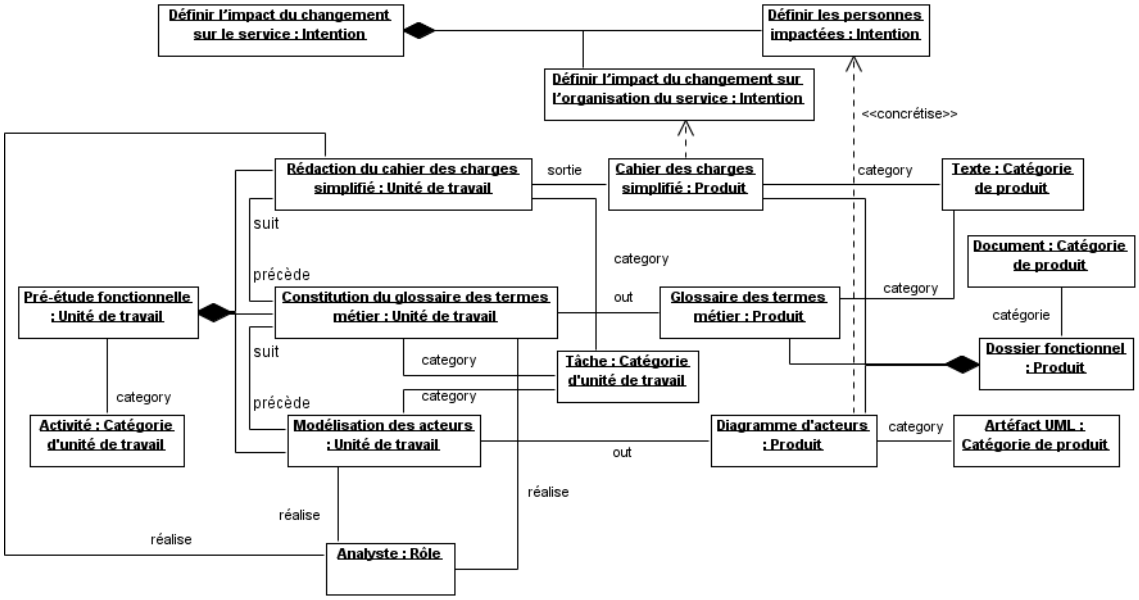


Figure 5-17. Modèle de processus sous forme de diagramme d'objets UML.

Nous présentons dans la Figure 5-18 quelques objets du diagramme d'objets avec des attributs valués. Par exemple, l'objet Rédaction du cahier des charges simplifié instance de la classe *Unité de travail* permet de définir les principales activités attendues. Cette unité de travail n'est pas optionnelle. Les autres attributs dont l'exposant était de 2 dans le méta-modèle seront instanciés au niveau de modélisation inférieur, c'est-à-dire lors de l'exécution des modèles.

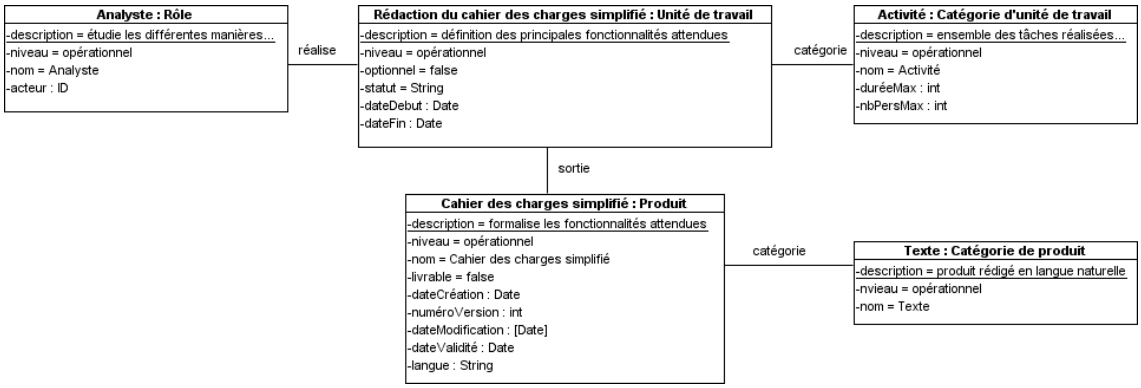


Figure 5-18. Extrait du diagramme d'objets avec des attributs valués.

5.2.2. Utilisation de formalismes

Les diagrammes d'objets étant difficilement compréhensibles, nous proposons une base de formalismes afin de représenter les objets instances d'une même classe de la même manière. Les formalismes associés aux classes sont présentés dans le Tableau 5-5. Ils proviennent des notations d'UML, SPEM 1.1 et 2.0, NATURE, MAP, Potts.

Classes	Formalismes possibles
Unité de travail	Unité de travail (OMG, 2009)

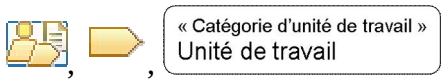
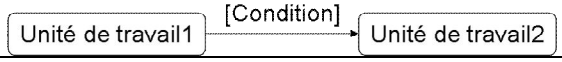
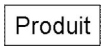
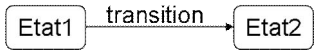
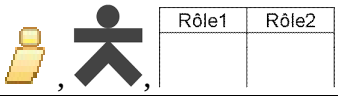
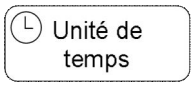
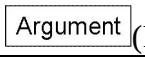
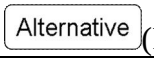
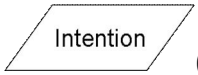
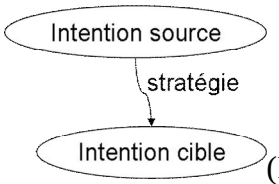
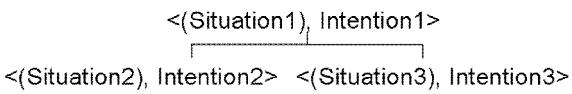
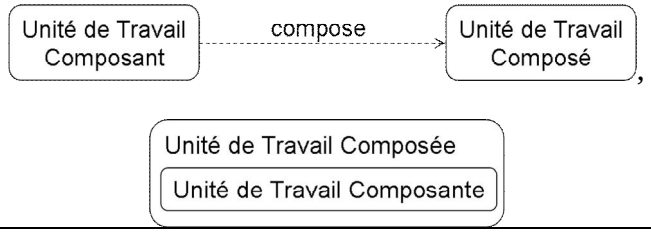
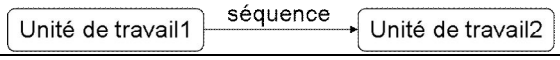
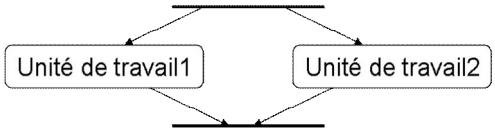
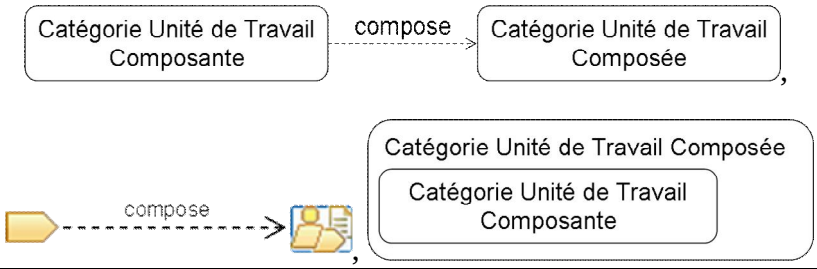
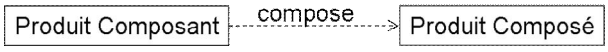
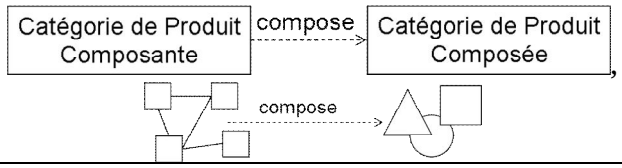
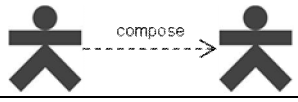
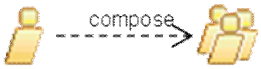
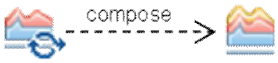
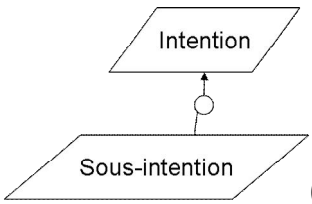
Catégorie d'unité de travail	 (OMG, 2008)
Condition	
Produit	 (OMG, 2009)
Catégorie de produit	 (OMG, 2005 et 2008)
État et Transition	 (OMG, 2009)
Rôle	 (OMG, 2005, 2008 et 2009)
Catégorie de rôle	 (OMG, 2008)
Unité de temps	
Catégorie d'unité de temps	 (OMG, 2008)
Problème	 (Potts, 1989)
Argument	 (Potts, 1989)
Alternative	 (Potts, 1989)
Intention	 (Objectiver, 2007)
MAP	 (Rolland et al., 1999)
NATURE	 (Rolland, 1994)

Tableau 5-5. Proposition de formalismes pour les classes

Le Tableau 5-6 propose une liste de formalismes pour les différentes associations.

Associations	Formalismes possibles
Composition d'unités de travail	
Unité de travail séquence	
Unité de travail parallèle	
Composition de catégories d'unité de travail	
Composition de produits	
Composition de catégories de produit	
Composition de rôles	
Composition de catégories de rôle	
Composition de catégories d'unité de temps	
Composition d'intentions	 (Objectiver, 2007)

Concrétise	« concrétise »
Réalise, entrée, sortie, cite, etc.	nom de l'association

Tableau 5-6. Proposition de formalismes pour les associations.

Nous proposons aux ingénieurs de méthodes de suivre la méthode présentée dans la Figure 5-19 afin de les aider dans le choix des formalismes. À partir du méta-modèle de processus complet (MPC) et d'une base de formalismes telle que présentée dans le Tableau 5-5 et le Tableau 5-6, une analyse des classes et associations permet de pré-sélectionner certains formalismes, notamment grâce à des règles présentées ci-après. Il faut affecter un formalisme à chaque classe et association du méta-modèle, parmi la liste des propositions. La légende contenant l'élément du méta-modèle ainsi que le formalisme correspondant sont enrichis jusqu'à ce que chaque élément du méta-modèle ait un formalisme.

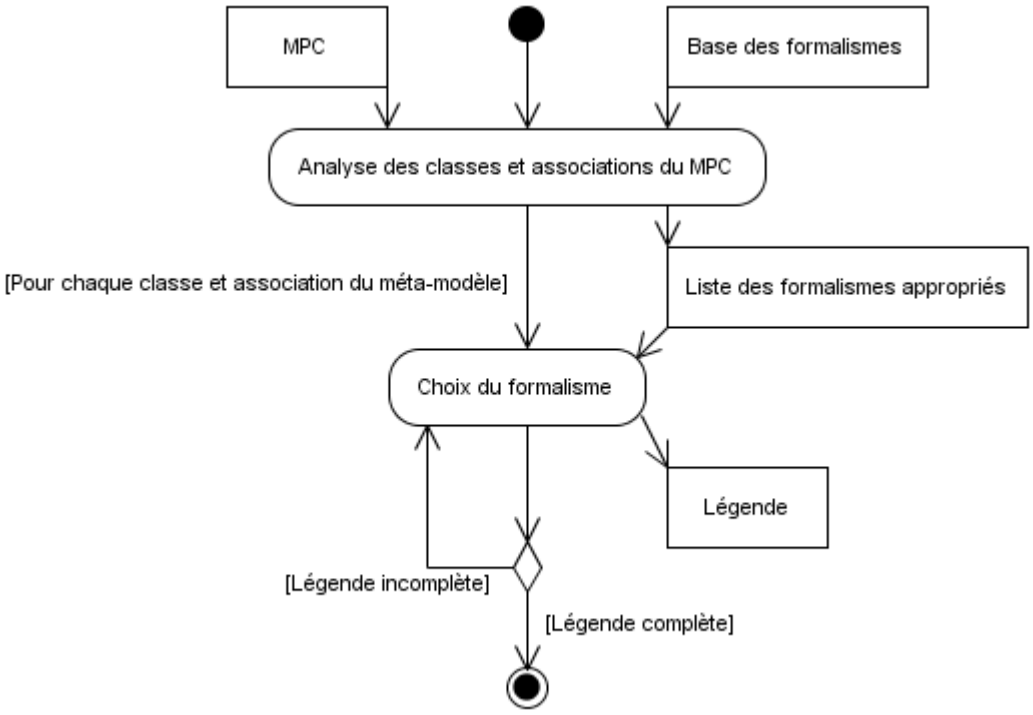


Figure 5-19. Guide pour la sélection des formalismes.

Il est possible d'affecter un formalisme pour chaque classe du méta-modèle, mais plusieurs classes du méta-modèle peuvent être également formalisées en utilisant le formalisme d'un méta-modèle de processus existant dans son ensemble. C'est ce que décrit le Tableau 5-7.

Classes	Formalisme
MAP	MAP
NATURE	NATURE

Intention	KAOS
Unité de travail	Unité de travail (Diagramme d'activités UML)
Unité de travail ^ Rôle	Unité de travail et couloirs (Diagramme d'activités UML à couloirs) ou Rôle
Unité de travail ^ Rôle ^ Produit	Unité de travail et Produit et couloirs ou Rôle (Diagramme d'activités UML à couloirs)
Problème ^ Alternative ^ Argument	Problème et Alternative et Argument (Potts)
Produit^ État ^ Transition	Etat-Transition (Diagramme états-transitions UML)

Tableau 5-7. Classes et formalismes associés.

Pour notre cas d'étude, nous réalisons une première légende, utilisée dans le modèle de processus à la Figure 5-20 :

- les unités de travail sont représentées par des rectangles arrondis. Pour représenter la composition, les rectangles sont imbriqués. La séquence d'unités de travail est présentée par une flèche pleine ;
- les catégories d'unité de travail sont représentées par un stéréotype dans les unités de travail ;
- les produits sont représentés par des rectangles ; lorsque des produits composent d'autres produits, on utilise une flèche en pointillé nommée « compose » ;
- les catégories de produit sont représentées par des stéréotypes ;
- pour représenter les produits en entrée des unités de travail, nous définissons une flèche pleine nommée « entrée », de même pour les produits en sortie ;
- les rôles sont représentés par des bonshommes. La réalisation d'une unité de travail par un rôle est modélisée par une flèche pleine nommée « réalise » ;
- les intentions sont représentées par des parallélogrammes, la composition d'intentions est modélisée par un rond, selon le formalisme de KAOS.

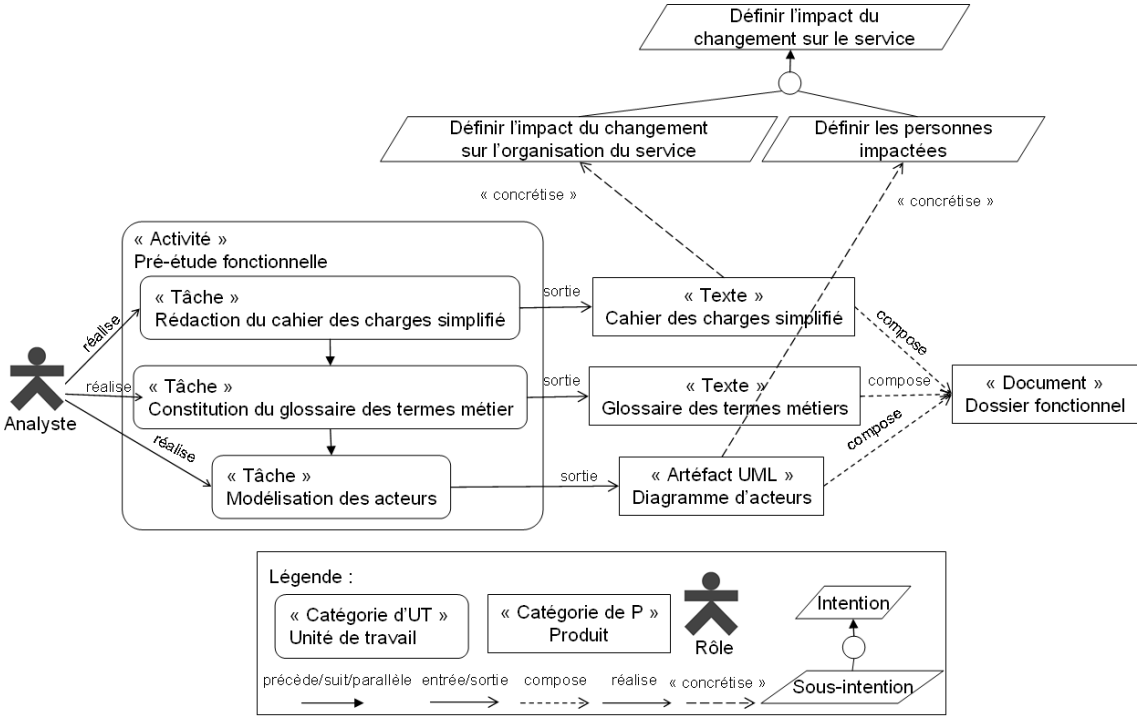


Figure 5-20. Modèle de processus avec les formalismes KAOS et UML.

Nous aurions pu définir une autre légende, comme celle utilisée pour le modèle de processus de la Figure 5-21 :

- les catégories d'unité de travail sont représentées par des symboles différents, ici ces catégories sont Activité et Tâche, le nom des unités de travail correspondant est précisé sous chaque catégorie du modèle ;
- les catégories de produit sont également symbolisées par des icônes différentes pour Texte, Artéfact UML et Rôle, le nom des produits est spécifié sous chaque catégorie dans le modèle de processus ;
- les symboles restants sont identiques au modèle précédent.

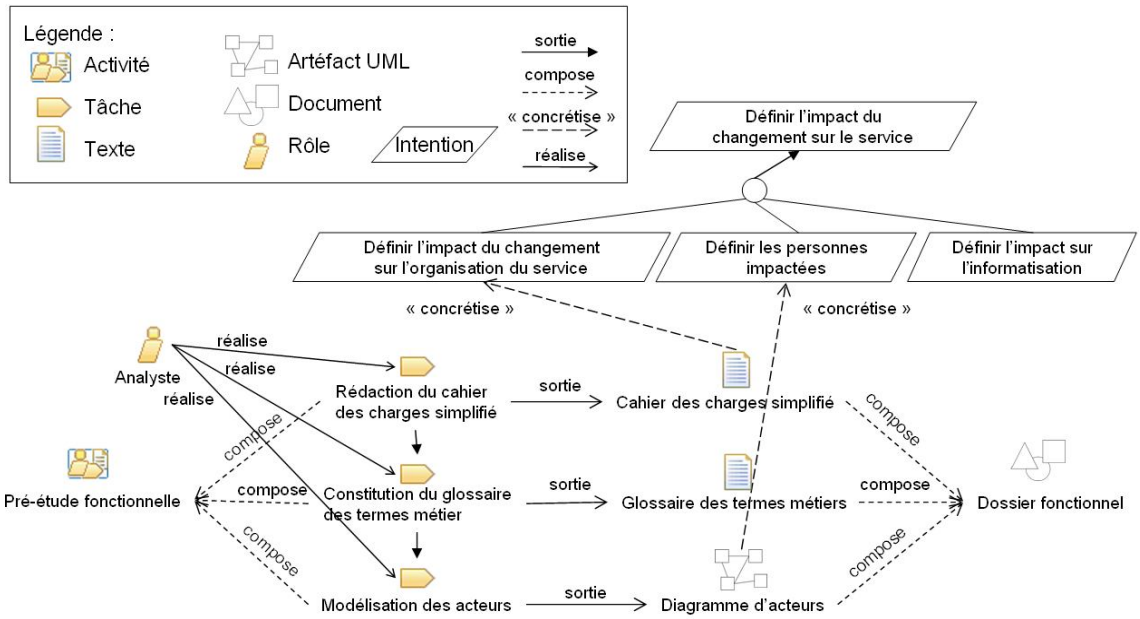


Figure 5-21. Modèle de processus avec les formalismes KAOS et SPEM.

Les légendes réalisées ne correspondent pas forcément à un formalisme classique proposé pour la modélisation des processus d'ingénierie de SI, ils peuvent être assimilés à des Domain Specific Language, où chaque ingénieur de méthodes adapte son formalisme selon les habitudes de l'organisation.

5.2.3. Itérations méta-modélisation/modélisation

Il est très difficile de réaliser le méta-modèle de processus adéquat en une seule itération. De nombreux manques peuvent apparaître au moment de l'instanciation du méta-modèle de processus. Il est donc possible d'itérer la boucle méta-modélisation/modélisation autant de fois que nécessaire. Ainsi, le méta-modèle de processus pourra être modifié et l'instanciation pourra être continuée en prenant en compte les évolutions du méta-modèle.

5.3. Synthèse

Dans ce chapitre, nous avons présenté notre méthode pour construire des méta-modèles de processus pour l'ingénierie des SI.

La méthode est basée sur un graphe conceptuel pour permettre aux ingénieurs des méthodes de sélectionner les concepts dont ils ont besoin dans leurs méta-modèles de processus. Les concepts sélectionnés sont ensuite intégrés dans le Méta-modèle de Processus en Cours de Construction, soit sous forme de classes issues du méta-modèle de domaine, soit sous forme d'associations ou de classes issues de l'imitation de patrons génériques ou de domaine.

Nous avons ensuite présenté la phase d'instanciation de la méthode, qui permet d'instancier le méta-modèle de processus créé pour définir des modèles de processus à l'aide d'une base de formalismes.

CONCLUSION DE LA PROPOSITION

Dans ces deux chapitres de proposition, nous avons tout d'abord présenté les supports utilisés par la méthode :

- le méta-modèle de processus de domaine contenant les principales classes et relations des méta-modèles de processus existants pour l'ingénierie des SI. Ce méta-modèle de domaine prend en compte les différents points de vue des processus d'ingénierie de SI (activité, produit, décision, contexte et stratégie) ainsi que les niveaux d'abstraction (intentionnel et opérationnel) ;
- des patrons génériques et de domaine, qui permettent d'enrichir des méta-modèles de processus par des imitations de structures de méta-modélisation ou des ajouts de classes issues de fragments de méta-modèles de processus existants ;
- le graphe conceptuel présentant l'ensemble des concepts possibles pour construire les méta-modèles de processus.

Dans le second chapitre, nous avons décrit la méthode qui s'exécute en deux temps :

- la sélection de concepts est guidée par le graphe conceptuel. Puis, l'intégration des concepts sous forme de classes ou d'associations dans le méta-modèle de processus en cours de construction se base sur le méta-modèle de domaine et les patrons génériques et de domaine. Cette première étape permet également l'ajout d'attributs aux classes du méta-modèle en cours de construction ;
- l'instanciation du méta-modèle de processus réalisé en utilisant une base de formalismes pour décrire des modèles de processus pour l'ingénierie des SI.

La méthode présentée permet donc aux ingénieurs des méthodes de créer des méta-modèles :

- multi-points de vue : tous les points de vue peuvent être modélisés ainsi que leur complémentarité ;
- adaptables : les ingénieurs des méthodes construisent les méta-modèles selon les contraintes et spécificités de leur organisation ;
- fédérateurs : un seul méta-modèle regroupe tous les concepts nécessaires à la modélisation des processus d'ingénierie de SI.

Le graphe conceptuel et la méthode ont été testés dans le milieu de la recherche, par des utilisateurs ayant un niveau avancé en ingénierie des SI. Des expériences ont également été menées avec des professionnels. Ces expériences sont présentées dans le chapitre suivant. D'autre part, une partie de la méthode a été implémentée dans un outil, présenté dans le Chapitre 7.

IMPLEMENTATION ET VALIDATION

INTRODUCTION A L'IMPLEMENTATION ET A LA VALIDATION

Dans cette partie, nous présentons l'implémentation de la méthode ainsi que sa validation.

1. L'implémentation

Une partie de la méthode a été automatisée dans un prototype que nous présentons brièvement. Un état de l'art des outils pour la modélisation et méta-modélisation des processus d'ingénierie de SI précède la présentation de ce prototype.

2. La validation

Afin de valider la méthode, nous avons réalisé des expériences auprès de focus group et des entretiens qualitatifs auprès d'industriels du domaine de l'ingénierie des systèmes d'information.

L'expérience avec le focus group nous a permis de mesurer la compréhension du graphe conceptuel, l'usage et l'utilisabilité du graphe conceptuel et de la méthode à travers deux exercices.

Suite à ces expériences, nous avons réalisé des entretiens qualitatifs avec des industriels. Ces entretiens avaient pour but de recueillir des informations sur les habitudes de travail de ces industriels par rapport aux méthodes d'ingénierie de systèmes d'information et leur perception du prototype de l'outil que nous leur avons présenté.

6. IMPLEMENTATION DE LA METHODE

De nombreux outils pour la modélisation des processus d'ingénierie de SI existent. Dans un premier temps, nous avons testé différents outils afin de déterminer s'ils permettaient de supporter notre méthode. Cette étude est présentée dans la section 6.1.

Les résultats de cette étude n'étant pas satisfaisants, un prototype a été développé, nous le présentons dans la seconde partie de ce chapitre.

6.1. État de l'art des outils de modélisation de processus d'ingénierie de SI

Nous avons étudié différents outils existants dans le domaine de la modélisation des processus d'ingénierie de SI. En testant ces outils, nous avons cherché à vérifier s'ils permettaient de :

- modéliser les modèles que notre méthode permet de décrire, c'est-à-dire des modèles de processus adaptés, multi-points de vue et fédérés ;
- et/ou créer des méta-modèles de processus et les instancier.

6.1.1. Spearmint

Spearmint (Becker et al., 1999) est un outil graphique basique pour réaliser des diagrammes d'activités prenant en compte les concepts suivant : activité, rôle, artefact et outil. Il est possible de créer des relations entre ces concepts. Les relations sont différentes selon les concepts reliés, mais il est impossible de les nommer. Les activités permettent de créer ou d'utiliser des artefacts et utilisent des outils. Les activités peuvent s'exécuter en parallèle ou en séquence (split/join). Les rôles exécutent les activités.

La Figure 6-1 présente le modèle de processus réalisé avec Spearmint, selon l'étude de cas développée dans le Chapitre 5. Il n'est pas possible de tout modéliser, comme la composition d'activités et d'artéfacts. D'autre part, le nombre de concepts proposé par Spearmint est restreint aux concepts de base des modèles de processus orientés activité.

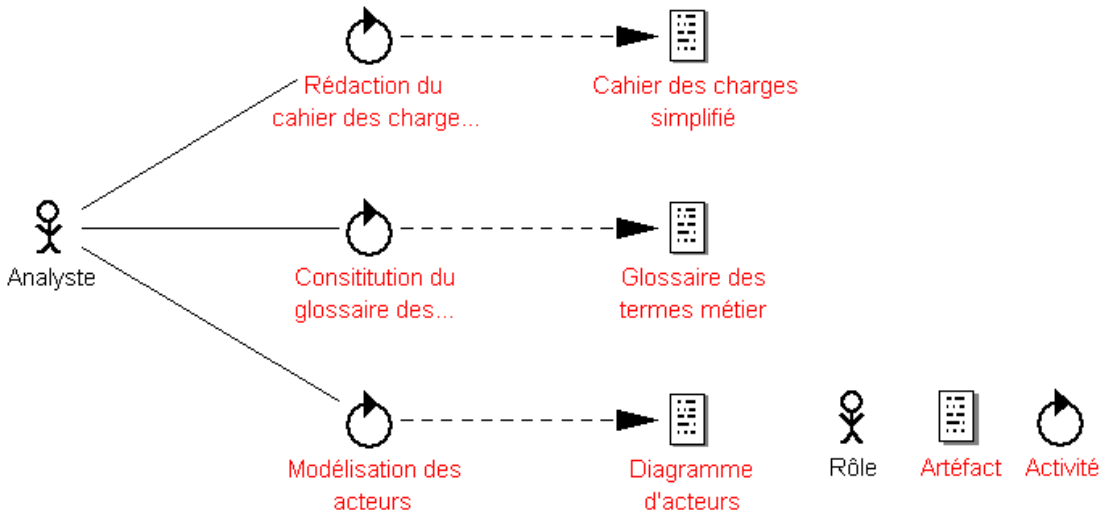


Figure 6-1. Modèle de processus réalisé avec Spearmint.

Le méta-modèle de Spearmint est donc limité et l'outil proposé ne permet pas d'étendre ce méta-modèle. Cet outil ne répond donc pas à nos besoins, car il n'a pas la couverture de notre proposition, en termes de niveaux de modélisation et de recouvrement des concepts.

6.1.2. Method Composer

Method Composer est un outil qui a été développé par le laboratoire COTAR[11] de Sydney en 2005. Cet outil est divisé en trois parties, destinées à des acteurs différents :

- pour les ingénieurs des méthodes : permet la création et la maintenance d'un répertoire de fragments de méthode ;
- pour les « méthodologistes » : permet la composition de fragments de méthode par parcours du répertoire et la sélection des fragments pour créer de nouvelles méthodes ;
- pour les développeurs : permet l'exécution d'une méthode pour un projet particulier (non disponible à l'heure actuelle).

L'outil est basé sur les méta-modèles (AS, 2004) et (ISO/IEC, 2007). Ces méta-modèles de processus étant orientés activités, les fragments de méthodes instanciés à partir du méta-modèle utilisé dans l'outil sont également orientés activité. L'outil n'offre donc pas la couverture souhaitée. De plus, le méta-modèle de l'outil n'est pas modifiable.

Comme le montre la Figure 6-2, il n'est possible de manipuler que les concepts de phases, d'activités, de tâches et de documents. La partie gauche de l'interface correspond au répertoire de fragments de méthode, la partie droite correspond aux fragments sélectionnés. Le modèle obtenu est donc très incomplet par rapport à ce que nous souhaitons modéliser avec notre méthode.

[11] http://www.cotar.uts.edu.au/

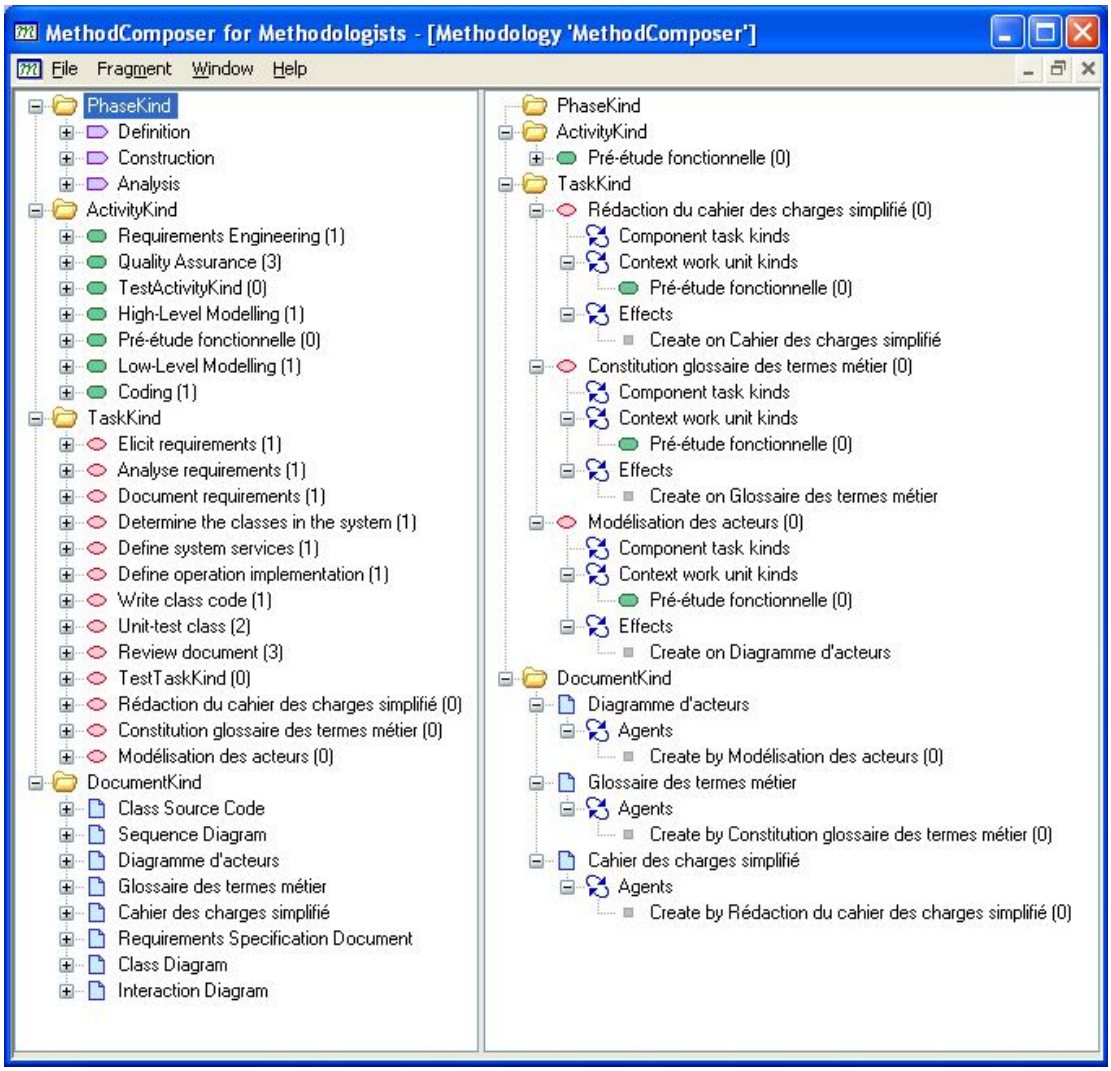


Figure 6-2. Modèle de processus réalisé avec Method Composer.

6.1.3. Eclipse Process Framework Composer

Eclipse Process Framework Composer (EPF, 2009) est un outil développé par Eclipse et Rational. Il existe une version payante Rational Method Composer (RMC, 2009) qui comporte les mêmes fonctionnalités avec quelques extensions supplémentaires, dont le déploiement et le suivi des processus.

EPF est construit à partir du méta-modèle SPEM 2.0 (OMG, 2008), les concepts manipulés sont donc restreints à ceux des méta-modèles de processus orientés activité. La Figure 6-3 détaille l'activité de Pré-étude fonctionnelle qui définit le rôle, les tâches et les produits réalisés.

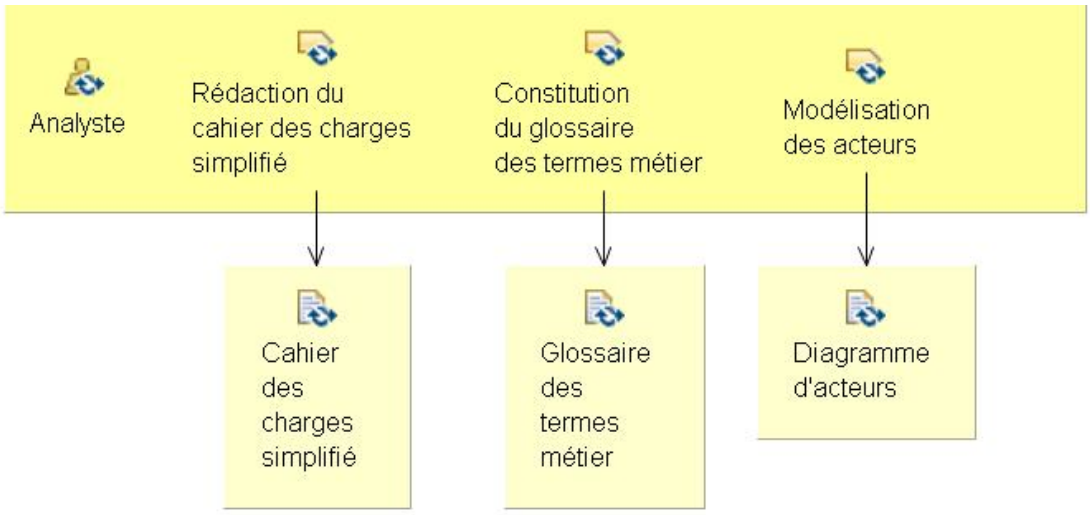


Figure 6-3. Modèle de processus réalisé avec EPF Composer.

La manipulation des éléments des modèles de processus est réalisée principalement via une arborescence, comme présenté dans la Figure 6-4.

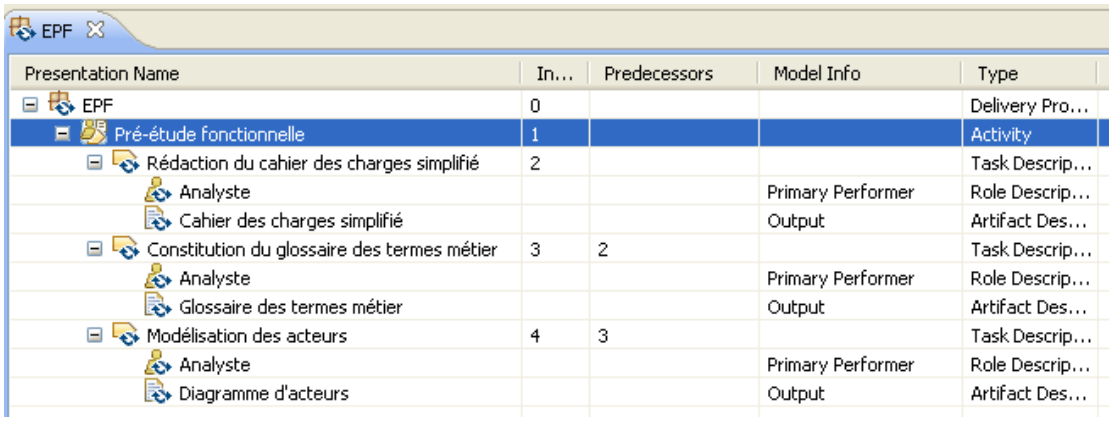


Figure 6-4. Arborescence réalisée avec EPF Composer.

EPF permet également de définir un répertoire de fragments de méthode pouvant être réutilisé dans des méthodes définies par les utilisateurs.

EPF est un outil très complet et solide, mais le méta-modèle utilisé ne permet pas d'être adapté pour prendre en compte tous les concepts des différents points de vue que nous souhaitons traiter.

6.1.4. MetaEdit+

MetaEdit+[12] est un environnement très puissant permettant de définir des modèles avec un formalisme propre. Un grand nombre de formalismes est proposé comme UML, BPMN ou GOPRR[13] pour la méta-modélisation. Il est donc possible de définir des méta-

[12] http://www.metacase.com/
[13] Graph-Object-Property-Role-Relationship

modèles, de les instancier en affectant un formalisme à chaque classe et association du méta-modèle.

Nous avons réalisé le méta-modèle correspondant à notre étude de cas avec le formalisme GOPRR, présenté à la Figure 6-5. Ce formalisme est équivalent à celui du diagramme de classes UML.

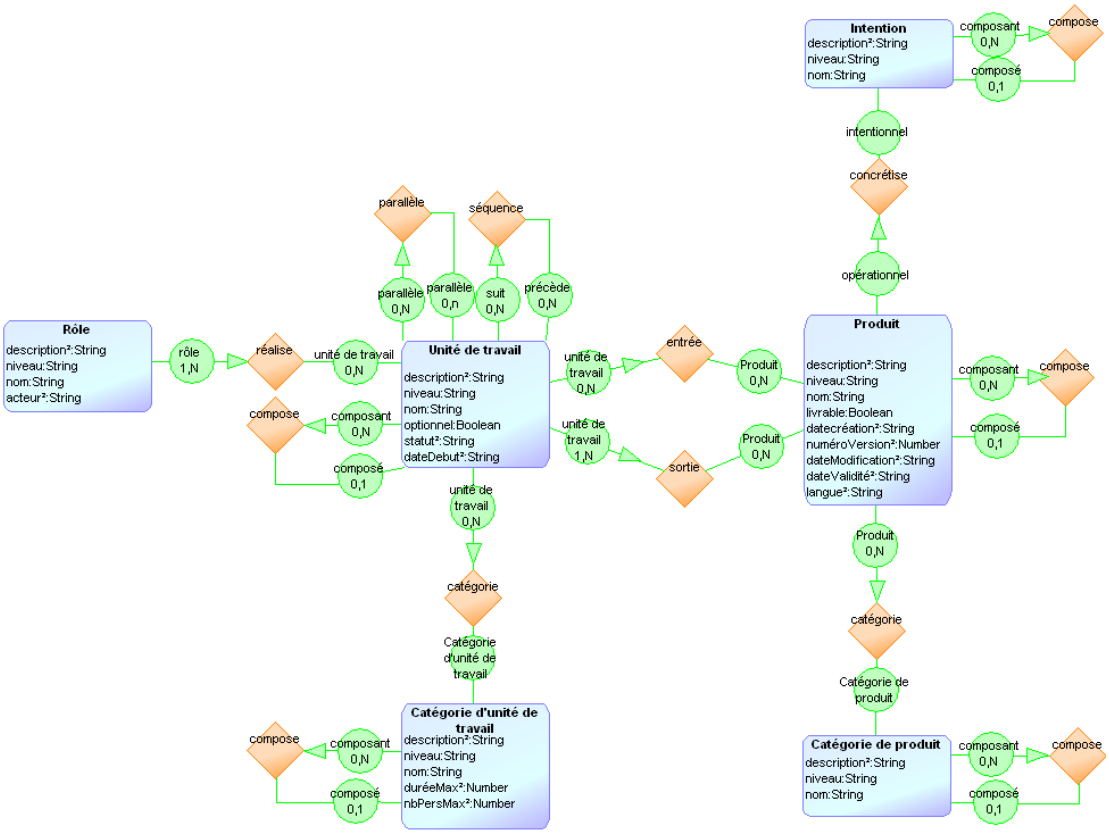


Figure 6-5. Méta-modèle de processus réalisé avec MetaEdit+.

Le méta-modèle peut ensuite être instancié dans le même outil, en ayant préalablement affecté un formalisme à chaque élément du méta-modèle. Le formalisme de chaque élément est créé via un éditeur graphique.

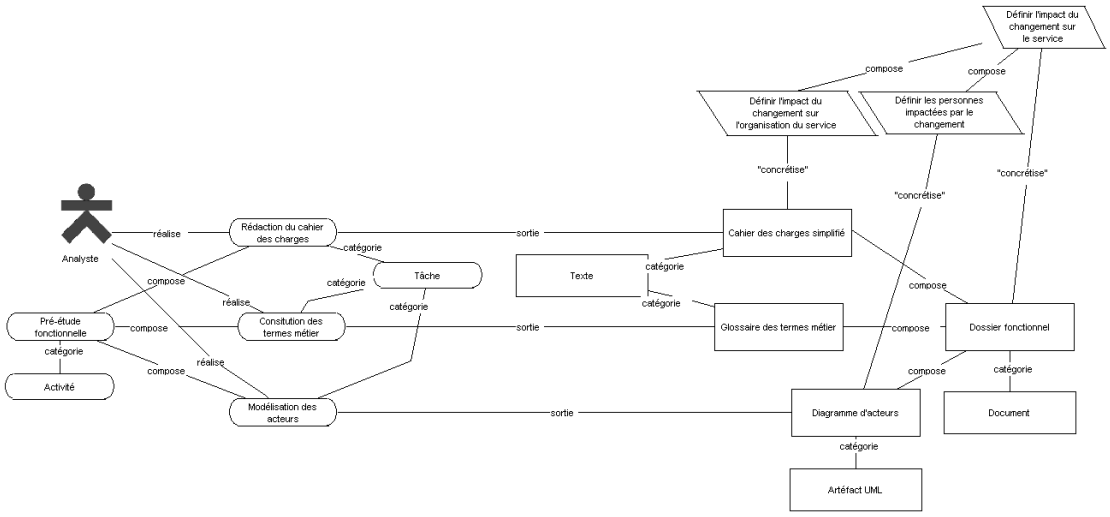


Figure 6-6. Modèle de processus avec Meta-Edit+.

MetaEdit+ permet également de générer du code exécutable Java ou C++ par exemple, qui permet l'exécution de modèles dans un programme. Cela permet donc d'obtenir une structure pour la création et le suivi de projets, basés sur des modèles de processus définis dans MetaEdit+. Par contre, les modèles réalisés dans l'outil ne peuvent être exécutés directement, il faut passer par un IDE.

MetaEdit+ permet donc de réaliser des méta-modèles et de les instancier en affectant le formalisme de notre choix. Ces opérations sont relativement complexes, mais permettent une grande liberté à l'ingénieur des méthodes, comme nous le souhaitons également.

Cependant, l'outil ne propose aucun guidage pour la création des méta-modèles et modèles. La création de formalismes est facile, mais l'éditeur graphique est relativement limité. L'interface est complexe, car l'outil propose beaucoup de fonctionnalités, inutiles dans notre domaine, ce qui rend l'utilisation de l'outil difficile. Enfin, la génération de code est également complexe.

6.1.5. Synthèse

Le Tableau 6-1 présente les différents outils étudiés et leurs fonctionnalités.

Au niveau des méta-modèles, nous souhaiterions que les ingénieurs des méthodes puissent créer leurs propres méta-modèles, de préférence avec un éditeur graphique. Les points de vue activité, produit, etc. devraient être couverts. La vérification de la cohérence du méta-modèle produit ainsi que l'export en XMI pour une réutilisation dans des AGL seraient un plus.

Au niveau des modèles de processus, les ingénieurs des méthodes doivent créer leurs propres modèles, avec un éditeur graphique, tout en ayant la possibilité d'intégrer des concepts des différents points de vue existants, en se basant sur des formalismes prédéfinis. Enfin, si l'outil propose un répertoire de fragments de modèles de processus,

cela permettrait à l'ingénieur des méthodes de réutiliser des fragments existants et ainsi de gagner du temps.

Pour permettre une instanciation plus facile des modèles de processus, les ingénieurs des méthodes pourraient générer du code pour exécuter le modèle.

L'outil le plus complet et présentant le plus d'intérêt pour appliquer notre méthode est MetaEdit+, cependant, il ne fournit aucun guidage. De plus cet outil étant un logiciel propriétaire, il est impossible d'y apporter des modifications.

Outils / Critères	Spearmint	Method Composer	Eclipse Process Framework Composer	MetaEdit+
Méta-modèle				
Création				✓
Éditeur graphique				✓
Couverture des cinq points de vue				✓
Vérification de la cohérence				✓
Export XMI				✓ (XML)
Modèle				
Création	✓	✓	✓	✓
Éditeur graphique	✓		✓	✓
Couverture des cinq points de vue				✓
Base de formalismes				✓
Répertoire de fragments		✓	✓	
Instanciation				
Génération de code				✓

Tableau 6-1. Synthèse des outils et de leurs fonctionnalités.

Les outils étudiés ne conviennent donc pas entièrement pour permettre la modélisation de modèles de processus adaptés, multi-points de vue et fédérés tels que nous le préconisons. D'une part, les méta-modèles sont restreints au point de vue activité des processus. D'autre part, les outils ne permettent pas de modifier les méta-modèles utilisés, leur utilisation ne peut donc couvrir le champ de notre recherche.

Nous souhaitons donc proposer un outil permettant aux ingénieurs des méthodes de :

– définir des méta-modèles de processus en suivant la méthode que nous avons proposée ;

– permettre l'instanciation de ces méta-modèles en guidant le choix des formalismes.

De plus, cet outil doit permettre aux chefs de projets de créer et suivre l'exécution d'un processus dans le cadre d'un projet. Cet outil fait l'objet de la section suivante.

6.2. Un outil pour la définition de processus d'ISI

6.2.1. Fonctionnalités de l'outil

La Figure 6-7 représente les trois grandes fonctionnalités que nous souhaitons offrir aux ingénieurs des méthodes et chefs de projets sous forme d'un diagramme de cas d'utilisation.

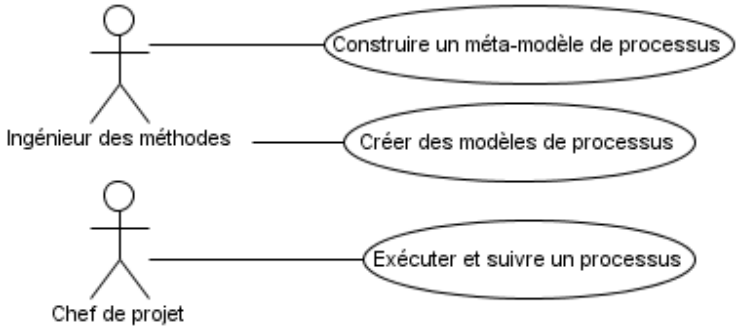


Figure 6-7. Fonctionnalités souhaitées de l'outil.

La première version de l'outil présentée ici se focalise sur la construction des méta-modèles de processus.

6.2.2. Architecture fonctionnelle et technique

L'outil réalisé permet de créer des méta-modèles de processus adaptés, multi-points de vues et fédérés, selon la méthode décrite dans le Chapitre 5, en utilisant les supports définis dans le Chapitre 4. La Figure 6-8 présente les fonctionnalités de l'outil pour créer des méta-modèles de processus pour l'ingénierie des SI.

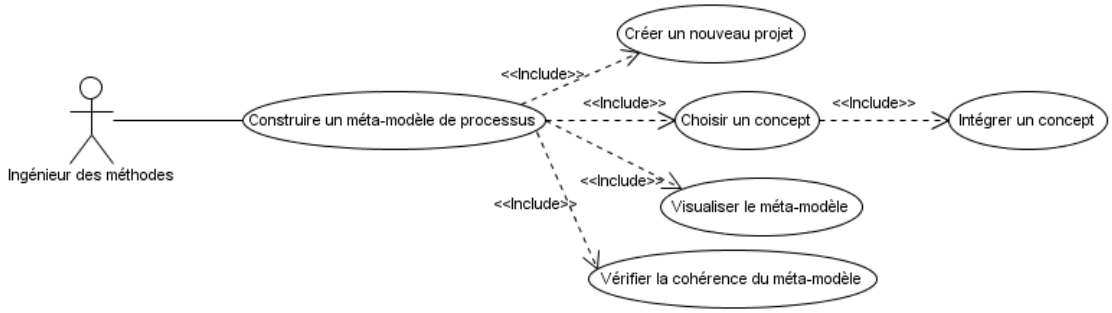


Figure 6-8. Fonctionnalités de la première version de l'outil.

La Figure 6-9 présente l'architecture technique retenue pour implémenter l'outil développé en Java :

- le graphe conceptuel et le méta-modèle de domaine sont stockés dans des fichiers XMI (XML Metadata Interchange) (OMG, 2007c), dans le but d'avoir un outil dynamique et évolutif ;
- les patrons génériques et de domaine sont codés en Java. Chaque patron est implémenté sous la forme d'une méthode appelée quand un patron doit être appliqué ;
- le méta-modèle en cours de construction est stocké dans un fichier XMI afin de pouvoir être réutilisé dans d'autres AGL ;
- le graphe conceptuel peut être visualisé grâce à l'outil Prefuse (Prefuse, 2009) ;
- le méta-modèle de processus est visualisable grâce à l'outil UMLJGraph (UMLJGraph, 2005).

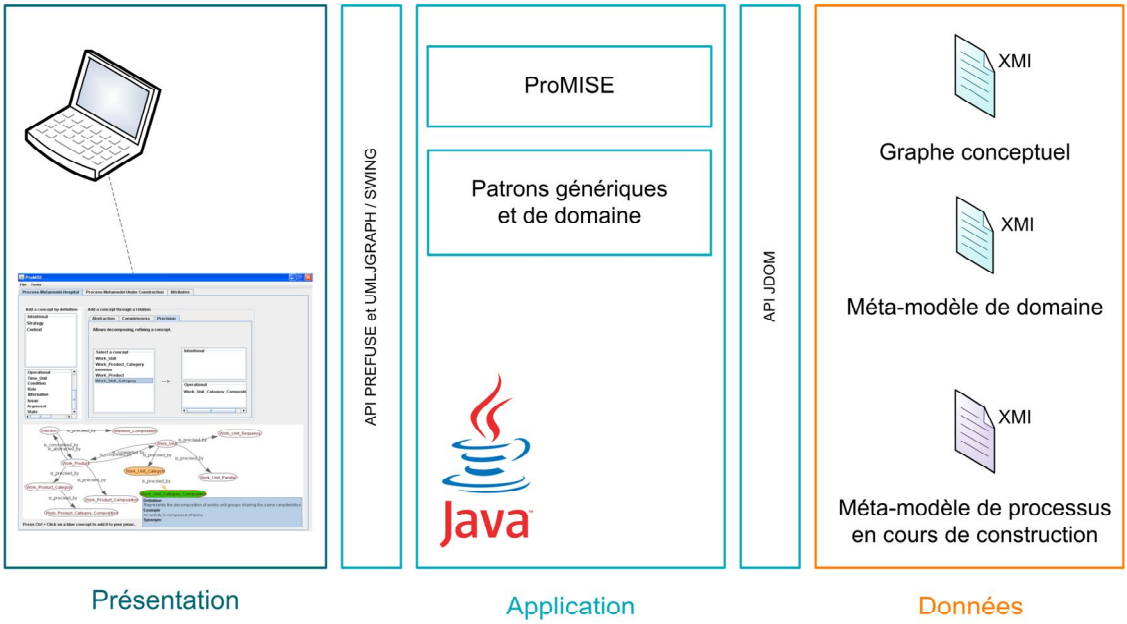


Figure 6-9. Architecture technique de l'outil.

L'implémentation a été réalisée par deux stagiaires de DUT informatique de l'IUT2 de Grenoble (Monjo, 2009) (Lefèvre, 2009) sur la base des fonctionnalités prévues et présentées ci-dessus.

Nous présentons les différentes fonctionnalités permettant la construction de méta-modèles de processus.

6.2.3. Construction de méta-modèles de processus

a) Sélection d'un concept par définition

À l'ouverture de l'application, l'ingénieur des méthodes peut créer ou ouvrir un projet existant.

L'interface d'initialisation d'un projet permet de visualiser la liste des concepts principaux. Les définitions, synonymes et exemples de chaque concept apparaissent lors

du survol de ces concepts avec le curseur, cf. Figure 6-10. La sélection des concepts se fait par clic du concept dans la liste.

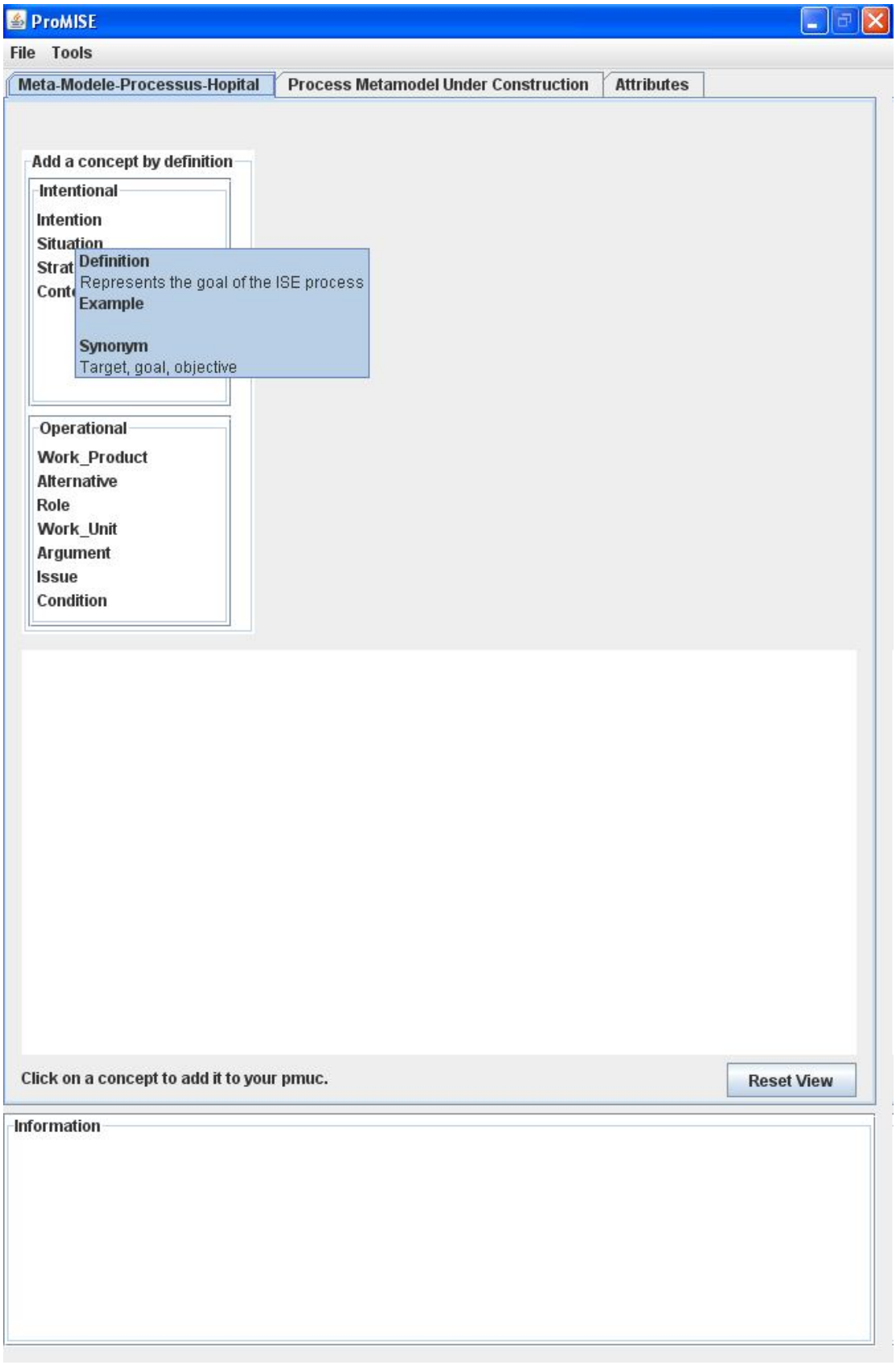

ProMISE
File Tools
Meta-Modele-Processus-Hopital
Process Metamodel Under Construction
Attributes
Add a concept by definition
Intentional
Intention
Situation
Strat
Conte
Definition
Represents the goal of the ISE process
Example
Synonym
Target, goal, objective
Operational
Work_Product
Alternative
Role
Work_Unit
Argument
Issue
Condition
Click on a concept to add it to your pmuc.
Reset View
Information

Figure 6-10. Interface d'initialisation d'un projet.

b) Intégration d'un concept

Le concept sélectionné est donc intégré au méta-modèle de processus en cours de construction et le graphe conceptuel affiche le concept ajouté, ici Intention, cf. Figure 6-11.

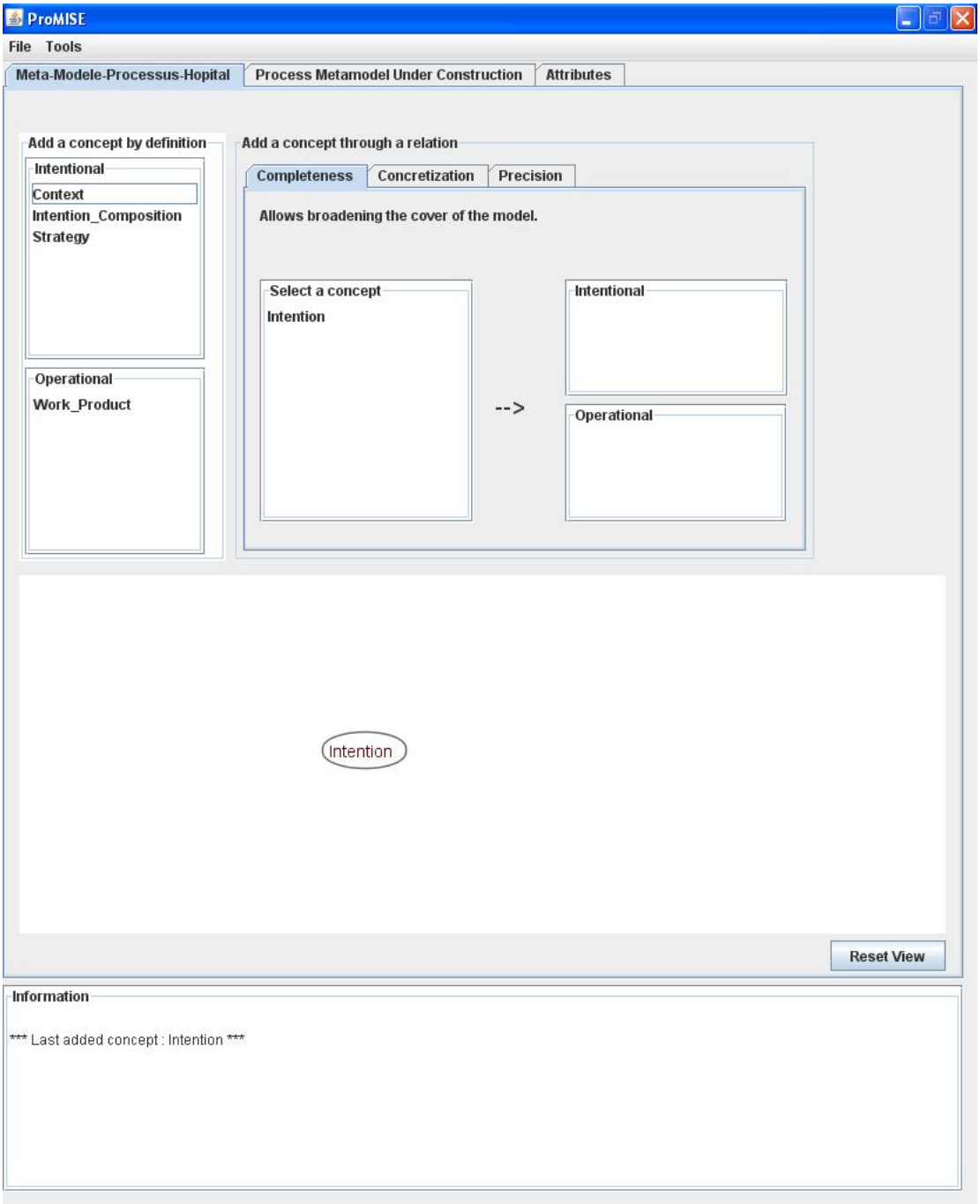


Figure 6-11. Interface après ajout du concept Intention.

c) Sélection d'un concept par relation

Une fois que le premier concept est intégré, il est possible de sélectionner de nouveaux concepts en utilisant à nouveau les définitions des concepts cibles d'Intention dans le graphe conceptuel ou en utilisant les relations sources : complétude, précision, abstraction ou concrétisation, à l'aide des onglets.

Il est possible d'utiliser directement le graphe conceptuel, qui affiche les concepts pouvant préciser, compléter ou concrétiser Intention en bleu, cf. Figure 6-12. Selon l'onglet sélectionné, les concepts en bleu sont mis à jour.

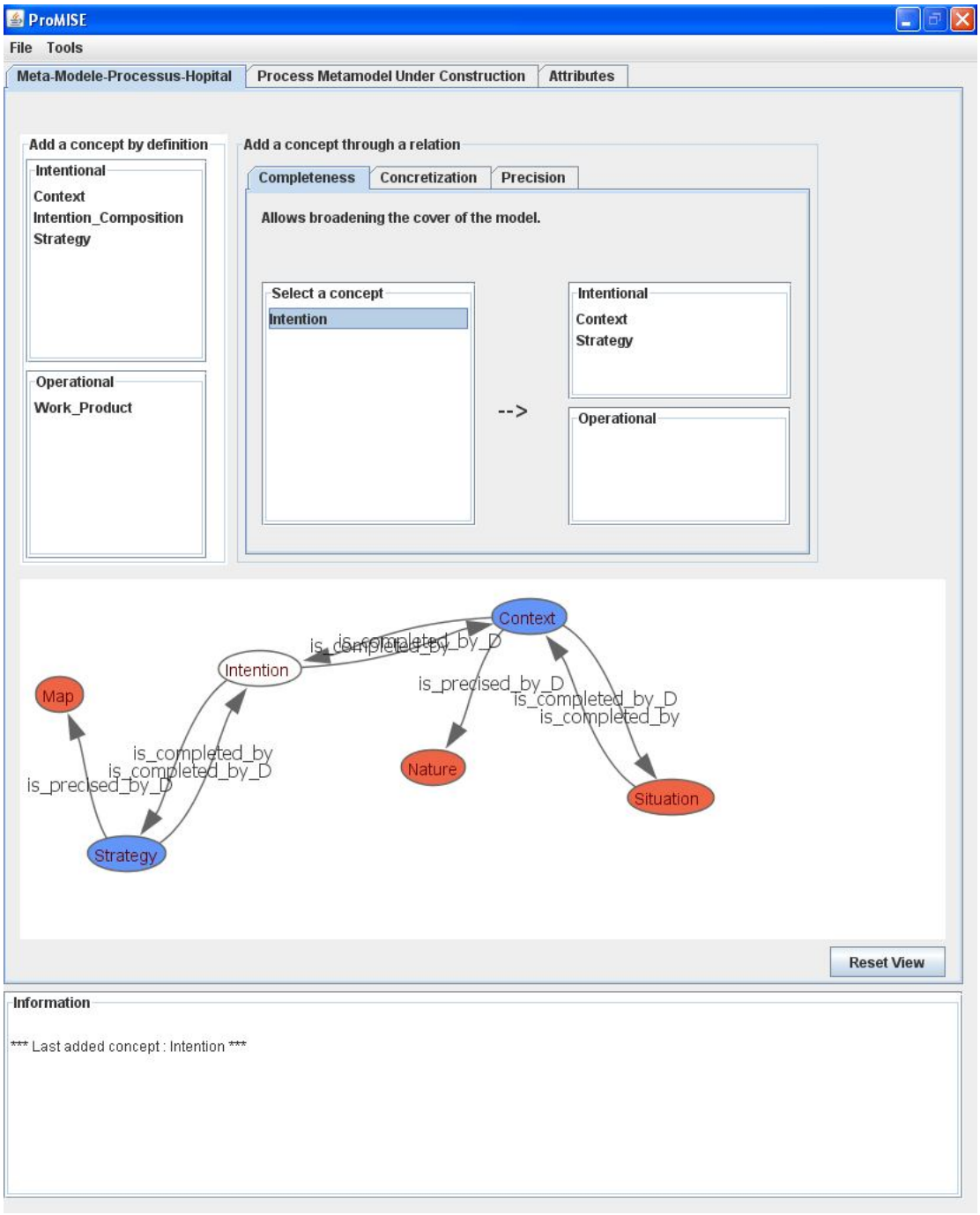


Figure 6-12. Interface de visualisation des concepts liés par la relation de complétude.

Il est également possible d'ajouter un concept par relation en passant par les listes de concepts mises à jour selon le concept sélectionné, cf. Figure 6-13. La liste de gauche permet de sélectionner le concept que l'on veut préciser, concrétiser ou compléter. La liste de droite affiche les concepts pouvant être sélectionnés et ajoutés dans le méta-modèle en cours de construction. Dans notre cas d'étude, nous souhaitons préciser le

concept Intention. Le seul concept pouvant préciser Intention est le concept Intention_Composition.

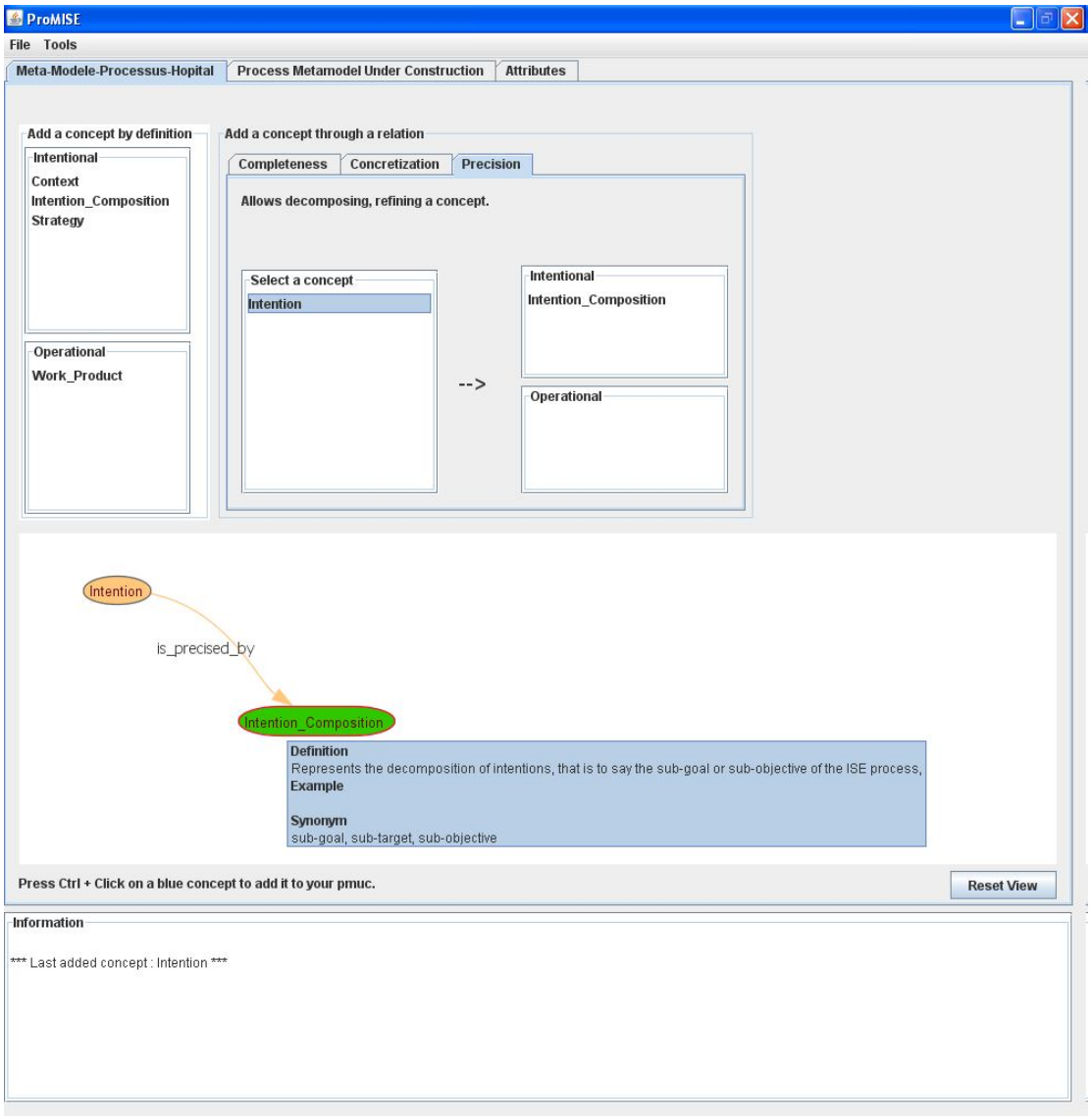


Figure 6-13. Ajout du concept Intention_Composition par la relation de précision.

Lorsqu'Intention_Composition est sélectionné, l'outil propose une interface pour l'adaptation de l'imitation du patron Composition – Agrégation réflexive sur la classe *Intention* du méta-modèle de processus en cours de construction, cf. Figure 6-14. Comme précisé dans le patron présenté au Chapitre 4, il est possible de choisir entre la composition et l'agrégation, de modifier les cardinalités et de nommer l'association.

Les patrons génériques ou de domaine ne nécessitant aucune adaptation sont appliqués tels que définis dans le Chapitre 4.

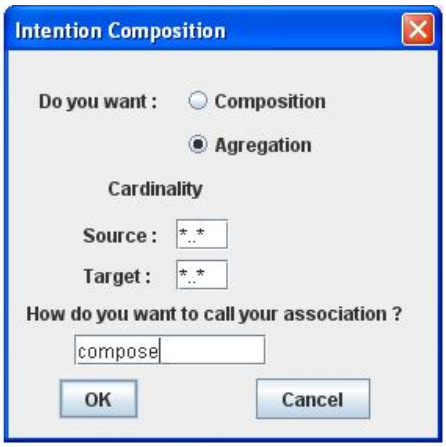


Figure 6-14. Interface d'adaptation de l'imitation du patron Composition – Agrégation réflexive sur Intention.

d) Visualisation du méta-modèle de processus en cours de construction

Lorsque tous les concepts nécessaires ont été sélectionnés, le chemin complet parcouru dans le graphe conceptuel est visible.

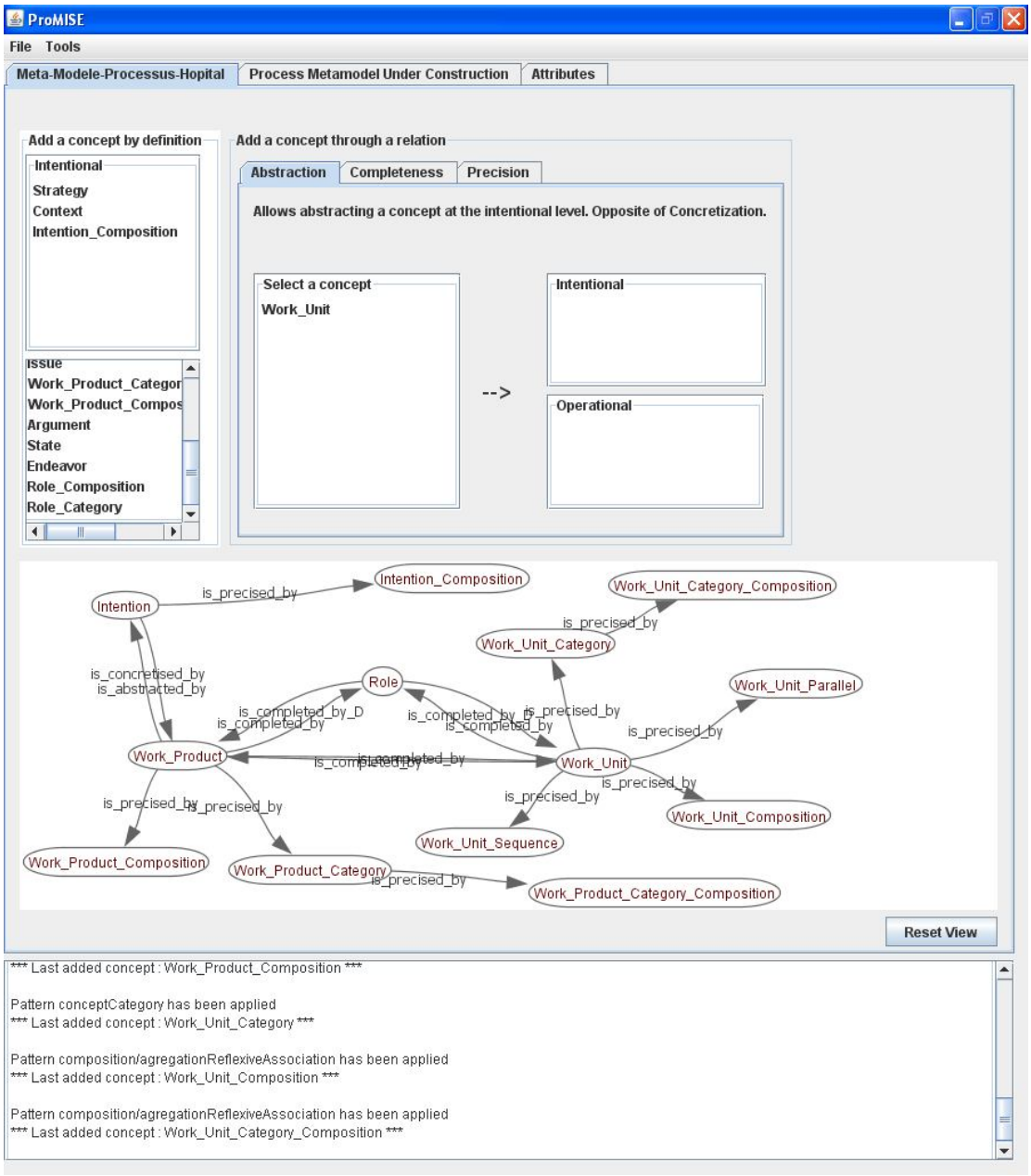


Figure 6-15. Chemin correspondant au méta-modèle de processus final.

Le méta-modèle de processus en cours de construction peut être visualisé à tout moment sous forme d'un diagramme de classes UML dans l'onglet « Process Meta-model Under Construction ».

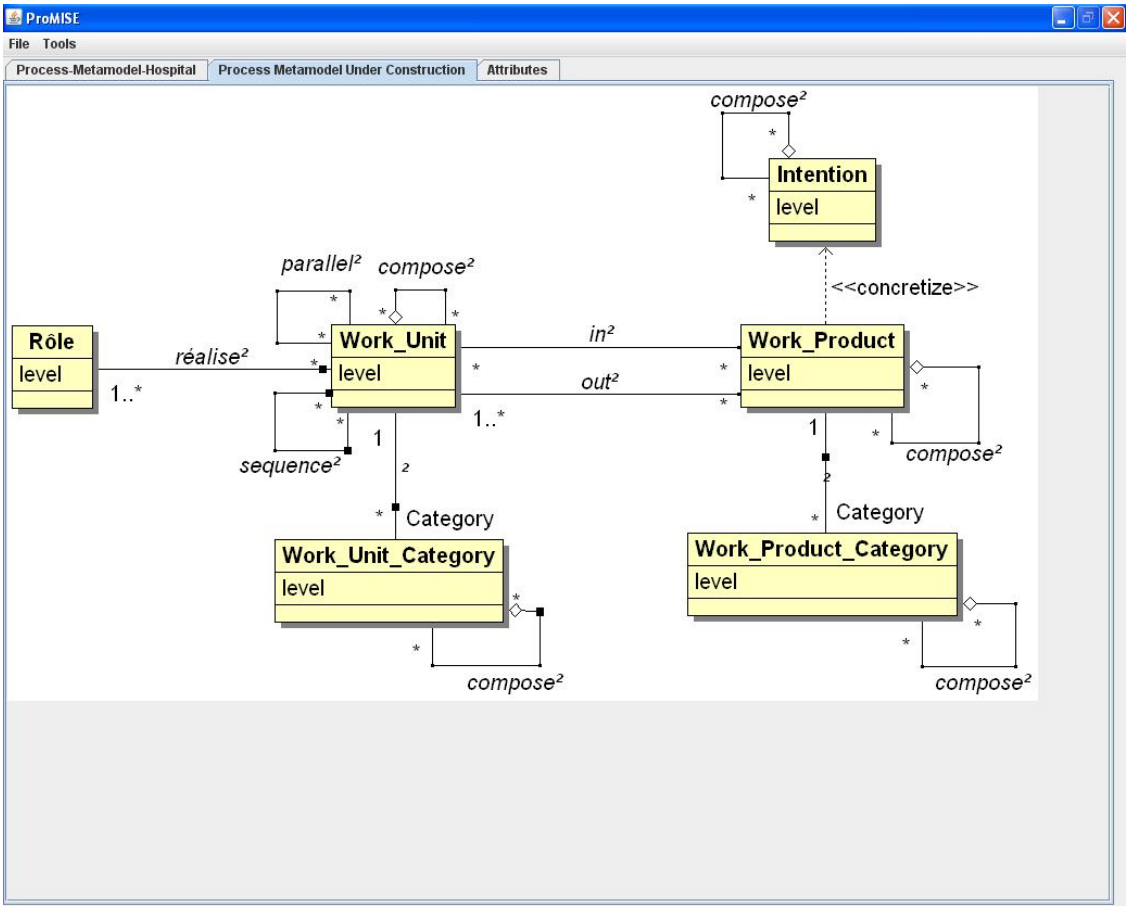


Figure 6-16. Méta-modèle de processus final.

e) Autres fonctionnalités

L'outil permet également de vérifier le méta-modèle en cours de construction. Il contrôle si aucune classe n'est esseulée et si toutes les dépendances entre classes sont bien respectées. Cette fonctionnalité a peu d'intérêt pour le moment, étant donné que la construction du méta-modèle est déjà contrainte par ces règles. Cependant, si l'outil est amené à évoluer et à proposer la modification directe du méta-modèle, il faudra pouvoir réaliser ces vérifications.

L'outil permet aussi d'exporter le méta-modèle de processus en cours de construction au format XMI. XMI est un standard de l'OMG permettant notamment l'échange de diagrammes UML. Ainsi, le fichier XMI obtenu à partir de notre outil peut être importé dans des AGL, pour être éventuellement visualisé et modifié manuellement.

f) Évolutivité de l'outil

L'outil a pour point d'entrée le graphe conceptuel défini au format XMI. Si le graphe évolue, l'outil prend en compte les modifications immédiatement. Cependant, si de nouveaux concepts secondaires font appel à de nouveaux patrons, ceux-ci devront être implémentés sous forme de méthodes dans le programme Java.

6.3. Synthèse

Dans ce chapitre, nous avons présenté un état de l'art de différents outils pour la modélisation des processus d'ingénierie de SI et étudié comment ces outils pouvaient servir pour l'implémentation de la méthode que nous proposons. Aucun de ces outils ne permettant de créer des méta-modèles de processus adaptés, fédérés et multi-points de vue, nous avons ensuite présenté un prototype permettant de définir des méta-modèles de processus d'ingénierie de SI selon notre méthode.

L'outil guide les ingénieurs des méthodes dans la création de leurs méta-modèles de processus, via le graphe conceptuel.

L'outil est pour le moment limité à la création de méta-modèles de processus. La fonctionnalité permettant l'ajout d'attributs aux classes du méta-modèle en cours de construction est en cours de développement. À long terme, il devra permettre d'instancier des modèles de processus en définissant des formalismes propres à chaque organisation ou projet. L'outil devra également permettre d'instancier les modèles définis pour suivre l'exécution de projets particuliers.

Enfin, un effort sur l'ergonomie sera fait, pour faciliter l'appropriation et l'utilisation de l'outil par les ingénieurs des méthodes.

7. VALIDATION

7.1. Contexte des expériences

Afin de valider et d'évaluer notre travail de thèse auprès d'utilisateurs potentiels, nous avons mis en place deux expériences différentes :

- la première expérience réalisée auprès d'experts des systèmes d'information a permis de tester le graphe conceptuel et la méthode, dans le but d'améliorer la définition des concepts et des relations ;
- la seconde expérience réalisée auprès de professionnels du domaine de l'ingénierie des SI a permis de mesurer la perception de la méthode et de recueillir des avis sur le prototype de l'outil, présenté au Chapitre 6.

Pour mettre en œuvre ces expérimentations, nous avons collaboré avec Mme Nadine Mandran, ingénieure méthodes et qualité de la plateforme Marvelig. Marvelig[14], plateforme d'expérimentations scientifiques du Laboratoire d'Informatique de Grenoble, a pour but de capitaliser les prototypes réalisés par les différentes équipes de recherche, de mutualiser les potentiels de chaque équipe, de communiquer sur les produits de la recherche du laboratoire et de mettre en œuvre des méthodes d'évaluation de concepts et d'outils. C'est grâce à ce dispositif que nous avons pu mettre en place les expériences présentées dans ce chapitre.

7.2. Évaluation du graphe conceptuel et du méta-modèle

7.2.1. Description du protocole

Cette expérience avait pour but de mesurer la compréhension, l'usage et l'utilisabilité du graphe conceptuel ainsi que l'usage du méta-modèle de processus. L'usage est défini comme « l'utilisation, [l'] emploi de quelque chose ; [la] possibilité de se servir de quelque chose » (CNTL, 2009). L'utilisabilité définit « le degré selon lequel un produit peut être utilisé, par des utilisateurs identifiés, pour atteindre des buts définis avec efficacité, efficience et satisfaction, dans un contexte d'utilisation spécifié » (ISO, 1998).

L'expérience a été réalisée auprès de 10 experts (doctorants et enseignants-chercheurs) en modélisation et méthodes pour l'ingénierie des systèmes d'information, sous la forme de focus group. Le focus group, ou groupe focalisé, est une méthode qualitative permettant le recueil d'informations. Il s'agit d'un groupe de discussion semi-structuré, modéré par un animateur neutre, qui a pour but de collecter des informations sur un nombre limité de thèmes définis à l'avance (APES, 2004). Le Tableau 7-1 résume le profil des sujets.

[14] https://marvelig.imag.fr

Nombre	Moyenne d'âge	Sexe	Fonctions
10	34 ans (min. 26 ans, max. 47 ans)	6 femmes, 4 hommes	7 doctorants, 2 enseignants-chercheurs, 1 stagiaire CNAM

Tableau 7-1. Profil des sujets.

Deux questionnaires ont été distribués :

- le premier questionnaire a été distribué avant de commencer les expériences, il s'agissait de connaître les pratiques des sujets en matière de développement de systèmes d'information et d'utilisation de méthodes de conception et de développement (cf. Annexe C1) ;
- le second questionnaire a été distribué à la fin de la deuxième expérience. Il s'agissait ici de mesurer l'opinion des sujets quant à l'usage et l'utilisabilité du graphe conceptuel et de la méthode (cf. Annexe C2).

De plus, un ensemble d'exercices a été réalisé afin de tester les hypothèses que nous souhaitions vérifier. Les sections suivantes décrivent ces hypothèses, les exercices réalisés et les résultats obtenus pour chaque point à tester.

7.2.2. Compréhension du graphe conceptuel

a) Hypothèses

Dans un premier temps, nous voulions tester la compréhension du graphe conceptuel. La compréhension du graphe conceptuel passe par la compréhension des concepts et des relations. Il s'agissait de faire construire un graphe conceptuel, par binôme, en fournissant la liste des concepts, leur définition ainsi que la définition des relations.

Notre hypothèse était la suivante :

- Les binômes vont construire un graphe conceptuel identique à celui que nous avons défini dans le Chapitre 4.

b) Exercice

Les sujets devaient construire leur propre graphe conceptuel à l'aide de papiers autocollants représentant les concepts et de stylos effaçables pour tracer les relations entre les concepts, comme le montre la Figure 7-1. Ils s'appuyaient sur le dictionnaire des concepts (cf. Annexe C3) ainsi que la définition des relations Précision, Complétude et Abstraction pour construire leur graphe conceptuel.

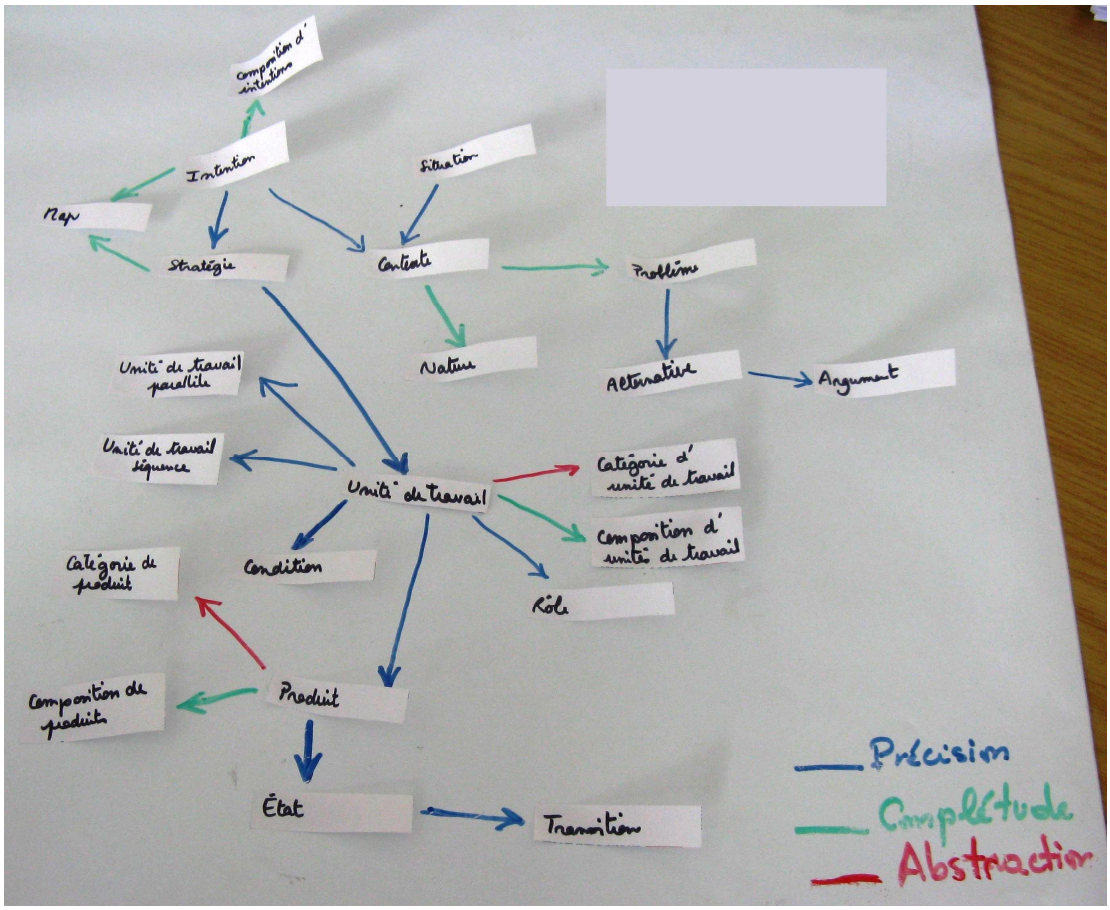


Figure 7-1. Graphe conceptuel réalisé par un des binômes.

c) Résultats

Les résultats ont montré que la compréhension des concepts n'était pas évidente pour tout le monde, car les définitions étaient parfois ambiguës. La difficulté réside principalement dans le vocabulaire employé qui peut être très spécifique en fonction des domaines : *« Je ne viens pas de l'informatique. La stratégie est quelque chose dans une entreprise qui est défini à long terme de manière globale. Je pense que si tu fais la même expérience avec des gens qui sont en mécanique ou ce genre de choses, tu vas avoir d'autres biais qui sont liés au métier »* (sujet 5).

D'autre part, nos définitions utilisaient des concepts qui n'étaient pas définis, ce qui a posé quelques problèmes : *« On a eu du mal à identifier l'action qui était donnée dans les définitions, mais c'était l'unité de travail »* (sujet 2), *« Le vocabulaire utilisé dans les définitions n'est pas parmi les concepts comme action et processus »* (sujet 6), *« Il manquait la notion d'action et de processus »* (sujet 8).

Les synonymes proposés ont parfois été difficiles à comprendre et ont créé une certaine confusion. De même, la liste des définitions des concepts commençait par des concepts abstraits (Intention, Stratégie, etc.), concepts qui sont moins abordables dans une phase de première utilisation du graphe conceptuel. Les sujets ont déclaré que les

concepts les plus concrets, c'est-à-dire du niveau opérationnel, devaient apparaitre au début du dictionnaire.

Néanmoins, tous les binômes ont réussi à regrouper les concepts par point de vue, autour des concepts centraux comme Unité de travail, Produit et MAP essentiellement.

Cependant, le point le plus délicat a concerné les relations qui n'ont pas été comprises de façon satisfaisante. Chaque binôme a en effet interprété les relations différemment et les relations entre concepts ont été finalement toutes différentes d'un groupe à l'autre : *« Est-ce qu'Abstraction est le contraire de Précision ? »* (sujet 1), *« Le terme de complétude pour moi dans le cadre de la qualité c'est le côté entier, qui décrit tout, là ça n'a pas l'air d'y correspondre »* (sujet 5), *« Il y a la notion de composition/décomposition dans les définitions, mais il n'y a pas de relation qui permet de le faire »* (sujet 2), *« On a beaucoup pataugé avec les relations »* (sujet 9), *« On ne savait pas si la décomposition correspondait à de la précision ou de l'abstraction »* (sujet 8).

Ces résultats nous ont conduits à modifier le dictionnaire des concepts en fonction des remarques de ces utilisateurs. La définition des concepts et des relations a été améliorée et rédigée à nouveau. Les modifications apportées ont été soumises à évaluation auprès des mêmes sujets, cf. 7.2.6.

7.2.3. Usage du graphe conceptuel

a) Hypothèses

Dans un second temps, nous avons fait réaliser un exercice au focus group pour appliquer la méthode sur un cas d'étude. L'un des objectifs de cet exercice était de tester l'usage du graphe conceptuel. Les hypothèses étaient les suivantes :

- le graphe conceptuel couvre les principaux concepts des processus d'ingénierie de SI. Le graphe conceptuel est complet, tous les concepts nécessaires à la modélisation des processus d'ingénierie de SI sont proposés ;
- le graphe conceptuel est utile pour la construction de méta-modèles de processus d'ingénierie de SI. Il est plus facile de créer un méta-modèle de processus à partir de notre méthode et du graphe conceptuel que sans aucune base.

b) Exercice

Par binôme, les sujets devaient sélectionner les concepts décrits dans le cas d'étude (cf. Annexe C4), à partir de notre version du graphe conceptuel. Au fur et à mesure que les concepts étaient sélectionnés dans le graphe, le méta-modèle de processus était construit manuellement par les expérimentateurs, en se basant sur le méta-modèle de domaine et les patrons, sans que les sujets s'en préoccupent.

c) Résultats

Les résultats de cet exercice sont divers. Pour 3 sujets sur 10, trouver la correspondance entre les concepts abordés dans le cas d'étude et les concepts du graphe a été facile, notamment pour les concepts du niveau opérationnel (1 sujet sur 10).

Cependant, 4 autres sujets ont trouvé difficile d'établir la correspondance entre les concepts du cas d'étude et les concepts du graphe surtout au niveau intentionnel (2 sujets sur 10).

De plus, les définitions des concepts n'étant pas suffisamment claires, les sujets ont ressenti un besoin d'aide en cours d'utilisation. Les sujets soulignent l'utilité des définitions, mais soulignent aussi que celles-ci sont confuses. Le point faible le plus important du graphe conceptuel concerne donc le vocabulaire des concepts et des relations.

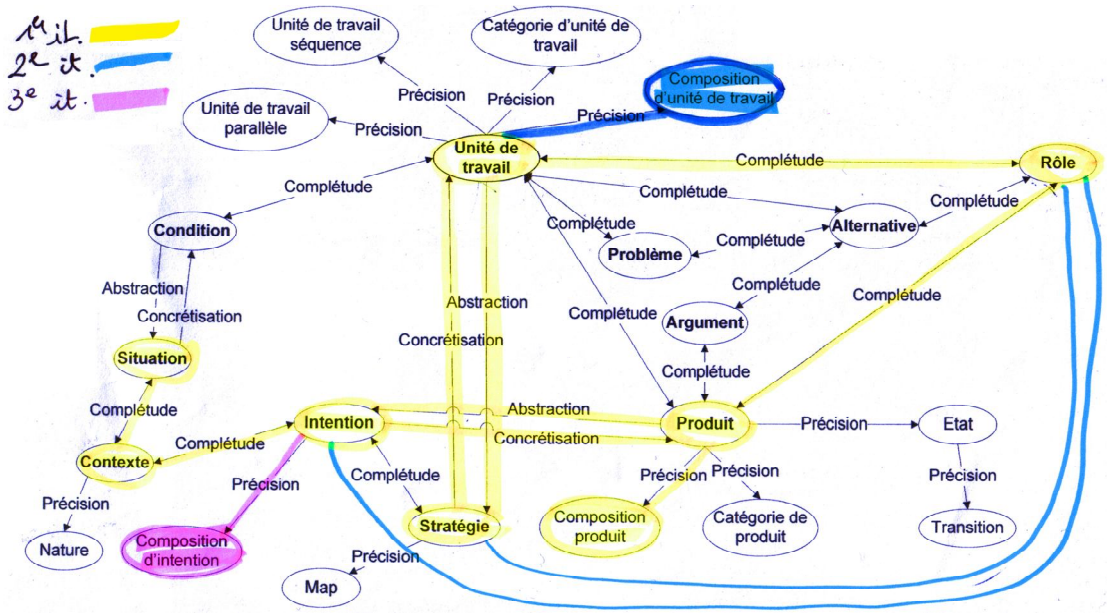


Figure 7-2. Graphe conceptuel après 3 itérations réalisé par un des binômes.

Les résultats obtenus suite à cette expérience confirment les conclusions de l'expérience sur la compréhension du graphe conceptuel. Il était donc nécessaire et indispensable de revoir les définitions des concepts et des relations pour améliorer l'utilisabilité du graphe conceptuel.

7.2.4. Usage du méta-modèle de processus

a) Hypothèses

Afin de tester l'usage du méta-modèle de processus, nous avons émis deux hypothèses :

– plusieurs itérations sont nécessaires pour arriver au méta-modèle complet. Le méta-modèle de processus ne peut être complet en une seule itération, car des

concepts manquants seront découverts lors de l'instanciation du méta-modèle. Les utilisateurs devront donc revenir au graphe conceptuel pour sélectionner de nouveaux concepts ;

– le cercle vertueux méta-modèle/instance permet l'innovation. Les itérations entre le méta-modèle et le modèle permettent aux ingénieurs des méthodes de trouver de nouveaux éléments à modéliser, d'enrichir les modèles de processus réalisés et ainsi de mieux cadrer l'exécution des processus dans le cadre de leurs projets.

b) Exercice

L'exercice permettant de valider ces hypothèses était le suivant : à partir de son méta-modèle de processus, chaque binôme devait ensuite choisir un formalisme pour chaque élément du méta-modèle de processus (classe et association) à l'aide (ou pas) de la liste de formalismes présentée dans le Chapitre 5.

Lorsque le formalisme était défini, les binômes pouvaient instancier le méta-modèle pour réaliser le modèle de processus. Si les binômes se rendaient compte qu'il manquait un concept en cours d'instanciation, ils avaient le droit de retourner au graphe conceptuel pour sélectionner le concept manquant. Le méta-modèle de processus était remis à jour par l'expérimentateur et les sujets pouvaient continuer l'exercice.

c) Résultats

9 sujets sur 10 pensent que le méta-modèle de processus est indispensable pour instancier le modèle, contre 1 sujet sur 10 (sujet non habitué à la méta-modélisation). Comme le précisent certains sujets, le méta-modèle permet de structurer le modèle, de comprendre les concepts du modèle, de guider son instanciation et d'éviter les oublis et les erreurs. *« C'est un guide important, je crois que sans le méta-modèle la réalisation du modèle serait plus difficile »* (sujet 3) et *« un méta-modèle permet de fixer les possibilités et les limites du modèle »* (sujet 6).

Certains sujets auraient cependant aimé pouvoir modifier directement le méta-modèle de processus réalisé, car certaines relations manquaient dans le graphe conceptuel (4 sujets sur 9) : par exemple, entre Rôle et Intention, et entre Rôle et Stratégie.

Le fait de pouvoir itérer entre le méta-modèle et le modèle de processus permet effectivement de compléter ou de raffiner le méta-modèle de processus (6 sujets sur 7). L'itération a permis *« de rajouter d'autres relations »* (sujet 1) ou de *« gommer certains concepts inutiles »* (sujet 4).

Finalement, les sujets étaient plutôt satisfaits de l'exercice réalisé et des résultats obtenus. Le méta-modèle créé en amont permet d'instancier le modèle plus facilement et de manière plus rigoureuse : *« la méthode définit un cadre assurant la complétude du modèle produit et la cohérence »* (sujet 2) et *« le diagramme UML est plus facile à instancier »* (sujet 6).

La Figure 7-3 présente un des modèles de processus instanciés à partir du méta-modèle de processus.

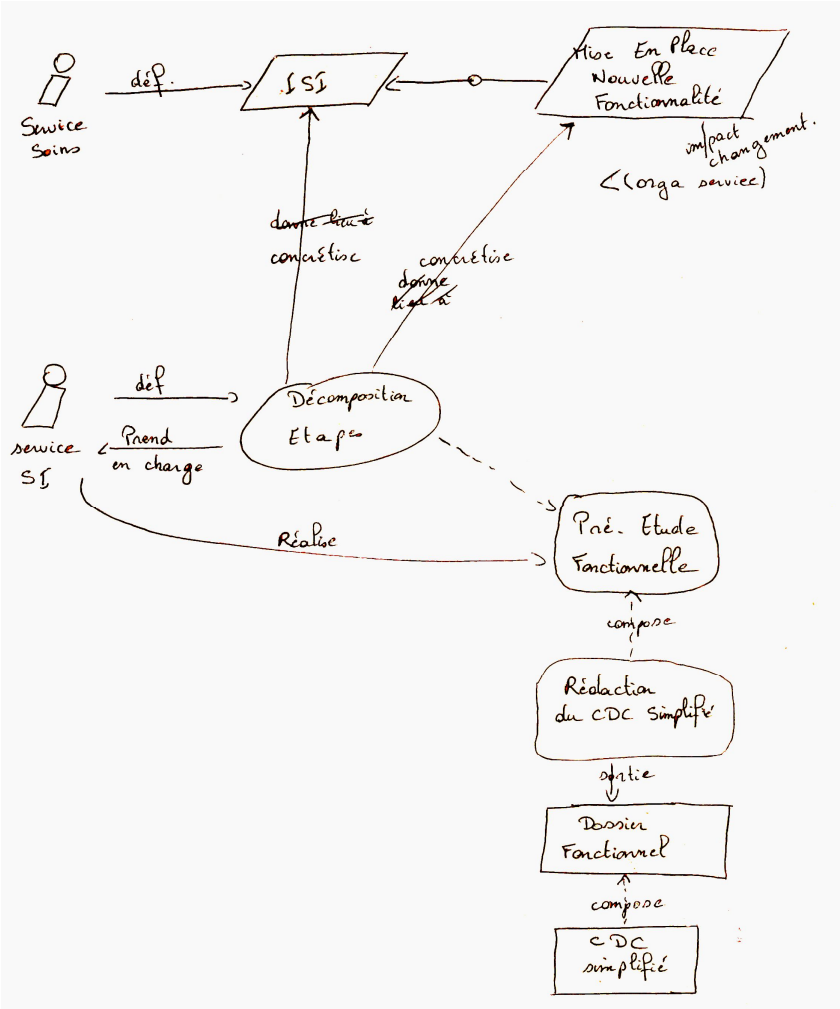


Figure 7-3. Modèle de processus réalisé par un des binômes.

7.2.5. Utilisabilité du graphe conceptuel

a) Hypothèses

La dernière hypothèse permettait de mesurer l'utilisabilité du graphe conceptuel :

– le graphe conceptuel est facile à utiliser et est efficace. Il permet aux ingénieurs des méthodes de sélectionner facilement les concepts dont ils ont besoin.

b) Exercice

L'utilisabilité a été testée grâce à l'exercice précédent.

c) Résultats

Les sujets pensent que la méthode peut leur faire gagner du temps (7 sujets sur 9, 1 sans opinion), en particulier pour un guidage/cadrage du problème à résoudre et ainsi leur éviter des erreurs lors de la conception du modèle.

Les sujets pensent qu'un apprentissage est nécessaire pour utiliser cette méthode (9 sujets sur 10). Cependant, le coût d'apprentissage est faible, quelques heures suffiraient (8/10), comme le précise un des sujets : *« Cette méthode s'apprend rapidement, on peut être rapidement opérationnel dessus. Je ne vais pas passer 15 jours avant de sortir un modèle »* (sujet 2).

Tous les sujets se sont déclarés prêts à utiliser cette méthode dans leur contexte professionnel, pour sa rigueur et sa facilité de compréhension : *« elle permet d'obtenir un résultat de qualité au travers d'une approche rigoureuse »*. De plus, certains sujets déclarent que le graphe conceptuel proposé semble utilisable dans d'autres domaines que l'ingénierie des SI.

Enfin, 9 sujets sur 10 pensent pouvoir décrire le fonctionnement de la méthode à une tierce personne. Ce résultat semble confirmer la facilité d'appropriation de la méthode.

7.2.6. Validation des définitions des concepts et des relations

a) Hypothèses

Au vu des résultats concernant la compréhension des concepts et des relations, il a été nécessaire de modifier les définitions et le dictionnaire proposés lors du premier exercice. Nous devions alors tester les modifications apportées aux définitions des concepts et des relations. Notre hypothèse était la suivante :

– les définitions des concepts et des relations sont plus claires et plus précises.

b) Modalité

À travers un entretien, nous avons présenté les nouvelles définitions des concepts et des relations et questionné les sujets individuellement pour recueillir leur avis sur ces modifications.

Pour être tout à fait correct d'un point de vue expérimental, il aurait néanmoins fallu refaire le même exercice que celui présenté au 7.2.2, avec des experts différents.

c) Résultats

Sur les 7 sujets que nous avons questionnés, tous ont noté des améliorations. Au niveau des concepts, les définitions paraissent plus claires et cohérentes, notamment grâce « aux liens entre les concepts des différentes définitions qui permettent de comprendre facilement les définitions » (sujet 1). Les exemples sont indispensables à la meilleure compréhension des concepts : *« Ce sont les exemples qui m'ont aidé à bien voir les choses »* (sujet 1).

En ce qui concerne la définition des relations, l'usage des verbes au lieu des substantifs parait plus simple pour 6 sujets sur 7 : « Je préfère les verbes. La dernière fois, quand on voulait tracer les flèches, on se posait toujours la question du sens » (sujet 1), *« Avec les verbes, je vois dans quel sens ça va »* (sujet 6) et *« L'aspect verbe me parle mieux que l'aspect substantif »* (sujet 10). Il manque cependant des exemples de ces relations (3 sujets sur 7).

La séparation des niveaux intentionnels et opérationnels dans le dictionnaire des concepts a eu un impact positif sur les sujets : *« C'est bien d'avoir décomposé les niveaux intentionnel et opérationnel. Ça limite les sous-ensembles de concepts que je dois manipuler en même temps pour les assembler »* (sujet 10).

Enfin, la colonne des synonymes est elle aussi utile pour les sujets, bien qu'elle ne soit pas toujours complétée selon les concepts : *« La colonne des synonymes est intéressante et nécessaire. Elle permet de me conforter dans l'idée que je voulais exprimer »* (sujet 6), mais *« Il manque encore des exemples et des synonymes pour Nature »* (sujet 10).

7.2.7. Synthèse

Le Tableau 7-2 récapitule les hypothèses et les constats réalisés suite à l'expérience avec le focus group.

Hypothèses	Constats	
	Points forts	Points faibles
Les binômes vont construire le même graphe conceptuel.	Les concepts ont été regroupés correctement.	Les relations entre les concepts n'ont pas été bien comprises.
Le graphe est utile pour la construction de méta-modèles de processus d'ISI.	9 sujets sur 10 pensent que le passage par le méta-modèle était indispensable pour réaliser le modèle.	1 sujet pense le contraire, cependant il n'est pas habitué à la méta-modélisation.
Le graphe conceptuel couvre les principaux concepts de processus d'ingénierie de systèmes d'information.	Hypothèse vérifiée en ce qui concerne les concepts : aucun binôme n'a ajouté de nouveaux concepts.	Hypothèse non vérifiée en ce qui concerne les relations : il manque des relations entre Rôle et Stratégie, et Rôle et Intention.
Il faut faire plusieurs itérations pour arriver au méta-modèle complet.	Hypothèse vérifiée : les sujets ont réalisé le méta-modèle complet en 2 ou 3 itérations.	
Le cercle vertueux méta-modèle/instance permet l'innovation.	6 sujets sur 7 pensent que l'itération méta-modèle/instanciation leur a permis de trouver de nouvelles idées.	1 sujet pense que l'utilisation du graphe permet de trouver le bon modèle du premier coup.

Le graphe est facile à utiliser et est efficace.	Hypothèse vérifiée : les sujets ont tous été satisfaits de l'exercice réalisé (10/10) ainsi que de la méthode utilisée (10/10).	
Les définitions des concepts et des relations sont plus claires et plus précises.	Hypothèse vérifiée : les sujets sont satisfaits des nouvelles définitions (7/7).	

Tableau 7-2. Grille d'analyse de l'expérience du focus group.

Le dictionnaire des concepts présenté au Chapitre 5 prend en compte les remarques issues de l'expérimentation du focus group. Les relations entre Rôle et Stratégie, et Rôle et Intention n'ont pas été prises en compte dans le graphe conceptuel et dans le méta-modèle de domaine pour le moment, car elles impliquent des relations entre le niveau intentionnel et opérationnel qu'il faut étudier en détail.

7.3. Perception de la méthode par des industriels

Afin de cerner la perception de notre méthode par des professionnels, nous l'avons présenté à une dizaine d'industriels travaillant dans le domaine de l'ingénierie des systèmes d'information, de l'ingénierie logicielle ou dans la qualité et la méthodologie.

7.3.1. Description du protocole

Les entretiens individuels qualitatifs avaient pour but d'une part de connaître les habitudes de travail des sujets en matière de méthode d'ingénierie de SI et leur mise en œuvre dans leur organisation, d'autre part de recueillir leur point de vue sur notre méthode et notre outil. Une étude qualitative consiste à récolter un maximum d'avis et d'idées différentes, il ne s'agit pas de quantifier l'usage ou l'utilisabilité de l'outil ou de la méthode. Le Tableau 7-3 résume le profil des sujets.

Nombre	Moyenne d'âge	Sexe	Fonctions
9	39 (min. 28 ans, max. 49 ans)	2 femmes et 7 hommes	1 gérant en informatique industriel, 1 analyste programmeur, 1 ingénieur logiciel, 1 ingénieur assurance qualité, 1 responsable informatique, 1 consultant logiciel scientifique, 1 enseignant chercheur, 1 analyste, 1 chef de projet

Tableau 7-3. Profil des sujets.

Dans un premier temps, nous présentions la définition d'une méthode et des exemples de processus d'ingénierie de SI. Ensuite, nous questionnions les sujets sur leurs habitudes de travail (voir questionnaire en Annexe D1). Nous présentions ensuite les points clés de notre approche : la méta-modélisation des processus, la méthode d'une manière globale et simplifiée, les différents points de vue, les niveaux d'abstraction, le graphe conceptuel et les relations dans le graphe. Pour chacun de ces points, nous questionnions les sujets.

Enfin, nous présentions un prototype de l'outil en 3 étapes. Le prototype était en partie basé sur l'outil présenté dans le Chapitre 6. La première étape de présentation du prototype consistait à dérouler un exemple jusqu'à la création d'un graphe conceptuel correspondant au cas d'étude présenté au Chapitre 5. Nous posions ensuite un certain nombre de questions sur cette partie.

Dans un second temps, nous montrions le méta-modèle de processus réalisé en UML via l'outil, toujours en questionnant les sujets.

Enfin, nous simulions l'étape finale qui consiste en la sélection d'un formalisme pour le méta-modèle réalisé et à montrer un exemple d'instanciation d'un modèle de processus selon ce formalisme (notons que cette partie n'étant pas primordiale, car de nombreux outils d'exécution de processus existent). Deux captures d'écran sont présentées en Annexe D2. Le méta-modèle de processus et le modèle de processus présentés étaient ceux du cas d'étude du Chapitre 5. Nous posions ensuite des questions sur l'outil et la méthode dans leur ensemble.

7.3.2. Analyse des résultats

a) Bilan sur les habitudes de travail

La majorité des sujets (7 sujets sur 9) utilisent des méthodes ou pseudos-méthodes. D'une part, *« une méthode fournit un guide, définit les tâches à faire, les étapes à suivre qui sont essentielles pour cadrer l'esprit du concepteur, du développement »* (sujet 3). D'autre part, une méthode est définie dans un *« objectif de réutilisabilité, capitalisation du processus »* (sujet 2). L'aspect de la communication avec le client est également relevé : *« La méthode aide à la communication avec le client »* (sujet 1). Enfin, les méthodes ont également un *« aspect commercial : [elles permettent une] diminution des coûts de production, une baisse des prix pour le client »* (sujet 2).

Les sujets évoquent l'utilisation des nouvelles méthodes Agiles en opposition au cycle en V. : *« [Le] cycle en V n'a plus de sens aujourd'hui, car il n'y a plus de conception, mais de l'assemblage de composants déjà développés. Les méthodes Agiles correspondent au besoin de faire du prototypage, de faire des essais, de gérer les erreurs »* (sujet 1), *« On ne peut plus se permettre de dire qu'on termine quelque chose et après on démarre autre chose. On a atteint un niveau tellement complexe qu'on ne peut plus travailler en cycle en V »* (sujet 6), *« On essaie d'être agiles, mais c'est difficile avec*

le client. On fait plutôt une succession de cycles en V avec des livraisons toutes les semaines. On cherche à éviter l'effet tunnel du cycle en V » (sujet 4).

La qualité des processus est également abordée (2 sujets sur 9) avec le CMMI : *« On se base fortement sur des modèles assez expérimentés et assez connus comme le CMMI »* (sujet 6), *« On essaie de se rapprocher des processus semblables au CMMI, c'est un objectif de qualité vendu dans la proposition commerciale, c'était une exigence dans l'appel d'offres. Ça les rassure sur notre capacité à livrer de la qualité »* (sujet 4). La modélisation des processus est donc aussi un argument de vente et une garantie de qualité auprès du client : *« Le fait qu'il y ait un processus dans la proposition commerciale, ça rassure [les clients], des fois c'est contractualisé »* (sujet 2).

Le suivi des méthodes est plus ou moins formel. Dans la majorité des entreprises (6 sujets sur 9), le suivi passe par des échanges oraux avec le client et des échanges de documents. Des outils comme Excel (3 sujets sur 9) sont utilisés pour réaliser le suivi des processus. D'autres outils comme ChangePoint ou Visual Studio Team System sont utilisés pour le suivi des états d'avancement.

La veille technologique qui est un moyen d'obtenir de nouvelles informations sur les méthodes de conception n'est pas généralisée au niveau des organisations, mais des individus (4 sujets sur 9). 2 sujets évoquent le fait que faire de la veille technologique coûte cher aux entreprises et que celles-ci ne sont pas prêtes à investir. Les sujets consultent donc des flux RSS, des blogs, la presse spécialisée. Les discussions informelles entre collègues sont également fructueuses.

b) Ressenti sur la méta-modélisation

Tous les sujets (9 sujets sur 9) avaient déjà entendu parler de méta-modélisation, mais seuls 4 sujets sur 9 l'ont pratiquée. De nombreux sujets (3 sujets sur 9) soulignent que la méta-modélisation permet une capitalisation de la connaissance : *« Menée convenablement c'est du gain, de la capitalisation »* (sujet 2), *« [La méta-modélisation] permet de capitaliser plein d'informations qui ne sont pas forcément faciles à exprimer en termes de modèles classiques »* (sujet 3), *« Ça permet de penser de façon générique, pour de la réutilisation »* (sujet 7).

Deux sujets abordent l'aspect IDM : *« Un méta-modèle permet à une machine de comprendre les modèles et d'opérer sur ces modèles : transformation, comparaison, versionnement »* (sujet 4), *« L'avantage c'est de pouvoir projeter facilement des modèles vers des plateformes technologiques et passer d'un modèle à un autre »* (sujet 5).

Cependant, les sujets précisent que la méta-modélisation n'a pas d'intérêt immédiat dans un environnement industriel : *« Ça n'a aucun intérêt pour ce que nous on en voit »* (sujet 4), *« Les gens n'ont pas le temps de faire ce type d'études, il y a des réductions de coûts, du manque de ressources, ils ne peuvent pas »* (sujet 6).

D'autre part, une formation à la méta-modélisation est nécessaire pour que les gens y voient un intérêt : *« J'ai du mal à utiliser du méta. Je préfère un modèle tout fait plutôt*

qu'un méta-modèle. La personne qui est capable de méta-modéliser ses activités, c'est qu'elle a déjà fait beaucoup de projets pour avoir ce degré d'abstraction » (sujet 1), *« Les industriels ne sont pas du tout familiers avec ce genre de notions. Cela demande un apprentissage. Ce n'est pas à leur portée immédiate »* (sujet 3), *« Il y a un manque de formation, [la méta-modélisation] demande un effort supplémentaire »* (sujet 5), *« ce n'est pas à la portée de tous »* (sujet 9).

Un sujet aborde le problème du niveau de détail des méta-modèles : *« Si c'est trop général, ça ne sert à rien. La difficulté est de trouver le juste milieu en terme de granularité : à partir du moment où des consultants passent leur temps à remodéliser des choses qu'ils ont déjà modélisées, c'est qu'on est allé trop loin. Il faut que ça reste applicable, il faut que le savoir soit exploitable »* (sujet 2).

La méta-modélisation a donc un intérêt, mais il faut avant tout qu'elle soit utile : *« Il faut montrer quelque chose d'utile. Les gens ont tellement de tâches à faire que proposer des choses automatiques dans la gestion de leur quotidien, ce sera magnifique »* (sujet 6).

c) Ressenti sur la méthode

L'ensemble des sujets (9 sujets sur 9) trouve les points de vue intéressants : *« Ils sont très adéquats »* (sujet 3), *« Tous ces points de vue sont des principes qu'on peut exploiter. Ça couvre assez, ça colle assez avec le principe qu'on chercher à appliquer »* (sujet 6), *« Ça permet d'apprendre rapidement avec un référentiel »* (sujet 2), *« Ce qui est pas mal sur ces points de vue, c'est de voir comment ils s'imbriquent les uns avec les autres »* (sujet 9).

Les sujets reconnaissent leur travail au niveau du point de vue activité : *« On le voit sur les plannings, ça fait partie du genre de représentation visuelle qu'on a l'habitude de voir »* (sujet 4). Le point de vue produit est également bien connu : *« On s'échange des documents qu'on versionne. C'est la validation interne ou du client qui fait qu'on change l'état du document »* (sujet 4).

Le point de vue décision est également courant : *« C'est la méthodologie problème – solution utilisée dans mon entreprise »* (sujet 1), *« La modélisation de l'aspect décision est assez classique »* (sujet 2), *« [L'aspect décision] peut être utilisé toutes les fois où il faut contrôler les dérives d'un plan projet »* (sujet 6). Mais ce point de vue n'est pas forcément modélisé : *« Ce n'est jamais modélisé, même pas représenté au tableau quand on a des problèmes, on ne fait qu'en discuter »* (sujet 4).

Le point de vue contexte est également utile : *« C'est un état des lieux de là où on est et de là où on veut aller »* (sujet 1), *« C'est à la réunion de lancement d'un projet : voilà où on en est, voilà ce qu'on veut obtenir. Réunion à la fin : ce qui a été atteint ou pas. C'est du verbal »* (sujet 4).

Tous s'accordent sur le fait que le point de vue stratégie concerne les décideurs : *« Ce n'est pas moi qui établis [les stratégies], je ne fais que les suivre. Ce sont les chefs de projet, le groupe de direction qui les établit »* (sujet 3), *« La stratégie c'est un plan*

d'exécution du projet, des objectifs du projet pas très détaillés définis par les program ou project managers » (sujet 6), *« La stratégie c'est pour celui qui achète »* (sujet 1), *« La stratégie est politique : elle va orienter ce qu'on veut faire, elle est donnée par la direction des systèmes d'information »* (sujet 8).

Les niveaux intentionnel et opérationnel sont bien reconnus par les différents sujets. Le niveau intentionnel s'adresse aux décideurs : *« C'est le niveau commercial »* (sujet 1), *« Ce qu'on voudrait faire. Il peut être influencé par la stratégie de l'entreprise, il n'est pas entièrement fixé par le développeur ou le chef de projet »* (sujet 2), *« Le chef de projet a intégré la demande du client, a fait sa réponse en fonction, il connaît les intentions et coordonne les taches de développement. Il donne les actions à mettre en place. C'est sa responsabilité »* (sujet 4).

Le niveau opérationnel correspond à *« ce qu'on a fait vraiment »* (sujet 3), *« Il est de notre responsabilité (analystes-programmeurs) : les actions à réaliser, dans le temps et avec le livrable attendu sont menées à bien ou alerter si on ne peut pas le faire, alors le chef de projet change l'intention et on revient à l'opérationnel »* (sujet 4).

Les deux niveaux sont *« utiles, je vois les deux niveaux dans la gestion de projet. C'est pour ça qu'on a plusieurs niveaux de responsabilités : ceux qui exécutent les actions, ceux qui sont responsables, qui définissent les objectifs »* (sujet 6).

d) Ressenti sur la construction du PMUC

Le graphe conceptuel présenté en milieu de démonstration du prototype a été accueilli différemment selon les sujets. Certains trouvent un intérêt dans le graphe lui-même : *« L'utilisateur dispose d'un référentiel auquel il peut se raccrocher »* (sujet 2), *« Il oblige à se poser des questions »* (sujet 7), *« Je crois que ça pourrait être une aide, c'est prêt, visuel, interactif, ça permet de naviguer dans les concepts de méta-modélisation »* (sujet 6). L'aspect visuel du graphe est un point important : *« Si le client veut une formalisation de ses habitudes, l'outil est assez structurant. Il permet d'avoir un résultat compréhensible, graphique, même si ça doit être assez difficile à faire passer pour quelqu'un qui n'est pas informaticien »* (sujet 2), *« C'est un support visuel, c'est plus facilement abordable »* (sujet 3).

Pour d'autres, il est difficilement compréhensible, car trop abstrait : *« Il faut le modèle pour mieux comprendre le méta-modèle »* (sujet 4), *« Il serait intéressant de pouvoir l'appliquer à quelque chose de concret, c'est-à-dire descendre d'un niveau »* (sujet 7), *« Ça reste dans l'abstraction »* (sujet 1), *« Pour moi c'est trop abstrait. À l'heure actuelle, on n'arrive déjà pas à modéliser à un niveau plus bas »* (sujet 8).

Les sujets soulignent que le graphe est une première étape pour aller plus loin dans la modélisation des processus : *« Il ne faut pas donner ça aux utilisateurs. Une fois qu'on a bien compris ce monde-là, il faut déclencher quelque chose qui va servir aux utilisateurs, qu'ils puissent se l'approprier et l'utiliser »* (sujet 6), *« J'ai besoin d'une instance de ce graphe-là »* (sujet 3), *« Il manque le M1 »* (sujet 4), *« Il faut que ce soit une étape qui*

nous amène à déclencher un niveau plus concret à proposer aux utilisateurs » (sujet 6), *« Si je peux ensuite baisser d'un niveau de modélisation, je l'utiliserai »* (sujet 7), *« Si les attentes sont plus pratiques, il manque le lien avec des outils de modélisation de processus existants dans cette optique de partager les mêmes choses, pour mettre en place le suivi de ces processus »* (sujet 2), *« Je serais intéressé si je pouvais instancier les niveaux du dessous. Si ça pouvait me délivrer un fichier dans un format utilisable par des outils capables d'instancier des processus. Sur le terrain, j'aimerai quelque chose de concret, où on génère une espèce de site web, où les gens collaborent sur un même processus avec l'aide de l'outil informatique. Je veux bien faire l'effort d'abstraction, mais derrière il faut que ça se concrétise. Méta-modéliser c'est bien, il faut forcément l'instancier et le mettre en situation »* (sujet 9).

Enfin, certains sujets ont émis des suggestions pour rendre l'outil plus intéressant : *« Il faudrait permettre aux ingénieurs d'annoter le choix des concepts au fur et à mesure du processus pour justifier, croiser les informations* » (sujet 2), *«Je fonctionnerai plutôt par suppression des concepts dans le graphe ou par check-list ou notion de priorités sur les concepts »* (sujet 1).

Le diagramme de classes UML correspondant a ensuite été présenté aux sujets. Certains sujets signalent la difficulté pour des industriels de comprendre UML : *« UML est inadapté comme outil de dialogue, c'est trop compliqué, il faut avoir quelque chose de plus facile »* (sujet 1), *« Le méta-modèle va faire fuir un non-informaticien. Le graphe conceptuel est plus rassurant »* (sujet 2) *« Tout le monde n'a pas le même niveau de compréhension du méta-modèle, alors s'il n'y a pas d'exemple derrière c'est difficile »* (sujet 4). Certains sujets soulignent l'intérêt du graphe par rapport au diagramme de classes : *« La connaissance fonctionnelle n'est pas toujours exploitée par les informaticiens, il y a des gens qui ont des connaissances très précises, qui pourraient comprendre un graphe conceptuel, mais qui ne comprendraient pas le diagramme de classes »* (sujet 2).

Le diagramme de classes UML est cependant plus précis que le graphe conceptuel : *« Il y a quelques renseignements intéressants, le diagramme de classes UML est plus riche, plus précis que le graphe conceptuel »* (sujet 2), *« Dans UML il y a du formalisme pour les entrées/sorties »* (sujet 6).

La complémentarité entre le graphe conceptuel et le diagramme de classes UML est remarquée : *« C'est bien de pouvoir passer de l'un à l'autre »* (sujet 1), *« C'est bien d'avoir les deux, car on ne comprend pas les choses au premier abord avec l'un ou avec l'autre. Ils sont très complémentaires »* (sujet 3), *« Les deux nous donnent un point de vue différent : sur le graphe conceptuel, c'est plutôt des principes. UML a un formalisme un peu plus détaillé »* (sujet 6).

La majorité des sujets préfèrent néanmoins le diagramme de classes (7 sujets sur 9) au graphe conceptuel.

e) Ressenti sur l'outil dans sa globalité

Les sujets pensent que la méthode et l'outil peuvent les aider dans la définition de leurs modèles de processus : *« La méthode va me permettre de faire le choix du vocabulaire avec lequel je vais exprimer la vision du processus »* (sujet 4), *« Il y a un support visuel très intéressant. Le fait que ce soit informatisé permet de ne pas gribouiller, on ne perd pas ses feuilles de papier »* (sujet 3), *« Je vois que l'outil peut m'aider à avoir plus de flexibilité, à avoir plus de modèles à disposition pour mieux implémenter mes flots »* (sujet 6), *« Valider l'ensemble des processus qui sont conformes à ce méta-modèle et ceux qui ne le sont pas. Ça ne sert à rien de faire un méta-modèle si seulement 50% des processus y sont conformes. Ça veut dire que ce n'est pas la bonne modélisation ou que les processus de l'entreprise sont très éclatés. Il faut raccrocher le méta-modèle à l'ensemble des processus déjà modélisés. Les entreprises grossissent, elles veulent des méthodes plus carrées, elles font l'effort de modélisation, ou pour la mise en place du cadre ISO, pour modéliser leurs activités, pour estimer la qualité de cette modélisation »* (sujet 2), *« l'outil est pratique et proche du terrain »* (sujet 1), *« L'avantage du méta-modèle tel qu'il est proposé c'est qu'on peut l'enrichir pour y mettre des aspects plus organisationnels, stratégiques. On peut avoir différentes facettes d'un processus pour avoir quelque chose de très consistent »* (sujet 9).

L'outil permet également de définir des modèles utiles pour la communication : *« les documents de travail pour échanger sont sympa »* (sujet 1), *« Ça peut servir de support de communication, y compris avec la direction qui n'a pas connaissance précise de ce que font les gens, mais qui veut avoir une vision globale sur la partie SI. C'est quelque chose qui est assez visuel et facile à comprendre »* (sujet 7).

L'outil peut également aider les utilisateurs à ne pas faire d'erreurs : *« Nous aider à modéliser de façon efficace, oui, il nous donne assez de catégories pour pouvoir les réutiliser dans les processus. Il est assez riche »* (sujet 6), *« Il évite d'introduire des concepts qui ne sont pas pertinents, il permet de cadrer »* (sujet 2). Une partie des erreurs peut être évitée dans la définition des modèles de processus grâce au méta-modèle défini en amont : *« à partir du moment où j'ai bien défini mon méta-modèle »* (sujet 3), *« s'il y a un méta-modèle derrière »* (sujet 4), *« l'instanciation du méta-modèle va permettre de générer quelque chose de cohérent. Et inversement, l'instanciation du méta-modèle permet de le valider »* (sujet 9).

Un sujet insiste sur la notion de check-list : *« s'il y a une check-list, ça évite d'oublier des concepts »* (sujet 1), un autre sujet rappelle qu'il y a toujours le problème *« du décalage entre l'activité modélisée et l'activité réelle »* (sujet 5).

Le fait de pouvoir choisir son propre formalisme est intéressant, *« Cela fait partie de la flexibilité de l'outil »* (sujet 6), et peut être utile *« dans une grande entreprise, ils doivent vouloir une homogénéisation de leurs modèles »* (sujet 7).

La démarche que nous proposons va du méta-modèle aux modèles. Certains sujets préfèreraient la démarche inverse, des modèles vers le méta-modèle : *« Beaucoup*

d'entreprises ont déjà des modèles de processus définis. Il est plus facile pour un utilisateur de sélectionner les concepts par rapport aux modèles qui sont déjà définis. C'est cyclique : construction du M1 puis du M2 » (sujet 2), *« partir d'un ou plusieurs modèles en M1 pour faire le M2* » (sujet 5).

La méta-modélisation voire la modélisation n'est cependant pas une habitude pour les sujets, ce qui peut expliquer pourquoi l'approche ne leur est pas naturelle : « *Je modélise peu »* (sujet 4), *« C'est la première fois que je vais expérimenter le niveau M2, j'ai toujours travaillé aux niveaux M0 et M1 »* (sujet 6), *« Je travaille directement au M1 »* (sujet 7).

La méthode et l'outil aideraient les sujets à *« structurer la communication avec le client, dire où est son rôle dans le processus, pour la transmission de la connaissance»* (sujet 1), ou pour *« définir un méta-modèle idéal du fonctionnement d'une entreprise, c'est un élément intéressant pour l'optimisation. Si on a des difficultés à modéliser les processus avec cet outil, ça peut vouloir dire qu'il y a des dysfonctionnements dans les processus comme deux processus incohérents, des deadlocks. La démarche va permettre d'identifier, réfléchir, mettre en évidence des processus incohérents »* (sujet 2), *« définir des processus à très haut niveau et pouvoir les proposer aux clients, pouvoir intégrer ceux du client dans la base des processus qu'on pourrait avoir, pour avoir une référence, ça apporterait un plus au client, pour aider le client à améliorer ses méthodes de travail. En tant que consultant c'est aussi mon rôle : en interne définir notre processus à nous, comparer ceux des clients et chez le client, implémenter ces processus et les suivre »* (sujet 3).

Le besoin d'adaptation au domaine est souvent cité : *« Un modèle comme ça doit être le plus générique possible, mais il y a certaines spécificités liées au business des entreprises. Il faut faire de la personnalisation par rapport au domaine »* (sujet 6), *« Je ne sais pas si on obtiendrait un résultat suffisamment générique, mais pas trop »* (sujet 2).

Tous les sujets (9 sujets sur 9) pensent que l'utilisation de la méthode et de l'outil nécessite une phase d'apprentissage. La phase d'appropriation peut aller de quelques heures pour des personnes connaissant la méta-modélisation à une semaine pour les autres. Deux sujets soulignent que les diagrammes de classes UML seront difficilement compris par des non-informaticiens : *« Pour les décideurs dans les entreprises qui ne sont pas du tout familiers avec UML, ça risque d'être plus chaud »* (sujet 3), *« Un décideur pourra modéliser assez vite parce que c'est assez intuitif, mais la méta-modélisation jamais de la vie ou presque »* (sujet 7).

Quant à l'interface du prototype, certains sujets notent son manque d'ergonomie. Enfin, il est important que l'outil permette l'exécution et le suivi du processus : *« Tout ce qu'on fait doit servir à quelqu'un et à quelque chose. Le but est de donner un outil aux utilisateurs qui soit efficace pour décrire leur projet, pour l'utiliser, l'exécuter »* (sujet 6).

7.3.3. Synthèse

Les entretiens qualitatifs avec les industriels ont permis de mettre en avant, d'une part, un besoin d'outils efficaces de modélisation des processus, et d'autre part, une nécessité d'exécution et de suivi de ces processus. Notre méthode et l'outil sous-jacent répondent en partie aux difficultés et besoins énoncés. Les concepts que nous proposons ainsi que leur assemblage via le graphe conceptuel et le méta-modèle de processus permettent d'obtenir un résultat structuré et structurant.

De nombreuses perspectives ont été évoquées par les utilisateurs, pour rendre l'outil plus attractif.

CONCLUSION DE L'IMPLEMENTATION ET DE LA VALIDATION

Dans cette partie, nous avons présenté, dans un premier temps, un état de l'art des outils pour la modélisation des processus d'ingénierie de systèmes d'information. Aucun des outils présentés ne permettant de répondre aux exigences de notre méthode, nous avons implémenté un prototype.

Ce prototype permet de construire des méta-modèles de processus en se basant sur les concepts et les relations définis dans le graphe conceptuel. Au fur et à mesure de la sélection des concepts par l'ingénieur des méthodes, le diagramme de classes correspondant est construit. Le diagramme peut ensuite être exporté en XMI pour être utilisé dans un AGL ou un outil de modélisation et d'exécution de processus.

Dans le second chapitre de cette partie, nous avons présenté les deux expériences menées dans le cadre de la plateforme Marvelig du Laboratoire d'Informatique de Grenoble afin de valider notre méthode. La première expérience consistait à faire construire un graphe conceptuel au focus group, en partant des définitions des concepts et des relations. Ensuite, le focus group a dû réaliser un méta-modèle de processus et l'instancier en se basant sur notre méthode et ses différents supports.

La seconde expérience consistait en des entretiens qualitatifs auprès d'industriels du domaine de l'ingénierie des systèmes d'information pour recueillir des informations sur leurs habitudes de travail et leur opinion sur la méthode que nous proposons et le prototype de notre outil.

8. CONCLUSION ET PERSPECTIVES

8.1. Contributions

Cette thèse propose un ensemble de supports pour la méta-modélisation des processus d'ingénierie de systèmes d'information.

Tout d'abord, nous avons présenté un méta-modèle de domaine des processus d'ingénierie de SI qui contient les concepts principaux du domaine. Ce méta-modèle a été obtenu en réalisant une analyse des classes équivalentes des différents méta-modèles de processus existants et en les pondérant afin de ne sélectionner que les classes les plus importantes. Les relations définies entre ces classes proviennent également d'une analyse des différents méta-modèles de processus existants. Ce méta-modèle de domaine prend en compte les cinq points de vue de la modélisation des processus d'ingénierie de SI : activité, produit, décision, contexte et stratégie. Il prend également en compte deux niveaux d'abstraction des processus : le niveau intentionnel et le niveau opérationnel.

Nous avons également proposé des patrons génériques et de domaine spécifiques à la méta-modélisation des processus d'ingénierie de SI. Les patrons génériques sont des patrons produit qui peuvent être imités sur des classes du méta-modèle de domaine. Les patrons de domaine sont des fragments issus de méta-modèles de processus existants qui permettent d'enrichir les classes du méta-modèle de domaine.

Enfin, nous avons proposé un graphe conceptuel regroupant les concepts du méta-modèle de domaine et les concepts issus de l'imitation des patrons génériques et de domaine. Ce graphe représente l'ensemble des concepts pouvant être intégrés aux méta-modèles de processus. Il permet à la fois de cadrer la création des méta-modèles de processus et de guider les ingénieurs des méthodes dans le choix des concepts, via différentes relations. Les relations définies entre les concepts sont de trois types : complétude, abstraction et précision. La complétude permet d'étendre le méta-modèle en cours de construction à un autre point de vue. L'abstraction permet d'atteindre des concepts du niveau intentionnel et la précision permet de raffiner un concept, via l'imitation de patrons. Les relations entre les concepts ont été définies de manière cohérente, afin que les méta-modèles sous-jacents soient également cohérents.

Dans un second temps, nous avons proposé une méthode pour la construction de méta-modèles de processus. Cette méthode est basée sur le graphe conceptuel qui permet la navigation et le choix de concepts à intégrer aux méta-modèles de processus de manière cohérente. Les ingénieurs des méthodes sont également guidés dans la sélection des attributs des classes de leurs méta-modèles. La méthode propose ensuite l'instanciation des méta-modèles de processus, en passant par la sélection de formalismes adaptés au contexte de l'organisation.

Dans une dernière partie, nous avons proposé un outil permettant de créer des méta-modèles de processus en suivant notre méthode. L'outil permet d'exporter le méta-

modèle créé au format XMI pour être exploitable par d'autres outils comme des AGL ou des outils de modélisation et d'exécution de processus.

Nous avons enfin présenté les résultats de deux expériences réalisées dans le cadre de la pépinière d'expérimentations scientifiques du Laboratoire d'Informatique de Grenoble, Marvelig. La première expérience a permis de mesurer l'usage et l'utilisabilité du graphe conceptuel et de la méthode auprès d'un focus group composé de 10 experts en modélisation du monde universitaire. La seconde expérience a permis de réaliser des entretiens qualitatifs auprès d'industriels du domaine de l'ingénierie des systèmes d'information.

8.2. Perspectives

Nous pouvons évoquer différentes pistes pour poursuivre le travail commencé, certaines ont été évoquées durant les expérimentations, notamment par les industriels.

À court terme, le graphe conceptuel doit être complété. Il doit être amélioré, pour mieux prendre en compte le concept de Rôle dans le niveau intentionnel des processus par exemple, et il faudrait introduire la notion d'« unité de coût » qui permettrait d'évaluer le coût d'une unité de travail. Ces nouveaux concepts permettraient de mieux modéliser les processus du niveau intentionnel. L'outil doit également être complété et amélioré du point de vue de l'ergonomie et du guidage des ingénieurs des méthodes dans la définition de leurs méta-modèles de processus. Un couplage avec des outils de modélisation/exécution de processus devra être réalisé pour permettre l'instanciation de modèles de processus et leur exécution dans le cadre de projets.

À moyen terme, le graphe conceptuel devra être étendu à d'autres domaines. Constitué à partir de concepts de la méta-modélisation des processus d'ingénierie de SI, il semble être suffisamment général pour permettre la définition d'autres types de processus comme la définition de démarches qualité, ou des processus métier comme la logistique ou l'ingénierie mécanique. Il s'agirait dans un premier temps de sélectionner les concepts identiques à tous les domaines, qui seront probablement les concepts principaux de notre graphe conceptuel. Dans un second temps, il faudra spécialiser le graphe conceptuel pour prendre en compte les concepts secondaires spécifiques au domaine étudié. Ces graphes permettraient à chaque métier de définir ses processus.

Enfin, à long terme, nous étudierons comment détecter et corriger les dysfonctionnements en ingénierie des systèmes d'information, via la méta-modélisation et la modélisation. Les dysfonctionnements peuvent provenir de modèles de processus incohérents entre eux, de modèles incohérents par rapport au méta-modèle défini ou de méta-modèle mal défini par rapport aux modèles utilisés. Il faudra donc s'intéresser à l'intégration de modèles de processus existants et leur prise en compte dans les méta-modèles. Comme l'ont spécifié de nombreux industriels, les modèles de processus sont déjà souvent définis, il faut donc pouvoir utiliser la connaissance déjà établie dans les organisations afin de créer les méta-modèles correspondants. Nous étudierons enfin

comment les méta-modèles et modèles de processus peuvent faciliter la communication avec le client à travers les projets et comment les méta-modèles et modèles de processus peuvent être personnalisés et spécifiés pour des projets ou des domaines métier précis.

L'enjeu primordial de ces travaux est de guider les ingénieurs des méthodes dans la création de leurs méta-modèles et modèles de processus d'ingénierie de SI mais également dans l'exécution de ces processus et dans leur suivi, pour pouvoir guider, contrôler, mesurer et optimiser la production des SI.

BIBLIOGRAPHIE

APES, 2004
APES. *Les groupes focalisés*. Fiche méthodologique F4, Université de Lièges, Février 2004.

AS, 2004
Australian Standard. Standard Metamodel for Software Development Methodologies. AS 4651-2004, Australian Standard, 2004.

Atkinson, 1997
Colin Atkinson. Meta-Modeling for Distributed Object Environments. In *EDOC '97*: Proceedings of the 1st International Conference on Enterprise Distributed Object Computing, IEEE Computer Society, pp. 90-101, 1997.

Atkinson et Kühne, 2001
Colin Atkinson et Thomas Kühne. The Essence of Multilevel Metamodeling. In *UML '01*, Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools, Springer-Verlag, pp. 19-33, 2001.

Atkinson et Kühne, 2002
Colin Atkinson et Thomas Kühne. Rearchitecting the UML infrastructure. ACM, 12, pp. 290-321, 2002.

Beck, 1999
Kent Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional, Longman Publishing Co., Inc. Boston, Massachusetts, 1999.

Becker et al., 1999
Ulrike Becker-Kornstaedt, Dirk Hamann, Ralf Kempkens, Peter Rösch, Martin Verlage, Richard Webby, Jörg Zettel. Support for the Process Engineer: The Spearmint Approach to Software Process Definition and Process Guidance. In *CAiSE'99, Proceedings of the 11th International Conference on Advanced Information Systems Engineering*, Springer-Verlag, LNCS 1626, pp. 119-133, 1999.

Boehm, 1988
Barry W. Boehm. A Spiral Model of Software Development and Enhancement. *IEEE Computer*, 21 (5), pp. 61-72, 1988.

Booch, 1991
Grady Booch. *Object Oriented Analysis and Design with Application*, Benjamin-Cummings Publishing Co., Inc., Redwood City, California, 1991.

Brinkkemper et al., 1999
Sjaak Brinkkemper, Motoshi Saeki et Frank Harmsen. Meta-Modelling Based Assembly Techniques for Situational Method Engineering. *Information Systems*, 24 (3), pp. 209-228, 1999.

Cauvet, 2006
Corine Cauvet. Modélisation des processus d'ingénierie des systèmes d'information. *Encyclopédie de l'Informatique et des Systèmes d'Information*, Vuibert, pp 1412-1423, 2006.

CMMI, 2009
Software Engineering Institute, http://www.sei.cmu.edu/cmmi/, consulté en juillet 2009.

CNTL, 2009
Centre national de Ressources Textuelles et Lexicales. http://www.cnrtl.fr, consulté en juillet 2009.

Coad, 1992
Peter Coad. Object-Oriented Patterns. *Communication of ACM, 35 (9),* ACM Press, pp. 152-159, 1992.

Conklin et Begeman, 1988
Jeff Conklin et Michael L. Begeman. gIBIS: a hypertext tool for exploratory policy discussion. *ACM Transactions on Information Systems*, ACM Press, 6 (4), pp. 303-331, 1988.

Conte et al., 2001
Agnès Conte, Mounia Fredj, Jean-Pierre Giraudin, Dominique Rieu. P-Sigma : un formalisme pour une représentation unifiée de patrons. *Actes du XIXème Congrès INFORSID*, pp. 67-86, 2001.

Dahchour et al., 2002
Mohamed Dahchour, Alain Pirotte et Esteban Zimányi. Materialization and Its Metaclass Implementation. IEEE Transaction on Knowledge and Data Engineering, IEEE Educational Activities Department, 14 (5), pp. 1078-1094, 2002.

Ecore, 2009
Package org.eclipse.emf.ecore.
http ://download.eclipse.org/modeling/emf/emf/javadoc/2.5.0/org/eclipse/emf/ecore/package-summary.html, consulté en avril 2009.

EPF, 2008
Eclipse Process Framework. Open Up 1.5.0.1. http://epf.eclipse.org/wikis/openup/index.htm, octobre 2008.

EPF, 2009
Eclipse Process Framework, http://www.eclipse.org/epf/, mai 2009.

Etien, 2006
Anne Etien. *Ingénierie de l'alignement : Concepts, Modèles et Processus – La méthode ACEM pour l'alignement d'un système d'information aux processus d'entreprise*. Thèse, Université Paris 1 – Sorbonne, Mars 2006.

Finkelstein et al., 1990
Anthony Finkelstein, Jeff Kramer et Michael Goedicke. ViewPoint oriented software development. In *3rd International Workshop on Software Engineering and Its Applications*, pp. 374-384, 1990.

Gonzalez-Perez et Henderson-Sellers, 2006
Cesar Gonzalez-Perez et Brian Henderson-Sellers. A powertype-based metamodelling framework. *Software and System Modeling*, 5(1), pp. 72-90, 2006.

Gonzalez-Perez et Henderson-Sellers, 2008
Cesar Gonzalez-Perez et Brian Henderson-Sellers. A work product pool approach to methodology specification and enactment. *Journal of Systems and Software*, 81(8), pp. 1288-1305, 2008.

Grosz et al., 1997
Georges Grosz, Colette Rolland, Sylviane Schwer, Carine Souveyet, Véronique Plihon, Samira Si-Said, Camille Ben Achour et Christophe Gnaho. Modelling and Engineering the Requirements Engineering Process: An Overview of the NATURE Approach. *Requirements Engineering*, 2 (3), Springer-Verlag, London, pp. 115-131, 1997.

Gzara, 2000
Lilia Gzara. *Les Patterns Pour l''Ingénierie des Systèmes d'Information Produit*. Thèse de L'Institut National Polytechnique de Grenoble, Décembre 2000

Harel, 1987
David Harel. Statecharts: A Visual Formulation for Complex Systems. *Science of Computer Programming*, 8 (3), pp. 231-274, 1987.

Harmsen, 1997
Frank Harmsen. *Situational Method Engineering*. Thèse de l'Université de Twente, Pays-Bas, Moret Ernst & Young Management Consultants, Janvier 1997.

Hassine, 2005
Ibtissem Hassine. *Spécification et formalisation des démarches de développement à base de composants métier : la démarche Symphony*. Thèse de L'Institut National Polytechnique de Grenoble, Septembre 2005.

Henderson-Sellers et al., 2007
Brian Henderson-Sellers, Cesar Gonzalez-Perez et Jolyta Ralyté. Situational Method Engineering: Fragments or Chunks? CAiSE'07 Forum, pp. 89-92, 2007.

Henderson-Sellers et Edwards, 1990
Brian Henderson-Sellers et Julian M. Edwards. The Object-Oriented Systems Life Cycle. *Communication of the ACM*, 33(9), pp. 142-159, 1990.

Henderson-Sellers et Gonzalez-Perez, 2005
Brian Henderson-Sellers et Cesar Gonzalez-Perez. A comparison of four process metamodels and the creation of a new generic standard. *Information & Software Technology*, 47 (1), pp. 49-65, 2005.

Henderson-Sellers et Gonzalez-Perez, 2005b
Brian Henderson-Sellers et Cesar Gonzalez-Perez. Connecting Powertypes and Stereotypes. *Journal of Object Technology*, 4 (7), pp. 83-96, 2005.

Hug et al., 2007
Charlotte Hug, Agnès Front et Dominique Rieu. Ingénierie des processus: une approche à base de patrons. *Actes du XXVème Congrès INFORSID*, pp. 471-486, 2007.

Hug et al., 2008
Charlotte Hug, Agnès Front et Dominique Rieu. Ingénierie des processus : une approche à base de patrons. *Modèles, Formalismes et outils pour les systèmes d'information*, RSTI série Ingénierie des systèmes d'Information, 13, 4, 2008.

Hug et al., 2008b
Charlotte Hug, Agnès Front et Dominique Rieu. A Process Engineering Method Based on Ontology and Patterns. In *ICSOFT 2008*, pp. 29-36, 2008.

Hug et al., 2008c
Charlotte Hug, Agnès Front et Dominique Rieu. A Process Engineering Method based on a Process Domain Model and Patterns. In *MoDISE-EUS 2008*, pp. 26-137, 2008.

Hug et al., 2009
Charlotte Hug., Agnès Front, Dominique Rieu et Brian Henderson-Sellers. A method to build information systems engineering process metamodels. *Journal of Systems and Software*, 82 (10), pp 1730-1742, 2009.

Humphrey et Kellner, 1989
Watt S. Humphrey et Marc I. Kellner. Software Process Modeling: Principles of Entity Process Models. In *ICSE 1989*, IEEE Computer Society/ACM Press, pp. 331-342, 1989.

ISO, 1998
ISO 9241-11 :1998. Exigences ergonomiques pour travail de bureau avec terminaux à écrans de visualisation (TEV) – Partie 11 : lignes directrices relatives à l'utilisabilité, 1998.

ISO/IEC, 2007
ISO/IEC. Software Engineering — Metamodel for Development Methodologies. ISO/IEC 24744, 2007.

Jarke et al., 1992
Matthias Jarke, John P. Mylopoulos, Joachim W. Schmidt et Yannis Vassiliou. DAIDA: an environment for evolving information systems. ACM *Transactions on Information Systems*, 10 (1), ACM Press, pp. 1-50, 1992.

Johnson et Woolf, 1996
Ralph Johnson et Bobby Woolf. The Type Object Pattern. In *PLOP 1996*, Third Annual Conference on the Pattern Languages of Programs, 1996.

Jouault et Bézivin, 2006
Frédéric Jouault et Jean Bézivin. KM3: A DSL for Metamodel Specification. In Roberto Gorrieri et Heike Wehrheim (Eds.), *Formal Methods for Open Object-Based Distributed Systems*, 8th IFIP WG 6.1 International Conference, FMOODS 2006, Springer, LNCS 4037, pp. 171-185, 2006.

Kruchten, 2000
Philippe Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley, Longman Publishing, Co., Inc. Boston, Massachusetts, 2000.

Kumar et Welke, 1992
Kuldeep Kumar et Richard J. Welke. Methodology EngineeringR: a proposal for situation-specific methodology construction. W. W. Cotterman et J. A. Senn (Eds.), *Challenges and strategies for research in systems development*, John Wiley & Sons, pp. 257-269, 1992.

Kunz et Rittel, 1970
Werner Kunz et Horst W. J. Rittel. Issues as elements of information systems. Working Paper 131, Heidelberg-Berkeley, 1970.

Lefèvre, 2009
Michaël Lefèvre. Évolution fonctionnelle et graphique d'un logiciel de modélisation des processus d'ingénierie de systèmes d'information, Rapport de stage d'IUT, août 2009.

Martin, 1991
James Martin. *Rapid Application Development*. Macmillan Coll. Div., 1991.

Mc Dermid et Ripken, 1984
John McDermid et Knut Ripken. *Life cycle support in the ADA environment*. University Press, 1984.

Monjo, 2009
Robin Monjo. Réalisation d'un logiciel de modélisation des processus d'ingénierie de systèmes d'information. Rapport de stage d'IUT, juin 2009.

Objectiver, 2007
Objectiver. A KAOS tutorial. Respect-It, version 1.0, 2007.

Odell, 1994
James J. Odell. Power Types. *Journal of Object Oriented Programming*, 7 (3), pp. 8-12, 1994.

OMG, 2002
OMG. Software Process Engineering Metamodel Specification, Version 1.0. Formal/2002-11-14, Object Management Group, 2002.

OMG, 2005
OMG. Software Process Engineering Metamodel Specification, Version 1.1. Formal/2005-01-06, Object Management Group, 2005.

OMG, 2006
OMG. Meta Object Facility (MOF) Core Specification, Version 2.0. Formal/2006-01-01, Object Management Group, 2006.

OMG, 2007
OMG. Unified Modeling Language: Superstructure, Version 2.1.2, formal/2007-11-02, Object Management Group, 2007.

OMG, 2007b
OMG. Unified Modeling Language: Infrastructure, Version 2.1.2, formal/2007-11-04, Object Management Group, 2007.

OMG, 2007c
OMG. MOF 2.0/XMI Mapping, Version 2.1.1, formal/2007-12-01, Object Management Group, 2007.

OMG, 2008
OMG. Software & Systems Process Engineering Meta-Model Specification, Version 2.0. Formal/2008-04-01, Object Management Group, 2008.

OMG, 2009
OMG. Unified Modeling Language, Superstructure, Version 2.2, formal/2009-02-02, Object Management Group, 2009.

OOSPICE, 2002
Software Process Improvement and Capability dEtermination for Object Oriented/component based software development. http ://www.oospice.com/, Dernière mise à jour le 7 octobre 2002.

OPF, 2005
OPEN Process Framework Repository Organization. http ://www.opfro.org/. Dernière mise à jour le 16 décembre 2005.

Panet et Letouche, 1994
George Panet et Raymond Letouche. *Merise/2 Modèles et techniques Merise Avancés*. Les Éditions d'Organisation, 1994.

Plihon, 1996
Véronique Plihon. *Un environnement pour l'ingénierie des méthodes*. Université Paris 1 – Sorbonne, Janvier 1996.

Potts, 1989
Colin Potts. A generic model for representing design methods. In *ICSE '89*, ACM Press, pp. 217-226, 1989.

Potts et Bruns, 1998
Colin Potts, Glenn Bruns. Recording the Reasons for Design Decisions. In *ICSE'88*, IEEE Computer Society Press, pp. 418-427, 1988.

Prefuse, 2009
Prefuse. http ://prefuse.org/, consulté en juin 2009.

Ralyté, 1999
Jolyta Ralyté. Reusing Scenario Based Approaches in Requirement Engineering Methods: CREWS Method Base. CREWS Report Series 99 – 12, CRI, Université Paris1- Sorbonne, 1999.

Ralyté, 2001
Jolyta Ralyté. *Ingénierie des méthodes à base de composants*. Thèse, Université Paris 1 – Sorbonne, Janvier 2001.

Ralyté et al., 2007
Jolita Ralyté, Sjaak Brinkkemper et Brian Henderson-Sellers *(Eds.). Situational Method Engineering: Fundamentals and Experiences*, Proceedings of the IFIP WG 8.1 Working Conference, *Situational Method Engineering, Springer*, 2007.

RMC, 2009
Rational Method Composer, http://www-01.ibm.com/software/awdtools/rmc/, mai 2009.

Rolland, 1994
Colette Rolland. A Contextual Approach For The Requirements Engineering Process. In *SEKE'94: Proceedings of the 6th International Conference on Software Engineering and Knowledge Engineering*, pp.25-35, 1994.

Rolland, 1998
Colette Rolland. A Comprehensive View of Process Engineering. In *CaiSE'98: Proceedings of the 10th International Conference on Advanced Information Systems Engineering*, Springer-Verlag, London, 1998.

Rolland, 2005
Colette Rolland. « L'ingénierie des méthodes : une visite guidée », *e-TI*, 1, 2005.

Rolland et al., 1999
Colette Rolland, Naveen Prakash, et A. Benjamen. A multi-model view of process modelling. *Requirements Engineering*, 4(4), Springer-Verlag London, 1999.

Rolland et al., 2000
Colette Rolland, Selmin Nurcan et Georges Grosz. A decision-making pattern for guiding the enterprise knowledge development process. *Information & Software Technology*, 42 (5), pp. 313-331, 2000.

Rolland et Grosz, 1994
Colette Rolland et Georges Grosz. A General Framework for Describing the Requirements Engineering Process. In *ICSMC'94: Proceedings of the International Conference on Systems, Man, and Cybernetics*, IEEE Computer Society Press, 1, pp. 818-823, 1994.

Rose et al., 1991
Thomas Rose, Matthias Jarke, Michael Gocek, Carlos Maltzahn et Hans W. Nissen. A decision-based configuration process environment. *Software Engineering Journal*, 6 (5), Michael Faraday House, pp. 332-346, 1991.

Royce, 1970
Winston W. Royce. Managing the Development of Large Software Systems. *IEEE Wescon*, pp 1-9, 1970

Schwaber et Beedle, 2001
Ken Schwaber et Mike Beedle. *Agile Software Development with SCRUM*. Prentice Hall, Upper Saddle River, New Jersey, 2001.

Souveyet, 2006
Carine Souveyet. *Contributions à l'amélioration de l'ingénierie des SI*. Habilitation à diriger les Recherches, Paris1-Panthéon Sorbonne, Décembre 2006.

UMLJGraph, 2005
UMLJGraph. http://umljgraph.sourceforge.net/, dernière modification effectuée en août 2005.

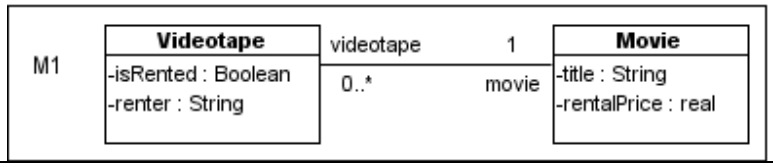
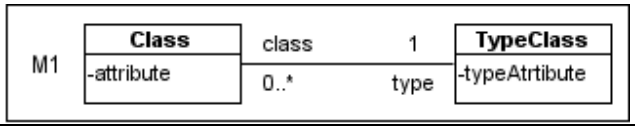
XP, 2006
Extreme Programming: a gentle introduction. http://www.extremeprogramming.org. Dernière mise à jour le 17 février 2006.

Zoukar, 2005
Iyad Zoukar, *MIBE : Méthode d'Ingénierie des Besoins pour l'implantation d'un progiciel de gestion intégré (ERP)*, Thèse, Université Paris 1 – Sorbonne, Avril 2005.

ANNEXES

Annexe A : Patron Type Object

Le patron Type Object a été introduit par (Johnson et Woolf, 1996). Son principe est identique au patron Item-Description.

Identifiant	Type Object
Classification	Produit
Contexte	Ne nécessite aucun patron pour être appliqué.
Problème	Ce patron est utilisé lorsque : – Les instances d'une classe doivent être groupées selon des attributs et/ou comportements communs. – Une classe doit être spécialisée pour chaque groupe, afin d'implémenter les attributs/comportements communs de ce groupe.
Force	Ce patron permet de gérer des objets et leurs propriétés qui sont communes à d'autres objets. Les propriétés communes des objets sont séparées des propriétés spécifiques de chaque objet.
Solution Modèle	Le patron est composé de deux classes concrètes *TypeClass* et *Class*. *Class* permet d'instancier des objets, et *TypeClass* permet d'instancier le type des objets. Chaque objet instance de *Class* pointe vers son type. 
Cas d'Application	Une classe *Movie* contient les informations concernant un film : titre et prix de location. Par exemple, un objet de type *Movie* peut être le film intitulé « Star Wars ». Une classe *Videotape* représente une cassette vidéo qui peut être louée (*isRented*) par un client (*renter*). Chaque objet de type *Videotape* fait référence à un objet de type *Movie*. 
Conséquence d'application	Les avantages du patron Type Object sont : – La possibilité d'ajouter dynamiquement autant d'objets instances de *TypeClass* que l'on veut, sans modifier la

	structure du programme. – La non-prolifération de sous-classes. – La séparation entre l'objet et son type est cachée au client. – Le changement dynamique du type d'un objet. – Un objet peut avoir plusieurs types (variante du patron sur la cardinalité du rôle type à 0..*) Les inconvénients du patron sont : – La difficulté de compréhension : la séparation objet et type d'objet peut porter à confusion. – La complexité d'implémentation : les objets de type *TypeClass* implémentent les méthodes d'une seule manière et les objets de types *Class* doivent gérer leurs propres comportements. – La gestion des références entre un objet et son type doit être faite par l'application.
Alternative	Item-Description (Coad, 1992)

Annexe B : Attributs des classes issues des concepts secondaires

Nous présentons ici les attributs des classes issues des concepts secondaires du graphe conceptuel.

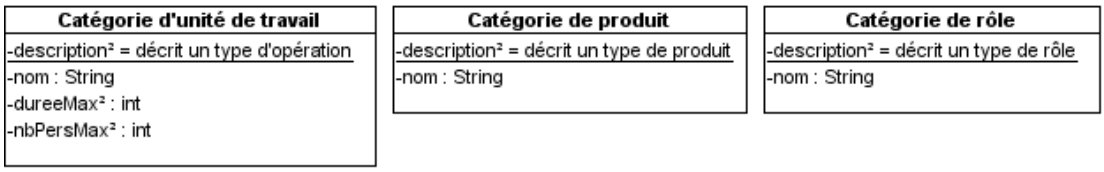


Figure 4. Attributs des classes Catégorie d'unité de travail, Catégorie de produit et catégorie de Rôle.

Une *Catégorie d'unité de travail* a une description, un nom, une durée maximale et un nombre de personnes maximal sur un projet donné. Par exemple, une tâche dans un projet particulier doit durer au maximum 10 jours et ne peut avoir plus de 3 rôles intervenants. *Catégorie de produit* et *Catégorie de rôle* ont une description et un nom, par exemple « Document », « Texte », « Diagramme » pour *Catégorie de produit* et « Équipe » ou « Organisation » pour *Catégorie de rôle*.

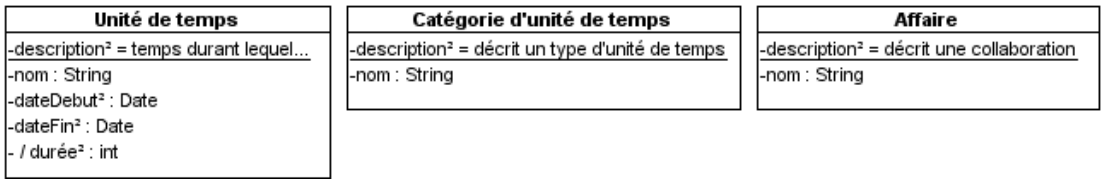


Figure 5.Attributs des classes Unité de temps, Catégorie d'unité de temps et Affaire.

Une *Unité de temps* a une description, un nom, une date de début, une date de fin et une durée qui peut être calculée à partir des dates. Par exemple, l'exécution d'Inception, une unité de temps définie dans le modèle de processus du RUP, commence le 15 juin et qui termine le 19 septembre, a une durée de 69 jours ouvrés. Une *Catégorie d'unité de temps* a une description et un nom, comme « Phase » ou « Cycle de vie ». Une *Affaire* a une description et nom, qui peut être le nom d'un projet ou d'un contrat.

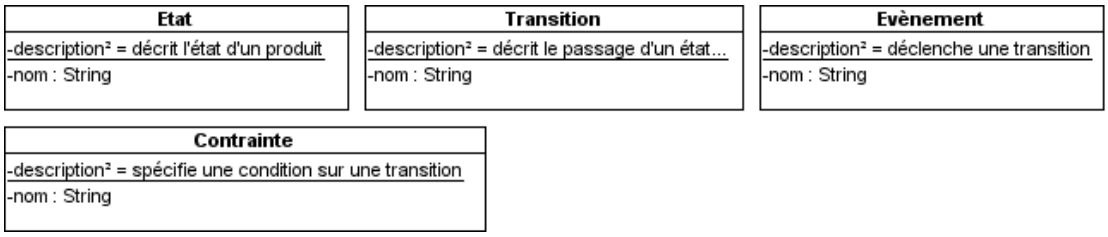


Figure 6.Attributs des classes issues du patron Etat-Transition.

Un *État* a une description et un nom, comme « Validé » ou « En réalisation ». Une *Transition* a également une description et un nom. Un *Évènement* a une description et un

nom, « Après 5 ans » ou « À la signature » par exemple. Une *Contrainte* a une description et un nom qui correspond à la contrainte sur la transition.

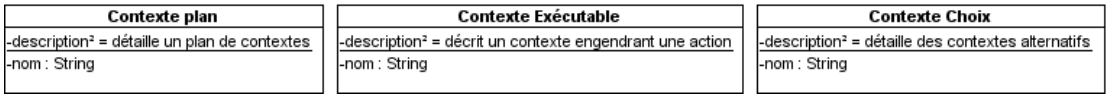


Figure 7.Attributs des classes issues du patron NATURE.

Contexte plan, *Contexte exécutable* et *Contexte choix* ont tous une description et un nom qui permet de décrire le contexte.

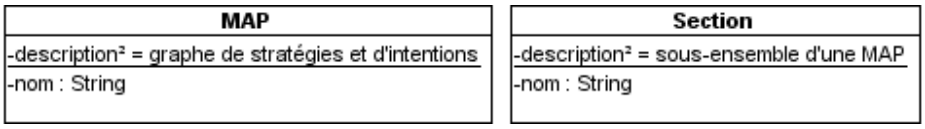


Figure 8. Attributs des classes issues du patron MAP.

MAP et *Section* ont toutes deux une description et un nom. Le nom d'une MAP est significatif comme « Écrire un scénario », le nom d'une section peut être un code comme « S2 ».

Annexe C : Expérimentation du focus group

Nous présentons ici les questionnaires distribués aux sujets du focus group, le dictionnaire des concepts et des relations à partir desquels les sujets ont dû construire leur propre graphe conceptuel, ainsi que le cas d'étude sur lequel les sujets ont dû créer le méta-modèle de processus et l'instancier.

1. Questionnaire sur les pratiques du focus group

La dernière fois que vous avez développé un logiciel ou un système, quelles étaient les grandes étapes par lesquelles vous êtes passé ?
Avez-vous l'habitude d'utiliser des méthodes de conception de systèmes d'information ou de développement logiciel ? Pourquoi en utilisez-vous ? Lesquelles ? Pourquoi, n'en utilisez-vous pas ?
Quand vous utilisez une méthode de conception, vous... 1. vous l'appliquez à la lettre 2. vous l'adaptez à vos besoins 3. vous en utilisez plusieurs en même temps
Est-ce que vous avez l'habitude de présenter, même partiellement ; vos méthodes à d'autres personnes ? 1. Très souvent 2. Souvent 3. Parfois 4. Jamais
Est-ce que vous avez l'habitude de décrire, même partiellement, vos propres méthodes ? 1. Très souvent 2. Souvent 3. Parfois 4. Jamais
Quels sont les outils et langages que vous utilisez pour décrire vos méthodes ?
Comment mettez-vous en place le suivi de votre méthode ?
Dans quels contextes vous arrive-t-il de présenter ou décrire vos méthodes ?
Vous utilisez une méthode de conception… 1. pour toutes les phases 2. pour les phases importantes 3. pour les phases de collaboration

2. Questionnaire focus group usage et utilisabilité

Globalement qu'avez-vous pensé de cet exercice et de sa réalisation ? Qu'est-ce qui a été le plus facile ? Qu'est-ce qui a été le moins facile ?
Est-ce que le passage au méta-modèle était indispensable pour réaliser le modèle attendu ? 1. oui, tout à fait 2. oui plutôt 3. non plutôt pas 4. non pas du tout Pourquoi ?
Auriez-vous préféré modifier le méta-modèle directement ? Pourquoi ?
L'itération méta-modèle/modèle vous a-t-elle permis de trouver de nouvelles idées ?
Par rapport à l'exercice que vous venez de réaliser, vous êtes 1. Très satisfait 2. Plutôt satisfait 3. Plutôt pas satisfait 4. Pas du tout satisfait
Est-ce que cette méthode a été satisfaisante par rapport aux résultats obtenus ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Pouvez-vous préciser en quoi elle a été satisfaisante ou insatisfaisante ?
Est-ce que vous pensez que cette méthode peut vous faire gagner du temps ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Pourquoi ?
Est-ce que vous pensez que cette méthode peut vous éviter des erreurs ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Pourquoi ?
À votre avis, ce graphe conceptuel est ... 1. Tout à fait complet 2. plutôt complet

3. plutôt incomplet 4. tout à fait incomplet Si pour vous ce graphe n'est pas complet, que lui manque-t-il ? Si vous pensez qu'il manque des nœuds, lesquels rajouteriez-vous ?
Faites-vous des différences entre les concepts proposés ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non plutôt pas 4. Non pas du tout Pouvez-vous préciser ?
Est-ce que vous pensez que cette méthode nécessite une phase d'apprentissage ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Vous pensez que la phase d'appropriation de cette méthode demande... 1. quelques minutes 2. quelques heures 3. quelques jours 4. quelques semaines 5. plus longtemps
Seriez-vous prêt à utiliser cette méthode dans votre contexte professionnel ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Pourquoi ?
Le graphe vous aiderait-il à formaliser un ou des problèmes ? Si oui le(s)quel(s) ?
Conseilleriez-vous l'utilisation de cette méthode ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout
Seriez-vous prêt à décrire le fonctionnement de la méthode à une tierce personne ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout
Est-ce que cette méthode est agréable à utiliser ? 1. Oui, tout à fait

2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout
Est-ce que vous auriez eu besoin de conseil en cours d'utilisation ? 1. Oui, tout à fait 2. Oui, plutôt 3. Non, plutôt pas 4. Non, pas du tout Pourquoi ?

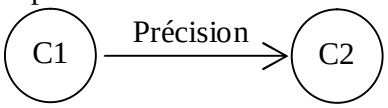
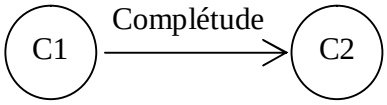
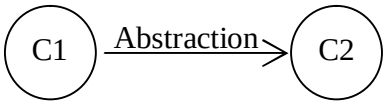
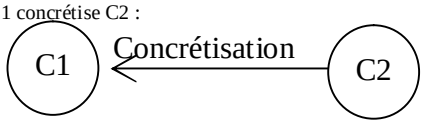
3. Dictionnaire des concepts et des relations

Dictionnaire des concepts

Concepts	Définitions	Synonymes	Exemples
Intention	Objectif d'un processus d'ISI.	Objectif, but	
Composition d'intentions	Décomposition d'intentions, c'est-à-dire des sous-objectifs d'un processus d'ISI.	Sous-objectif, sous-but	
Stratégie	Manière dont une intention est atteinte.	Tactique, approche	
Situation	Situation du projet à un instant donné du processus d'ISI.		
Map	Architecture de plusieurs stratégies et d'intentions.		
Contexte	Intention et situation à un instant donné du processus d'ISI.		
Nature	Architecture de contextes.		
Problème	Problème rencontré durant le processus d'ISI.	Question, difficulté, dilemme	
Alternative	Réponse à un problème rencontré durant le processus d'ISI.	Choix, possibilité, éventualité, option	
Argument	Argument pour supporter ou rejeter une alternative.	Preuve, raison	
Unité de travail	Action exécutée durant le processus d'ISI.		Définir le système, sélectionner les besoins des utilisateurs, implémenter les composants, analyse structurelle, analyse dynamique
Rôle	Quelqu'un ou quelque chose qui réalise une action durant le processus d'ISI ou qui est responsable d'un produit.	Acteur	Développeur, analyste, système informatique
Condition	Condition sur une action.	Contrainte	Pré-condition, post-condition
Produit	Quelque chose qui est produit, utilisé ou modifié durant le processus d'ISI.		Modèle de conception, cahier des charges, modèle d'analyse, user stories
Composition de produits	Décomposition d'un produit en sous-produits.		Le modèle d'analyse est composé de la description des objets métier et de la cartographie des objets métier.
Catégorie de produit	Ensemble de produits ayant des caractéristiques communes.		Document, texte, modèle, diagramme, logiciel, image
État	Différents états d'un produit.		Validé, en cours de rédaction, en diffusion
Transition	Transitions entre les différents états d'un produit.		
Unité de	Fait que deux unités de travail		

travail parallèle	peuvent s'exécuter en parallèle.		
Unité de travail séquence	Fait que deux unités de travail peuvent s'exécuter en séquence.		
Composition d'unités de travail	Décomposition d'unités de travail en unités plus petites.		Analyse est composée de l'analyse structurelle et de l'analyse dynamique
Catégorie d'unité de travail	Ensemble d'unités de travail ayant des caractéristiques communes.		Cycle de vie, Phase, activité, tâche.

Définitions des relations

Précision	Permet de préciser un concept. C2 précise C1 : 
Complétude	Permet d'élargir le modèle. C2 complète C1 : 
Abstraction	Permet d'abstraire un concept. C2 abstrait C1 : 
Concrétisation	Permet de concrétiser un concept. Inverse d'abstraction. C1 concrétise C2 : 

4. Cas d'étude du focus group

Sujet

Le service des Systèmes d'Information (SI) d'un hôpital gère un système d'information composé d'une quarantaine d'applications destinées aux services de l'hôpital. Ces applications sont utilisées par des assistants médicaux, des médecins et du personnel administratif.

Le responsable du service SI souhaite modéliser les processus d'ingénierie de SI utilisés dans l'hôpital pour pouvoir ensuite améliorer leur cohérence et leur performance.

Un des chefs de projet du service est chargé de réaliser l'analyse des processus d'ingénierie de SI existants. Ces processus sont analysés à deux niveaux :

- auprès des chefs de service de soin et supports ;
- auprès du service SI.

Chefs de services de soin et supports

Ces chefs de service s'intéressent aux objectifs d'un processus d'ingénierie de systèmes d'information. Par exemple, dans le cas de la mise en place d'une nouvelle fonctionnalité, les chefs de service veulent connaître :

- l'impact de la nouvelle fonctionnalité du SI sur leur service ;
- cet objectif peut être détaillé en plusieurs points dont :
- connaître l'impact du changement sur l'organisation de leur service.

Service SI

Le service SI s'intéresse plus à l'aspect concret des processus d'ingénierie de SI, c'est-à-dire aux différentes étapes à suivre pour réaliser le SI et aux produits réalisés.

Par exemple, l'activité de pré-étude fonctionnelle est composée de plusieurs tâches dont :

- la rédaction du cahier des charges simplifié qui permet de produire un cahier des charges simplifié qui est un document textuel.
- L'ensemble des documents produits par l'activité de pré-étude fonctionnelle constitue un document appelé dossier fonctionnel.

Consigne

Vous êtes chargé de la modélisation des processus d'ingénierie de SI évoqués.

- En utilisant le graphe conceptuel, le dictionnaire de concepts et des relations, sélectionnez les concepts qui vous semblent les plus pertinents pour modéliser les processus décrits. Tous les concepts sélectionnés doivent être reliés.
- Lorsque vous pensez avoir sélectionné tous les concepts nécessaires, construisez une légende pour chaque élément du méta-modèle fourni par l'expert : associez un formalisme à chaque classe et association. Vous pourrez vous inspirer de la

fiche de formalisme. Attention : tous les éléments du méta-modèle (classes et associations) doivent avoir leur correspondance dans la légende.

- Lorsque la légende est terminée, utilisez le méta-modèle donné et la légende pour instancier la partie du problème traité dans votre modèle de processus.
- Si un élément manque pour l'instanciation, vous devez revenir au graphe conceptuel, prendre un stylo de couleur différente et sélectionner le concept manquant. Revenez ensuite au modèle de processus et continuez l'instanciation.

Annexe D : Expérimentation des industriels

Nous présentons ici les questions posées aux industriels durant l'entretien ainsi que deux captures d'écran de la démonstration du prototype.

1. Questionnaire industriels

Habitudes de travail

Tout d'abord, pouvez-vous me décrire rapidement les activités de votre entreprise et de votre équipe ?

Si la personne travaille seule : quand vous développez un logiciel ou un système quelles sont les étapes par lesquelles vous passez ?

Quelles sont les méthodes de conception que vous mettez en place dans votre équipe ?

Avez-vous l'habitude d'utiliser des méthodes de conception de systèmes d'information ou de développement logiciel ? Pourquoi en utilisez-vous ? Lesquelles ? Pourquoi, n'en utilisez-vous pas ?

Dans quels contextes, pour quels types de projets utilisez-vous des méthodes de conception ?

Quels sont les outils et langages que vous utilisez pour décrire vos méthodes ?

Comment vos collaborateurs les mettent en œuvre ?

Comment intégrez-vous de nouvelles méthodes ? Lesquelles ? Quelles sont vos sources d'information ?

Comment mettez-vous en place le suivi de votre méthode ?

Quand vous utilisez une méthode de conception, vous l'appliquez à la lettre, l'adaptez à vos besoins, en utilisez plusieurs en même temps ?

Vous utilisez une méthode de conception pour toutes les phases, pour les phases importantes, pour les phases de collaboration

Est-ce que vous avez l'habitude de présenter, même partiellement ; vos méthodes à d'autres personnes ?

Est-ce que vous avez l'habitude de décrire, même partiellement, vos propres méthodes ?

Présentation de la méta-modélisation

Qu'en pensez-vous ? Quelles peuvent être les avantages de la méta-modélisation par rapport à vos habitudes de travail ? Les inconvénients ?

Avant ma présentation, est-ce que vous aviez entendu parler de la méta-modélisation ? Si oui, l'utilisez-vous et dans quels contextes ?

Présentation de la méthode

Plus précisément, que pensez-vous de l'approche avec les différents points de vue ?

Que pensez-vous du niveau d'abstraction intentionnel ?

Que pensez-vous du niveau d'abstraction opérationnel ?

Que pensez-vous des concepts ?

Que pensez-vous des relations ?

Présentation de l'outil

Présentation du graphe conceptuel

Qu'en pensez-vous ?

Est-ce que cet outil vous aide à comprendre la méta-modélisation ? Pourquoi ?

Est-ce que vous l'utiliseriez ? Pour quel type d'application l'utiliseriez-vous ?

Présentation du méta-modèle de processus en UML

Qu'en pensez-vous ?

Par rapport au graphe précédent ? Que pensez-vous des niveaux d'abstraction sur le méta modèle ?

Quel schéma préférez-vous ? Pourquoi ?

Présentation du modèle de processus instancié

L'ensemble des concepts manipulés vous parait-il complet ?

Pensez-vous que cette méthode peut vous aider dans la définition de vos modèles de processus ? Pourquoi ?

Pensez-vous que l'outil peut vous éviter de faire des erreurs ? Pourquoi ?

Le formalisme proposé vous parait-il utile ? Intéressant ?

Dans la méthode proposée, le méta modèle est construit avant le modèle, qu'en pensez-vous ?

Est-ce une méthode habituelle pour vous ?

(Le fait d'avoir d'abord construit le méta-modèle de processus vous aide-t-il à mieux définir les modèles ?)

Pensez-vous que l'utilisation de la méthode nécessite une phase d'apprentissage ?

Pensez-vous que la phase d'appropriation de la méthode demande : quelques minutes, quelques heures, quelques jours, quelques semaines, plus longtemps

Seriez-vous prêt à utiliser la méthode dans votre environnement professionnel ?

Conseilleriez-vous l'utilisation de cette méthode ?

Seriez-vous prêt à décrire le fonctionnement de la méthode à une tierce personne ?

Pensez-vous que l'outil présenté est en adéquation avec la méthode ?

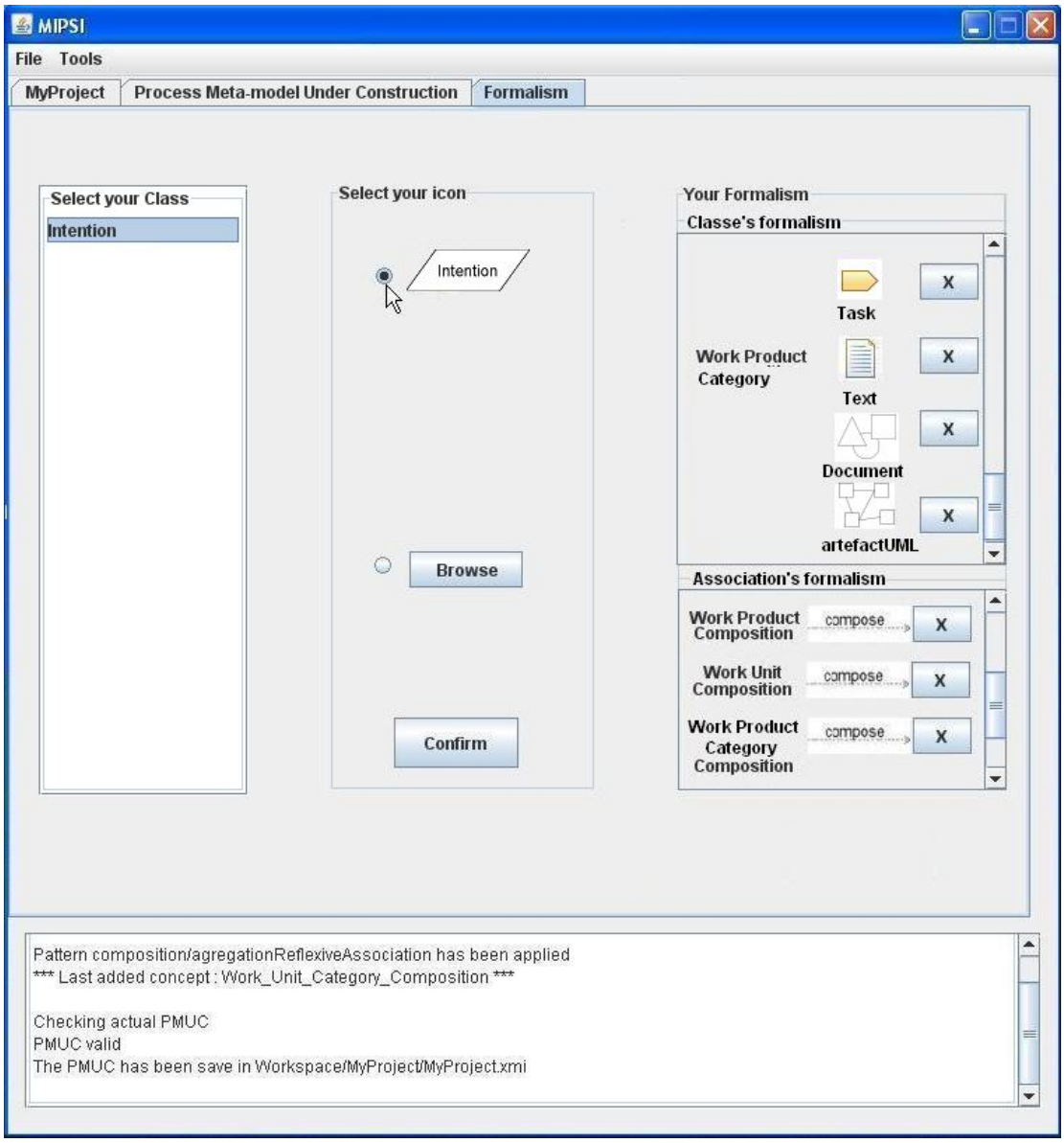
Est-ce que cet outil est agréable à utiliser ?

Pensez-vous avoir besoin d'aide durant l'utilisation de l'outil ?

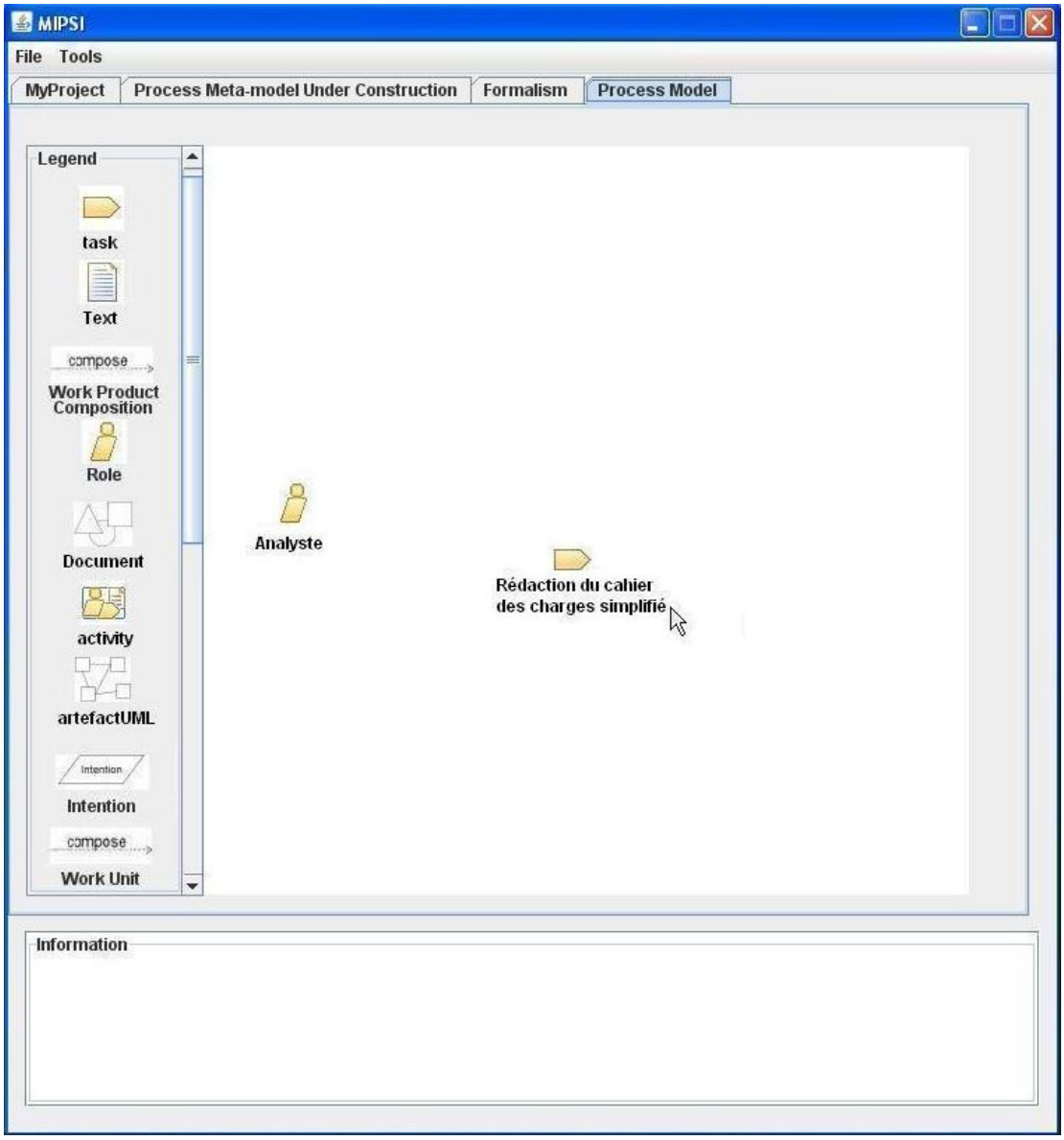
L'interface de l'outil est-elle claire et facile à comprendre ?

La méthode et l'outil vous aideraient-ils à formaliser un ou des problèmes ? Lesquels ?

2. Captures d'écran de la démonstration de l'outil



Cette capture d'écran montre l'interface de sélection du formalisme par rapport aux classes du méta-modèle en cours de construction. Un symbole est choisi pour toutes les classes et associations du méta-modèle.



Cette capture d'écran montre la construction d'un modèle de processus à partir du formalisme choisi.

Résumé

L'ingénierie des systèmes d'information propose de nombreuses méthodes et modèles produit et processus pour concevoir efficacement des systèmes d'information. Cependant, les modèles de processus définis ne correspondent pas forcément aux contraintes et spécificités des organisations.

De plus, différents points de vue (activité, produit, décision, contexte, stratégie) sont pris en compte dans la modélisation et la méta-modélisation des processus d'ingénierie de systèmes d'information, mais ceux-ci sont définis indépendamment les uns des autres. De par leur nombre et leur manque de flexibilité, les modèles de processus existants sont difficilement adaptables par les ingénieurs des méthodes pour prendre en compte les spécificités des organisations.

Cette thèse propose une méthode permettant aux ingénieurs des méthodes de définir leurs propres méta-modèles de processus en tenant compte des contraintes et spécificités des organisations. Cette méthode est guidée par un graphe conceptuel comprenant l'ensemble des concepts pour la méta-modélisation des processus. Les méta-modèles créés prennent en compte les différents points de vue. De plus, la construction des méta-modèles de processus est basée sur l'imitation de patrons génériques et de patrons de domaine.

La méthode a été outillée et expérimentée auprès d'experts en ingénierie des systèmes d'information.

Mots-clefs

Ingénierie des systèmes d'information, processus d'ingénierie, méta-modélisation, patron

---

Title

Method, models and tool for information systems engineering process metamodelling

Abstract

Information systems engineering offers many methods, product and process models to efficiently carry out information systems. However, the defined process models do not necessarily meet the organizations constraints and specificities.

Moreover, different points of view (activity, product, decision, context, strategy) are considered in information systems engineering process modelling and metamodelling but they are defined independently one from another. Due to their number and lack of flexibility, the existing process models are hardly adaptable by method engineers to consider the organizations specificities.

This thesis proposes a method allowing method engineers to define their own process metamodels, taking into account their organizations constraints and specificities. It is guided by a conceptual graph including the set of process metamodelling concepts. The created metamodels include the different points of view. Furthermore, the construction of process metamodels is based on the reuse of generic and domain patterns. The method has been implemented and experimented with information systems engineering experts.

Keywords

Information systems engineering, process engineering, metamodelling, pattern