HAL
open science

Ingénierie des processus. Une approche à base de patrons

Charlotte Hug, Agnes Front, Dominique Rieu

To cite this version:

Charlotte Hug, Agnes Front, Dominique Rieu. Ingénierie des processus. Une approche à base de patrons. Revue des Sciences et Technologies de l'Information - Série ISI : Ingénierie des Systèmes d'Information, 2008, 13 (4), pp.11. 10.3166/isi.13.4.11-34 . hal-00450713

HAL Id: hal-00450713

https://hal.science/hal-00450713

Submitted on 19 Feb 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Ingénierie des processus : une approche à base de patrons

Charlotte Hug — Agnès Front — Dominique Rieu

LIG, équipe SIGMA – Université de Grenoble
220, rue de la Chimie 38400 Saint Martin d'Hères
{Prenom.Nom}@imag.fr

RESUME. *Il existe de nombreux modèles de processus dédiés à l'ingénierie des systèmes d'information. Ils sont basés sur des concepts (activité, produit, décision, contexte, stratégie), et donc sur des méta-modèles de processus différents mais complémentaires. En ce sens, ces méta-modèles ne permettent qu'une vision partielle des processus d'ingénierie. Nous proposons dans cet article un patron de conception qui permet de modéliser la plupart des concepts de méta-modèles de processus. Ce patron constitue une première étape vers la construction de méta-modèles de processus unifié, adaptés et multi-vues.*

ABSTRACT. *There are many process models for information systems engineering, based on different concepts (activity, product, decision, context, strategy) and then on different process meta-models. These meta-models allow a partial vision of the process engineering. This paper presents a design pattern to model concepts of various process meta-models. This pattern is a first step towards the construction of unified, fitted and multi-views process meta-models.*

MOTS-CLÉS : *modèle de processus, ingénierie des processus, patron, méta-modélisation.*

KEYWORDS: *process model, process engineering, pattern, meta-modelling.*

1. Introduction

De nombreuses définitions d'une méthode d'ingénierie de systèmes d'information existent. Pour Harmsen (Harmsen, 1997), il s'agit d'une « collection de procédures, de techniques, de descriptions de produit et d'outils pour le support effectif, efficace et consistant du processus d'ingénierie d'un système d'information ». Pour Booch (Booch, 1991), une méthode est « un processus rigoureux permettant de générer un ensemble de modèles qui décrit divers aspects d'un logiciel en cours de construction en utilisant une certaine notation bien définie ». Toutes ces définitions mettent en évidence la nécessité d'offrir des langages permettant de modéliser des produits et une démarche correspondante, c'est-à-dire, un méta-modèle de produit et un modèle de processus. Les modèles de processus proposent des démarches de développement de systèmes logiciels. Ils prescrivent une démarche méthodologique pour atteindre la cible que constituent les produits (Rolland, 2005). Dans cet article, nous nous intéresserons aux processus et particulièrement à la modélisation des processus d'ingénierie de systèmes d'information.

Un processus d'ingénierie de systèmes d'information est long et lourd à gérer. (Cauvet, 2006) présente plusieurs arguments pour la modélisation des processus d'ingénierie. Le processus doit être modélisé afin de prévoir ses différentes étapes, pour guider les divers acteurs qui interviennent tout au long du cycle d'ingénierie. La modélisation d'un processus permet également de suivre sa progression en temps réel, c'est-à-dire son exécution. Enfin, modéliser les processus permet d'historiser ce qui a été fait, pour une réutilisation des parties les plus génériques ou pour le contrôle et l'amélioration du processus. Les organisations souhaitant modéliser leurs modèles de processus le plus précisément possible et sur toutes les étapes du cycle de vie d'ingénierie, sont confrontées à différents problèmes. La définition d'un modèle de processus doit être dirigée par des concepts, des règles et des relations, un méta-modèle est donc nécessaire pour l'instanciation de modèles. Il existe de nombreux méta-modèles de processus permettant de représenter des points de vue différents du processus. Cependant, chaque méta-modèle de processus décrit un unique point de vue du processus et il n'existe aucune correspondance entre ces points de vue. Ils sont pourtant liés puisqu'ils représentent le même processus sous des points de vue différents. Ils sont, de plus, souvent trop précis ou spécifiques pour être adaptés au vocabulaire d'une organisation. La majorité d'entre eux ne propose pas de mécanismes d'extension. Comment choisir, alors, le bon méta-modèle ? Lequel choisir si l'on veut représenter plusieurs points de vue ? Comment gérer la complémentarité de ces méta-modèles ? Il est essentiel de permettre aux organisations de créer ou d'adapter des modèles de processus en fonction des contraintes et caractéristiques de chaque projet.

Pour permettre une vision globale et complète du processus d'ingénierie des systèmes d'information, c'est-à-dire, pour représenter un processus d'ingénierie avec des points de vues multiples, leur correspondance sous la forme d'un seul modèle (et

non différents modèles fragmentés), il est nécessaire de disposer de méta-modèles de processus :

- unifiés : un seul méta-modèle de processus permet de représenter tous les points de vue et leur correspondance,

- multi-vues : le méta-modèle de processus contient tous les points de vue,

- adaptés : le méta-modèle répond aux besoins, contraintes des organisations et projets.

L'objectif de cet article est de proposer un patron de conception pour la création de méta-modèles de processus unifiés, adaptés et multi-vues.

D'autre part, dans les modèles et méta-modèles de processus, certains concepts ne peuvent pas être modélisés par une seule classe car des informations les concernant manqueraient. Ces informations ne peuvent cependant pas être stockées dans la même classe car elles sont partagées par d'autres concepts. Dans un souci de factorisation de l'information, il faudrait donc permettre la modélisation de concepts et de catégories de concepts dans les modèles et méta-modèles de processus. De plus, les informations modélisées, ou propriétés, ont besoin d'être instanciées à différents niveaux : certaines propriétés sont identiques à certains concepts au niveau de la modélisation des modèles de processus, d'autres ne concernent qu'un seul concept concret au niveau de l'exécution du processus. L'instanciation de propriétés devrait donc se faire sur plusieurs niveaux de modélisation. C'est aussi un des buts de notre proposition.

Cet article est organisé comme suit : la section 2 définit les différents niveaux de modélisation. La section 3 discute les modèles et méta-modèles de processus existants et expose la problématique. La section 4 présente le patron « Item-Description » de (Coad, 1992) et l'instanciation profonde (Atkinson et al., 2001) qui servent de base à notre proposition de patron pour la méta-modélisation des processus. La section 5 montre comment le patron proposé facilite la définition de méta-modèles de processus. Enfin, la section 6 conclut l'article et présente les perspectives de notre travail.

2. Les niveaux de modélisation

La double facette produit/processus replacée dans l'architecture de l'OMG (OMG-MOF, 2005) permet de définir une méthode comme étant composée :

– d'un méta-modèle de produit : par exemple, UML (OMG, 2006).

– d'un modèle de processus : par exemple, le modèle de processus de la méthode RUP (Kruchten, 2000).

Dans cet article, nous nous concentrons sur les processus. Un exemple d'architecture à quatre niveaux définie par l'OMG pour les processus est détaillé ci-

après (cf. Figure 1). Le méta-méta-modèle permet de définir un méta-modèle de processus. «Activité», « Acteur » et « Ressource » sont des méta-classes[1], instances de la méta-méta-classe[2] « Classifier » définie dans le MOF (OMG-MOF, 2005). Ces méta-classes appartiennent ici au méta-modèle de processus de l'OMG, SPEM (OMG, 2005). Le modèle instancie le méta-modèle pour présenter un processus : l'analyse des processus métier, réalisée par un analyste, produit un modèle de processus métier. « Analyse des processus métier » et «Description informelle du processus métier» sont des instances de la méta-classe «Activité». Enfin, le niveau instance représente le processus réel, c'est-à-dire l'exécution du modèle de processus, dans le cadre du développement d'un système d'information particulier.

SPEM se focalise sur un point de vue des processus particulier nommé activité. Les modèles de processus instanciés à partir de SPEM sont donc aussi orientés activité. Il existe d'autres types de modèles de processus que nous présentons dans la section suivante.

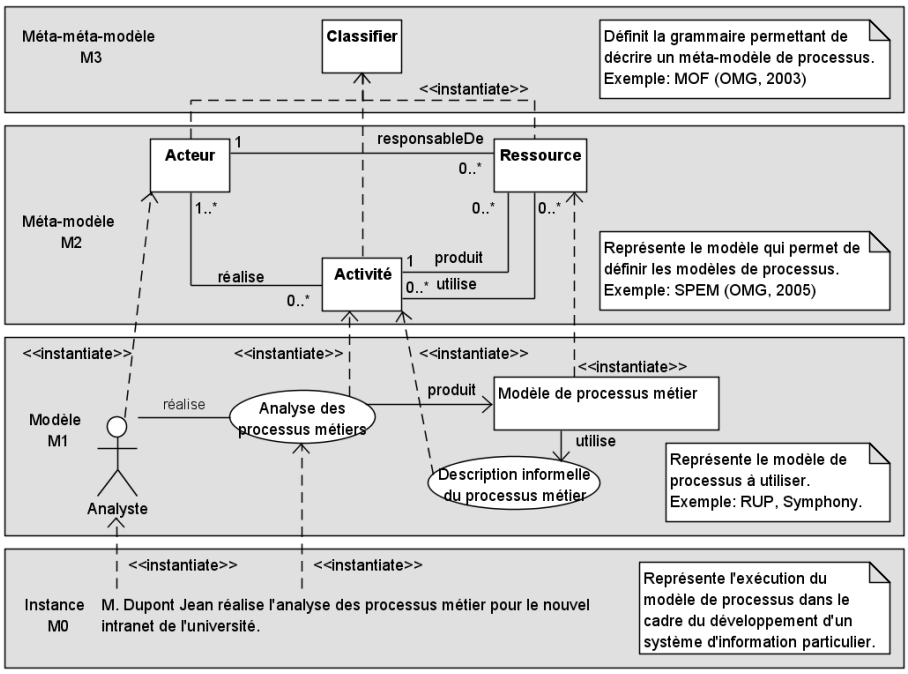


Figure 1. *Exemple d'architecture à quatre niveaux pour les processus.*

3. Les modèles et méta-modèles de processus existants

D'après (Dowson, 1987), un modèle de processus peut se focaliser sur :

[1] M2 – level class (OMG-MOF, 2005).
[2] M3 – level class (OMG-MOF, 2005).

– les activités,

– les objets qui résultent de ces activités (les produits),

– les décisions qui engendrent le traitement des activités et des produits.

D'autres types de modèles de processus ont été définis :

– les modèles de processus orientés contexte (Rolland *et al.*, 1995),

– les modèles de processus orientés stratégie (Rolland *et al.*, 1999).

Nous représentons dans les sections 3.1 à 3.5 ces différentes approches sous la forme de diagrammes d'objets où :

– les classes des modèles de processus (niveau M1) sont représentées par des objets,

– les classes des méta-modèles de processus (niveau M2) sont représentées par des classes instanciées.

3.1. *Modèles de processus orientés activité*

Les modèles de processus orientés activité représentent les activités et leur ordonnancement pour la réalisation d'un produit. Des exemples de tels modèles sont RUP (Kruchten, 2000), 2TUP (Roques, 2000) et Symphony (Hassine, 2005).

La Figure 2 présente le modèle de processus Symphony dans le formalisme préconisé par SPEM (OMG, 2005). Dans cet exemple, l'analyste réalise trois activités dont l'établissement des cartographies des objets métier qui produit une cartographie des objets métier qui est un diagramme UML.

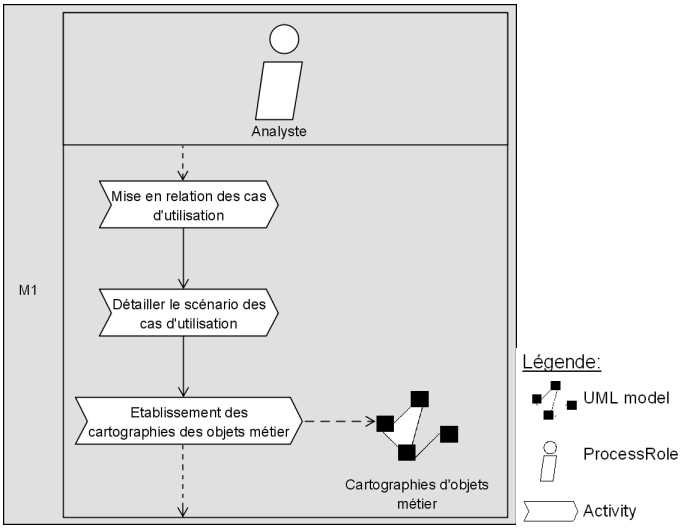


Figure 2. *Extrait du modèle de processus orienté activité Symphony.*

De nombreux méta-modèles permettent de définir des modèles de processus orientés activité : SPEM (OMG, 2005), l'OPEN Process Framework (OPF, 2005), OOSPICE (OOSPICE, 2002), SMSDM (Australian Standard, 2004)... La Figure 3 présente un extrait du méta-modèle SPEM. Une activité est réalisée par un ProcessRole qui est responsable de WorkProducts. La classe ActivityParameter permet de préciser quels sont les paramètres (WorkProduct) d'une activité.

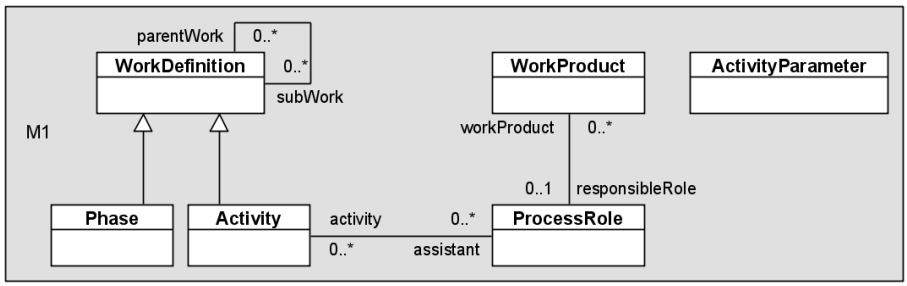


Figure 3. *Extrait du méta-modèle SPEM.*

Le modèle de processus Symphony est une instance du méta-modèle de processus SPEM. Par exemple (cf. Figure 4), « Spécification organisationnelle des besoins », est une phase de Symphony. Elle est instance de la classe « Phase » telle que SPEM la définit. De même, «Détailler le scénario des cas d'utilisation», qui est une activité de Symphony est instance de «Activity» définie dans SPEM.

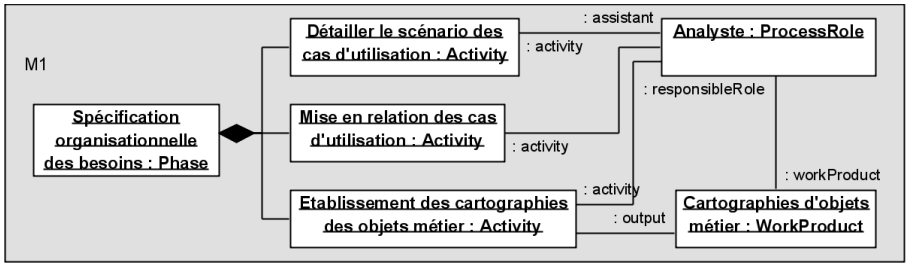


Figure 4. *Extrait du modèle de processus orienté activité Symphony comme instance de SPEM.*

3.2. *Modèles de processus 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).

La Figure 5 montre, par exemple, la transition permettant de passer d'un produit en cours de développement à un produit en cours de validation.

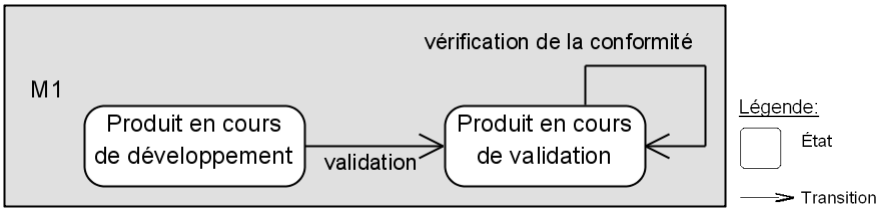


Figure 5. *Extrait d'un modèle de processus orienté produit représenté avec le formalisme des diagrammes états-transitions d'UML.*

Il existe différents méta-modèles de processus orientés produit comme le méta-modèle du modèle Statecharts de Harel (Harel, 1987), le méta-modèle State-Transition (un template ViewPoint présenté dans (Finkelstein et al., 1990)), le méta-modèle de processus Entity (Humphrey et al., 1989), et le méta-modèle du state machines d'UML (OMG, 2006). Un méta-modèle de processus orienté produit définit comment un produit passe d'un état à un autre, c'est-à-dire par quelle transition. La Figure 6 présente un extrait du méta-modèle State-Transition.

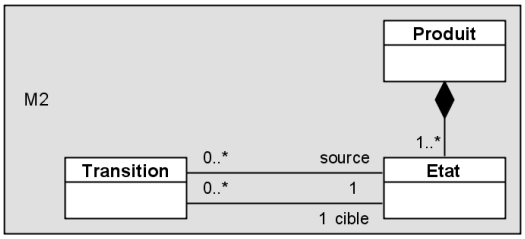


Figure 6. *Extrait du méta-modèle du State-Transition.*

La Figure 7 présente le même diagramme que la Figure 5 sous forme de diagramme d'objets.

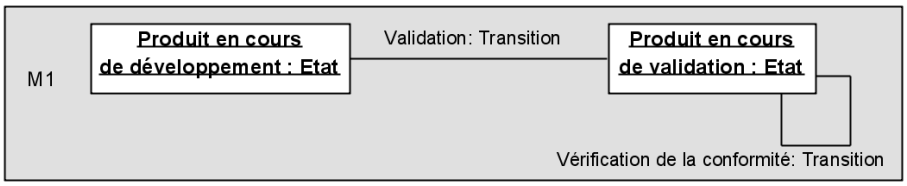


Figure 7. *Extrait d'un modèle de processus orienté produit comme instance du State-Transition.*

3.3. *Modèles de processus orientés décision*

Les modèles de processus orientés décision présentent les transformations ou les élicitations successives du produit dues à des décisions (Rolland, 2005).

La Figure 8 montre un modèle de processus orienté décision concernant la création d'une entité dans un schéma entités/associations, en utilisant le formalisme prescrit par Potts (Potts *et al.*, 1988).

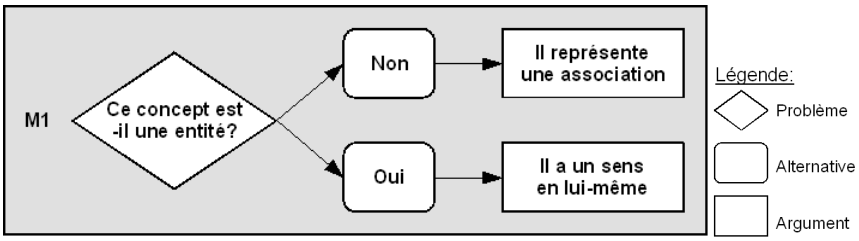


Figure 8. *Extrait d'un modèle de processus orienté décision, dans le formalisme de Potts.*

CAD° (Conversations among Agents on Decisions over Objects) du projet DAIDA (Jarke *et al.*, 1992) inspiré de (Potts *et al.*, 1988), et IBIS (Kunz *et al.*, 1970) sont des méta-modèles de processus orientés décision. La Figure 9 présente le méta-modèle de processus orienté décision de Potts et al. Ce méta-modèle permet de représenter des alternatives répondant à un problème, les arguments sur lesquels ces alternatives sont basées et les produits concernés par ce problème.

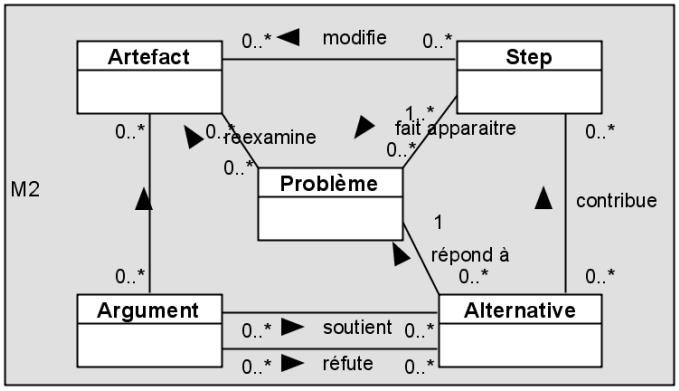


Figure 9. *Méta-modèle de processus orienté décision de Potts et al.*

Le diagramme d'objets de la Figure 10 présente quelques concepts instanciés du méta-modèle de Potts et al.

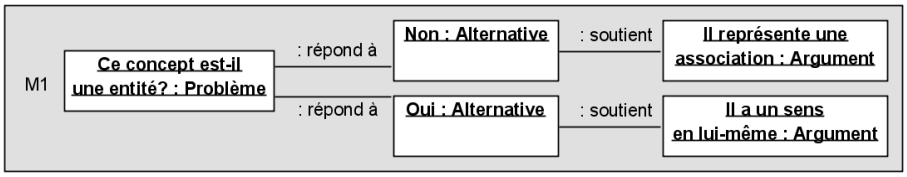


Figure 10. *Extrait d'un modèle de processus orienté décision comme instance du méta-modèle de (Potts et al.).*

3.4. *Modèles de processus 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).

Cette catégorie de modèles de processus a été introduite lors du projet européen NATURE, un méta-modèle du même nom a été défini (Rolland *et al.*, 1995).

Dans l'exemple de la Figure 11, la racine de l'arbre des contextes est composée d'une situation vide et d'une intention qui est de définir le modèle de processus. Le contexte racine est décomposé en plusieurs contextes, jusqu'à l'obtention d'un modèle de processus. Ce modèle est représenté dans le formalisme NATURE (Rolland *et al.*, 1995).

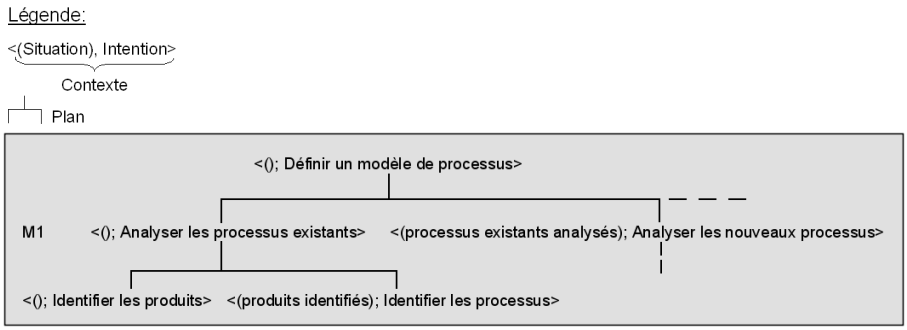


Figure 11. *Extrait d'un modèle de processus orienté contexte, dans le formalisme NATURE.*

Dans le méta-modèle NATURE (cf. Figure 12), le contexte est composé d'une situation (par rapport à un produit) et d'une intention (un objectif par rapport à un produit). Un contexte peut être exécutable, dans ce cas, il engendre une action sur un produit, un contexte peut être de type « choix », il permet alors de choisir entre plusieurs contextes et enfin, il peut être de type « plan », il impose une succession de contextes.

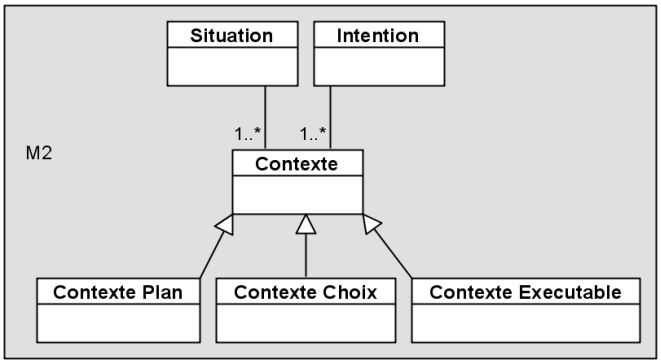


Figure 12. *Extrait du méta-modèle de processus orienté contexte NATURE.*

La Figure 13 présente un extrait du modèle de processus donné en Figure 11 sous forme d'un diagramme d'objets.

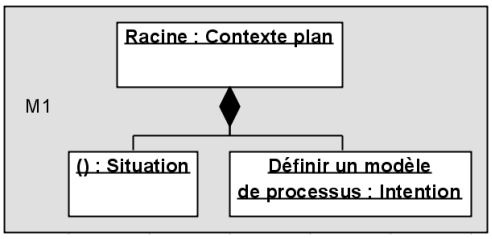


Figure 13. *Extrait d'un modèle de processus orienté contexte comme instance de NATURE.*

3.5. *Modèles de processus 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).

Le modèle de processus de la méthode d'ingénierie des besoins CREWS-L'Écritoire (Rolland *et al.*, 1999) est orienté stratégie. Par exemple, « Éliciter un but » et « Écrire un scénario » correspondent à des intentions de CREWS (cf. Figure 14). « Start » et « Stop » sont les intentions de départ et d'arrivée. La stratégie « Prose libre » permet de passer de l'intention « Éliciter un but » à « Écrire un scénario ». Les intentions sont représentées par des nœuds, les stratégies par des arcs orientés.

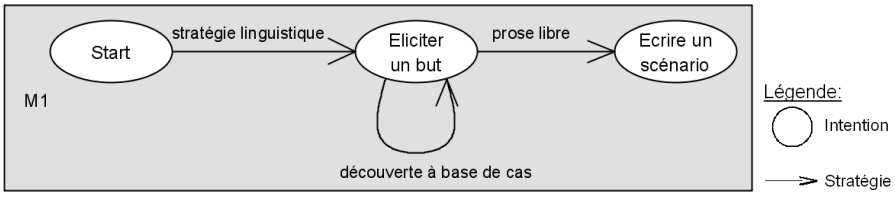


Figure 14. *Extrait du modèle de processus orienté stratégie, CREWS- l'Écritoire dans le formalisme MAP.*

MAP est, à notre connaissance, le seul méta-modèle de processus orienté stratégie, il a été défini par (Rolland *et al.*, 1999). Ce méta-modèle représente des intentions pouvant être atteintes selon différentes stratégies, cf. Figure 15.

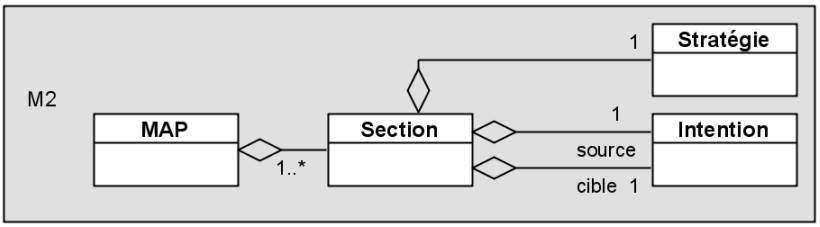


Figure 15. *Extrait du méta-modèle de processus orienté stratégie MAP.*

La Figure 16 présente le modèle de processus de la Figure 14 sous forme d'un diagramme d'objets.

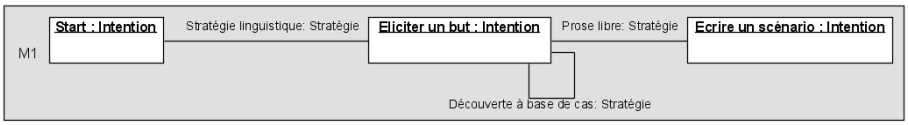


Figure 16. *Extrait du modèle de processus orienté stratégie CREWS- l'Écritoire, instance de MAP.*

3.6. *Synthèse*

Les points précédents montrent qu'il existe une multitude de modèles et de méta-modèles de processus représentant différents points de vue d'un même processus (orientés activité, produit, décision, contexte, stratégie). Les méta-modèles existants ne couvrent pas tous les points de vue : chacun d'eux se concentre sur un ou deux points de vue en particulier. Par exemple, SPEM permet de définir correctement des modèles de processus orientés activité, mais ne permet pas de définir des modèles de processus orientés décision, car il ne contient aucun des concepts liés au point de

vue « décision ». Nous pensons que dans le cadre d'une méthode d'ingénierie des systèmes d'information, voire de son adaptation dans une organisation, les différents points de vue ne sont pas antagonistes mais complémentaires. Par exemple, l'utilisation d'un modèle de processus orienté activité devrait être complémentaire avec celle d'un modèle de processus orienté produit. Il faut pouvoir suivre quelle activité influe sur quel produit, comment et à quel moment du processus. Il est donc nécessaire d'unifier les différents méta-modèles pour faciliter l'utilisation des points de vue multiples.

Ces points de vue se situent à différents niveaux d'abstraction. Le niveau intentionnel (stratégie et contexte) représente les objectifs d'un processus, le niveau opérationnel (produit, activité et décision) représente les actions pour concrétiser ces objectifs. Les différents points de vue ont des relations entre eux ; par exemple, un modèle de processus orienté activité peut être vu comme un raffinement d'un modèle de processus orienté stratégie : une activité concrétise une stratégie.

D'autre part, les méta-modèles de processus existant n'utilisent pas les mêmes concepts ou un même concept n'a pas le même sens d'un méta-modèle à un autre. Par exemple, les principaux concepts de SPEM sont « Phase, Life Cycle, Iteration et Activity » alors que ceux de l'OPF sont « Workflow, Technique, Task, et Activity ». Les modèles de processus résultant ne sont pas très différents car ils sont tous orientés activité, mais aucune relation n'étant établie entre les concepts des différents méta-modèles, il est impossible de déterminer une correspondance entre les modèles de processus. Dans le méta-modèle de processus orienté décision de Potts, le concept « Step » correspond au concept « Activity » de SPEM ; cette correspondance entre concepts n'est pas explicite.

De plus, certains méta-modèles de processus sont trop spécifiques, les modèles de processus instanciés ne sont pas adaptables au vocabulaire d'une organisation. Par exemple, SPEM définit des concepts spécifiques « Phase, Life Cycle, Iteration et Activity » et la façon dont ils doivent être utilisés. Ces concepts devraient être définis à un niveau inférieur de modélisation afin de laisser aux organisations le choix de leur concept et de leur usage.

Enfin, la plupart des méta-modèles n'offre pas de mécanismes d'adaptation, sauf SPEM qui est extensible grâce au mécanisme de profils d'UML.

Pour résoudre ces problèmes, il convient de disposer de techniques de modélisation pour construire des méta-modèles de processus :

– unifiés : afin de n'avoir qu'un seul méta-modèle de processus à instancier et faire évoluer,

– adaptés : construit par des experts de l'organisation et donc adaptés au vocabulaire et aux contraintes propres à cette organisation,

– multi-vues : proposant les cinq points de vue (activité, produit, etc.) d'un processus et les relations entre ces points de vue, et sur différents niveaux d'abstraction, intentionnel ou opérationnel.

Ces techniques de modélisation doivent être accompagnées d'un guide méthodologique permettant de guider leur utilisation.

Nous présentons quelques unes de ces techniques dans la suite de cet article. Nous ne traitons ici que l'aspect « unificateur » et « adaptable », nous détaillerons l'aspect « multi-vues » dans un article ultérieur.

4. Un patron pour la méta-modélisation des processus

La construction d'un méta-modèle de processus passe par l'ajout et la suppression de concepts (activité, produit,...), selon les projets ou les organisations. Chaque organisation utilise son propre vocabulaire, ses propres méthodes, ceci étant dû à l'histoire de l'organisation et son fonctionnement. Traditionnellement, dans un méta-modèle, un concept est modélisé par une méta-classe principale liée à d'autres méta-classes par des associations (par exemple, une activité produit et consomme des ressources, cf. Figure 1). Certains concepts ne peuvent cependant pas être modélisés par une seule méta-classe. En effet, comme nous l'avons vu dans l'introduction, il faut distinguer les concepts de leur catégorie. Nous proposons donc un patron permettant de catégoriser des concepts dotés de propriétés spécifiques et partageant des propriétés communes et ceci sur deux niveaux de modélisation :

– au niveau des modèles de processus (M1) pour valuer des propriétés applicables à chaque concept du méta-modèle de processus (par exemple, le fait qu'une phase puisse être optionnelle ou non) et à l'ensemble des concepts d'une même catégorie (par exemple, le fait que toute phase soit composée d'activités).

– au niveau du processus (M0) pour valuer des propriétés applicables à chaque occurrence de concept du modèle de processus (par exemple, la durée de la phase d'analyse d'un projet donné) et à l'ensemble des phases d'un projet (par exemple, la durée maximale d'une phase).

Un certain nombre de concepts ont été identifiés. Notre solution prend en compte chacun de ces concepts, ainsi que leur catégorie : unité de travail et catégorie d'unité de travail (le terme activité utilisé précédemment nous paraît trop limitatif, une activité est une sorte d'unité de travail, au même titre qu'une phase), produit et catégorie de produit, agent et catégorie d'agent, décision et catégorie de décision, contexte et catégorie de contexte, stratégie et catégorie de stratégie, contrainte et catégorie de contrainte, guidage et catégorie de guidage... L'association d'un concept à sa catégorie de concepts évoque le problème résolu par le patron « Item-Description ».

Dans cette partie, nous présentons le patron « Item-Description » sur lequel nous nous basons pour construire notre patron appelé « Concept- Catégorie de Concepts ». Nous expliquons ensuite l'instanciation profonde qui est utilisée dans le patron « Concept- Catégorie de Concepts » et enfin, nous présentons ce dernier dans le formalisme P-SIGMA (Conte *et al., 2001).*

4.1. *Le patron « Item-Description »*

Ce patron a été introduit par Peter Coad dans (Coad, 1992). Il est utilisé lorsque des valeurs d'attributs sont attribuées à plusieurs objets instances d'une même classe. La classe « ItemDescription » (telle que représentée par (Gzara, 2000) dans la Figure 17) possède des valeurs d'attributs qui s'appliquent à plusieurs Items. Un « Item » a ses propres valeurs d'attributs et possède des méthodes pour accéder aux valeurs d'attributs de la classe « ItemDescription ». La Figure 18 montre une imitation de ce patron pour des voitures et leur modèle. Toutes les voitures d'un même modèle partagent les mêmes propriétés définies pour ce modèle.

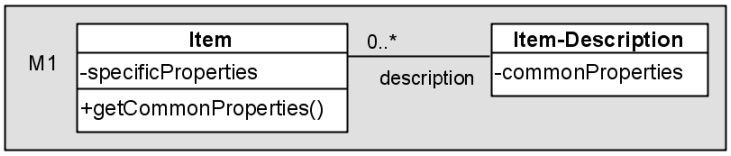


Figure 17. *Le patron « Item-Description ».*

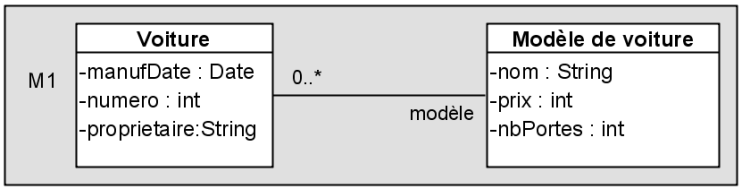


Figure 18. *Imitation du patron « Item-Description ».*

Le patron « Item-Description » a été initialement proposé pour construire des modèles de produit (niveau M1) d'une application (cf. Figure 18). Pour la modélisation des processus, nous devons néanmoins l'utiliser sur trois niveaux de modélisation (cf. Figure 19) :

– niveau M2 : pour représenter le méta-modèle de processus,

– niveau M1 : pour représenter le modèle de processus,

– niveau M0 : pour représenter le projet, c'est à dire l'exécution du processus.

D'autre part, les propriétés des différentes classes doivent pouvoir être instanciées sur différents niveaux de modélisation : au niveau M1 pour définir les propriétés des classes du modèle de processus et au niveau M0 pour définir les propriétés des objets du processus réel. Afin de permettre l'instanciation d'attributs sur des niveaux de modélisation différents, la Figure 19 propose de combiner instanciation et héritage. Cette notion (héritage et instanciation d'attributs de méta-classes) est très proche de la notion de Powertype (Odell, 1994), (Atkinson, 2001), (Henderson-Sellers et al., 2005).

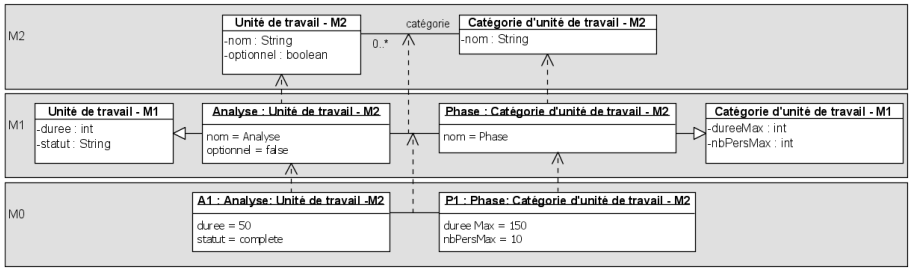


Figure 19. *Instanciation et héritage d'attributs de méta-classes.*

La classe Phase (une catégorie d'unité de travail parmi d'autres) instancie les propriétés de la méta-classe « Catégorie d'unité de travail -M2 » et hérite des propriétés de la classe « Catégorie d'unité de travail-M1 ». Les propriétés héritées (par exemple duréeMax,...) sont instanciées au niveau M0 par P1, modélisant l'ensemble des phases d'un projet donné, dans l'exemple toutes les phases auront une duréeMax égale à 150 jours.

La classe « Analyse » (une unité de travail) instancie les propriétés de la méta-classe « Unité de Travail -M2 » et hérite des propriétés de la classe « Unité de travail -M1 ». Les propriétés héritées (par exemple durée) sont instanciées au niveau M0 par A1, modélisant une phase d'analyse dans un projet donné, dans l'exemple, A1 a une durée de 50 jours.

Afin d'alléger les schémas de modélisation tels que ceux obtenus en Figure 19, nous proposons d'utiliser le mécanisme d'instanciation profonde.

4.2. *L'instanciation profonde*

L'instanciation profonde (« Deep Instantiation ») a été introduite par (Atkinson et al., 2001). Elle permet d'instancier un attribut ou une association au niveau de modélisation que l'on souhaite et non plus au niveau strictement inférieur. L'instanciation profonde consiste à affecter un exposant à un attribut ou à une association. Chaque fois que la classe à laquelle l'attribut appartient est instanciée, l'exposant de l'attribut est retranché de 1. Tant que l'exposant est supérieur à 0, l'attribut n'est pas instancié. Lorsque l'exposant est égal à 0, l'attribut est instancié. La notation adoptée pour les attributs est la suivante :

– « attribut : type », lorsque l'attribut a un exposant égal à 1 (P=1). Il est instancié normalement au niveau strictement inférieur de modélisation.

– « $attribut^P$: type », lorsque l'attribut a un exposant supérieur à 1 (P>1). Il est instancié lorsque P=0, au $P^{ième}$ niveau inférieur de modélisation.

– « attribut=valeur », lorsque l'attribut a un exposant égal à 0 (P=0), il est alors instancié.

Les associations ont un exposant qui est décrémenté lorsque les classes auxquelles elles sont associées sont instanciées. Lorsque l'exposant vaut zéro, un lien est créé entre les objets.

La Figure 20 présente l'adaptation du patron « Item-description » pour la méta-modélisation de processus, pour unité de travail et catégorie d'unité de travail et leur instanciation, en utilisant le mécanisme d'instanciation profonde. Au niveau du méta-modèle (M2), l'attribut «durée» de la classe « Unité de travail » a un exposant égal à 2. Cela signifie que cet attribut est instancié au deuxième niveau inférieur, c'est à dire au niveau de l'exécution du processus (M0).

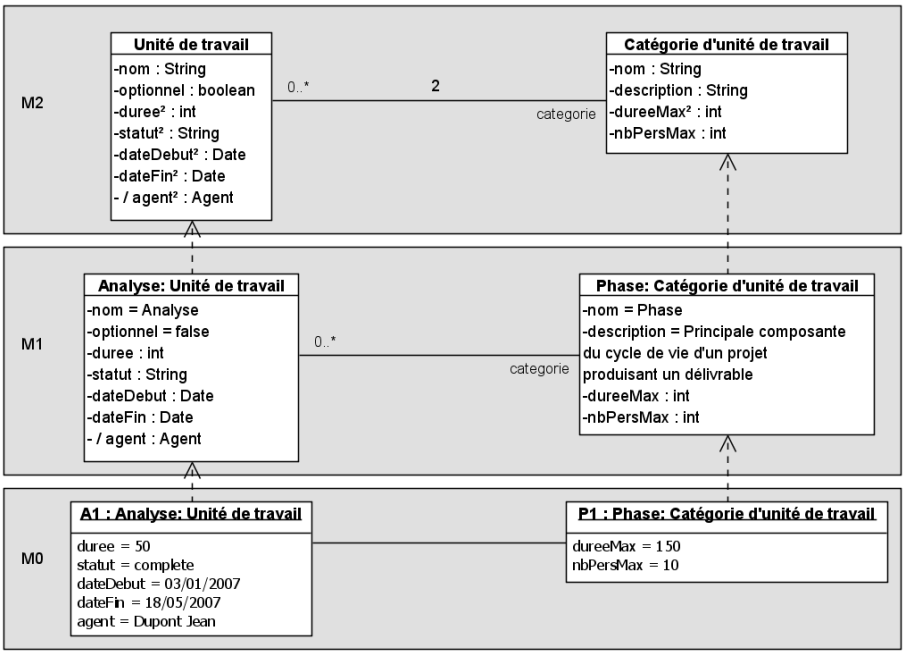
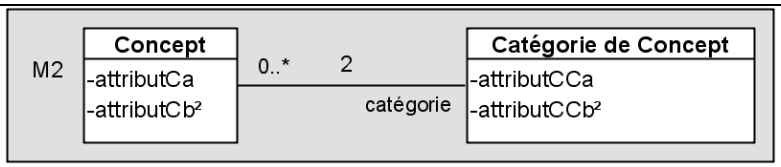


Figure 20. *Utilisation de l'instanciation profonde pour la méta-modélisation de processus.*

4.3. *Le patron « Concept- Catégorie de Concepts »*

Le patron « Concept- Catégorie de Concepts » présenté dans le tableau ci-dessous est une adaptation du patron « Item-Description » pour la méta-modélisation de processus et utilise notamment l'instanciation profonde.

Le patron est présenté avec le formalisme P-SIGMA (Conte *et al., 2001)* proposé par l'équipe Sigma. Ce formalisme permet l'uniformisation de la représentation des patrons produits et processus, la clarification de l'interface des patrons et une meilleure organisation des systèmes de patrons.

Nom	Concept – Catégorie de Concepts
Classification	{patron 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 à deux 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 catégorie de concepts font parties du méta-modèle de processus. Leurs instances définissent les éléments faisant partie des modèles de processus et des catégories de modèles de processus auxquels ils se rattachent. Ce patron permet également d'instancier les propriétés des concepts à deux niveaux de modélisation : modèle (M1) et exécution (M0).
Solution-modèle	 Les attributs « attributCa » et « attributCCa » sont classiquement instanciés au niveau M1. Les attributs « attributCb² » et « attributCCb² » et l'association entre « Concept » et « Catégorie de Concept » bénéficient de l'instanciation profonde et sont instanciés au niveau M0.
Cas d'application	Nous souhaitons définir le concept de produit. Ce concept doit être décliné sous la forme de deux méta-classes de niveau M2: – Catégorie de produit définit les propriétés instanciables par les catégories de produit (par exemple UML Model) et les instances des catégories de produit (UMLM1 dans la figure). – Produit définit les propriétés instanciables par les produits (par exemple modèle des processus métier) et les instances des produits (MPM1 dans la figure).

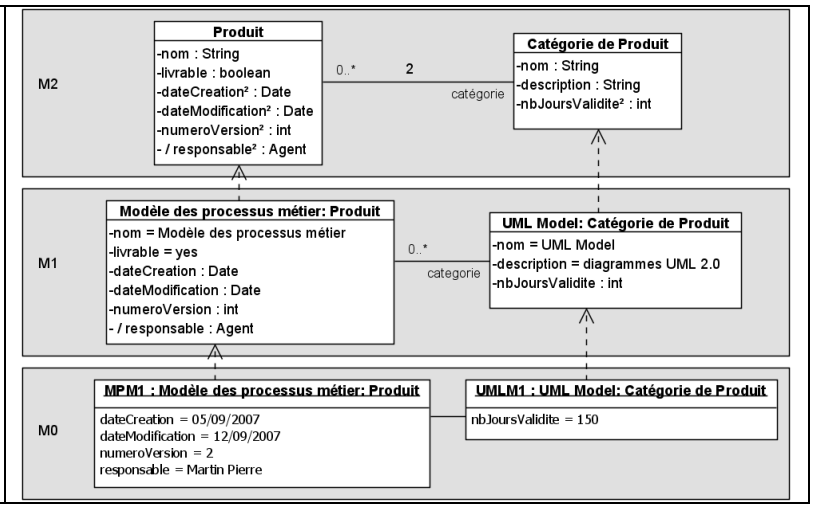


Tableau 1. *Le patron « Concept – Catégorie de Concept ».*

5. Ingénierie des processus par imitation du patron « Concept-Catégorie de Concepts »

Nous présentons ici un exemple d'un méta-modèle de processus unifié (tous les concepts des méta-modèles de processus existants peuvent être représentés dans un unique méta-modèle) et adapté aux contraintes et caractéristiques d'un projet particulier, défini principalement par imitation du patron « Concept-Catégorie de Concepts ». Puis nous présentons un exemple partiel d'instanciation de ce méta-modèle.

5.1. Exemple de méta-modèle de processus

Nous souhaitons ici représenter les concepts du modèle de processus de la démarche Symphony. Le modèle de processus de la démarche Symphony est instance du méta-modèle de processus SPEM. La Figure 21 illustre les concepts de Symphony et les catégories de concepts correspondantes de SPEM. Il apparaît clairement que chaque concept de Symphony, par exemple, « Spécifications conceptuelles des besoins », correspond à une catégorie de concepts de SPEM, ici « Phase ».

D'autre part, il faudra définir des propriétés générales à la démarche dans le modèle de processus et des propriétés spécifiques pour chaque exécution du processus dans le cadre d'un projet particulier. Le patron « Concept-Catégorie de Concepts » répond bien à ces deux problèmes : catégorisation et instanciation de propriétés à deux niveaux de modélisation.

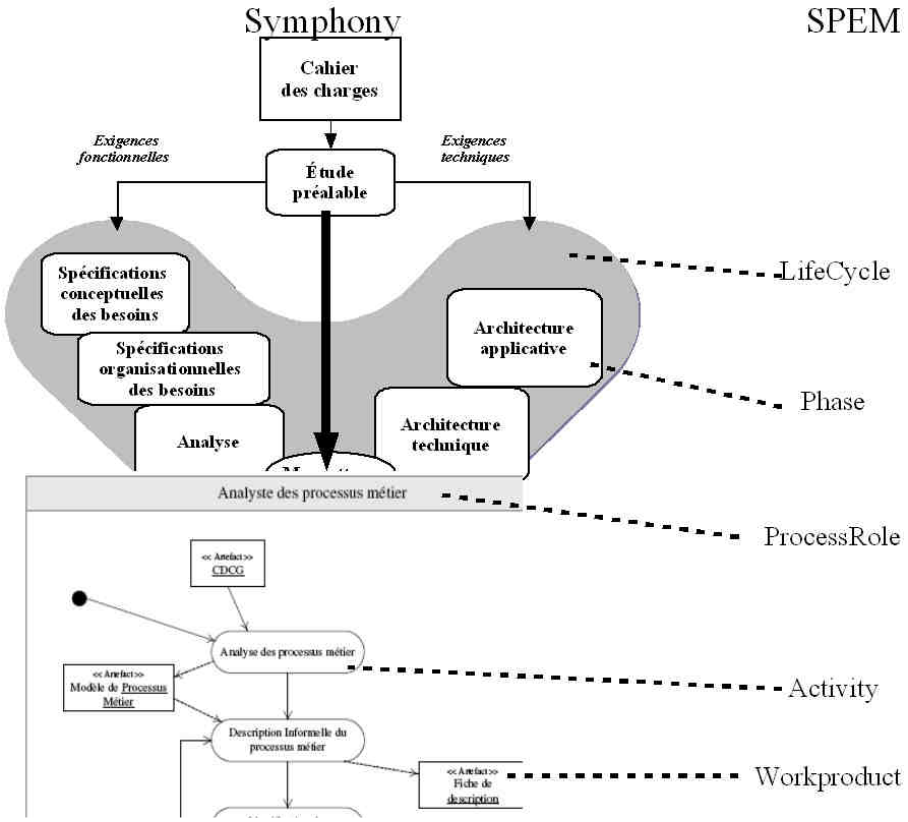


Figure 21. *Illustration des catégories de concepts de SPEM et des concepts de Symphony.*

Dans un premier temps, il faut donc définir un méta-modèle de processus qui permettra d'instancier les concepts de Symphony et leurs catégories de concepts. Le méta-modèle, Figure 22, contient trois imitations du patron « Concept- Catégorie de Concepts » pour les concepts unité de travail et catégorie d'unité de travail, produit et catégorie de produit, agent et catégorie d'agent.

Une catégorie d'agent peut participer à une catégorie d'unité de travail, de même qu'un agent peut participer à une unité de travail. Une unité de travail peut avoir des produits en entrée ou en sortie. Chaque produit a un ou plusieurs agents responsables. Des unités de travail peuvent s'effectuer en séquence ou en parallèle.

Les classes définies dans le méta-modèle de processus vont pouvoir être instanciées pour définir SPEM et Symphony.

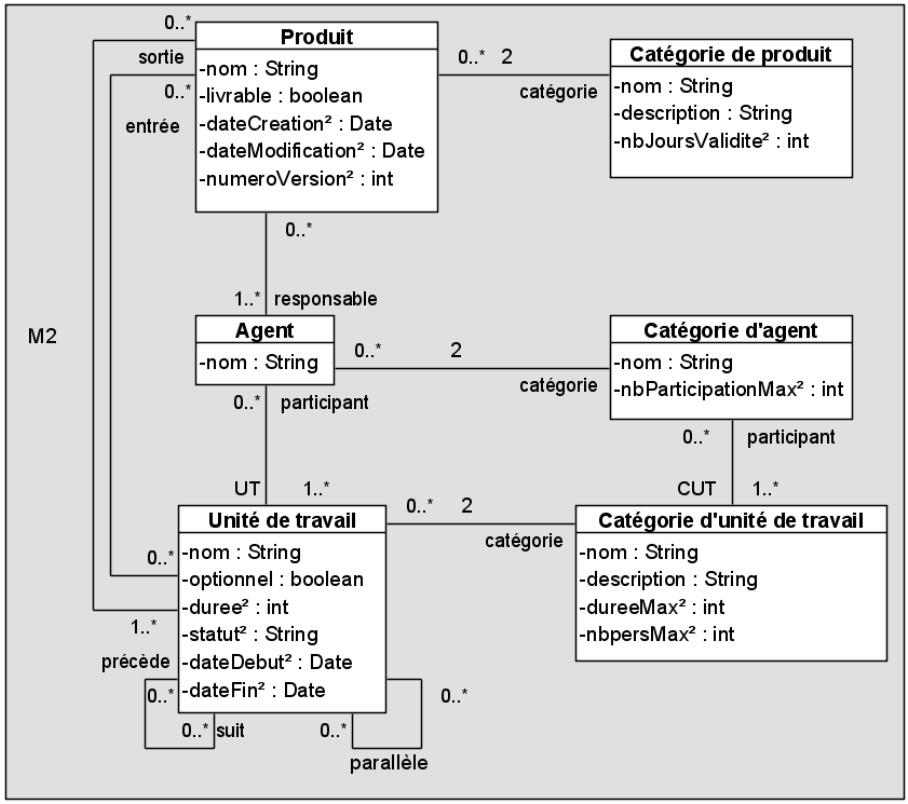


Figure 22. *Extrait d'un méta-modèle de processus : trois imitations du patron « Concept – Catégorie de Concept ».*

5.2. Instanciation pour SPEM et Symphony

La Figure 23 illustre la construction de SPEM et de Symphony à partir du méta-modèle de processus de la Figure 22. Pour plus de clarté, les classes de cet exemple ne contiennent ni attribut ni méthode.

Dans un premier temps, il est nécessaire d'instancier les catégories de concepts qui constituent la catégorie de modèle de processus, ici SPEM. Lorsque SPEM est instancié, nous pouvons instancier les concepts qui sont liés à une catégorie de concepts. L'ensemble de ces concepts forme le modèle de processus. Ici, nous avons instancié certains concepts de Symphony (plus clairs sur la figure). Les concepts de Symphony sont rattachés aux catégories de concepts de SPEM. Par exemple, le concept « Spécifications conceptuelles des besoins » est rattaché à la catégorie de concepts « Phase ».

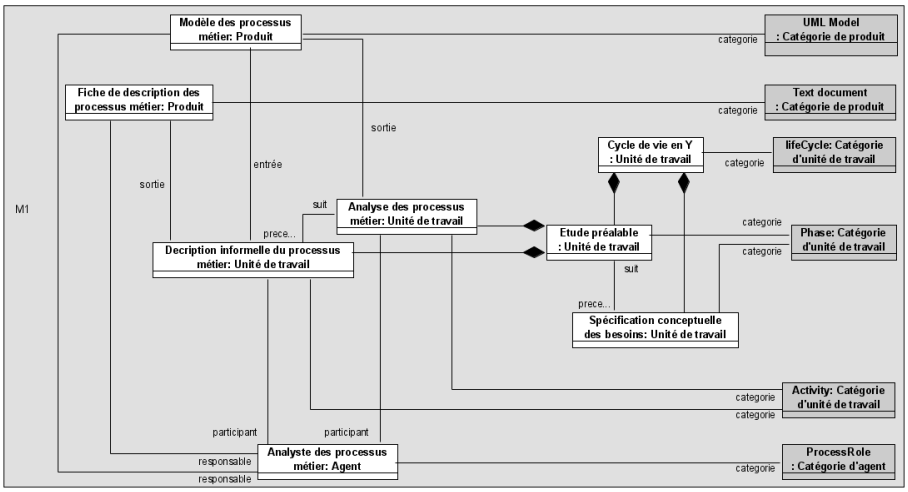


Figure 23. *Construction d'une partie de SPEM et de Symphony avec le méta-modèle de processus.*

Ce modèle de processus diffère de la vision de l'OMG, étant donné que certains concepts de SPEM, qui est considéré comme un méta-modèle de processus, sont « descendus » au niveau des modèles de processus, comme LifeCycle, Phase, Activity et ProcessRole. Une catégorie de concept d'un modèle de processus n'est pas forcément un concept issu d'un méta-modèle de processus existant, cela dépend du méta-modèle de processus et de son niveau de précision : avec notre solution, plus le méta-modèle est précis au niveau des concepts, plus il y a de chance que ces concepts se situent au niveau du modèle de processus.

6. Conclusion

Dans cet article nous avons présenté les différents types de méta-modèles de processus existants et les différents problèmes liés à leur utilisation : manque de consensus autour des concepts utilisés, complémentarité des points de vue non explicite et manque d'adaptabilité au vocabulaire des organisations. Tous ces problèmes conduisent à l'instanciation de modèles de processus inadaptés aux besoins des organisations. Nous avons proposé le patron « Concept-Catégorie de Concepts » qui permet, par imitation, de créer des méta-modèles de processus unifiés et adaptés : tous les concepts des méta-modèles de processus existants peuvent être représentés dans un unique méta-modèle qui ne présente que des concepts généraux qui pourront être instanciés pour des besoins spécifiques, en utilisant le vocabulaire adéquat. Le patron permet de créer des concepts et leur catégorie de concepts au niveau des méta-modèles de processus, dont les propriétés

peuvent être instanciées au niveau des modèles (M1) ou au niveau de l'exécution du processus (M0).

De manière plus générale, nous devons étudier les patrons spécifiques à la méta-modélisation des processus. Des patrons de conception existants tels que ceux du GoF (Gamma et al., 1995) doivent être étudié pour déterminer si leur utilisation a un sens pour la méta-modélisation des processus. Ils devront ensuite être adaptés pour pouvoir être réutilisés dans ce cadre précis.

7. Bibliographie

Atkinson C., Kühne T., « The Essence of Multilevel Metamodeling », *UML '01: Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools*, Toronto, Canada, October 1-5, 2001, Lecture Notes in Computer Science, vol. 2185 , p. 19-33.

Australian Standard, Standard metamodel for software development methodologies, AS 4651—2004, August 2004.

Booch G., *Object Oriented Analysis and Design with Application*, Benjamin/Cummings, 1991.

Cauvet C., « Modélisation des processus d'ingénierie des Systèmes d'Information », *Encyclopédie de l'Informatique et des Systèmes d'Information*, Hermès, 2006.

Coad P., « Object-Oriented Patterns », *Communication of the ACM*, ACM Press, vol. 35, n° 9, 1992, p. 152-159.

Conte A., Fredj M., Giraudin J.-P., Rieu D., « P-Sigma : un formalisme pour une représentation unifiée de patrons », *Actes du XIXème Congrès INFORSID*, Martigny, Suisse, 24-27 mai, 2001, p. 67-86.

Dowson M., « Iteration in the software process », *Review of the 3rd International Software Process Workshop*, Proceedings of the 9th International Conference on Software Engineering, Los Alamitos, CA, USA, 1987, IEEE Computer Society Press, p. 36-41.

Finkelstein A., Kramer, J., Goedicke, M., « ViewPoint oriented software development », *3rd International Workshop on Software Engineering and Its Applications*, 1990, p. 374-384.

Gamma E., Helm R., Johnson R., Vlissides J., *Design patterns -Elements of reusable object-oriented software*, Professional Computing Series. Addison Wesley, 1995.

Gzara L., Les patterns pour l'ingénierie des systèmes d'information, Thèse de l'INPG, décembre 2000.

Harmsen, A.F., *Situational Method Engineering*, Moret Ernst & Young, 1997.

Harel D., « Statecharts: A Visual Formulation for Complex Systems », *Science of Computer Programming*, vol. 8, n° 3, 1987, p. 231-274.

Hassine I., Spécification et formalisation des démarches de développement à base de composants métier : la démarche Symphony, Thèse de l'INPG, septembre 2005.

Henderson-Sellers B., Gonzalez-Perez C., « The rationale of powertype-based metamodelling », *Second Asia-Pacific Conference on Conceptual Modelling*, 30 January–4 February 2005. Australian Computer Science Communications, vol. 7, no. 6. Australian Computer Society, p. 7–16.

Humphrey W. S., Kellner M. I., « Software Process Modeling: Principles of Entity Process Models », *ICSE '89: Proceedings of the 11th International Conference on Software Engineering*, ACM Press, Pittsburgh, PA, USA, May 15-18 1989, p. 331-342.

Jarke M., Mylopoulos J., Schmidt J. W., Yannis Vassiliou, « DAIDA: An Environment for Evolving Information Systems », *ACM Transactions on Information Systems*, January 1992, vol. 10, n° 1, p. 1-50.

Kunz W., Rittel H.W.J., Issues as elements of information systems, Working Paper n° 131, Heidelberg-Berkeley, 1970.

Kruchten P., *The Rational Unified Process: An Introduction*, Addison-Wesley, 2000.

Odell J., « Power Types », *Journal of Object-Oriented Programming*, vol. 7, n° 2, May 1994, p. 8-12.

OMG, MetaObjectFacility (MOF) Specification, Version 1.4.1, July 2005.

OMG, Software Process Engineering Metamodel Specification, Version 1.1, January 2005.

OMG, Unified Modeling Language: Superstructure, Version 2.1, April 2006.

OOSPICE: Software Process Improvement and Capability Determination for Object Oriented/ Component Based Software Development, http://oospice.com, 7 October 2002.

OPF: Open Process Framework, http://www.opfro.org/, 16 December 2005.

Potts C., Bruns G., « Recording the Reasons for Design Decisions », *ICSE '88: Proceedings of the 10th International Conference on Software Engineering*, IEEE Computer Society Press, Singapore, April 11-15, 1988, p. 418-427.

Roques P., Vallée F., *UML en action : de l'analyse des besoins à la conception en Java*, Eyrolles, 2000.

Rolland C., « L'ingénierie des méthodes : une visite guidée », *e-TI*, n° 1, (http://www.revue-eti.net/document.php?id=726), 25 octobre 2005.

Rolland C., Prakash N., Benjamen A., « A Multi-Model View of Process Modelling », *Requirements Engineering*, Springer-Verlag London Limited, vol. 4, n° 4, 1999, p. 169-187.

Rolland C., Souveyet C., Moreno M., « An Approach for defining ways-of-working », *Information System Journal*, vol. 20, n° 4, 1995, p. 337-359.