Garantir la complétude des cartes de patrons à l'aide de méta-patrons

Rebecca Deneckere

Garantir la complétude des cartes de patrons à l'aide de méta-patrons

Rébecca Deneckère

Université Paris 1 Panthéon-Sorbonne, Centre de Recherche en Informatique

90 rue de Tolbiac 75013 PARIS, France

denecker@univ-paris1.fr

RESUME.

La notion de patron a été largement utilisée ces dernières années. Elle permet de définir des techniques autorisant la réutilisation de la connaissance existante. Un patron résout un problème spécifique du cycle de vie d'un système. La connaissance encapsulée dans ces patrons est généralement stockée dans des bibliothèques surpeuplées. Pour résoudre ce problème engendrant une difficulté d'accès et de choix des patrons, nous avons proposé, dans [DEN 01B], l'utilisation des cartes de processus pour aider l'ingénieur de méthodes à les organiser et les sélectionner. Cependant, la complétude des cartes est également un très gros problème. Les cartes doivent être complètes pour permettre d'offrir une aide utile à l'ingénieur de méthodes. Cet article traite de ce problème et offre une technique de construction des patrons lors de la construction des cartes de processus.

ABSTRACT.

The pattern notion defines techniques allowing the existing knowledge reuse. A pattern solves a specific problem of a software system life cycle. The knowledge encapsulated in these patterns is generally stored in classic library repositories that quickly become overcrowded. As a result, [DEN 01B] proposes the use of process maps in order to help the method engineer to sort and select them. But the completeness of the maps are a very important problem that has to be solved in order to offer a useful guidance to the method engineer. This paper deals with this problem with a pattern construction technique guiding engineers when creating the maps.

MOTS CLEFS :

Ingénierie des méthodes, méthode situationnelle, extension de méthode, construction de patron, carte de processus, méta-patron.

KEYWORDS :

Method engineering, situational method, method extension, pattern construction, process map, meta-pattern.

1 Introduction

Il n'est plus réellement nécessaire de présenter le concept de patron. Il a été utilisé et décrit dans différents domaines : [COA 92] [BEC 97] [BUS 96] [COP 95] [GAM 94] [HAY 96] [FOW 97]. Dans l'ingénierie des méthodes, les patrons génériques aident à construire des méthodes situationelles. Ils permettent de savoir quels sont les meilleurs processus dans telle ou telle situation et guident l'ingénieur de méthodes lors de la construction de sa méthode.

Suite à cet engouement pour la notion de patron est apparu sur le marché une quantité incroyable de langages de patrons entraînant les problèmes définis dans [DEN 01B]. Premièrement, les catalogues de patrons deviennent rapidement surpeuplés. Deuxièmement, certains patrons ont des relations de précédence qu'il serait utile de mettre en évidence. [DEN 01B] proposait une technique d'organisation de ces patrons en utilisant les cartes de processus [BEN 99] [ROL 99]. Ces cartes sont une formalisation du processus d'utilisation des patrons qui aident l'ingénieur en le guidant à travers chaque transformation.

Cependant, un des problèmes restant était la complétude de ces cartes de patrons. Cette technique est parfaite si les cartes sont complètes mais complètement inutile si elles ne le sont pas. En effet, si seulement l'un des patrons n'a pas été défini et que l'ingénieur en ai besoin pour modifier sa méthode, alors il se retrouve dans une impasse et ne peut plus naviguer à l'intérieur de sa carte. Pour résoudre ce problème, nous proposons ici une technique de construction des patrons par l'utilisation de méta-patrons.

Les notions de patrons et de cartes de processus sont décrites dans la section suivante. La section 3 se concentre sur la technique des méta-patrons et des stratégies. Un exemple illustre ces propos en quatrième section. Nous concluons dans la cinquième et dernière section.

2 Patrons et Cartes de Processus

Nous utilisons ici le formalisme défini dans [DEN 01A][DEN 98]. Un patron est composé de deux parties : la connaissance réutilisable (le corps) et les aspects applicatifs (la signature). Le corps encapsule la description du processus à appliquer au produit en modification. La signature représente la situation avant modification, l'intention à réaliser et la cible de cette modification. On appelle également ce concept une *interface* [ROL 98]. Elle est vue comme un triplet <situation, intention, cible> associé au corps.

[DEN 01B] utilise le concept de carte de processus pour organiser les patrons d'un langage. Une carte est un graphe composé de nœuds et d'arcs. Les nœuds sont les intentions à atteindre et les arcs les différentes stratégies applicables pour les atteindre. Une navigation dans la carte représente un chemin particulier commençant toujours par l'intention *Démarrer* et terminant par l'intention *Arrêter*. Chaque

patron est représenté dans la carte par une section spécifique (une intention source, une intention cible et une stratégie pour aller de l'une à l'autre). Le patron à appliquer sera différent selon la stratégie utilisée par l'ingénieur de méthodes. Il y aura donc autant de patrons applicables, entre deux intentions, que de stratégies (i.e. que de sections dans la carte).

La figure suivante illustre l'application de cette technique. Cette partie de carte comprend trois intentions: les deux intentions obligatoires *Démarrer* et *Arrêter*, ainsi qu'une intention d'extension d'une méthode pour y intégrer le concept de *Référentiel Temporel*. Cette carte possède trois sections différentes : l'une permettant d'accéder à l'intention d'arrêt de la navigation dans la carte, les deux autres permettant de satisfaire l'intention d'extension. Ces deux sections représentent deux patrons utilisant deux stratégies différentes : la généralisation et la spécialisation. L'intégration de ce nouveau concept sera donc effectuée de manière différente selon la section (i.e. le patron) choisie lors de la navigation.

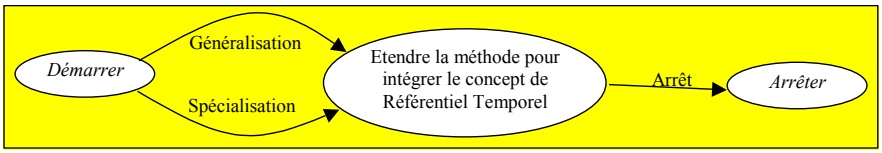

Figure 1. *Exemple de carte de processus*

3 Méta-patrons et Stratégies

Les patrons peuvent donc être regroupés et organisées dans une carte de processus spécifique. Cependant, la construction de ces cartes pose un problème de complétude.

La problématique, lors de la construction d'une carte, est tout d'abord de définir les différentes intentions de celle-ci selon le champ d'application considéré (par exemple, dans le cadre d'une extension de méthode, il faudra faire l'inventaire de tous les éléments que l'on souhaite intégrer dans celle-ci). Ensuite, l'ingénieur doit faire l'inventaire de toutes les stratégies applicables pour atteindre ces intentions. Finalement, chacune des sections définies doit être représentée par un patron différent. Cependant, cet inventaire est un travail long, difficile et surtout stratégique. Si l'on oublie une section, la carte n'est plus complète et l'on risque de se trouver dans une impasse lors de la navigation. Nous proposons donc ici de définir un ensemble de stratégies génériques selon le champ d'application considéré. Appliquer toutes ces stratégies sur toutes les intentions de la carte permet de certifier que celle-ci sera complète.

De plus, pour faciliter la réutilisation de ces stratégies génériques, toute la connaissance relative est encapsulée dans un patron spécifique que l'on appelle méta-patron. Chaque application d'un méta-patron permettra de construire un patron spécifique permettant d'atteindre une intention cible, en partant d'une intention

source, par l'application d'une stratégie particulière. La figure suivante illustre l'instanciation du méta-patron utilisant la stratégie *Spécialisation* pour la construction de trois patrons différents (trois sections différentes de la carte).

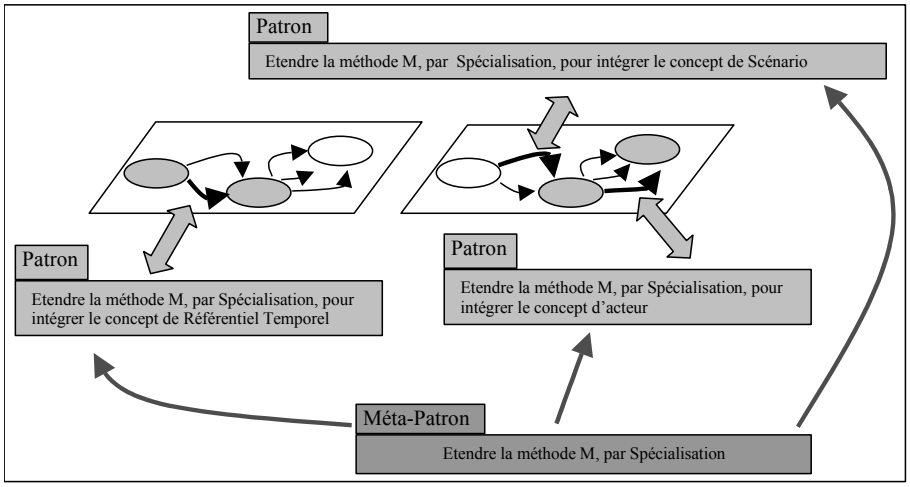

Figure 2. *Trois instanciations d'un même méta-patron*

Une description complète de ce méta-patron est faite en [DEN 01A].

Pour restreindre le champ d'application de cet article et offrir une formalisation concise des stratégies, nous allons nous concentrer ici sur les stratégies permettant d'étendre une méthode [DEN 01A]. Ce type de modification permet de modifier une méthode existante pour l'adapter à la situation en cours selon les besoins de l'ingénieur de méthodes.

3.1 Stratégie d'extension de méthode

Les stratégies décrivent une façon de modifier une méthode. Elles prennent trois paramètres en compte : l'élément de méthode à modifier - [Elément X] (cet élément fait partie de la méthode avant l'extension), l'élément que l'on souhaite intégrer dans la méthode - [Elément Y] (cet élément fait partie de la méthode après extension) et l'élément rendu caduc à la suite de la modification – [Elément D]. Cet élément n'est pas obligatoire. En effet, toutes les modifications ne modifient pas la méthode d'origine au point de rendre l'un de ces éléments inutile. Cependant, si tel est le cas, la stratégie doit prendre en compte la suppression de cet élément.

Chaque stratégie peut donc être formalisée de la manière suivante: **Stratégie ([Elément X], [Elément Y], [Elément D])**.

Prenons l'exemple suivant: un patron permettant d'intégrer le concept de *Classe Acteur* dans une méthode qui contient déjà le concept de *Classe*. Il suffit alors de

pratiquer une spécialisation dans le but d'offrir un nouveau sous-type à ce concept. Ici, l'[Elément X] - l'élément existant - est le concept de *Classe*, l'[Elément Y] – l'élément à intégrer - est la *Classe Acteur*. La stratégie est la *Spécialisation*. Dans ce cas, aucun élément ne devient inutile du fait de la modification donc l'[Elément D] est nul. L'application de la stratégie sera donc représentée ainsi: **Spécialisation (Classe, Classe Acteur, ϕ).**

La connaissance relative à chaque stratégie est encapsulée dans un patron spécifique appelé méta-patron. Ceux-ci ont le même type de description que les patrons.

3.2 Méta-patron d'extension de méthode

Les méta-patrons, comme les patrons, ont une signature et un corps. La signature représente la partie formelle de sa description, i.e. son interface.

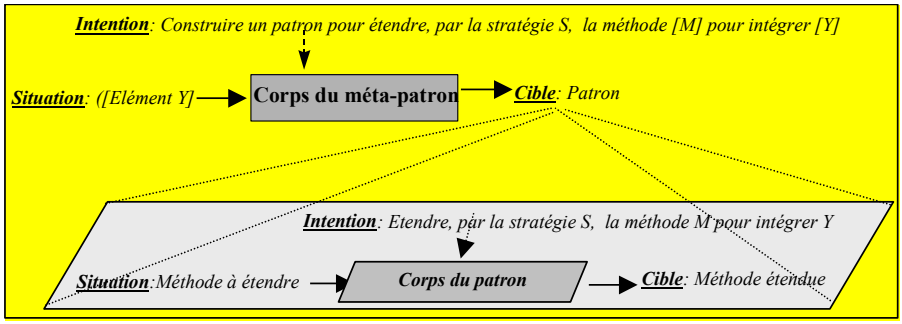

Figure 3. *Interface d'un méta-patron*

La situation représente l'élément Y (celui que l'on veut intégrer dans la méthode). L'intention est de construire un patron qui intégrera cet élément dans la méthode, ceci en suivant une stratégie spécifique. La cible est donc ce patron, construit pour intégrer cet élément spécifique dans une méthode en suivant cette stratégie.

Le corps du méta-patron représente toutes les opérations permettant la construction de ce patron, i.e. la définition de sa situation, de son intention, de sa cible et de son corps. Celui-ci est construit par un ensemble d'opérateurs représentant la stratégie utilisée. Chaque stratégie représente un ensemble d'opérateurs utilisant les trois paramètres cités plus haut. L'élément Y est connu dés que l'on utilise le méta-patron pour construire le patron. Au contraire, les éléments X et D ne seront connus que lors de l'instanciation de ce patron à la méthode spécifique que l'on souhaite étendre.

De même que pour les patrons, les méta-patrons peuvent être différenciés selon deux types : ceux permettant de construire un patron modifiant la partie Produit d'une méthode et ceux modifiant sa partie Processus.

[DEN 01A] définit une extension de méthode comme une technique donnant à l'ingénieur de méthodes les moyens de représenter un plus grand nombre de phénomènes réels qu'il ne le pouvait auparavant avec la méthode qu'il appliquait. Une extension de la partie Produit d'une méthode permet donc d'introduire de nouveaux concepts dans un modèle existant. Une extension de la partie Démarche d'une méthode permettra d'introduire la construction de ces nouveaux concepts dans la même méthode.

En effet, si la partie Produit de la méthode ne correspond pas aux besoins de l'ingénieur, la méthode peut être modifiée par l'application d'un patron de Produit. Celui-ci est défini dans le langage de patrons peuplé grâce à l'application des méta-patrons de Produit. Ils permettent de construire des Patrons pertinents adaptés à l'application en cours. Cela permettra à l'ingénieur de méthodes de travailler avec un plus grand nombre de concepts dans le but de construire une application plus complète et plus cohérente.

De plus, si l'ingénieur de méthodes applique un patron de Produit alors que la méthode modifiée contient une partie Processus, il devient préférable de modifier également cette partie pour préserver la cohérence de la méthode. Nous définissons donc également des méta-patrons permettant de construire des patrons de Processus appelés méta-patrons de Processus.

Prenons l'exemple suivant d'extension du Produit d'une méthode. La **Figure 4** illustre les instanciations successives permettant d'obtenir, à partir d'un méta-patron, un patron pour l'intégration d'un élément spécifique, puis un patron instancié permettant d'étendre une méthode particulière. Ici, le concept à intégrer est celui de *Classe Calendrier* et la méthode à modifier est O*.

- La première étape représente l'instanciation du méta-patron Spécialisation pour le concept de *Classe Calendrier*.

- La seconde étape représente l'instanciation du patron (obtenu à l'étape précédente) à la méthode O*.

Nous pouvons constater ici que le concept de *Classe Calendrier* est inséré en tant que sous-type du concept de *Classe* et que le concept déjà existant de *Granule* est supprimé.

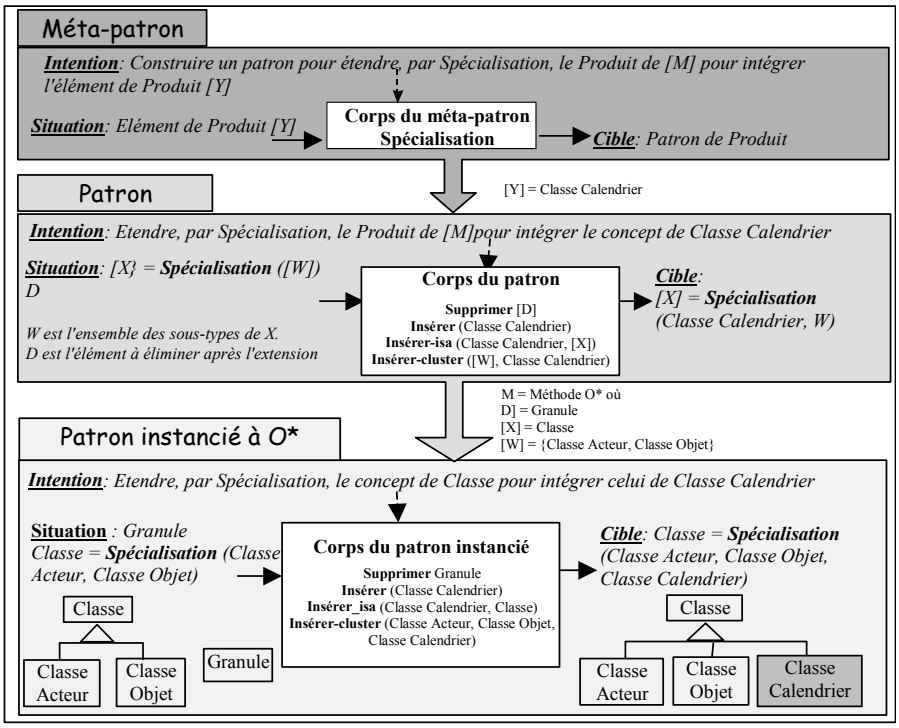

Figure 4. *Exemple d'utilisation d'un méta-patron*

4 Exemple d'application aux extensions de méthodes orientées objets

Pour restreindre encore un peu le champ d'application de cet article, nous allons nous concentrer ici surtout sur le langage de patrons utilisé lors de l'extension de méthodes orientées objets. Ces patrons sont obtenus à l'aide de méta-patrons définis grâce à un ensemble de stratégies qui leur sont spécifiques. Nous les appellerons des « stratégies EMOO » (stratégies d'extension de méthodes orientées objets).

Nous avons vu qu'il y avait deux types de méta-patrons: les méta-patrons de Produit et les méta-patrons de Processus. En effet, la manière d'intégrer un nouveau concept dans le modèle d'une méthode et d'intégrer sa construction dans son arbre de processus n'est pas la même. Les stratégies seront donc également différentes.

4.1 Stratégies EMOO de Produit

L'intégration d'un nouveau concept dans le modèle de Produit d'une méthode peut se faire de deux manières différentes : soit le concept à intégrer est un nouvel élément, soit il est déjà représenté dans la méthode mais de façon incomplète ou incorrecte.

4.1.1 Introduction d'un nouveau concept dans la méthode

L'introduction d'un nouveau concept doit être suivi de sa connexion avec le reste du modèle. En effet, un concept n'est important que si il est relié aux autres concepts. Ces greffes sont réalisées en instanciant les différents types de liens définis dans le méta-modèle de Produit de la méthode. La Figure 5 montre le méta-modèle générique de plusieurs méthodes objets.

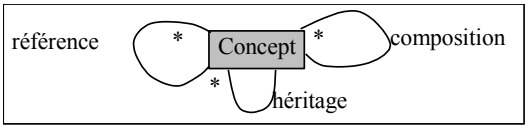

Figure 5. *Méta-modèle générique de plusieurs méthodes objets*

Un nouveau concept peut donc être relié au reste du modèle par l'utilisation des liens d'héritage, de composition ou de référence.

4.1.1.1 Lien d'héritage

Si l'ingénieur de méthode veut utiliser le lien d'héritage pour relier son concept au modèle, deux cas s'offrent à lui. Dans le premier cas, le concept est rattaché en tant que spécialisation d'un concept déjà existant dans la méthode. Dans le second cas, le concept est inséré en tant que spécialisation d'un concept non existant. Cela veut simplement dire que la méthode d'origine contient un concept très proche sémantiquement du concept que l'on souhaite intégrer. On insère donc un autre concept (le super-type) qui permettra d'être spécialisé, soit en notre nouveau concept, soit en ce concept existant déjà dans la méthode.

Par exemple, l'intégration du concept d'*Evénement Temporel* dans la méthode OMT se fera par spécialisation puisque celle-ci contient déjà le concept d'*Evénement* qui peut être considéré comme un super-type de l'élément à intégrer. Par contre, si l'on souhaite y intégrer le concept de *Classe Acteur*, on ne pourra le faire qu'en ajoutant un concept généralisé *Classe* qui aura deux sous-types: la *Classe Acteur* que l'on souhaite insérer et le concept existant de *Classe Objet*.

Insérer un concept par le lien d'héritage peut donc être vue de deux manières différentes: lien d'héritage (le nouveau concept est intégré en tant que spécialisation d'un élément déjà existant) ou lien de généralisation (le nouveau concept et un concept existant deviennent les sous-types d'un nouvel autre concept). On a donc deux stratégies EMOO : la Spécialisation et la Généralisation.

4.1.1.2 Lien de composition

L'utilisation du lien de composition peut également être vue de manière double. En effet, le nouveau concept est intégré soit comme un composé d'un élément existant, soit comme un composant.

Prenons de nouveau l'exemple de la méthode OMT. Y insérer le concept de *Contrainte d'objet* peut se faire en considérant ce concept comme un composant de celui de *Classe Objet*. De manière inverse, l'intégration du concept de *Classe Acteur* peut se faire en considérant le concept d'*Evénement Externe* déjà existant comme un composé du nouvel élément.

Intégrer un concept avec ce lien peut donc être vu comme l'utilisation d'un lien de composition (le nouveau concept est intégré comme composé) ou de décomposition (le nouveau concept est intégré comme composant). Chacune de ces deux techniques peut être considérée comme une stratégie EMOO : la Composition et la Décomposition.

4.1.1.3 Lien de référence

Le troisième lien présent dans le méta-modèle est celui de référence. Dans le cas de l'utilisation de ce lien, seul un cas est à considérer. En effet, le lien de référence est présent à la fois du côté de l'élément référençant et du côté de l'élément référencé.

Par exemple, lier le concept d'*Agent* dans une méthode contenant déjà celui d'*Action* peut être fait par le lien de référence (une *Action* est faite par un *Agent* qui exécute une ou plusieurs *Actions*).

Insérer un élément avec un lien de référence peut être considéré comme l'utilisation de la stratégie EMOO : la Référence.

4.1.2 Introduction d'un concept déjà représenté mais de façon incomplète ou incorrecte

L'ingénieur de méthodes peut également souhaiter remplacer un concept existant car il ne correspond plus à ces besoins et il souhaite en changer la description.

Prenons l'exemple d'une méthode contenant le concept de *Granule*. Si l'on souhaite intégrer le concept de *Classe Calendrier*, l'ancien concept devient incomplet par rapport à nos besoins et il est donc nécessaire de le remplacer.

Etendre une méthode avec cette technique peut être vue comme l'application de la stratégie EMOO suivante : le Remplacement.

En résumé, nous avons l'ensemble suivant de stratégies EMOO de Produit: Héritage, Généralisation, Composition, Décomposition, Référence et Remplacement. Chacune de ces stratégie sera définie dans un méta-patron spécifique.

4.2 Stratégies EMOO de Processus

Etendre la partie Processus d'une méthode pour y intégrer la construction d'un nouvel élément revient à greffer une nouvelle branche dans l'arbre de processus de

la partie Démarche[1] de la méthode. Deux manières de greffer sont possibles : soit la branche correspondant à la construction initiale du schéma est laissée intacte et la greffe est faite de manière contrôlée, soit l'arbre de processus est modifié de façon à ce que l'extension soit faite de manière transparente pour l'utilisateur.

4.2.1 Intégration contrôlée

Dans ce type d'extension, les processus permettant d'étendre la méthode sont greffés de manière à ne pas modifier l'arbre de processus initial. L'extension ne prendra donc place qu'une fois tous les éléments d'origine définis. L'arbre de processus conserve le processus initial (« Construire X ») mais est incrémenté d'un nouveau processus d'extension (« Etendre X »). Ces deux branches étant exécutées en séquence, nous appelons cette technique une extension séquentielle (**Figure 6**).

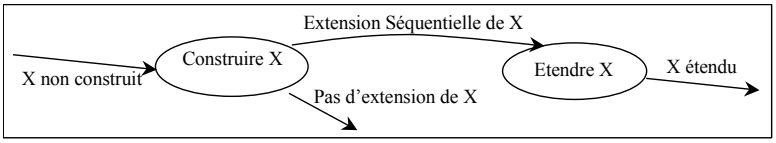

Figure 6. *Principe de l'extension séquentielle*

Ce principe représente la séquence entre le processus de construction de X et son extension. Cependant, le terme X peut avoir différentes significations selon la granularité utilisée. A la granularité la plus basse, X représente un simple élément à étendre, i.e. la construction du concept de *Classe* ou de *Granule*, etc. Au contraire, à la granularité la plus haute, X représente la construction entière du schéma de la méthode. Ces deux cas sont illustrés dans la figure ci-dessous.

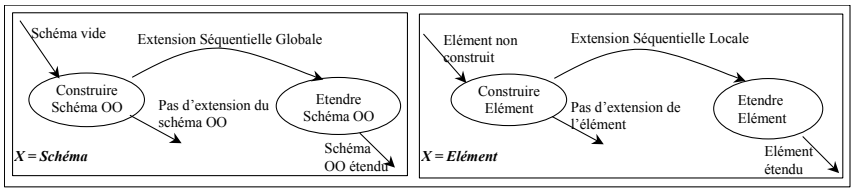

Figure 7. *Principes des extensions globales et locales*

Ces deux possibilités représentent deux stratégies EMOO différentes: l'Extension séquentielle Locale et l'Extension Séquentielle Globale.

[1] Le formalisme utilisé pour représenter la partie Démarche d'une méthode est celle du projet Esprit « NATURE » décrit en [ROL 95][ROL 96][JAR 99].

4.2.2 Intégration transparente

Ce type d'extension ne laisse pas le processus initial intact. Les nouveaux processus sont totalement intégrés dans l'arbre de processus, l'extension n'apparaît donc plus comme une étape artificielle mais comme une étape plus complète (**Figure 8**).

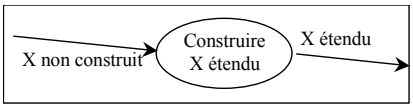

Figure 8. *Principe de l'extension intégrée*

Ce type d'extension représente la stratégie EMOO d'Extension Intégrée.

La partie Processus d'une méthode peut donc être étendue par l'application de l'ensemble des stratégies suivantes : Extension séquentielle globale, Extension séquentielle locale et Extension intégrée. Chacune de ces stratégies sera intégrée dans un méta-patron de Processus spécifique.

4.3 Stratégies EMOO de Processus Versus *Stratégies EMOO de Produit*

Notons que l'utilisation d'une stratégie de Produit va avoir un impact sur l'utilisation des stratégies de Processus. Il y a trois types de répercutions possibles.

- Si la stratégie de Produit est *Spécialisation* ou *Généralisation*, il y aura alors une modification des alternatives (choix) de l'arbre de processus. Celui-ci aura alors une augmentation de ces types spécialisés (en cas de Spécialisation) ou une étape supplémentaire (en cas de Généralisation).

- De façon similaire, l'utilisation d'une *Composition* ou *Décomposition* amènera une modification de séquence (plan).

- La stratégie de *Remplacement* entraînera également un remplacement d'une des branche de l'arbre.

Nous pouvons également différencier deux cas différents pour l'*Extension séquentielle globale* : soit l'exécution de cette stratégie est la première (« Première greffe »), soit ce n'est pas le cas (« Greffe Additionnelle »). Dans le premier cas, l'ingénieur de méthodes doit complètement réévaluer le graphe de précédence associé à l'arbre de processus pour conserver la cohérence de la méthode.

La dernière différence se situe lors de l'utilisation des stratégies de Produit *Spécialisation* ou *Généralisation*. L'extension sera en effet différente selon la spécialisation utilisée. Une spécialisation « par type » entraîne une complète différentiation de la description des sous-types. Au contraire, une spécialisation « par état » entraîne une description commune et un choix du type de concept pour connaître le sous-type à décrire.

Nous avons donc défini un ensemble de stratégies de Produit et un ensemble de stratégies de Processus. Cependant, selon la stratégie de Produit utilisée, l'utilisation de la stratégie de Processus sera différente. La **Figure 9** illustre les différentes possibilités de combinaisons.

Stratégie EMOO de Processus / **Stratégie EMOO de Produit**	**EXTENSION SEQUENTIELLE GLOBALE**		**EXTENSION SEQUENTIELLE LOCALE**	**EXTENSION INTEGREE**	
	Première greffe	**Greffe additionnelle**		**Elément ayant une spécialisation par type**	**Elément ayant une spécialisation par état**
SPECIALISATION	SPECIALISATION GLOBALE (Première greffe)	SPECIALISATION GLOBALE (Greffe Additionnelle)	SPECIALISATION LOCALE	SPECIALISATION INTEGREE (par type)	SPECIALISATION INTEGREE (par état)
GENERALISATION	GENERALISATION GLOBALE (Première greffe)	GENERALISATION GLOBALE (Greffe Additionnelle)	GENERALISATION LOCALE	GENERALISATION INTEGREE (par type)	GENERALISATION INTEGREE (par état)
COMPOSITION	COMPOSITION GLOBALE (Première greffe)	COMPOSITION GLOBALE (Greffe Additionnelle)	COMPOSITION LOCALE	COMPOSITION INTEGREE	
DECOMPOSITION	DECOMPOSITION GLOBALE (Première greffe)	DECOMPOSITION GLOBALE (Greffe Additionnelle)	DECOMPOSITION LOCALE	DECOMPOSITION INTEGREE	
REFERENCE	REFERENCE GLOBALE (Première greffe)	REFERENCE GLOBALE (Greffe Additionnelle)	REFERENCE LOCALE	REFERENCE INTEGREE	
REMPLACEMENT				REMPLACEMENT INTEGRE	

Figure 9. *Stratégies EMOO de Processus Versus Stratégies EMOO de Produit*

Il est à noter que l'on n'autorise pas l'utilisation des stratégies d'extension séquentielle si la stratégie de Produit utilisée était le *Remplacement*. En effet, ce serait une perte de temps que de décrire un élément pour ensuite le remplacer.

Nous obtenons donc 23 stratégies EMOO de Processus différentes.

5 Conclusion et Perspectives

Nous avons proposé ici une technique de méta-patrons pour construire des patrons. Cette technique a été définie pour résoudre le problème de complétude posé lors de la construction des cartes de processus.

Nous avons donc défini comment construire des stratégies encapsulées dans des méta-patrons spécifiques. Ceux-ci entraînent la création des patrons présents dans les cartes. En conséquence, toutes les sections de la cartes sont définies à l'aide d'un patron, la carte est complète et l'ingénieur de méthodes ne peut aboutir à une impasse lors de sa navigation.

Nous avons illustré notre propos par la définition de méta-patrons et de stratégies spécifiques au domaine des extensions de méthodes orientées objets.

Les problèmes suivants restent à résoudre :

- Un plus grand champ d'application : l'application de cette technique à d'autres domaines que les extensions de méthodes et à d'autres types de méthodes que les méthodes orientées objets.

- L'aide supportée par des logiciels: l'un permettrait de construire la carte (et les patrons associés) avec la technique décrite dans cet article et l'autre aiderait à son exécution.

6 Références

[BEC 97] Beck, K., *Smalltalk, Best Practice Patterns. Volume 1, Coding,* Prentice Hall, Englewood Cliffs, NJ, 1997

[BEN 99] Benjamen, A., *Une approche multi-démarches pour la modélisation des démarches méthodologiques,* PhD thesis, University of Paris 1, Paris, 1999

[BUS 96] Buschmann, F., Meunier, R., Rohnert, et al., *Pattern-Oriented Software Architecture - A System of Patterns,* John Wiley, 1996

[COA 92] Coad, P., *Object-Oriented Patterns,* Communications of the ACM, Vol. 35, No. 9, 1992, pp 152-159

[COP 95] Coplien, J.O., and Schmidt, D.O. (ed.), *Pattern Languages of Program Design,* Addison-Wesley, Readind, MA, 1995

[DEN 01A] Deneckere, R., *Approche d'extension de méthodes fondée sur l'utilisation de composants génériques,* PhD thesis, University of Paris1-Sorbonne, 2001

[DEN 01B] Deneckere, R., Souveyet, C., *Organising and Selecting Patterns in Pattern Languages with Process Maps,* Proceedings of OOIS'2001 Conference, Springer-Verlag, Calgary (Canada), 2001

[DEN 98] Deneckere, R., Souveyet, C., *Patterns for extending an OO model with temporal features*. Proceedings of OOIS'98 conference. Springer-Verlag, Paris (France), 1998

[FOW 97] Fowler, M., *Analysis Patterns: Reusable Object Models,* Addison-Wesley, 1997

[GAM 94] Gamma, E., Helm, R., Johnson, R., et al., *Design Patterns: Elements of Reusable Object-Oriented Software,* Addison Wesley, 1994

[HAY 96] Hay, D., *Data Model Patterns: Conventions of Thought,* Dorset House, NY, 1996

[JAR 99] Jarke, M., Rolland, C., Sutcliffe, A., Dömges, R. (Hsrg.), *The NATURE of Requirements Engineering,* Shaker Verlag, Aachen, 1999

[ROL 95] Rolland, C., Souveyet, C., Moreno, M., *An Approach for Defining Ways-Of-Working*. Information Systems, Vol 20, No4, pp337-359, 1995

[ROL 96] Rolland, C., Plihon, V., *Using generic chunks to generate process models fragments,* Proceedings of the 2nd IEEE International Conference on Requirements Engineering, ICRE, ICRE'96, Colorado Spring, 1996

[ROL 98] Rolland, C., Plihon, V., Ralyté, J., *Specifying the reuse context of scenario method chunks,* Proceedings of the conference CAISE'98, Springer-Verlag, Pisa Italy, 1998

[ROL 99] Rolland, C., Prakash, N., Benjamen, A., *A multi-model view of process modelling,* Requirements Engineering Journal, p. 169-187, 1999