Famille de méthodes : la flexibilité au cœur du processus de construction de méthode

Deneckère, Rébecca; Kornyshova, Elena; Ralyte, Jolita

How to cite

DENECKÈRE, Rébecca, KORNYSHOVA, Elena, RALYTE, Jolita. Famille de méthodes : la flexibilité au cœur du processus de construction de méthode. In: Ingénierie des systèmes d'information, 2014, vol. 19, n° 1, p. 67–95. doi: 10.3166/isi.19.1.67-95

This publication URL: https://archive-ouverte.unige.ch//unige:39679
Publication DOI: 10.3166/isi.19.1.67-95

Famille de méthodes : la flexibilité au cœur du processus de construction de méthode

Rébecca Deneckère[1], Elena Kornyshova[2], Jolita Ralyté[3]

1. Centre de Recherche en Informatique, Université Paris 1 Panthéon-Sorbonne
90 Rue de Tolbiac, 75013 Paris, France
rebecca.deneckere@univ-paris1.fr

2. CEDRIC-CNAM
2, rue Conté, 75003, Paris, France
elena.kornyshova@cnam.fr

3. Institut de Sciences des Services, Université de Genève
CUI, Battelle bâtiment A, 7 Route de Drize, 1227 Carouge, Suisse
jolita.ralyte@unige.ch

RESUME. *Les difficultés d'utilisation de l'Ingénierie des Méthodes Situationnelles (IMS) dans l'industrie sont dues essentiellement à la complexité des solutions proposées et l'IMS cherche de nouvelles voies pour faciliter la construction des méthodes spécifiques au contexte. Une piste possible vient de l'ingénierie des lignes de produits logiciel (LdP) qui développe des solutions adaptables au contexte en réutilisant la connaissance existante. Pour utiliser les concepts de réutilisation, d'adaptabilité et de variabilité dans l'IMS, nous proposons la notion de famille de méthodes qui permet d'organiser de multiples composants de méthode de diverses façons et de configurer une ligne de méthode pour une situation donnée. Nous détaillons ici le processus de construction de ces familles. La variabilité étant la notion sous-jacente à la notion de famille, notre proposition inclut l'analyse de variabilité inspirée des LdP. Nous illustrons notre approche par la construction d'une famille de méthodes spécialisée dans le lancement de projets agiles.*

ABSTRACT. *Situational Method Engineering (SME) was missing its rendezvous with industry mainly because of the complexity of proposed solutions and the need for advanced method engineering skills to apply them in practice. Today, SME is looking for new ways to facilitate situation-specific method construction. The inspiration comes from Software Product Line Engineering (SPLE), which is developing solutions that can be easily adapted to a specific context and by reusing existing knowledge. In order to deal with method reuse, adaptability and variability in SME, we suggest the notion of method family that allows to organize multiple method components in different ways and to configure a particular method line for a given situation. In this paper, we specify the process guiding the construction of method families. As variability is underlying the method family notion, our proposal includes variability analysis inspired from SPLE. We illustrate our approach with the construction of a method family dealing with the agile projects launch.*

MOTS-CLES : *Processus, Famille de méthodes, Flexibilité, Variabilité, Composant de méthode*

KEYWORDS: *Process, Method Family, Flexibility, Variability, Method Component*

1. Introduction

Une méthode de développement de systèmes d'information (SI) est un ensemble d'idées, d'approches, de techniques et d'outils dont un analyste se sert pour transcrire les besoins d'une organisation en un SI approprié. Cependant, il est maintenant clair qu'il n'existe pas de méthode universelle pouvant s'appliquer à toutes les situations. Le domaine de l'ingénierie des méthodes situationnelles (IMS), depuis les années 90 (Kumar, 1992), s'est attaqué à cette problématique pour offrir des techniques permettant la création ou l'adaptation de méthodes selon la situation du projet en cours (la situation d'ingénierie de chaque projet est différente et exige un support méthodologique différent). À cette fin, l'IMS promeut la construction de méthodes spécifiques aux contextes en réutilisant des parties de méthodes existantes conçues comme des composants autonomes et stocké dans des bases de méthodes. En effet, dans toutes les approches proposées, les méthodes sont décomposées en morceaux modulaires pouvant ensuite s'assembler ou s'intégrer les uns avec les autres. Cette notion fondamentale permet d'optimiser la création de méthodes en réutilisant des différents morceaux, appelés composants. (Ralyté et Rolland, 2001) décrivent le processus d'IMS en deux étapes : (1) la réingénierie des méthodes existantes en composants réutilisables qui seront stockés dans une base de méthodes, (2) la construction de méthodes situationnelles par composition de composants adéquates, ce qui inclut la spécification de la situation du projet et de ses besoins, la recherche des composants qui correspondent le mieux aux spécificités du projet et la construction d'une nouvelle méthode avec les composants sélectionnés (Mirbel et Ralyté, 2006). La notion de composant de méthode a été introduite par (Harmsen, 2007) par analogie avec la notion de composants logiciels (la construction modulaire d'une méthode à base de composants de méthodes stockés dans une base de composants est similaire à la notion de construction de logiciels orientée-composants logiciels).

Cependant, malgré la diversité des différentes approches et des techniques que l'IMS propose (par assemblage (Brinkkemper *et al*., 1998 ; Firesmith, 2002 ; Ralyté *et al*., 2001b), basée sur la configuration (Karlsson, 2004), basée sur l'adaptation de processus (Rossi *et al*., 2004) ou encore l'orientée service (Guzelian *et al*., 2007)), leur usage n'est pas très répandu dans les entreprises car leur mise en œuvre semble difficile. Les entreprises tardent à adopter les approches et techniques proposées bien que conscientes de l'importance du rôle que les méthodes jouent dans leurs activités d'ingénierie. Nous pouvons imaginer les causes probables de cette réticence à l'utilisation des approches IMS. Tout d'abord, les approches ne sont pas interopérables car toutes les approches sont extrêmement liées à la structure même de leurs composants, ce qui fait que les techniques développées dans une approche ne sont pas utilisables dans les autres. Ensuite, les techniques pour retrouver un composant sont assez complexes et peuvent nécessiter d'effectuer des recherches dans plusieurs bases de méthodes différentes. De plus, comme la plupart des approches ne proposent pas de processus organisationnel pour la gestion des composants, cela complique encore le processus d'exécution de la construction de la méthode. Qui dit méthode situationnelle dit situation, or la prise en compte de la caractérisation d'un projet n'est pas toujours très formalisée, de même que la caractérisation d'un composant et la comparaison possible entre les deux. Mais le frein majeur que rencontre la popularisation des méthodes en entreprise est que celles-ci

soutiennent que les processus IMS consomment trop de temps et sont trop chers en termes de connaissances et de ressources pour les mettre en œuvre dans des projets d'entreprise. Ils exigent des approches et des outils simples et agiles, faciles à mettre en œuvre et ne requérant pas de vaste connaissance en ingénierie des méthodes.

Le concept de famille de méthodes (Asadi *et al.*, 2011 ; Kornyshova *et al.*, 2011b ; Rolland, 2004) apparaît comme une manière de fournir des processus de construction de méthodes aux entreprises de développement de systèmes d'information, chacune consacrée à un certain domaine d'application particulier et/ou un but d'ingénierie spécifique, pouvant être facilement configuré dans des lignes de méthodes spécifiques à un projet selon les caractéristiques et les exigences de ce projet. L'inspiration initiale de ce concept vient de l'ingénierie des lignes de produits logiciel (LdP) et en particulier du concept de Famille de Produit (Clements, 2001) (Weiss, 1999). L'ingénierie des LdP est un paradigme permettant de développer des applications (systèmes logiciels et produits logiciels) avec de la personnalisation massive. Il est basé sur l'identification et la spécification des similitudes et différences entre applications d'une ligne de produits (Pohl *et al*., 2011).

La spécification d'un produit logiciel prend toutes les exigences communes de la ligne de produits et quelques exigences spécifiques du logiciel à développer. De même dans l'IMS, nous considérons que la notion de famille de méthodes simplifierait la tâche d'ingénierie de méthode pendant l'exécution du projet en la réduisant à une configuration de la famille de méthodes, c'est-à-dire la sélection de composants de méthode communs et spécifiques de la famille de méthodes. Bien sûr, ceci présume que la famille de méthodes existe avant le démarrage du projet. À cette fin, nous proposons une approche d'ingénierie basée sur les familles de méthodes et inspirée des LdP (la variabilité) et de l'IMS (la réutilisation et la composition de composants de méthode). L'IMS est en général basée sur la décomposition de méthodes existantes en composants de méthode et leur réutilisation pour la construction de nouvelles méthodes (Ralyté et Rolland, 2001). Notre proposition tente de répondre à la problématique des entreprises en envisageant la construction de méthodes situationnelles sous un autre angle de vue, celui de la configuration. D'après les principes de l'IMS, notre approche d'ingénierie par famille de méthodes réutilise des méthodes et/ou des composants de méthode existants, dans le but de fournir des familles de méthodes satisfaisant des organisations et/ou des projets de développement divers. La tâche de l'ingénieur de méthode sera réduite à une sélection de ligne de méthode simple de la famille de méthodes (processus appelé configuration de ligne de méthode), ce qui limitera également le risque d'erreurs et réduira le temps nécessaire pour produire des méthodes spécifiques à un projet.

Notre notion de famille de méthode a été introduite dans (Kornyshova et al, 2011). Nous proposons ici (a) un méta-modèle plus complet (prenant en compte la description plus détaillée des concepts de composants de méthodes et d'interface de famille de méthodes), (b) le processus de construction d'une famille et (c) la représentation d'une famille (textuellement et graphiquement). Après un rapide tour d'horizon des travaux connexes (partie 2), nous entrons dans le cœur de notre proposition (partie 3) en expliquant le concept de famille de méthodes. La deuxième partie de notre proposition (l'approche de construction et de configuration de familles de méthodes) est décrite (partie 4) et illustrée (partie 5).

2. Travaux connexes au concept de Famille de méthodes

Le concept de famille de méthodes a été présenté par Rolland (2007) comme une nouvelle approche d'IMS basée sur l'organisation d'un ensemble de composants de méthode dans une multi-méthode configurable. La variabilité étant un concept central pour permettre la réutilisabilité et l'adaptabilité de méthodes aux situations spécifiques, une famille de méthodes doit considérer diverses situations d'ingénierie des systèmes et capturer de multiples techniques pour leur faire face. L'adaptation de la famille de méthodes à un projet particulier consiste dans la sélection d'une ligne de méthode selon les caractéristiques et les exigences du projet.

Asadi *et al*. (2011) ont déjà exploré le concept de famille de méthodes et ont proposé une approche pour modéliser des familles d'architectures orientées méthodes. Leur approche est basée sur la notion de service de méthode utilisé pour construire des familles de méthodes. Des techniques de modélisation des variantes sont appliquées pour gérer la variabilité en ingénierie de méthodes et des techniques de web sémantique sont utilisées pour permettre la découverte de services méthode.

La notion de famille de méthodes est l'une des dernières propositions de l'IMS. Il existe déjà différentes approches d'IMS qui ont été proposées cette dernière quinzaine d'années. Historiquement, la première approche proposée est celle des fragments de méthodes (Brinkkemper, 1996 ; Harmsen *et al.,* 1994), fragments de produit ou de processus, stockés dans une base puis réassemblés selon un ensemble de règles. L'approche par morceau (chunk) de méthode (Ralyté, 2001 ; Mirbel et Ralyté, 2006) ajoute une notion intentionnelle aux composants pour permettre de les retrouver en les comparant avec les besoins du projet. L'approche OPEN utilise un méta-modèle pour générer les composants stockés dans la base appelés fragment OPF (Firesmith, 2002). L'approche de service méthode (Guzélian *et al*., 2007 ; Deneckère *et al.,* 2008) a développé un autre type de composant basé sur la notion de service et de composition de services. L'approche par extension (Deneckère, 2001) guide l'utilisateur en lui proposant des patrons d'extension applicables dans diverses situations et en argumentant sur leur choix. Il s'avère cependant que ces approches sont difficilement applicables en pratique. L'ingénieur de méthodes doit déjà avoir un haut niveau d'expertise et le coût de la création est finalement considéré comme trop important par les entreprises.

Pour contrer ce problème, certaines approches d'IMS ont pris en compte le concept de configuration. Karlsson *et al*. (2004) propose une approche basée sur les paquetages de configuration. Une méthode est décomposée en composants assemblés ensuite par paquetages selon certaines caractéristiques. Ces paquetages sont réutilisés et améliorés selon le projet pour obtenir une nouvelle méthode situationnelle. Cette approche optimise la réutilisation puisqu'il n'est plus nécessaire de réassembler systématiquement une méthode à chaque nouveau projet puisque l'expérience est capitalisée (Wistrand *et al.*, 2004), mais elle ne prend cependant en compte qu'une seule méthode initiale pour en permettre la configuration. Mirbel (2004) s'est aussi intéressée au problème de la configuration de méthodes en proposant une base de fragments réutilisables associés à des facteurs organisationnels, techniques et humains. Ce travail introduit la notion de cadre de réutilisation (reuse frame), structure ontologique permettant aux membres d'un projet de spécifier le contexte. Ces facteurs sont ensuite utilisés pour définir la méthode

la plus adaptée au projet et au contexte. Cette technique peut être utilisée combinée avec les autres approches de construction de méthode. C'est ce qui a été effectué dans (Mirbel et Ralyté, 2006) pour proposer une nouvelle technique permettant de construire tout d'abord une méthode par assemblage et adaptée au projet en cours puis de choisir ensuite les parties de cette méthode en fonction du contexte de l'utilisateur de la méthode. Cependant, cette approche se focalise sur la construction d'une méthode particulière adaptable sans chercher à avoir la variabilité que peut offrir une famille de méthodes.

Le concept sous-jacent à celui de la configuration est celui de la variabilité. Celle-ci s'est avérée être un concept central dans différents domaines d'ingénierie comme la fabrication, le développement de logiciels, etc. afin de développer des solutions pouvant être facilement adaptables à différents contextes organisationnels et/ou à différents types de clients pour un meilleur coût. La notion de variabilité du logiciel est donc définie comme la capacité d'un système logiciel à être modifié, adapté ou configuré dans un certain contexte (Van Gurp, 2000). La notion de lignes de produits est apparue suite à la prise en compte de cette variabilité croissante dans le domaine du génie logiciel. L'ingénierie des lignes de produits correspond au développement d'applications logicielles utilisant de la personnalisation de masse, les points communs et différences des applications de la ligne de produit devant être modélisés de manière générique (Pohl *et al.*, 2005). Un produit d'une famille est défini en sélectionnant ou désélectionnant des caractéristiques (features) selon les préférences de l'utilisateur. De manière similaire, une famille de processus est configurable pour être personnalisée en lignes de processus, eux-mêmes adaptables à un projet (Rolland *et al.*, 2007).

Cette adaptation de plus en plus grande à des particularités de situation est également une conséquence des travaux sur la sensibilité au contexte. Le terme de sensibilité au contexte vient des travaux sur l'informatique pervasive (ou *informatique ubiquitaire)* où les systèmes utilisent les modifications de l'environnement. Ce terme a également été appliqué à la théorie des entreprises dans le domaine de la gestion des processus métiers (Rosemann *et al.*, 2006), la réingénierie des processus métiers (Bessai *et al.*, 2008), l'informatique (Bradley *et al.*, 2005), la sélection de services (Kirsh Pinheiro *et al.*, 2008) et la prise de décision en situation militaire (Rosen *et al.*, 2008 ; Drury *et al.*, 2008). Dans ces derniers cas, le modèle de contexte est perçu comme une manière d'analyser une *situation* donnée afin de guider une exécution de tâches. Ainsi, les modèles de contexte sont principalement utilisés pour résoudre le problème de manque de flexibilité et d'adaptabilité au sein des processus.

3 Définition d'une famille de méthodes

Nous définissons une famille de méthodes comme une composition de composants alternatifs ou complémentaires, venant de méthodes diverses et variées mais applicables au même domaine d'application et ayant des objectifs similaires. Une famille de méthodes peut être considérée comme une multi-méthode (multi-processus et multi-produits) dédiée à un domaine ou un but d'ingénierie particulier et cherchant à satisfaire un ensemble de projets de ce domaine. Une famille est multi-processus puisqu'elle offre un ensemble de démarches possibles pour atteindre le même objectif,

et elle est multi-produits puisque les composants peuvent être combinés de différentes manières. L'adaptation d'une famille de méthodes à un projet, (configuration), se fait en sélectionnant une ligne de méthode à l'intérieur de la famille. Une ligne de méthode est composée d'un sous-ensemble des composants de la famille, sélectionnés pour un contexte donné (un projet particulier). Cette démarche permet de simplifier la construction de la méthode en réduisant la complexité de la recherche de composants.

3.1. Concepts principaux

La notion de famille de méthodes peut être décrite avec le méta-modèle de la figure 1 (selon le formalisme UML). Ce méta-modèle a été introduit dans (Kornyshova et al, 2011) de manière plus succincte. La figure 1 détaille de manière plus complète certains concepts (e.g. composants de méthodes, interface de famille de méthodes).

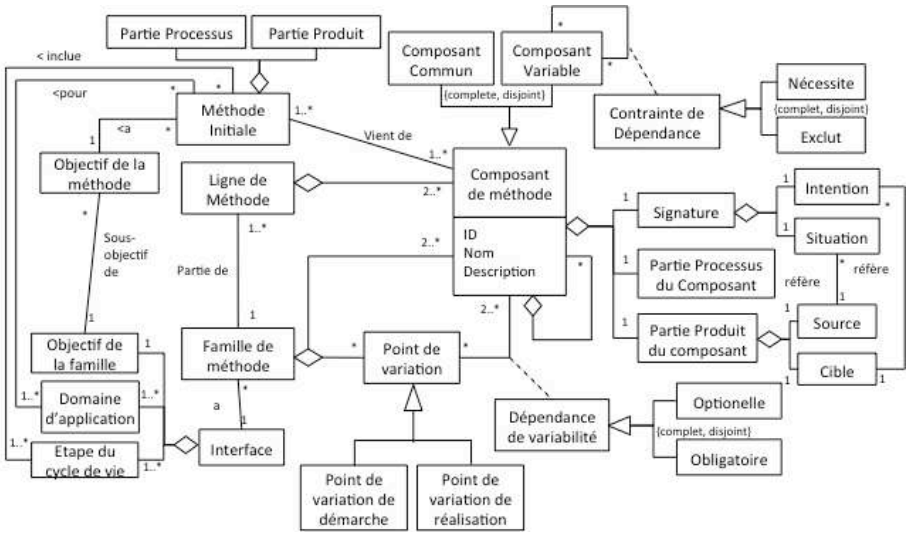


Figure 1. Méta-modèle de famille de méthodes.

Une famille de méthodes est définie par son *interface*, identifiant l'objectif principal de la famille, son domaine d'application et sa portée fonctionnelle. Elle peut inclure des composants venant de diverses méthodes (les méthodes initiales) qui correspondent à cette interface. Une analyse informelle des différents éléments permet d'identifier si une méthode initiale correspond à l'interface d'une famille de méthode (si son objectif est un sous-objectif de celui de la famille, si elle s'applique dans le même domaine d'application ou une sous-partie du même domaine et si l'étape du cycle de vie pris en compte fait partie de la portée fonctionnelle de la famille).

Un *composant de méthode* est une partie d'une méthode suffisamment autonome pour pouvoir être appliquée de manière indépendante mais qui peut également être

combinée avec d'autres composants. Il existe beaucoup de types de composants dans le domaine de l'IMS. Le concept de composant de méthode a ici la même sémantique que dans ces approches de construction de méthodes situationnelles classiques, il s'agit d'un bloc de construction réutilisable, sous-ensemble d'une méthode, qui peut être composé d'un ensemble d'autres composants. Cependant, les différentes approches existantes se focalisent essentiellement sur la description du type de composant utilisé, alors que notre proposition, au contraire, considère que le composant n'est qu'une boite noire représentant une partie de méthode, indépendamment de sa structure. Cette distinction permet donc de combiner des composants venant de méthodes différentes, indépendamment de leur structure, ce qui n'est pas possible avec les approches situationnelles classiques (voir (Henderson-Sellers *et al*., 2008) pour plus de détails).

Un composant est défini par sa *signature* qui spécifie la situation dans laquelle il peut être appliqué et le but qu'il doit satisfaire (*l'intention*). Son contenu est formalisé en termes de produit et processus. Un composant peut être composé d'un ensemble de composants plus petits (différentes manières de réaliser le même but ou ensemble d'étapes considérées séparément comme des composants indépendants). Un composant est également caractérisé par son *contexte* de réutilisation permettant d'identifier les éléments de contexte favorables à son utilisation (Kornyshova et al, 2011a).

L'organisation des composants dans la famille consiste à déterminer ce qui est similaire et ce qui est différent dans les méthodes composant la famille. A l'intérieur d'une famille de méthodes, les composants peuvent donc être considérés comme variables ou communs. Les composants communs correspondent à des composants qui seront présents dans toutes les situations du domaine. Les composants variables, au contraire, sont des composants qui ne seront présents que dans certaines situations. Ces situations engendrant une certaine variabilité sont appelées des *points de variation*. Ceux-ci peuvent être de deux types : de démarche ou de réutilisation. Les points de variation de démarche concerne la variabilité attachée à une situation donnée et permet d'identifier les composants qu'il est possible d'utiliser dans ce contexte précis – les points de variation de réutilisation sont plus axés sur la situation cible et l'identification des composants pouvant atteindre cet objectif. La relation entre le point de variation et le composant définit la *dépendance de variabilité*, pouvant être obligatoire ou optionnelle. Cette représentation de la variabilité est inspirée de (Pohl et *al.*, 2005) où l'on trouve la définition de la variabilité des lignes de produits logiciels.

Un composant de méthode est considéré comme un composant *commun* s'il apparaît dans toutes les méthodes initiales composant la famille. Il sera alors considéré comme obligatoire et devra être présent dans toute ligne de méthode générée à partir de la famille. Un composant de méthode *variable* est optionnel – il peut exister d'autres composants permettant d'atteindre le même but mais avec des techniques différentes, ou la famille permet d'utiliser d'autres composants pour réaliser l'objectif principal.

Les points de variation sont utilisés pour spécifier le contexte d'utilisation des composants lorsqu'une alternative est possible. Les composants variables associés à un point de variation peuvent appartenir à plusieurs méthodes différentes ou encore à la même méthode initiale, incluant cette notion d'alternative, de variabilité.

La distinction entre composants variables et communs permet d'évaluer le degré de variabilité d'une famille et fournit les outils nécessaires à la configuration des lignes de méthodes puisque celles-ci sont composées des composants communs et d'un certain nombre de composants variables, choisis pour un projet donné selon le contexte.

3.2. Spécification de la variabilité dans les familles de méthodes

Pour spécifier la flexibilité des familles de méthodes, nous utilisons les concepts définis dans le modèle de variabilité orthogonal (Orthogonal Variability Model – OVM) proposé dans (Pohl *et al.*, 2005). Nous utilisons en particulier les concepts de point de variation, de dépendances de variabilité et de contraintes de dépendances.

Dans le domaine des lignes de produits, les points de variation identifient les endroits où se produisent des variations (Deelstra *et al*., 2004). Ces endroits sont reconnus comme des éléments centraux dans la gestion de la variabilité puisqu'ils en facilitent la documentation, la traçabilité, la réutilisabilité et l'évolution (Deelstra *et al*., 2004). De la même manière, dans les familles de méthodes, la modélisation des points de variation permet une configuration flexible des lignes, adaptée au contexte d'utilisation, une évaluation de la réutilisabilité des composants et une évolution facilitée de la famille.

La dépendance de variabilité définie la relation entre un point de variation et un composant variable. Elle spécifie si le composant est obligatoire ou optionnel pour cette situation spécifique. Pohl *et al*. (2005) spécifient que la dépendance de variabilité optionnelle indique qu'un variant peut être (mais n'est pas nécessairement) une partie de la ligne configurée. De la même manière, une famille peut contenir des composants optionnels représentant les alternatives offertes à l'ingénieur à un point de variation. Une dépendance de variabilité obligatoire indique qu'un variant précis est nécessaire lors d'un point de variation (Pohl *et al*., 2005). Cependant, il est à noter que, même si un composant de méthode est considéré comme obligatoire dans un point de variation spécifique, il peut également ne pas se trouver dans certaines lignes de méthodes si le point de variation lui-même est optionnel dans la famille de méthodes.

En plus de la dépendance de variabilité, il y a plusieurs contraintes possibles selon la compatibilité des composants entre eux (y compris les composants appartenant à différents points de variation). Par exemple, l'utilisation d'un composant peut nécessiter l'utilisation d'un autre composant ou, au contraire, l'exclure complètement. La contrainte de dépendance permet de définir ces contraintes. Nous reprenons cette notion de dépendance de variabilité (Pohl *et al*., 2005) dans la définition des familles de méthodes en caractérisant certaines relations entre composants comme 'nécessaire' ou 'exclusif'. La relation 'nécessite' exprime la complémentarité obligatoire de deux composants (relation ET), alors que la relation 'exclusif' ne permettra pas d'utiliser les deux dans la même ligne de méthode (relation XOR).

3.3. Concepts liés à la configuration des familles de méthodes

(Kornyshova et al, 2011a) détaille la notion de contexte utilisable dans le cadre des composants de méthodes. Deux contextes doivent être définis : le contexte du

composant de méthode qui caractérise toutes les situations dans lequel le composant peut être appliqué et le contexte du projet qui inclut toutes les caractéristiques de la situation en cours (peut correspondre à un ensemble de contextes de composants variables). Ces deux contextes sont formalisés de la même manière, ce qui permet une meilleure correspondance entre la situation et les composants lors de la configuration. Un composant de méthode variable est sélectionné selon la situation en cours (selon le contexte du projet). L'ensemble des composants de méthodes sélectionnés, avec les composants de méthodes communs, correspond à une ligne de méthode du projet. Une ligne de méthode peut être de deux types : soit une méthode initiale (une méthode déjà connue), soit une ligne de méthode du projet (définie avec notre processus). La ligne de méthode projet inclut les composants obligatoires et ceux sélectionnés pour le projet en cours selon ses caractéristiques. Une famille de méthodes est composée d'un ensemble de lignes de méthode. Chaque configuration de la famille implique la création d'une ligne de méthode (en d'autres termes, chaque chemin potentiel dans le processus organisationnel de la famille est une ligne potentielle). Ce concept permet de regrouper plusieurs lignes pour un domaine.

4. Construction d'une famille de méthodes

Suivant les principes de l'ingénierie des méthodes situationnelles, la construction d'une famille de méthodes est basée sur la réutilisation des composants extraits de méthodes existantes. Ces méthodes doivent avoir le même domaine d'application et des objectifs similaires. Cependant, alors que les approches habituelles de construction de méthode se concentrent sur le processus de construction lui-même et sur ses nombreuses difficultés (comme la difficulté d'effectuer une intégration lorsqu'il y a un chevauchement de concepts par exemple), les familles de méthodes offre une manière de simplifier le travail des utilisateurs avec un détachement du processus de construction. L'ingénieur de méthode construit la famille de méthodes en décomposant en premier lieu les méthodes (avec les approches IMS classiques) et en les regroupant en une famille que l'utilisateur configurera pour obtenir une méthode adaptée à ses besoins. Le processus de construction d'une famille de méthode abordé dans (Kornyshova et al, 2013) est ici détaillé de manière approfondie et illustré sur un exemple d'utilisation. Nous identifions trois étapes principales dans le processus de construction d'une famille de méthodes, comme illustré dans la figure 2.

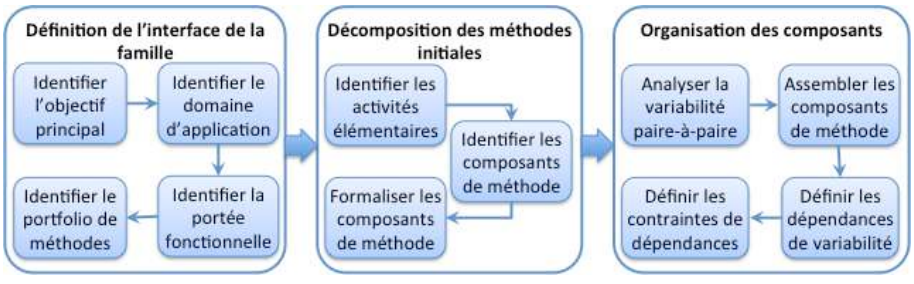


Figure 2. Processus de construction d'une famille de méthodes

4.1. Définition de l'interface de la famille de méthodes

On ne peut construire une famille de méthodes qu'avec des méthodes initiales ayant la même unité téléologique (i.e. qui ont le même objectif, la même intention de réalisation), celle-ci représentant le degré de similarité entre les méthodes selon leur usage et leurs objectifs (Kornyshova, 2011). Cela signifie que seules les méthodes ayant la même utilisation et créées dans le même but pourront être intégrées dans une même famille. Bien entendu, ces méthodes, même si elles sont proches du point de vue téléologique, contiennent cependant des différences qui représentent les différentes manières de réaliser leurs objectifs. Selon (Asadi *et al.*, 2011), la première phase de la construction d'une famille consiste à définir son interface, qui inclut l'identification du portfolio de méthodes initiales, la spécification du domaine d'application et l'identification des fonctionnalités principales. Le modèle de processus générique défini dans (Ralyté *et al.*, 2003) spécifiait que la première phase consistait à identifier l'objectif principal de la méthode à construire. Nous reprenons ces idées et définissons quatre étapes précises, comme l'indique la figure 2.

Identifier l'objectif principal. Avant de se lancer dans la construction d'une famille de méthodes, il est nécessaire de comprendre quel sera son objectif principal. Une famille de méthodes peut être dédiée à un certain type de projets (e.g. le développement de systèmes ERP) ou à un certain type d'entreprise (e.g. les fournisseurs d'applications Web), ou encore pour des audiences très larges (e.g. le développement de services dans le secteur public). L'unification de méthodes similaires ou leur standardisation sont autant d'exemples d'objectifs de famille de méthodes.

Identifier le domaine d'application. Le domaine d'application d'une famille de méthodes peut être plus ou moins étendu. Quelques exemples de domaines pourraient être : l'ingénierie du logiciel, la conception de services mobiles, l'ingénierie des besoins, la conception de systèmes d'information, etc.

Identifier la portée fonctionnelle. Cette étape consiste à identifier la partie du cycle de vie prise en compte. Celle-ci peut couvrir l'ensemble du processus de développement ou juste une partie de processus (e.g. la phase d'analyse des besoins), ou encore une activité très spécifique (e.g. l'identification des besoins préliminaires).

Identifier le portfolio de méthodes. La dernière étape consiste à identifier le spectre de méthodes correspondant aux précédents éléments identifiés, i.e. l'objectif principal, son domaine d'application et sa portée fonctionnelle. Seules les méthodes correspondant à cette même unité téléologique pourront être intégrées (Kornyshova, 2011).

4.2. Décomposition des méthodes initiales en composants de méthodes

L'objectif de cette phase est de décomposer les méthodes existantes en un ensemble de composants de méthodes autonomes. Il existe différentes manières d'exécuter cette décomposition (comme dans (Ralyté *et al*., 2001)) mais nous proposons ici une décomposition en trois étapes.

Identifier les activités élémentaires. La décomposition d'une méthode consiste à décomposer son processus en un ensemble d'activités élémentaires. Une activité décrit un ensemble d'actions élémentaires comme : définir une classe objet, calculer une somme pondérée, etc. La description des activités peut être réalisée à l'aide de différents modèles de processus (e.g. avec des diagrammes d'activités UML).

Identifier les composants de méthodes. Les activités élémentaires définies dans l'étape précédente sont regroupées sous la forme de composants de méthodes. Cette étape suit deux grands principes : l'indépendance et la complétude. L'indépendance signifie que chaque composant obtenu doit être autonome – il peut être appliqué séparément, dans différentes situations. La complétude signifie que si une méthode a été décomposée, la réunion de tous les composants issus de celle-ci doit pouvoir aboutir à la méthode initiale – aucune activité ne doit avoir été omise.

Formaliser les composants de méthodes. La formalisation des composants dépend de l'approche de construction de méthode situationnelle choisie (orientée morceau, fragment, service ou autre) étant donné que notre proposition ne se limite pas à un type de composant précis. Dans cet article, nous illustrerons notre proposition à l'aide des composants de type morceau (Ralyté et al., 2001) qui incluent plusieurs éléments : l'identification du composant (ID, nom, description, composants – si le composant est un agrégat – et signature comprenant la situation et l'intention), la partie produit (avant et après exécution du composant) et la partie processus. Un exemple de composant de méthode est donné en figure 3.

DSDM-5 Planifier le projet par plan directeur

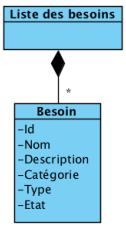
Description: l'objectif de ce composant est d'aider un chef de projet à établir une première planification des livrables du projet en fonction des besoins à développer dans un plan directeur.

Situation source : Liste des besoins
Situation cible : Livrables identifiés et ordonnés dans un planning des livrables.

Contexte :
- Taille du projet: grande
- Type de planification : {au produit}
- Longueur d'un cycle itératif: long
- Complexité du projet : forte
- Respect des délais: facilement

Processus (PPR)
1. Répartir les besoins en différents livrables avec le client en fonction de leurs priorités.
2. Estimer l'effort et le temps de développement des besoins à implémenter dans chaque livrable.
3. Fixer les dates des livrables et les organiser dans le Plan Directeur (première planification).

Produit Source (PPS)



Produit cible (PPC)

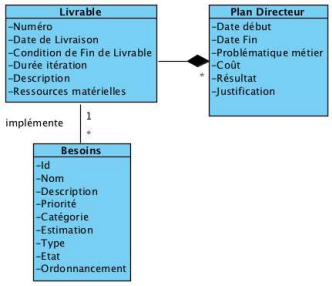


Figure 3. Exemple de composant de méthode DSDM-5

4.3. Organisation des composants de méthodes dans une famille

L'organisation des composants de méthodes dans la famille est basée sur l'analyse de la variabilité des composants, utilisant des mesures de similarité. Comme dans (Ralyté et Rolland, 2001b) nous proposons de mesurer la similarité des concepts (en prenant en compte deux aspects : leur sémantique et leur structure.

La similarité sémantique entre deux concepts de produit est basée sur la ressemblance de leurs noms, cette métrique est appelée Affinité de Noms (AN). Elle utilise la relation de synonyme (SYN) entre les noms des éléments du produit pour l'évaluation de la ressemblance. SYN définit une relation symétrique entre deux termes T_i et T_j ($T_i \neq T_j$) considérés comme synonymes (e.g. <Objectif SYN But>).

La similarité structurelle (SSC) entre deux concepts (c_1,c_2) d'un modèle de produit est basée sur la similitude de leurs propriétés et liens avec les autres concepts (1).

$$SSC(C_1, C_2) = \left(\frac{2*(Nb\ de\ propriétés\ communes)}{\sum_{i=1}^{2} Nb\ de\ propriétés\ dans\ C_i} + \frac{2*(Nb\ de\ liens\ communs)}{\sum_{i}^{2} Nb\ de\ liens\ de\ C_i}\right) \div 2 \qquad (1)$$

Nous définissons des métriques de similitudes entre les concepts du modèle de processus. Leur définition dépend du formalisme de modélisation de processus utilisé (des exemples de métriques de similarité sont donnés dans (Ralyté, 2001)).

Toutes ces métriques aident à évaluer le degré de variabilité des composants de méthodes (commun, variable ou différent), à identifier les points de variation et à spécifier les dépendances de variabilité entre les composants de méthode. De plus, nous utilisons des techniques d'assemblage pour connecter et ordonner les composants à l'intérieur de la famille. Cette phase est formalisée en quatre étapes.

Analyser la variabilité paire-à-paire. Nous utilisons l'analyse paire-à-paire pour définir la variabilité des composants de méthode (composants variables, composants communs et points de variation), comme illustré dans la figure 4.

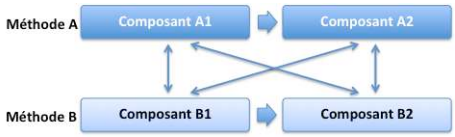


Figure 4. Comparaison paire-à-paire des composants de méthode

En utilisant le type de composant choisi et les mesures de similarité introduites précédemment, nous comparons les éléments suivants des composants de méthode : partie de produit source (PPS), partie de produit cible (PPC) et partie de processus (PPR), d'un point de vue sémantique et structurel. Par exemple, pour les deux composants de méthode cm_1 et cm_2, la formulation $PPS\ (cm_1) \approx PPS\ (cm_2)$ signifie que le produit source mc_1 est similaire au produit source de mc_2. Chaque flèche présente sur la figure 4 correspond à une comparaison paire-à-paire de ces trois éléments (i.e. PPS, PPC et PPR). Si le degré de similarité totale (structurelle et sémantique) entre deux composants de méthode est élevé, ils sont considérés comme composants communs. S'ils sont similaires

seulement du point de vue sémantique, ils sont considérés comme composants variables. Le tableau 1 montre les cinq cas principaux utilisés pour analyser la variabilité.

Assembler les composants de méthode. Cette étape consiste à finaliser l'organisation des composants de méthode en les intégrant dans une multi-méthode cohérente et complète. Nous utilisons les techniques d'assemblage définies dans (Ralyté et Rolland, 2001b) qui proposent deux démarches, l'association et l'intégration, pour assembler des composants de méthodes en fonction de leur similarité. L'association de composants est utilisée pour connecter des composants variables ou consécutifs selon leur produits source ou cible (par exemple, dans le cas de composants consécutifs, le produit cible du composant prédécesseur est connecté au produit source du composant successeur). Les composants variables sont connectés à un point de variation. L'intégration de composants est nécessaire pour assembler des composants communs en fusionnant leurs éléments similaires (ex. concepts, propriétés, activités, etc.). Les deux techniques (intégration et association) utilisent des opérateurs d'unification pour unifier les terminologies employées dans les composants et permettre leur assemblage.

Tableau 1. Comparaison paire-à-paire des composants de méthodes.

Cas de comparaison	Interprétation
• PPS (cm_1) ≈ PPS (cm_2) • PPC (cm_1) ≈ PPC (cm_2) • PPR (cm_1) ≈ PPR (cm_2)	Les composants cm_1 et cm_2 sont communs.
• PPS (cm_1) ≈ PPS (cm_2) • PPC (cm_1) ≈ PPC (cm_2) • PPR (cm_1) ≠ PPR (cm_2)	Les composants cm_1 et cm_2 sont variables. Le produit source est considéré comme un point de variation avec le même résultat mais différentes démarches pour l'obtenir.
• PPS (cm_1) ≈ PPS (cm_2) • PPC (cm_1) ≠ PPC (cm_2) • PPR (cm_1) ≠ PPR (cm_2)	Les composants cm_1 et cm_2 sont variables. Le produit source est considéré comme un point de variation avec différents résultats.
• PPS (cm_1) ≠ PPS (cm_2) • PPC (cm_1) ≈ PPC (cm_2) • PPR (cm_1) ≠ PPR (cm_2)	Les composants cm_1 et cm_2 sont variables. Le produit cible est considéré comme un point de variation avec différents produits source différents.
• PPS (cm_1) ≈ PPC (cm_2) • PPR (cm_1) ≠ PPR (cm_2)	Les composants cm_1 et cm_2 sont consécutifs.
"≈" signifie "similaire"; ≠ signifie "non similaire"	

Définir les dépendances de variabilité. Cette étape consiste, pour chaque composant variable d'un point de variation, à spécifier la dépendance de variabilité, pouvant être soit optionnelle, soit obligatoire.

Définir les contraintes de dépendances. Cette étape permet de compléter le processus d'assemblage en spécifiant quels composants de méthode sont exclusifs ou complémentaires les uns les autres.

Lorsque toutes ces étapes ont été effectuées, la famille de méthodes est finalisée. A ce stade, la variabilité est prise en compte par l'identification des points de variations et des composants communs et variables. Les composants sont identifiés et organisés

comme des variations potentielles à chaque point de variation. Est également prise en compte la variabilité des stratégies pour atteindre un même objectif.

4.4. Représentation d'une famille de méthodes

4.4.1. Représentation formelle

Une famille de méthodes *FM* peut être représentée (de façon réduite) sous la forme d'une liste de composants communs *CMC*, d'une liste de composants variables *CMV*, d'une liste de points de variation de démarche *PVD*, d'une liste de points de variation de réalisation *PVR*, d'une liste de contraintes *C* et d'un ensemble de dépendances *D*.

```
FM = ({CMC},{CMV},{PVD},{PVR},{C},{D})                    (2)
```

Représentation d'un composant :

Soit *Nomcomp* le nom du composant de méthode, *Desc* la description du but à atteindre par la réalisation de ce composant, *Contexte* les caractéristiques de contexte de réutilisation, *PPS* la situation du produit avant exécution du composant, *PPC* la situation du produit après son exécution et *PPR* son processus d'opérationnalisation, un composant *CM* sera défini comme suit :

```
CM = (Nomcomp,Desc,Contexte,PPS,PPC,PPR)                  (3)
```

Deux exemples de composants sont donnés aux figures 3 et 9.

Représentation d'un point de variation :

Soit *PV* le nom du point de variation, *ID* le nom du composant de méthode suivi de la dépendance de variabilité (*D*) de ce composant dans ce point de variation (+ signifie une dépendance obligatoire, * une dépendance optionnelle) alors PV correspondra à l'ensemble des composants de méthodes reliés à ce point de variation.

```
PV = {ID,D}                                               (4)
```

Par exemple, le point de variation de réalisation 'Définir la faisabilité du projet' est lié aux deux composants de méthodes DSDM-1 et DSDM-2, ces deux composants étant optionnels, le point de variation s'écrira formellement :

```
Définir la faisabilité du projet=(DSDM-1,*; DSDM-2,*)
```

Représentation d'une contrainte de dépendance de type 'nécessite' :

Soit CM_1 et CM_2 deux composants de méthode, une contrainte de dépendance *C* de type 'nécessite' entre ces deux composants sera représentée par le symbole '&'.

```
C = CM1 & CM2                                             (5)
```

Représentation d'une contrainte de dépendance de type 'exclut':

Soit CM_1 et CM_2 deux composants de méthode, une contrainte de dépendance *C* de type 'exclut' entre ces deux composants sera représentée par le symbole '⊕'.

$$C = CM1 \oplus CM2 \quad (6)$$

4.4.2. Représentation graphique

Pour plus de facilité d'utilisation, il est également possible de représenter graphiquement une famille de méthodes à l'aide d'un graphe orienté. Nous avons choisi une représentation graphique tirée du modèle de processus MAP (Rolland *et al.,* 1999) (Rolland, 2007b). Ce métamodèle est un système de représentation initialement développé pour représenter les modèles de processus en termes intentionnels (ce qui permet de savoir pourquoi le processus est exécuté). MAP permet de spécifier les processus d'une manière flexible en mettant l'accent sur les intentions de processus et sur les différentes manières d'atteindre chacune de ces intentions. Une carte (une instance du méta-modèle) est présentée comme un diagramme dont les nœuds sont les intentions et les arcs les stratégies. Les arcs orientés montrent quelles sont les intentions qui peuvent précéder ou suivre les autres. Un arc entre dans un nœud si sa stratégie associée peut être utilisée pour réaliser l'intention cible (le nœud). Comme il peut y avoir plusieurs arcs entrant dans un nœud, cela représente tous les moyens pour réaliser une intention. Le choix entre les différentes alternatives se fait en fonction de la situation. Cela permet une grande flexibilité lors de la réalisation des processus et la prise en compte pertinente de la sensibilité au contexte.

Le formalisme MAP initial ne permet pas de représenter graphiquement toutes les contraintes sur une carte, il nous a donc été nécessaire de modifier légèrement le graphisme afin de pouvoir les intégrer. Le tableau 2 définit les concepts inhérents aux familles de méthodes pouvant être représentés graphiquement dans une carte et le tableau 3 ceux qui ont été ajoutés pour permettre la représentation des contraintes.

Tableau 2. Concepts représentables graphiquement sur une carte

Concept de Famille	Représentation graphique avec la carte
Famille de méthodes	Carte
Ligne de méthode	Carte obtenue par sélection d'un sous-ensemble de sections
Composant de méthode	Section
- commun	- dessinée avec une ligne épaisse
- variable	- dessinée avec une ligne fine
Point de variation	Intention source

Tableau 3. Concepts représentables sur la carte de manière additionnelle

Concept de Famille	Représentation graphique avec la carte

Dépendance de variabilité	Lien entre un point de variation (intention) et un composant (section)
- Optionnelle	- une '*' sur la section correspondant au composant
- Obligatoire	- une '+' sur la section correspondant au composant
Contrainte de dépendance	Contrainte entre deux sections
- nécessite	- une ligne pointillée avec un '&' entre les deux sections (composants)
- exclut	- une ligne pointillée avec un '⊕' entre les deux sections (composants)

La figure 5 montre un exemple de carte avec l'intégration des contraintes et des dépendances. Dans cet exemple, le point de variation de démarche Intention1 est relié aux trois composants optionnels ID-1, ID-2 et ID-3. Les deux composants ID-2 et ID-3 s'excluent mutuellement alors que ID-1 et ID-4 sont nécessaires l'un à l'autre.

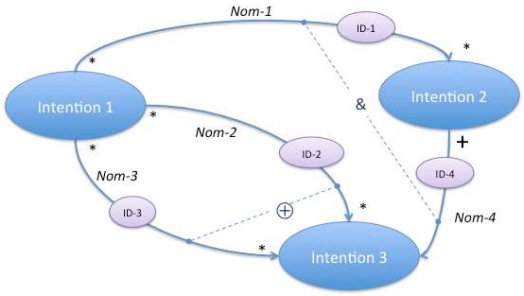


Figure 5. Exemple de carte avec contraintes.

4.5. Construction d'une méthode par la configuration d'une ligne de méthode

Lors de l'étape de la configuration de la famille dans le but d'obtenir une ligne adaptée au projet, l'utilisateur utilise la caractérisation des composants pour identifier ceux compatibles avec le contexte du projet. Il est également possible d'exécuter plusieurs configurations de manière séquentielle pour obtenir la méthode souhaitée.

La configuration d'une ligne se fait en comparant les caractéristiques du projet avec celle des composants alternatifs, ce qui permet de toujours choisir le composant le plus adapté à la situation en cours. Le choix à effectuer entre les alternatives est défini comme un problème de prise de décision. Les alternatives sont comparées entre elles selon un ou plusieurs critères (caractérisant le projet et le composant). Les méthodes de décision appliquées peuvent donc être monocritères ou multicritères. L'utilisation d'un seul critère est plus simple, mais elle ne suffit pas lorsque les conséquences des alternatives à analyser sont importantes (Roy, 1996). Les méthodes multicritères de prise de décision permettent une analyse plus approfondie du problème mais sont plus compliquées car les valeurs des indicateurs doivent être agrégées (Roy, 1996 ; Keeney *et al*., 1993).

En utilisant les méthodes de prise de décision pour configurer les familles de méthodes, il est possible d'identifier trois types de configuration (Kornyshova, 2011) : (a) sélectionner un sous-ensemble de composants de méthodes correspondant au contexte du projet, (b) choisir une ligne de méthodes parmi toutes les lignes possibles de la famille, ou (c) sélectionner les composants étape par étape, ce qui permet une plus grande flexibilité en tenant compte du contexte évolutif à l'exécution du processus.

La typologie des caractéristiques prises en compte peut être basée, par exemple, sur celle qui a été proposée dans (Deneckère et al., 2010) (Kornyshova et al, 2011a) (Deneckère et Kornyshova, 2010) où le contexte est caractérisé par des facettes génériques (organisationnelle, humaine, concernant le domaine d'application ou la stratégie de développement) ou spécifique au formalisme utilisé.

5. Illustration

Nous illustrons notre proposition avec la construction d'une famille de méthodes concernant les méthodes agiles mais en limitant cette famille à la phase de lancement d'un projet. L'objectif est de pouvoir proposer à l'entreprise un ensemble de composants organisés leur permettant de planifier un projet agile de manière adaptée à son projet en cours. Cette famille LPA - Lancement de Projet Agile - contient 13 composants de méthode. Les sous-sections suivantes reprennent les étapes de construction de famille de méthodes en les appliquant au cas LPA.

5.1. Identifier l'interface de la famille de méthodes

- *Identifier l'objectif principal.* L'objectif principal de la famille LPA est d'offrir un support méthodologique riche, flexible et configurable pour les activités de lancement des projets agiles.
- *Identifier le domaine d'application.* Le domaine d'application de la famille LPA est le développement de projets en suivant un cycle agile.
- *Identifier la portée fonctionnelle.* La portée fonctionnelle de la famille LPA est restreinte à la phase de lancement de projet (planification des releases).
- *Identifier le portfolio de méthodes.* Les méthodes étudiées pour créer les composants de méthodes sont les trois méthodes Scrum (Schwaber, 2011), XP (Wells, 2009) et DSDM (DSDM, 2009).

5.2. Décomposition des méthodes en composants de méthodes.

Les méthodes agiles inclues dans le portfolio ont été analysées et étudiées pour pouvoir être décomposées en composants de méthode. Les figures 6, 7 et 8 décrivent les processus de ces trois méthodes durant la phase de lancement d'un projet agile.

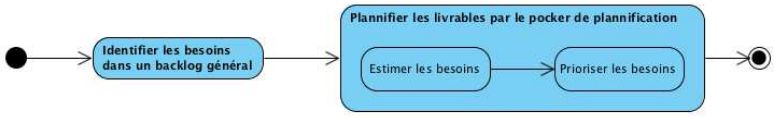


Figure 6. Processus de lancement de projet avec la méthode XP

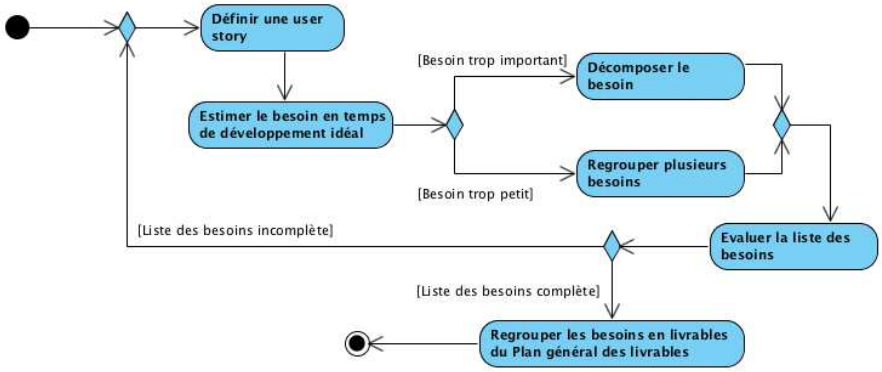


Figure 7. Processus de lancement de projet avec la méthode SCRUM

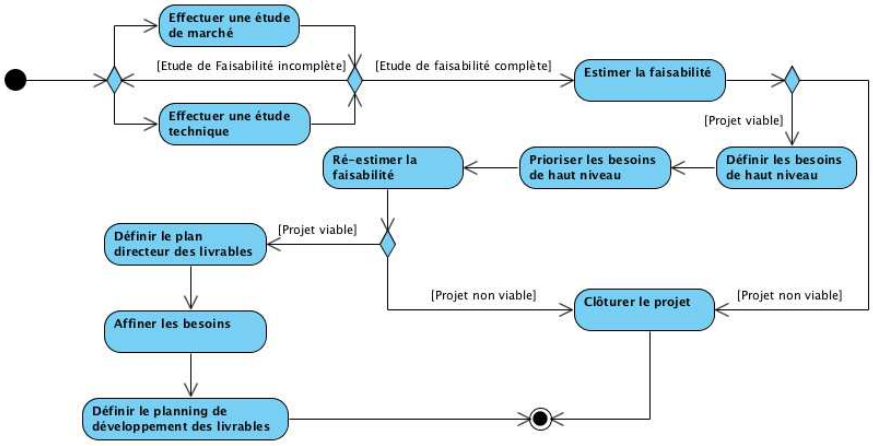


Figure 8. Processus de lancement de projet avec la méthode DSDM

Les trois méthodes agiles étudiées comportent clairement deux groupes principaux d'activités, celles liées à la définition des besoins du client vis-à-vis du système à réaliser et celles traitant de la planification globale de haut niveau du projet. La méthode DSDM préconise en plus de ces deux groupes d'activités d'effectuer une étude de marché et une étude technique avant toute chose dans le projet. Cette étape

supplémentaire permet de pouvoir abandonner le projet si l'étude de faisabilité montre un échec potentiel.

Chacune des trois méthodes agiles proposent une méthodologie différente pour définir les besoins (de haut niveau) du client. SCRUM propose d'identifier les besoins du client, d'estimer leur taille par la technique dite du poker de planification, puis de les prioriser par rapport aux risques et aux incertitudes qu'ils permettent de réduire. XP quand à lui, propose de réaliser une liste des besoins par une itération de définition de besoins et d'estimation. Chaque besoin est identifié de façon similaire à SCRUM mais il est ensuite estimé en temps de développement idéal. Si son temps estimé est trop important, le besoin est alors décomposé en plusieurs besoins. Inversement, s'il est trop petit, il est regroupé à d'autres besoins pour en former un plus important (XP-1). La méthode XP propose également un style moins formel pour la rédaction des besoins que la méthode SCRUM. La méthode DSDM propose d'inclure les futurs utilisateurs finaux du système directement dans le processus pour la définition et la priorisation d'une liste de besoins de haut niveau répondant à la problématique du projet.

Une fois la liste des besoins établie, la méthode DSDM propose encore une fois de s'interroger sur la viabilité ou la faisabilité du projet avec l'ensemble des besoins tels qu'ils sont définis. Si le projet n'est pas estimé viable, le projet est clôturé. Dans le cas contraire, le processus passe à la partie de planification du projet en utilisant diverses techniques.

L'utilisateur peut ensuite, s'il le désire, arrêter là le processus et terminer ici l'étape de lancement du projet pour ensuite lancer les cycles d'itération de développement. En DSDM cependant, il est possible de commencer un affinement des besoins de haut niveau du projet après avoir établi une première planification. Il peut alors affiner un besoin fonctionnel en divers sous-besoins fonctionnels et réaliser une ébauche de liste des besoins non fonctionnels. Il peut aussi revenir sur la planification du projet pour affecter des acteurs aux livrables et caractériser par un type (modèle fonctionnel, prototypage ou développement) les itérations dans les livrables. L'utilisateur peut exécuter ceci jusqu'à ce que les niveaux de granularité des besoins et du planning soient satisfaisants. Le processus de lancement d'un projet agile s'arrête lorsque le chef de projet estime la planification complète.

Les Tables 4, 5 et 6 montrent les composants issus des différentes méthodes.

Tableau 4. Composants de méthodes créés à partir de la méthode DSDM

ID	Signature	Description
DSDM-1	< {Description du problème}, Etudier la faisabilité du projet par une étude de marché >	L'objectif de ce composant est d'aider le chef de projet à déterminer si un projet de développement de logiciel est faisable ou non. Ce composant propose de réaliser une étude de marché pour le projet qui rentrera dans le cadre d'une étude de faisabilité.
DSDM-2	< {Description du problème}, Etudier la faisabilité du projet par une étude technique >	Ce composant aide un chef de projet à déterminer si un projet de développement de logiciel est faisable ou non. Ce composant propose de réaliser une étude technique du projet qui rentrera dans le cadre d'une étude de faisabilité.

DSDM-3	< {Document de faisabilité}, Abandonner le projet >	Ce composant permet de prendre acte de l'arrêt du projet lorsque l'étude de faisabilité montre un échec potentiel.
DSDM-4	< {Document de faisabilité}, Définir les besoins du projet par définition des besoins de haut niveau >	L'objectif de ce composant est d'aider le chef de projet et le client à définir les besoins de haut niveau vis-à-vis du système à développer, puis de prioriser l'ensemble des besoins en collaboration avec les utilisateurs.
DSDM-5	< {Liste des besoins}, planifier le projet par plan directeur >	L'objectif de ce composant est d'aider le chef de projet à établir une première planification des livrables du projet en fonction des besoins à développer dans un plan directeur.
DSDM-6	< {Planning des livrables}, Définir les besoins du projet par affinement >	Ce composant aide le chef de projet à affiner la liste des besoins de haut niveau en sous-besoins et établir une première liste de besoins non fonctionnels généraux.
DSDM-7	<{Liste des besoins}, Abandonner la planification >	Ce composant permet d'arrêter le processus lorsque le chef de projet estime que les besoins définis sont irréalisables selon les ressources allouées.
DSDM-8	< {Planning des livrables}, Planifier le projet par développement planning >	Ce composant permet au chef de projet d'affiner la première planification effectuée lors du planning des livrables en identifiant les premières itérations et en affectant des acteurs aux livrables.
DSDM-9	<{Planning des livrables}, Clôturer la planification >	Ce composant permet d'arrêter le processus lorsque le chef de projet estime que la planification est complète.

Tableau 5. Composants de méthodes créés à partir de la méthode SCRUM

ID	Signature	Description
SCRUM-1	< {Description du problème}, Définir la liste des besoins du projet par définition du backlog général >	L'objectif de ce composant est d'aider le chef de projet et le client à définir les besoins vis-à-vis du système à développer, sous la forme de besoins utilisateur. Une fois les besoins identifiés, le composant aide à prioriser l'ensemble des besoins utilisateur dans une liste ordonnée, la liste des besoins du produit.
SCRUM-2	< {Liste des besoins}, Planifier le projet par planification générale >	Ce composant aide le chef de projet à établir une première planification des livrables du projet.
SCRUM-3	<{Planning des livrables}, Clôturer la planification >	Ce composant permet d'arrêter le processus lorsque le chef de projet estime que la planification est complète.

Tableau 6. Composants de méthodes créés à partir de la méthode XP

ID	Signature	Description
XP-1	< {Description du problème}, Définir la liste des besoins du projet par définition de user stories >	Ce composant aide le chef de projet et le client à définir les besoins vis-à-vis du système à développer sous la forme de besoins utilisateur, puis de réaliser une première estimation du temps de développement de chaque besoin.
XP-2	< {Liste des besoins}, Planifier le projet par définition du plan général des releases >	Ce composant permet au chef de projet d'estimer le temps de développement des besoins utilisateurs puis d'établir une première planification des livrables du projet en utilisant la technique du poker de planification.
XP-3	<{Planning des livrables}, Clôturer la planification >	Ce composant permet d'arrêter le processus lorsque le chef de projet estime que la planification est complète.

5.3. Organisation des composants de méthode en famille de méthodes.

Analyser la variabilité paire-à-paire.

Nous illustrons l'organisation des composants avec l'exemple des deux composants XP-2 (figure 9) et DSDM-5 (figure 3). Il y a un choix exclusif entre ces deux composants de méthodes pour réaliser la planification du projet. Le composant XP-2 issu de XP propose d'effectuer une estimation précise du temps de développement des besoins et de répartir les besoins sans les ordonnancer sur un planning de livrables, ceci se faisant avec l'aide de la technique du poker de planification (jeu de cartes permettant d'estimer les besoins et de les répartir). Le composant DSDM-5 dérivé de la méthode DSDM propose de définir l'ensemble des livrables par rapport à la liste des besoins et d'organiser ensuite ces livrables dans un planning. La variabilité paire-à-paire de ces deux composants est présentée en table 7. Elle conclue que les deux composants sont variables puisque leur partie produit source et cible sont considérés comme similaires, en prenant en compte les mesures AN (Planning des livrables, Plan directeur) = 1 (< Planning des livrables SYN Plan directeur >) and SSC (Planning des livrables, Plan directeur) ≈ 1. La partie produit source est donc considérée comme un point de variation. Ces composants représentent en fait deux manières alternatives d'atteindre le même produit cible en appliquant des procédures différentes.

<table>
<tr><td colspan="2">XP-2 Planifier le projet par définition du plan général des releases</td></tr>
<tr><td colspan="2">Description: l'objectif de ce composant est d'aider un chef de projet à estimer le temps de développement des besoins utilisateur puis d'établir une première planification des livrables.</td></tr>
<tr><td colspan="2">Situation source : Liste des besoins
Situation cible : Livrables identifiés et ordonnés dans un planning des livrables.</td></tr>
<tr><td colspan="2">Contexte :
- Taille du projet: petite
- Type de planification : {au produit}
- Longueur d'un cycle itératif : court
- Complexité du projet : faible
- Respect des délais: difficilement</td></tr>
<tr><td colspan="2">Processus (PPR)
1. Ecrire chaque besoin utilisateur sur une carte.
2. Estimer le temps de développement idéal (temps idéal en semaine que l'équipe de projet mettra pour développer ce besoin) de chaque besoin, sans sous-estimation.
3. Regrouper les cartes pour former des livrables (incréments logiciels fonctionnels qui seront livrés au client), sans ordonnancement des besoins.
4. Recommander au client de sélectionner pour les premiers livrables les besoins utilisateur primordiaux au fonctionnement du système et ayant le plus de valeur ajoutée pour le domaine métier de la solution.
5. Calculer les dates des livrables en effectuant la somme des estimations des besoins de chaque livrable.</td></tr>
<tr><td>Produit Source (PPS)</td><td>Produit cible (PPC)</td></tr>
</table>

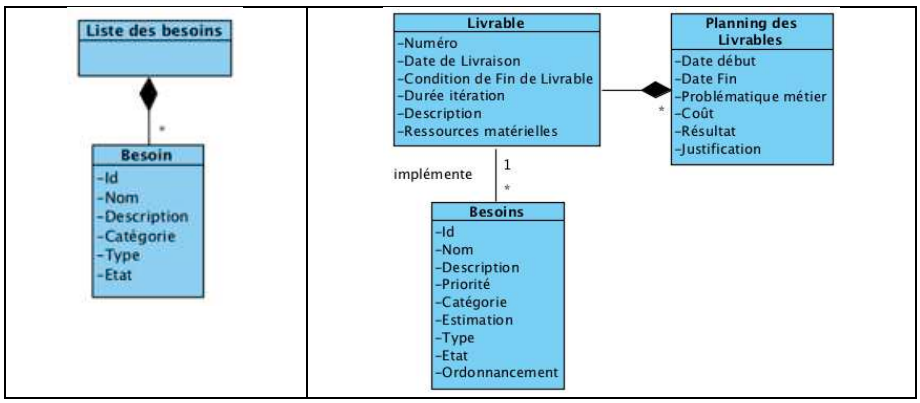


Figure 9. Composant de méthode XP-2 (Planifier le projet par définition du plan général des releases)

Tableau 7. Analyse de la variabilité des deux composants XP-2 et DSDM-5

	XP-2	DSDM-5	Comparaison
PPS	liste de besoins	liste de besoins	PPS (XP-2) ≈ PPS (DSDM-5)
PPC	planification des livrables avec estimation de temps et répartition des besoins dans un planning des livrables	planification des livrables avec répartition des besoins et ordonnancement temporel dans un plan directeur	PPC (XP-2) ≈ PPC (DSDM-5)
PPR	Estimation et priorisation par poker de planification	Estimation et priorisation manuelle	PPR (XP-2) ≠ PPR (DSDM-5)

En poussant ce raisonnement sur les autres paires de composants de méthode, on pourra déduire que le composant SCRUM-2 aura les mêmes caractéristiques que les deux précédents composants étudiés : même produit source, même produit cible mais processus différent. Ces trois composants représentent donc trois possibilités alternatives pour atteindre la même intention cible.

Il est également possible de se rendre compte qu'il existe des composants communs, quelque soit les méthodes initiales. C'est le cas, par exemple, des composants DSDM-9, SCRUM-3 et XP-3 qui ont :

- le même produit source (PPS (DSDM-9) ≈ PPS (SCRUM-3) ≈ PPS (XP-3)),
- le même produit cible (PPC (DSDM-9) ≈ PPC (SCRUM-3) ≈ PPC (XP-3)), et
- le même processus (PPR (DSDM-9) ≈ PPR (SCRUM-3) ≈ PPR (XP-3)).

Assembler les composants de méthode.

Pour assembler les différents composants, nous devons étudier la ressemblance des éléments et, le cas échéant, les combiner. Pour cela, nous pouvons par exemple appliquer l'opérateur de renommage qui consiste à renommer l'un des concepts

synonyme. Dans le cas des deux premiers composants étudiés précédemment (XP-2 et DSDM-5), le concept de 'Plan directeur' peut être renommé en 'Planning des releases' puisque l'on estime que ce terme est celui qui est le plus connu en pratique.

Dans le cas des composants communs (DSDM-9, SCRUM-3 et XP-3), ces composants sont combinés dans le même composant final, présent une seule fois dans la famille, en utilisant la démarche *d'intégration* (ce nouveau composant sera ici identifié par le numéro AGILE-1).

Définir les dépendances de variabilité.

La définition des contraintes de variabilité permet d'identifier certains composants optionnels ou obligatoires. Par exemple, étudier la faisabilité du projet est quelque chose d'optionnel, le chef de projet n'est pas obligé de passer par cette étape pour pouvoir planifier son projet. Cette variabilité est exprimée dans un point de variation de réalisation, comme illustré dans la figure 10.

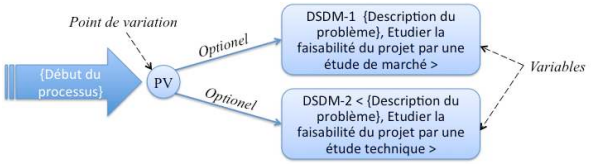


Figure 10. Les deux composants DSDM-1 et DSDM-2 assemblés

Définir les contraintes de dépendances.

Les deux composants XP-2, DSDM-5 et SCRUM-2 sont considérés comme exclusifs les uns avec les autres (la planification ne se fait que d'une seule manière). Cette contrainte de dépendance est indiquée entre chaque paire de composants de méthode, comme illustré dans la figure 11.

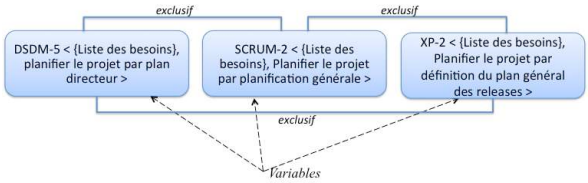


Figure 11. Contraintes de dépendance entre les composants de méthode DSDM-5, SCRUM-2 et XP-2

Ce processus est étudié pour toutes les paires de composants possibles. Une fois la variabilité étudiée, il est possible de positionner tous les composants dans une seule et même famille. Le détail des composants de méthodes peut être trouvé dans (Iacovelli, 2012) sur le modèle présenté dans les figures 3 et 9. La représentation graphique de la famille de méthodes est donnée à la figure 12.

```
LPA = ({CMC},{CMV},{PVD},{PVR},{C},{D})
```

où
CMC = {AGILE-1}
CMV = {DSDM-1, DSDM-2, DSDM-3, DSDM-4, DSDM-5, DSDM-6, DSDM-7, DSDM-8, XP-1, XP-2, SCRUM-1, SCRUM-2}
PVD = {(Commencer={DSDM-1,* ; DSDM-2,* ; SCRUM-1,* ; XP-1,*}), (Etudier la faisabilité du projet={DSDM-3,*;DSDM-4,*}), (Définir les besoins du projet={DSDM-5,*;XP-2,*; SCRUM-2,*;DSDM-7,*;DSDM-8,*}), (Planifier le projet={DSDM-6,* ; AGILE-1,*})}
PVR = {(Etudier la faisabilité du projet={DSDM-1,*;DSDM-2,*}) (Définir les besoins du projet={XP-1,*;SCRUM-1,*; DSDM-4,*;DSDM-6,*}) (Planifier le projet={DSDM-5,*;XP-2,*;SCRUM-2,*; DSDM-8,*}) (Arrêter={DSDM-3,*;DSDM-7,*;AGILE-1,*})}
C = {(C1=DSDM-5⊕SCRUM-2),(C2=DSDM-5⊕XP-2),(C3=SCRUM-2⊕XP-2)}

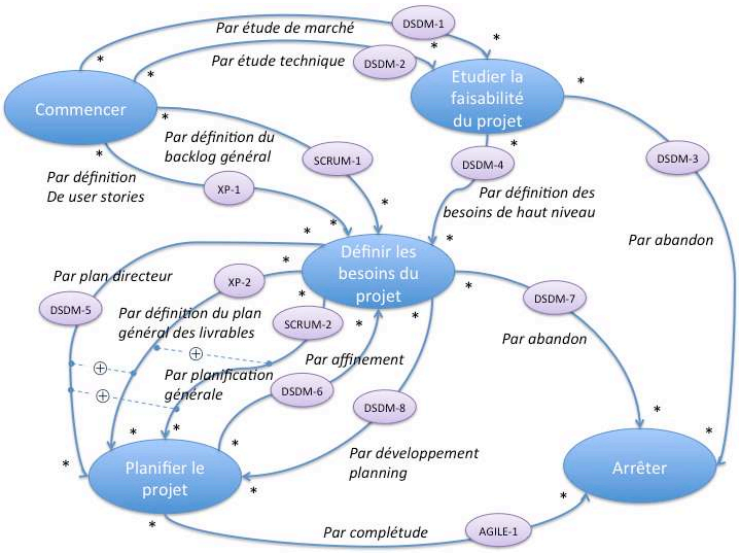


Figure 12. Famille de méthodes LPA

5.4. Construction d'une ligne de méthode.

Prenons le cas d'une petite entreprise voulant développer un nouveau logiciel. L'équipe de développeurs chargée du projet veut suivre une démarche agile et connaît bien la méthode XP. Elle souhaiterait cependant modifier quelques parties du processus en utilisant certaines fonctionnalités d'autres méthodes agiles. La famille LPA est utilisée pour la première phase du projet.

Les caractéristiques du projet sont les suivantes : *Taille du projet* : petite ; *Complexité du projet* : faible ; *Degré d'innovation du projet*: faible ; *Degré de risques liés au projet* : faible ; *Degré d'interaction avec les utilisateurs finaux du*

système : faible ; *Implication du client dans le processus de développement* : forte ; *Coût de mise en œuvre de l'alternative pour l'équipe* : 6 ; *Type de planification* : {au produit, au délai, au budget} ; *Respect des délais* : facilement ; *Longueur d'un cycle itératif :* court.

1ère étape : Le départ du processus offre un choix entre quatre composants de méthode : DSDM-1, DSDM-2, SCRUM-1 et XP-1. Les deux composants de la méthode DSDM ont pour objectif d'étudier la faisabilité du projet. Cependant, les caractéristiques du projet permettent d'éliminer cette étape puisque le projet est petit, d'une faible complexité et peu innovant. Le facteur de risque de ce projet est donc relativement faible. Les deux autres composants permettent d'identifier les besoins, celui de XP étant plus axé sur la définition de scénarios. Le composant SCRUM-1 étant plus utile lorsque le projet est complexe, le composant XP-1 est choisi dans cette situation de développement.

2ème étape : La planification du projet offre également un choix d'un des quatre composants : DSDM-5, DSDM-8, SCRUM-2 et XP-2. Les composants DSDM et XP n'offrent qu'une planification selon le produit alors que l'équipe de projet souhaite effectuer une planification axée à la fois sur le produit, le délai et le budget. Ce sera donc le composant SCRUM-2 qui sera choisi dans ce cas.

3ème étape : Deux possibilités sont offertes ensuite à l'équipe de projet : soit d'effectuer un affinement des besoins avec le composant DSDM-6, soit d'arrêter le processus avec AGILE-1. Ce dernier est un composant commun de la famille donc il sera intégré automatiquement. Le premier composant sera conservé dans la ligne de méthode pour permettre à l'équipe de projet plus de flexibilité dans la définition des besoins et la planification en autorisant des itérations successives de ces deux étapes.

La ligne de méthode obtenue est illustrée à la figure 13.

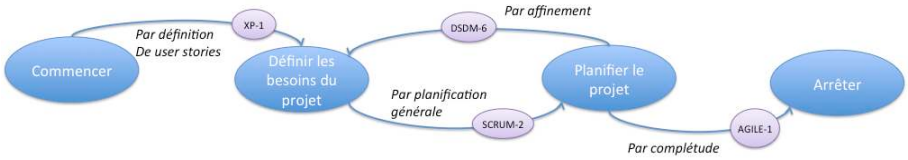


Figure 13. Ligne de méthode configurée pour le projet

Plus de détails sur ce cas d'étude, les caractéristiques de projet et les facteurs de contingence associés à ces différents composants sont dans (Iacovelli, 2012).

6. Conclusion

La notion de famille de méthodes a été définie dans le but de faciliter la réutilisation et l'adaptation des méthodes en pratique. Une famille de méthodes est composée d'un ensemble de composants, venant de méthodes diverses mais appartenant à la même unité téléologique. La combinaison des composants différents permet d'obtenir une assistance méthodologique bien adaptée au projet en cours. De plus, tandis que l'offre de composants de méthode est augmentée, l'effort

d'ingénierie est significativement réduit. Au lieu de créer une méthode spécifique à un projet à partir de zéro, comme il est suggéré par nombre d'approches de construction de méthodes situationnelles, l'utilisateur de la famille configurera une ligne de méthode en choisissant les composants de méthode appropriés, sans avoir à fournir d'effort supplémentaire pour les assembler. Le fait de ne pas être dépendant de la structure du composant permet de donner une plus grande facilité dans l'utilisation de la famille. De plus, la construction de la famille pour un domaine spécifique de projets facilite son acceptation du point de vue de l'utilisateur. En effet, le processus de recherche des composants est un processus parfois complexe qui devient complètement transparent dans une famille. Les familles de méthodes montrent de manière explicite les composants communs ou variables selon le contexte. La contextualisation permet de prendre en compte la caractérisation du projet, la caractérisation des composants et l'accord entre le composant et la situation. La notion de famille de méthodes et la formalisation du contexte associé aux composants et aux points de variation sont détaillées dans (Kornyshova *et al.,* 2011b).

Le processus de construction de famille de méthodes est fortement lié à la notion de variabilité, qui est à la base du concept de famille de méthodes. La combinaison des questions d'analyse de variabilité dans le domaine des LdP avec les techniques de l'IMS forme la base de notre approche de construction de famille de méthodes. La contribution principale de cette proposition consiste à tenir compte de la variabilité dans la construction de familles de méthodes. Nous y avons détaillé le méta-modèle de la famille de méthodes et la représentation de la variabilité ainsi qu'un modèle de processus pour la construction de famille de méthodes qui combine les principes de base de l'ingénierie des méthodes situationnelles et l'analyse de la variabilité. Ce modèle de processus est illustré dans cet article avec la construction d'une famille de méthodes de lancement de projets agiles.

Nos perspectives de recherche se concentreront sur l'analyse de la variabilité appliquée à la configuration de familles de méthodes, en portant une attention particulière aux manières de considérer les contraintes de variabilité (nécessite et exclut) pendant la configuration. Nous avons aussi l'intention d'adapter notre vision du processus de construction de famille aux méthodes utilisant des architectures de méthode orientées service. De plus, l'approche de construction de famille de méthodes sera étendue et testée en pratique dans des études de cas et projets divers, ce qui nous permettra de réaliser des évaluations qualitatives mais aussi quantitatives de cette approche.

7. Remerciements

Merci à Adrian Iacovelli pour avoir fourni la base de composants de méthodes agiles ayant servi à développer l'exemple de cet article.

8. Bibliographie

Asadi M., Mohabbati B., Gašević D., Bagheri E. (2011). Developing Families of Method-Oriented Architecture. *ME 2011*, pp. 168-183, IFIP AICT 351, Springer, Heidelberg.

Bessai B., Claudepierre B., Saidani O., Nurcan S. (2008). Context-aware business process evaluation and redesign, *Proceedings of BPMDS'08*, 2008.

Brinkkemper S., (1996). Method engineering: engineering of information systems development method and tools*, Information and Software Technology*, 38:7.

Brinkkemper S., Saeki M., Harmsen F. (1998). Assembly Techniques for Method Engineering. *Proceedings of CAiSE 1998*, pp. 381-400, LNCS 1413, Springer Verlag.

Clements P., Northrop L. (2001). *Software Product Lines: Practices and Patterns*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Deelstra S., Sinnema M., Nijhuis J., Bosch, J. (2004). COSVAM: A Technique for Assessing Software Variability in Software Product Families. *ICSM'04*, Chicago, USA, pp. 458-462.

Deneckere R. (2001). *Approche d'extension de méthodes fondée sur l'utilisation de composants génériques*. Thèse de doctorat, Université de Paris 1-Sorbonne, 2001.

Deneckère R., Iacovelli A., Kornyshova E., Souveyet C. (2008). From method fragments to method services, *Proceedings of EMMSAD 2008*, Montpellier, France.

Deneckere R., Kornyshova E. (2010). Process Line Configuration : an Indicator-based Guidance of the Intentional Model MAP, EMMSAD 2010, Hammamet, Tunisie.

Deneckere R., Kornyshova E., (2011), Processus téléologique et variabilité: Utilisation de la sensibilité au contexte, Journal *Ingénierie des Systèmes d'Information (ISI),* 16:1 p. 61-88.

Drury J. L., Scott S. D. (2008). Awareness in unmanned aerial vehicle operations, *International C2 journal*, Geoffrey N. Hone, 2:1.

Firesmith D.G., Henderson-Sellers, B. (2002). *The OPEN Process Framework: An Introduction.* Addison-Wesley, London, UK, 330 p.

Guzélian G., Cauvet C. (2007). SO2M : Towards a service-oriented approach for method engineering, *Proceeding of IKE'07*, Las Vegas, Nevada, USA.

Harmsen F., Brinkkemper S., Han Oei J.L. (1994). Situational method engineering for information system project approaches, *Methods and Associated Tools for the Information Systems Life Cycle conference*, 1994, pp 169-194.

Harmsen A. F. (2007) Situational Method Engineering. Doctoral dissertation University of Twente.

Henderson-Sellers B., Gonzalez-Perez C., Ralyté J. (2008). Comparison of Method Chunks and Method Fragments for SME. *ASWEC 2008*, pp. 479-488, Los Alamitos, CA, USA.

Iacovelli A. (2012). *Approche orientée service pour la configuration de méthodes outillées*. Thèse de doctorat, Université Paris 1 Panthéon-Sorbonne, 2012.

Karlsson F., Ågerfalk P.J. (2004). Method Configuration: Adapting to Situational Characteristics while Creating Reusable Assets, *Journal IST*, Vol.46 (9).

Keeney R.L., Raiffa H. (1993). *Decisions with Multiple Objectives : Preferences and Value Trade-Offs*, Cambridge University Press.

Kirsch Pinheiro M., Vanrompay Y., Berbers Y. (2008). Context-aware service selection using graph matching, *Proceedings of ECOWS 2008*, vol. 411.

Kornyshova E. (2011). *MADISE: Method Engineering-based Approach for Enhancing Decision-Making in Information Systems Engineering*., PhD thesis, Paris, France, 2011.

Kornyshova E., Deneckere R., Claudepierre B., (2011). Towards Method Component Contextualization, *Journal IJISMD,* vol 2, 4 p. 49-81.

Kornyshova E., Deneckère R., Rolland C. (2011). Method Families Concept: Application to Decision-Making Methods. *Enterprise, Business-Process and Information Systems Modeling*, pp. 413–427, LNBIP 81, Springer.

Kornyshova E., Ralyté J., Deneckère R. (2013). Constructing Method Families Based on the Variability Analysis, RCIS 2013 forum.

Kumar K., Welke R.J. (1992). Methodology Engineering: A Proposal for Situation Specific Methodology Construction. *Challenges and Strategies for Research in Systems Development*, Cotterman, W. and J. Senn (eds.), J. Wiley, Chichester, UK, pp. 257-266.

Mirbel, I. (2004). Rethinking ISD methods: fitting project team members profiles. *Proceedings of ISD 2004*, Vilnius, Lituania, pp. 103-113.

Mirbel I., Ralyte J. (2006) Situational method engineering: combining assembly-based and roadmap-driven approaches, *Requirement Engineering Journal*, 11(1), pp. 58—78.

Pohl K., Böckle G., van der Linden F. (2005). *Software product line engineering: foundations, principles and techniques,* Springer, Berlin Heidelberg New York.

Ralyté J. (2001) Ingénierie de méthodes par assemblage de composants. Thèse de doctorat en informatique, Université Paris 1 – Sorbonne, 2001.

Ralyté J., Deneckère R., Rolland, C. (2003). Towards a Generic Model for Situational Method Engineering. *Proceedings of CAiSE 2003*, pp. 95-110, LNCS 2681, Springer.

Ralyté J., Rolland C. (2001). An Approach for Method Reengineering. *Proceedings of ER 2001*, pp.471-484, LNCS 2224, Springer-Verlag.

Ralyté J., Rolland C. (2001b). An Assembly Process Model for Method Engineering. *Proceedings of CAISE 2001*, LNCS 2068, Springer, Berlin, pp. 267-283.

Rolland C., Prakash N., Benjamen A. (1999). A Multi-Model View of Process Modelling, Requirements Engineering Journal, 4(4), pp 169-187, Springer.

Rolland C., Prakash N. (2007). On the Adequate Modeling of Business Process Families, *Proceedings of BPMD 2007*, Trondheim, Norway.

Rolland C. (2007). Method Engineering: Trends & Challenges. Keynote Talk at IFIP WG 8.1 Working Conference ME'07, http://www.cs.uu.nl/groups/OI/me07/.

Rolland C (2007b). Capturing System Intentionality with Maps. In Conceptual Modelling in Information Systems Engineering, edited by John Krogstie, Andreas Lothe Opdahl, and Sjaak Brinkkemper, 141–158. Springer Berlin Heidelberg,

Rosemann M., Recker J. (2006). Context-aware process design: exploring the extrinsic drivers for process flexibility, *CAISE'06 workshops*, Luxembourg, p 149-158.

Rosen M. A., Fiore S. M., Salas E., Letsky M., Warner N. (2008). Tightly coupling cognition: understanding how communication and awareness drive coordination in teams, *International C2 journal*, 2:1.

Rossi, M., Ramesh, B., Lyytinen, K., Tolvanen, J-P. (2004). Managing evolutionary method engineering by method rationale, *Journal of the AIS*, 5(9): 356-391.

Roy B. (1996). Multicriteria Methodology for Decision Aiding, Dordrecht, Kluwer Ac. Pub.

Van Gurp J. (2000). *Variability in Software Systems, the key to Software Reuse*, Licentiate Thesis, University of Groningen, Sweden.

Weiss D.M. and Lai C.T.R. (1999). *Software product-line engineering: a family-based software development process*. Addison-Wesley.

Wistrand W., Karlsson F. (2004). Method components: rationale revealed, *Proceedings of CAISE'04*, Springer-Verlag. Riga, Latvia.