Vers une meilleure compréhension de la Composition de Services par Méta Modélisation d'un Service Composite

Anthony Hock-koon*, Mourad Oussalah*

*Université de Nantes, LINA, 2 rue de la Houssinière 44322 Nantes, France
anthony.hock-koon@univ-nantes.fr,
mourad.oussalah@univ-nantes.fr

Résumé. La composition de services est un des enjeux principaux des Architectures Orientées Services (AOS). Elle a pour vocation la maximisation des réutilisations en permettant les combinaisons de ressources existantes. Ces ressources, encapsulées sous la notion de service, collaborent afin de réaliser une tâche complexe. Quantité de travaux se focalisent sur la composition de services et la résolution de ses nombreux problèmes. Cependant, la multitude des approches et leur caractère souvent spécialisé ne permettent pas d'avoir une vision globale de la composition de services qui soit indépendante de toutes technologies ou tous domaines d'application. Notre article s'inscrit dans cette logique d'explicitation de la composition de services. Il propose un métamodèle de service composite qui réifie d'un seul tenant l'ensemble des caractéristiques d'une composition de services. Il définit leurs interdépendances et assure la capacité de réutilisation de cette composition. De plus, nous définissons un mécanisme d'auto composition qui permet des modifications dynamiques de l'architecture du composite et des logiques de compositions associées.

1 Introduction

Les Architectures Orientées Services ou AOS (OASIS, 2008; OpenGroup, 2009) sont un paradigme de développement logiciel couramment utilisé dans le design et le déploiement d'applications variées : allant de logiciels purement business (style ERP (Themistocleous et al., 2001)) à des services Web (Alonso et al., 2003), en passant par des environnements d'intelligence ambiante ou Iam (Weiser, 1991).

L'AOS repose sur la notion de service. Le service est utilisé pour encapsuler l'hétorégénéité des ressources et se focalise principalement sur les fonctionnalités. Il est associé à l'emploi de mécanismes spécifiques tels que la publication ou la composition de services. La philosophie de l'AOS peut être résumée par une volonté d'homogénéisation et d'automatisation de mécanismes permettant l'exploitation de ressources à nature hétérogène.

Cet article se focalise sur un de ces mécanismes : la *composition de services*. La composition est un des maillons essentiels de l'AOS. Il veut faciliter les réutilisations pour améliorer la productivité dans le développement. Ainsi, un architecte sera capable d'identifier les services disponibles qui correspondent à ses besoins, puis de les incorporer dans son application. Composer ces services sélectionnés, c'est définir les communications afin qu'ils coordonnent leurs

actions pour réaliser une tâche complexe de plus haut niveau. La définition de cette collaboration pose différents problèmes qui sont les sujets d'étude de nombreux travaux. Cependant la quantité importante de ces travaux et leur vision souvent parcellaire dressent un portrait morcellé de la composition.

Cet article veut combler ce manque de lisibilité. Il présente un métamodèle de service composite qui réifie une composition de service en tant que service à part entière. Ce métamodèle explicite l'ensemble des caractéristiques d'une composition de services au niveau architectural. Il identifie leurs interdépendances et permet une compréhension globale du mécanisme de composition. Cette compréhension est la base pour la définition d'un mécanisme d'auto composition. L'auto composition est la capacité du composite à modifier son architecture et à prendre en compte ces modifications dans ses logiques de compositions. De plus, notre métamodèle se veut indépendant de tous domaines d'application ou de technologies particulières.

La suite de cet article se déroule suivant ce plan. Dans la section 2, nous mettons en perspectives les carences théoriques existantes autour de la notion de composition de services et de sa réification en tant que service composite. Nous présentons les motivations de notre approche, les principaux challenges qu'un composite explicite doit surmonter et les objectifs que notre métamodèle veut atteindre. Nous présentons notre métamodèle dans la section 3. Nous définissons ses éléments architecturaux et leurs dépendances. La section 4 présente un court exemple d'instanciation d'un service composite. La section 5 aborde l'implémentation du prototype. La section 6 introduit la problématique du couplage faible. Enfin, la section 7 conclut et pose les perspectives futures.

2 Service composite : synthèse et perspectives

Il existe de nombreuses définitions formelles du paradigme AOS (listées dans Erickson et Siau (2008)) telles que OASIS (2008); OpenGroup (2009). Chacune d'elles tente d'expliciter l'ensemble des concepts de l'orienté services. D'une manière générale, elles définissent la composition de services comme décomposée en trois phases : la *découverte de services*, la *sélection de services* et la *composition de services*. La découverte de services correspond à l'identification parmi les *services concrets* disponibles d'un ensemble de services candidats qui peuvent répondre aux besoins, les *services abstraits*, exprimés par l'architecte. La sélection de services identifie parmi ces services candidats les plus adéquats. Enfin, la composition de services correspond à l'établissement de la coopération entre ces services sélectionnés.

Notre approche s'inscrit principalement dans cette troisième phase de définition de la coopération.

Pour organiser et illustrer les travaux existants autour de cette phase de composition des services nous utilisons un exemple très simple. Soit une composition entre deux services : un Microphone et un Jukebox. Un utilisateur exprime vocalement le titre d'une chanson qu'il veut écouter. Le microphone récupère son choix et le transfère au jukebox qui l'exécute. Ces services sont hétérogènes dans leur mode de d'implémentation et de communication. Le microphone gère des flux audio. Le jukebox récupère un string pour l'expression du choix de la chanson.

Un premier problème se pose sur la modélisation et la réalisation des collaborations entre ces services. C'est à dire définir l'ordonnancement entre les services et les échanges de données impliqués. Certains travaux existants ont des approches génériques (Pfeffer et al., 2008;

Moser et al., 2008) sur la modélisation des collaborations. D'autres ont une vision plus spécifique, délimitée par des problèmes ciblés. Par exemple les problèmes de distribution ou de sécurité dans les services Web (Milanovic et Malek, 2004; Mitra et al., 2008; Chang et al., 2008; Subramanian et al., 2008) ou encore la restriction des ressources et leur mobilité dans les Iam (Kalasapur et al., 2007; Lee et al., 2006).

Un second problème est posé par la nature hétérogène des services qui ne peuvent pas forcément se comprendre directement. Dans notre exemple, le microphone transfère des flux audio alors que le jukebox ne comprend que des chaînes de caractères. Ainsi différents travaux se focalisent sur les problèmes d'interopérabilité entre services (Roman et al., 2006; Beauche et Poizat, 2008; OASIS, 2009).

Cependant, un autre enjeux principal des AOS s'additionne aux problèmes précédents : la flexibilité des applications. En AOS, une composition de services doit être capable d'évoluer rapidement en fonction des contraintes de l'environnement ou des besoins de l'architecte (Bottaro et al., 2007; Chibani et al., 2008). Dans notre exemple, supposons que le service de jukebox n'est plus disponible et que deux services, lecteur MP3 et enceinte, sont disponibles et peuvent assurer le remplacement. Idéalement, le système doit être capable de modifier dynamiquement sa composition pour retirer le service défectueux et incorporer le ou les services remplaçants. Hors, la redéfinition des services présents dans la composition induit obligatoirement une coordination avec les deux aspects précédents de gestion de la collaboration et des hétérogénéités. Cependant un modèle qui réutilise les approches précédentes et qui explicite leur coordination dynamique reste encore à définir.

Ainsi, nous voulons définir ces règles de coordination et les encapsuler dans un seul modèle. Nous proposons donc un métamodèle de service composite qui réifie l'ensemble des aspects précédents afin de réaliser une composition flexible de services. De plus, cette approche par service composite assure :

- la réutilisation facile des compositions qui sont exposées comme des services à part entière ;
- une gestion hiérarchisée plus claire des niveaux de compositions (composition de composites) ;
- une utilisation simple des mécanismes prometteurs du paradigme AOS tel que la réplication de services ou la migration de services (Ameling et al., 2008; Grimshaw et al., 2009).

Cette définition d'un service composite se justifie également par son absence totale dans les spécifications formelles existantes (Erickson et Siau (2008) : OASIS, OpenGroup, W3C, IBM, etc.). Ces dernières sont incomplètes et définissent un service composite simplement comme un flux d'informations et de contrôle sur des services individuels (à la manière de BPEL). Elles ne rendent pas explicite la gestion de l'ensemble des notions issues des travaux sur la composition de services (hétérogénéité, adaptation, etc.).

Dans cet article, nous présentons un métamodèle de service composite qui encapsule l'ensemble des caractéristiques associées à une composition de services. Il explicite les dépendances entre ces caractéristiques pour assurer leur coordination lors de modifications de l'architecture du composite. Le but étant de mieux comprendre l'ensemble des interactions qui existent dans un service composite puis de maximiser sa flexibilité. La gestion des coordinations définit ce que l'on appelle l'auto composition.

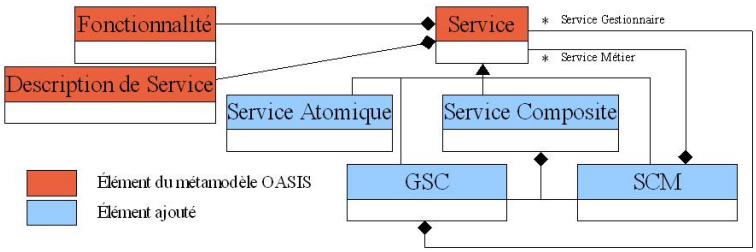

FIG. 1 – *Métamodèle du Service Composite*

3 Métamodèle : définition

La construction de notre métamodèle se base sur la réification des différentes propriétés d'une composition de services. Chacune de ces propriétés est clairement identifiée et définie. Puis nous les associons à un élément architectural précis de notre métamodèle. De plus, ce métamodèle est construit comme une extension du métamodèle de Service de l'OASIS (OASIS, 2008). Cela permet sa localisation dans l'ensemble des concepts portés par l'AOS.

Ainsi, un service composite est un service. Il en hérite donc l'ensemble des propriétés (OASIS, 2008; OpenGroup, 2009) telles que les notions de description de service ou de fonctionnalités (cf Figure 1). Par opposition à un service atomique, un service composite est une composition d'un ou plusieurs services : les *services constituants*. Nous répartissons ces services constituants en *services métiers* et *services gestionnaires* du composite :

- *services métiers* : services fonctionnels ou non fonctionnels (sécurité, etc.) qui sont sélectionnés pour répondre aux besoins de l'architecte lors de la définition de son composite. Ces services fournissent leurs fonctionnalités sans connaissance globale sur la composition où ils participent. Les services métiers sont réunis dans le *Service Composite Métier* ou SCM.
- *services gestionnaires* : services spécialisés dans la gestion des logiques de composition. Ils gèrent les autres services constituants et ont une connaissance globale de la composition. Les services gestionnaires sont réunis dans le *Gestionnaire de Service Composite* ou GSC.

Ainsi, un service composite peut être vu comme un ensemble de services métiers qui réalisent les tâches définies par l'architecte et les services gestionnaires qui assurent les bonnes coopérations. Cette distinction est rendue visible au niveau architectural par le service composite constitué des SCM et GSC (cf Figure 1). De plus, cette classification des services constituants offre une compréhension claire des niveaux d'exposition. Le premier niveau visible est la partie fonctionnelle qui portent concrètement sur la tâche du composite. Le second niveau représente le non fonctionnel tel que la sécurité, l'identification des client du composite etc. Et enfin le dernier niveau, la partie gestionnaire qui assure la mécanique opératoire entre tout ces éléments.

Notre travail d'explicitation d'une composition de services se focalise principalement sur la définition du GSC. Nous identifions clairement les caractéristiques de gestion attendues et les réifions au niveau de l'architecture de notre service composite en tant que services ges-

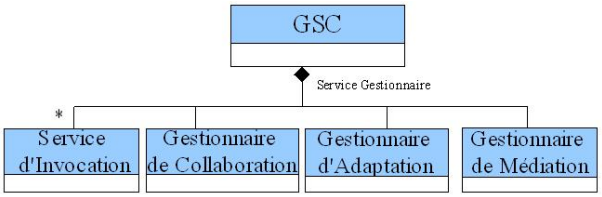

FIG. 2 – *Métamodèle du GSC*

tionnaires. Nos contributions portent non seulement sur cette clarification mais aussi sur la définition des interdépendances entre les rôles des gestionnaires. Par la suite, ces interdépendances seront la base de la définition du mécanisme d'auto composition.

3.1 Gestionnaire de Service Composite (GSC)

Le GSC réunit l'ensemble des services gestionnaires qui sont totalement transparents aux utilisateurs. Il représente la partie invisible du composite, chargée des mécaniques opératoires liées aux logiques de composition. À partir des travaux existants autour de la composition de services, nous pouvons abstraire quatre rôles principaux :

- l'invocation : la capacité du composite à déclencher l'exécution des services constituants (Roman et al., 2006; OASIS, 2009) ;
- la collaboration : la capacité du composite à connaître quels services métiers invoquer et de coordonner ces invocations (Pfeffer et al., 2008; Moser et al., 2008) ;
- la médiation : la capacité du composite à assurer la bonne compréhension des données échangées entre ses services constituants (Roman et al., 2006; Beauche et Poizat, 2008) ;
- l'auto adaptation : la capacité du composite à modifier son architecture, c'est à dire ajouter ou retirer des services constituants en fonction du contexte environnemental (OASIS, 2008; Bottaro et al., 2007; Chibani et al., 2008).

Cependant, on peut remarquer que les modifications de l'architecture d'un composite impliquent obligatoirement un changement sur les logiques de composition afin d'assurer la cohérence avec le nouvel ensemble de services. Les rôles d'invocation, de collaboration et de médiation doivent être correctement mis à jour pour garantir cette consistance. L'auto composition va correspondre à la gestion dynamique de ces dépendances. Dans un premier temps nous associons chacun des rôles précédents à des services gestionnaires précis. Dans un second temps nous définissons les dépendances entre ces services gestionnaires et localisons leurs gestions au niveau du GSC. Le GSC représente "l'intelligence" de notre composite chargé d'assurer sa consistance globale.

3.2 Services gestionnaires

Les quatre rôles précédents sont associés à quatre types de service gestionnaire.

Les **services d'Invocation** (cf Figure 2) : pour la capacité d'invocation du composite. Ils représentent les passerelles technologiques qui gèrent les hétérogénéités des services. Ce service d'Invocation est dépendant de l'interface du service constituant dont il assure l'invocation. Il

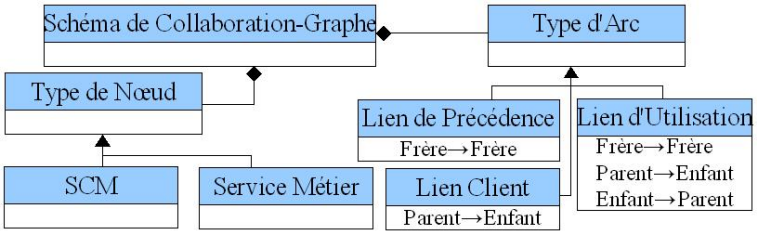

FIG. 3 – *Schéma de Collaboration*

construit le message d'invocation adéquat, en fonction des contraintes technologiques. Chaque service constituant du composite est donc lié à un service d'Invocation particulier qui peut être l'identité si des communications directes sont possibles.

Le **gestionnaire de collaboration** (cf Figure 2) : pour diriger les collaborations entre services métiers. Il maintient le *schéma de collaboration* qui spécifie le flux de travail (workflow) et le flux de données (dataflow). Le flux de travail définit l'ordonnancement des invocations, établissant les *liens de précédence* entre services métiers. Le flux de données définit les échanges de données entre services métiers, établissant les *liens d'utilisation*. Ce schéma définit aussi le *lien client* entre les services métiers et le SCM du composite. En effet, les services métiers ne sont pas forcément "physiquement" incorporer dans le composite mais établissent une relation client fournisseur avec ce dernier.

Nous modélisons ce schéma de collaboration par un graphe avec pour noeuds les services métiers et le SCM, et pour arcs les *liens de précédence*, *d'utilisation* et *client* (cf Figure 3). Suivant la hiérarchie de composition, les services métiers ont une relation frère-frère, et une relation parent-enfant avec le SCM (cf Figure 3).

Le **gestionnaire de médiation** (cf Figure 2) : pour diriger les médiations entre services métiers. L'hétérogénéité des services induit irrémédiablement des incompatibilités sur les données échangées. Pour assurer la compréhension des informations transmises d'un service à l'autre, des transformations peuvent être nécessaires. Le gestionnaire de médiation identifie ces besoins entre les services métiers. La réalisation de ces transformations met en évidence un type particulier de service métier : les *services de médiation* ou *Mservices*. Comme les services métiers, ces Mservices n'ont aucune connaissance globale de la composition et se limitent à l'exécution de leur fonction de transformation de données. Ainsi, le gestionnaire de médiation maintient un *schéma de médiation* qui définit des *liens utilisations* entre services métiers et Mservices. Ce schéma précise quel(s) Mservice(s) est utilisé pour garantir la compréhension de données échangées entre deux services métiers.

De la même façon que pour le schéma de collaboration, nous modélisons le schéma de médiation par un graphe avec pour noeuds les services métiers, le SCM et les Mservices et pour arcs le *lien d'utilisation*.

Le **gestionnaire d'adaptation** (cf Figure 2) : pour les modifications sur l'architecture. Le gestionnaire d'adaptation gère la présence des services métiers dans le composite. Il assure les ajouts ou les retraits des services du SCM (services métiers ou Mservices). Lors des ajouts de services, le gestionnaire d'adaptation suit les étapes classiques de construction d'une composition de services : la découverte, la sélection de services. Ainsi, il réutilise un moteur de

découverte capable de réaliser ces étapes à partir de la description des services requis par le composite. Le gestionnaire d'adaptation maintient un *schéma de composition* qui représente les services métiers présents dans le composite.

Le schéma de composition est un graphe avec pour noeuds les services métiers, le SCM et les Mservices et pour arcs le *lien de composition*.

La modélisation des trois schémas (collaboration, médiation, composition) est faîte de façon homogène. Les noeuds des graphes sont des services constituants présents dans notre métamodèle. Les arcs représentent des dépendances entre ces noeuds.

Cette identification des différents types de service gestionnaire et de leur rôle offre une vision claire du service composite. De plus, elle est la base pour l'explicitation de l'ensemble des dépendances imbriquées. Ces dépendances permettent la spécification de notre mécanisme d'auto composition : la capacité du composite à modifier son architecture en fonction du contexte et d'assurer la consistance avec les logiques de composition.

3.3 Dépendances entre services gestionnaires

Les modifications de l'architecture d'un composite ont des conséquences directes sur les logiques de composition. Les rôles d'invocation, de collaboration et de médiation, portés par les différents services gestionnaires, dépendent des services métiers présents dans le composite. Et cette présence des services métiers est sous l'influence du gestionnaire d'adaptation.

Nous avons localisé la gestion de ces dépendances au niveau du GSC qui regroupe les quatre types de service gestionnaire (cf Figure 2). Le GSC est donc en charge du mécanisme d'auto composition. Ainsi, à partir de la description des services identifiés comme défectueux, le GSC invoque le gestionnaire d'adaptation. Ce dernier retire ces services du composite et les remplace. Le schéma de composition est modifié. Le GSC doit ainsi gérer les impacts possibles sur :

- les autres schémas : les nouveaux services métiers peuvent nécessiter de nouveaux flux de travail ou de données. De même, les différences d'interfaces peuvent entraîner de nouveaux besoins en médiation.
- les services d'invocation : ces services étant basés sur les interfaces, les nouveaux services métiers ne sont pas forcément compatibles avec les services d'invocation présents.

Ainsi, nous avons défini trois types de **dépendance de schémas** entre les services gestionnaires : les liens de dépendance *AdaptationVersCollaboration*, *AdaptationVersMédiation*, *CollaborationVersMédiation*.

Les différents schémas maintenus par les services gestionnaires étant modélisés par des graphes, les dépendances de schémas expriment les impacts des modifications d'un graphe sur les autres graphes. Pour expliciter ces impacts, chacun des graphes est associé à un certain nombre de modifications admises. Ces modifications représentent des combinaisons de retraits/ajouts de noeuds/arcs qui portent une sémantique particulière tout en garantissant la consistance du graphe. Ainsi les dépendances de schémas sont définies par des implications des modifications admises d'un graphe vers des modifications admises sur les autres graphes.

En raison de la limitation du nombres de pages, nous ne détaillons pas les modifications admises sur les différents graphes. De plus, la description des trois dépendances de schéma précédentes va se limiter à des explicitations haut niveau et une simple présentation de leurs sous dépendances.

AdaptationVersCollaboration :

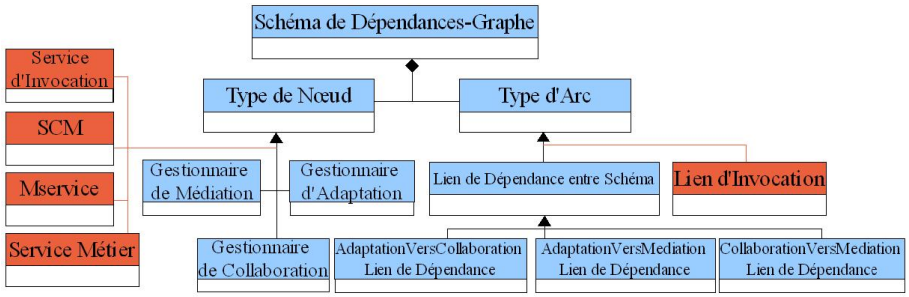

FIG. 4 – *Schéma de Dépendances*

La redéfinition de l'ensemble des services métiers par le gestionnaire d'adaptation implique des modifications au niveau du flux de travail et du flux de données. Ainsi, les nouveaux services métiers doivent être ajoutés dans le schéma de collaboration et de nouveaux liens d'utilisation, de précédence et de client doivent être définis. La dépendance AdaptationVersCollaboration est raffinée en trois sous dépendances différentes entre les modifications admises des deux graphes. Chacune des dépendances gèrent respectivement les conséquences impliquant le flux de travail, le flux de données, et les liens clients.

AdaptationVersMédiation :

La redéfinition de l'ensemble des services métiers induit des modifications dans le schéma de médiation qui possède ces services métiers comme noeuds. Ainsi, les services métiers doivent être mis à jour. Cependant cette mise à jour peut poser des problèmes de compatibilité avec les Mservices maintenus par le gestionnaire de médiation, les Mservices étant basés sur les interfaces. Ces nouveaux besoins en médiation peuvent déclencher une recherche de nouveaux Mservices par le gestionnaire d'adaptation. Ainsi la dépendance AdaptationVersMédiation est raffinée en deux sous dépendances qui gèrent respectivement les services métiers et les conséquences sur les Mservices.

CollaborationVersMédiation :

Une redéfinition du flux de données impliquent des échanges de données différents et donc des changements possibles au niveau des médiations. Une modification du schéma de collaboration qui impliquerait les liens d'utilisation a un impact direct sur la médiation portant sur la donnée échangée. Le schéma de médiation doit être mis à jour. Cette dépendance CollaborationVersMédiation est décrite par une seule dépendance précise entre certaines modifications admises.

D'une façon homogène avec les schémas de collaboration, médiation et composition, nous définissons un schéma maintenu par le GSC à partir des dépendances précédentes. Ce *schéma de dépendances* est défini comme un graphe avec pour noeuds les services gestionnaires et pour arcs les dépendances de schémas AdaptationVersCollaboration, AdaptationVersMédiation, CollaborationVersMédiation (cf Figure 4, éléments en bleu).

Enfin, le GSC doit gérer une dernière dépendance : la **dépendance d'invocation**.

En effet, les modifications de l'architecture du composite correspondent à des changements sur l'ensemble des services métiers présents. Pour pouvoir assurer la bonne exécution du com-

posite, le GSC doit être capable d'invoquer ces nouveaux services. Ainsi, de nouveaux services d'invocation peuvent être nécessaires pour pouvoir garantir cette capacité d'invocation.

Pour maintenir ces informations dans le GSC nous étendons le graphe des dépendances (cf Figure 4, éléments en rouge). Les noeuds services métiers, SCM, Mservices et service d'invocation sont reliés par les arcs *lien d'invocation*.

3.4 Mécaniques opératoires

Dans les sections précédentes nous avons décrit les éléments architecturaux de notre service composite et leurs dépendances. Nous apportons maintenant les mécaniques opératoires utilisant ces éléments qui permettent : d'une part d'exécuter le service composite pour qu'il réalise ses fonctionnalités, et d'autre part d'exécuter le processus d'auto composition.

Le GSC centralise l'intelligence de notre composite. Il est donc en charge de l'ensemble des exécutions. Lorsqu'un utilisateur du service composite demande l'exécution d'une des fonctionnalités, le GSC invoque le gestionnaire de collaboration. Ce dernier lui indique le premier service métier à déclencher pour débuter la réalisation de la fonction voulue. Le GSC fait appel au service d'invocation associé à ce service métier afin de pouvoir interagir avec lui. Le gestionnaire de collaboration indique aussi les échanges de données attendues. Pour assurer la compréhension dans ces échanges, le GSC invoque le gestionnaire de médiation qui lui précise quels Mservices vont effectuer les transformations.

Le mécanisme d'auto composition se déclenche dynamiquement en cas de services métiers défectueux. Le rôle d'identification de ces services est attribué aux services d'invocation. En effet, les services d'invocation sont en contact direct avec les services métiers et peuvent donc analyser leurs réponses (absence de réponses, qualité de services modifié, temps de réponse plus long, etc.). Ainsi, lorsque le GSC tente d'invoquer un service métier via le service d'invocation associé, si ce dernier lui indique ce service métier comme indisponible alors le GSC déclenche l'auto composition. Il fait appel au gestionnaire d'adaptation en lui fournissant la description du service défectueux. En fonction des services ajoutés par le gestionnaire d'adaptation, le GSC invoque les gestionnaires de collaboration et de médiation pour qu'ils mettent à jour leur schéma. Ces différentes invocations suivent les trois dépendances établies : AdaptationVersCollaboration, AdaptationVersMédiation, CollaborationVersMédiation.

4 Exemple d'instanciation

Dans cette section nous reprenons l'exemple du jukebox et présentons un cliché des différents schémas associés aux services gestionnaires.

Ainsi, le service composite est composé d'un SCM et d'un GSC. Le SCM groupe les services métiers Microphone et Jukebox ainsi que le Mservice qui assure la transformation du choix de la chanson (d'un flux audio vers un string). Le GSC est composé du gestionnaire de collaboration, du gestionnaire de médiation, du gestionnaire d'adaptation et d'un service d'invocation pour chaque service constituant. La figure 5 partie gauche Microphone-Jukebox représente les graphes des différents schémas et offre un cliché des dépendances entre services. Prenons l'exemple du schéma de collaboration. Le flux de données est représenté par un lien d'utilisation sur le choix de la chanson entre les services métiers Microphone et Jukebox. Le flux de travail est représenté par un lien de précédence définissant l'exécution du Microphone

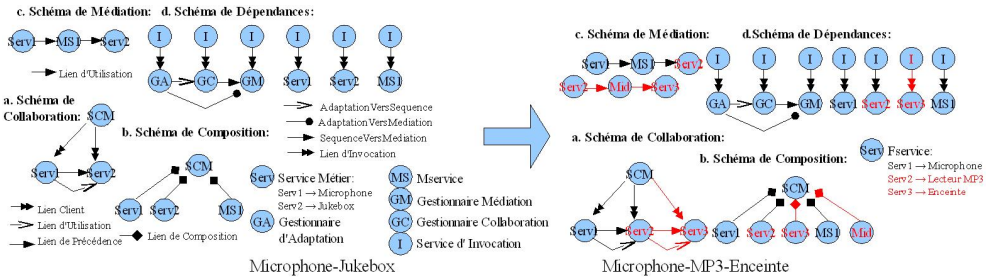

FIG. 5 – *Exemple d'instanciation du service composite Microphone Jukebox*

avant celle du Jukebox. Et enfin des liens clients représentent la relation client fournisseur entre composite et constituants.

La figure 5 partie droite Microphone-MP3-Enceinte représente ces même graphes après l'exécution d'un processus d'auto composition (les éléments en rouge représentant les changements). Dans ce cas, le service métier Jukebox a été identifié par son service d'invocation comme indisponible. Lors de la recherche de remplaçants par le gestionnaire d'adaptation, ce dernier a trouvé deux services capables de remplir les même fonctions : un lecteur MP3 et une enceinte. Le gestionnaire d'adaptation indique les modifications de son graphe (Figure 5 Schéma de composition droit) au GSC qui gère par la suite les impacts sur les autres graphes. On a supposé que le lecteur MP3 répond à la même interface que le jukebox. Ainsi le Mservice de transformation du choix de la chanson a pu être réutilisé (Figure 5 Schéma de médiation droit), de même que le service d'invocation précédemment lié au jukebox (Figure 5 Schéma de dépendances droit). Cependant un service d'invocation particulier a été nécessaire pour le service métier enceinte (Figure 5 Schéma de dépendances droit). Le nouveau lien d'utilisation qui existe en lecteur MP3 et enceinte (Figure 5 Schéma de collaboration droit), bien que ne nécessitant pas médiation, a été associé au Mservice identité (Figure 5 Schéma de médiation droit).

5 Implémentation

L'implémentation d'un prototype validant notre métamodèle se fait dans les technologies SCA (OASIS, 2009). SCA est une plate-forme de développement d'applications s'incrivant dans les AOS. Notre objectif est d'étendre le modèle SCA et de proposer une version améliorée de SCA Tools (Eclipse, 2009).

Le principe de l'extension va être d'incorporer notre métamodèle de service composite dans le métamodèle de SCA. Ainsi, un utilisateur aura le choix entre implémenter un composite SCA ou implémenter notre composite. Cette nouvelle entité logicielle proposera une gestion explicite des collaborations et médiations qui n'est pas présente dans SCA. De plus, le nouvel outil sera capable de générer dynamiquement nos services gestionnaires à partir des informations fonctionnelles fournies par l'architecte. Les différents services gestionnaires assureront la réalisation du mécanisme d'auto composition qui améliorera la robustesse des applications.

Notre extension de SCA Tools est toujours en cours de développement et les pistes d'implémentations que nous allons citer peuvent être amenées à changer. Cependant le principe général va être la réutilisation au maximum des technologies portées par SCA. Ainsi les services d'invocation seront directement supportés par la capacité de SCA Tools à faire interagir des technologies différentes. Le gestionnaire de collaboration sera défini par un composant SCA contenant un BPEL. La gestion des médiations sera clairement identifiée. Enfin, le gestionnaire d'adaptation réutilisera un annuaire (tel que pour les Web Services (Alonso et al., 2003)) et un GSC coordonnera l'ensemble pour la réalisation de l'auto composition.

6 Vers une définition du couplage faible

La notion de couplage faible (OASIS, 2008; OpenGroup, 2009) est une propriété visée par l'AOS. Elle est en rapport direct avec la composition de services. Diminuer le couplage entre services d'une composition, c'est limiter leurs dépendances. Ainsi, les propagations des erreurs se trouvent plus contenues et les remplacements de services deviennent plus faciles. Cette notion de couplage est intuitivement compréhensible et les bénéfices sur la qualité des applications sont immédiatement tangibles.

Cependant, cette intuitivité dans la définition cache un manque de théorisation formelle. En effet, les différents modèles de composition de services cherchent le plus souvent à diminuer ce couplage et réalisent cette diminution par l'apport de nouveaux mécanismes. Chacune de ces approches se focalise sur un aspect particulier de la composition. Il est donc d'autant plus difficile de les comparer qu'il n'existe pas de cadre globale de mesure du couplage entre services.

Dans la continuité du travail d'explicitation d'un service composite, nous voulons apporter une meilleure compréhension de la notion de couplage faible. Sa définition se base en partie sur les dépendances entre services mises en évidence par notre métamodèle. En effet, les concepts portés par notre métamodèle étant issus de la réification d'approches existantes, ils représentent des points de comparaisons communs. Brièvement, nous avons défini trois couplages différents :

- le couplage sémantique : est basé sur la vision haut niveau de l'architecte qui exprime les fonctionnalités de son composite par des ensembles de collaborations entre services abstraits. Il représente les dépendances sémantiques issues de l'expertise de l'architecte dans le domaine d'application ciblé.
- le couplage syntaxique : représente le couplage entre les services abstraits (l'expression des besoins de l'architecte) et les services concrets (les services candidats). Il explicite dans sa mesure les solutions alternatives possibles pour la réalisation du composite.
- le couplage lexical ou physique : représente l'ensemble des métriques classiques utilisées pour mesurer les dépendances physique entre entités logicielles implémentées (par exemple : nombre d'appels de méthodes, objets communs, etc.)

Chacun de ces couplages est associé à un étalonnage de son poids et à un ensemble métriques permettant de l'évaluer. Le couplage global d'un composite correspond à l'union de ces couplages.

7 Conclusion

Cet article présente un métamodèle de service composite qui apporte une meilleure compréhension du concept de composition de services. En effet, il rend explicite l'ensemble des mécanismes associés à cette composition et gère leurs interdépendances. À partir de ces dépendances nous avons défini le principe d'auto composition. Ainsi, notre métamodèle spécifie un service composite capable de modifier dynamiquement son architecture. Ces modifications de l'architecture sont prises en compte par les services gestionnaires afin d'assurer la bonne coopération de l'ensemble des services constituants.

De plus, nous montrons brièvement les lacunes dans la notion de couplage faible qui est un concept central dans la composition de services. Nous montrons comment notre métamodèle peut être réutilisé pour expliciter cette notion et apporter les méthodes de sa mesure.

Dans un premier temps, nos travaux futurs vont se focaliser sur la finalisation de notre outil de développement. Nous voulons rendre accessible à la communauté Eclipse une version étendue du SCA Tools. Ses utilisateurs pourront implémenter des services composites conformes à notre métamodèle. Cette version publique va permettre des applications grandeur nature de notre approche et ainsi avoir des retours sur son utilisation.

Dans un second temps, nous allons nous concentrer sur la publication de notre étude du couplage faible. Nous allons finaliser la formalisation des critères d'évaluation et ainsi fournir une étude comparative précise des approches existantes de composition de services. L'objectif est de compléter une boîte à outils de développement de service composite qui soit capable d'assister un architecte. Ces outils vont lui permettre d'appréhender simplement les services critiques qui sont liés à un couplage trop important ; et en dernier lieu, de lui proposer des guides de développement.

Références

Alonso, G., F. Casati, H. Kuno, et V. Machiraju (2003). *Web Services*. isbn 978-3-540-44008-6.

Ameling, M., M. Roy, et B. Kemme (2008). Replication in service oriented architectures. In *ICSOFT (PL/DPS/KE)*, pp. 103–110.

Beauche, S. et P. Poizat (2008). Automated service composition with adaptive planning. In *ICSOC*, pp. 530–537.

Bottaro, A., A. Gérodolle, et P. Lalanda (2007). Pervasive service composition in the home network. In *AINA*, pp. 596–603.

Chang, Y.-C., P. Mazzoleni, G. A. Mihaila, et D. Cohn (2008). Solving the service composition puzzle. In *IEEE SCC (2)*, pp. 387–394.

Chibani, A., K. Djouani, et Y. Amirat (2008). Semantic middleware for context services composition in ubiquitous computing. In *MOBILWARE*, pp. 9.

Eclipse (2009). Soa tools platform : Sca tools project. http ://www.eclipse.org/stp/sca/.

Erickson, J. et K. Siau (2008). Web services, service-oriented computing, and service-oriented architecture : Separating hype from reality. *J. Database Manag. 19*(3), 42–54.

Grimshaw, A. S., M. M. Morgan, et K. Sarnowska (2009). Ws-naming : location migration, replication, and failure transparency support for web services. *Concurrency and Computation : Practice and Experience 21*(8), 1013–1028.

Kalasapur, S., M. Kumar, et B. A. Shirazi (2007). Dynamic service composition in pervasive computing. *TPDS 18*.

Lee, S., J. Lee, et B. Lee (2006). Service composition techniques using data mining for ubiquitous computing environments. *IJCSNS 6*.

Milanovic, N. et M. Malek (2004). Current solutions for web service composition. *IEEE Internet Computing 8*(6), 51–59.

Mitra, S., R. Kumar, et S. Basu (2008). Optimum decentralized choreography for web services composition. In *IEEE SCC (2)*, pp. 395–402.

Moser, O., F. Rosenberg, et S. Dustdar (2008). Viedame - flexible and robust bpel processes through monitoring and adaptation. In *ICSE Companion*, pp. 917–918.

OASIS (2008). Reference architecture for service oriented architecture 1.0. http ://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.pdf.

OASIS (2009). Service component architecture assembly model specification version 1.1. http ://www.oasis-opencsa.org/.

OpenGroup (2009). Soa source book. http ://www.opengroup.org/projects/soa-book/.

Pfeffer, H., D. Linner, et S. Steglich (2008). Modeling and controlling dynamic service compositions. *ICCGI*. isbn 978-0-7695-3275-2.

Roman, D., J. de Bruijn, A. Mocan, H. Lausen, J. Domingue, C. Bussler, et D. Fensel (2006). Www : Wsmo, wsml, and wsmx in a nutshell. In *ASWC*, pp. 516–522.

Subramanian, S., P. Thiran, N. C. Narendra, G. K. Mostéfaoui, et Z. Maamar (2008). On the enhancement of bpel engines for self-healing composite web services. In *SAINT*, pp. 33–39.

Themistocleous, M., Z. Irani, R. M. O'Keefe, et R. J. Paul (2001). Erp problems and application integration issues : An empirical survey. In *HICSS*.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 94–104.

Summary

Service composition is a main challenge in the Service Oriented Architectures (SOA) paradigm. It provides mechanisms to combine available resources which are exposed as services. This mechanism tries to fulfill a key software engineering principle: maximize the reusability. Typically, a service composition specifies a cooperation between services which replies to some high level goals. Many service composition models were developed. Each of them tries to solve some particular problems. This paper intends to reify the relevant notions of these models in a Composite Service MetaModel. This composite approach provides an homogeneous reuse of existent compositions. This metamodel defines all interlaced features and provides a global and explicit vision of the service composition. Moreover, these specifications of interlaced features allow the definition of the auto composition mechanism. Our composite service can dynamically modify its architecture and its composition logics according to the environmental context.