# Une approche centrée exigences pour la composition de services web

Maha Driss, Yassine Jamoussi, Naouel Moha, Jean-Marc Jézéquel, Henda Hajjami Ben Ghézala

# Une approche centrée exigences pour la composition de services web

**Maha Driss*,** — Yassine Jamoussi** — Naouel Moha***
**Jean-Marc Jézéquel* — Henda Hajjami Ben Ghézala****

** IRISA/INRIA, Université de Rennes 1*
*Campus universitaire de Beaulieu*
*35042 Rennes, France*
*{mdriss, moha, jezequel}@irisa.fr*

*** ENSI, Laboratoire RIADI-GDL, Université de La Manouba*
*Campus universitaire de La Manouba*
*La Manouba, 2010, Tunisie*
*{yassine.jamoussi, henda.benghezala}@ensi.rnu.tn*

RÉSUMÉ. *Cet article présente une approche centrée exigences pour la composition de services web qui permet : (i) la modélisation des exigences des utilisateurs avec le formalisme la Carte et la spécification des services requis avec un modèle intentionnel de services (MIS) ; (ii) la découverte des services web pertinents en interrogeant le moteur de recherche de services Service-Finder ; (iii) la sélection automatique de services pertinents et de haute QdS par l'application de l'analyse formelle de concepts (AFC) ; et (iv) la génération automatique de processus de coordination BPEL par l'application de la technique de transformation de modèles. Dans cet article, nous illustrons notre approche par une application d'arrangement de conférences et nous la validons empiriquement en termes de précision et de rappel sur cette application.*

ABSTRACT. *This paper presents a requirement-centric approach for Web service composition which allows: (i) modeling users' requirements with the MAP formalism and specifying required services using an Intentional Service Model (ISM); (ii) discovering relevant Web services by querying the service search engine Service-Finder; (iii) selecting automatically relevant and high QoS services by applying Formal Concept Analysis (FCA); and (iv) generating automatically BPEL coordination processes by applying the model transformation technique. In this paper, we illustrate our approach with a conference arrangement application and we validate it empirically in terms of precision and recall on this application.*

MOTS-CLÉS : *composition de services web, exigences des utilisateurs, QdS, AFC, transformation de modèles.*

KEYWORDS: *web service composition, users' requirements, QoS, FCA, model transformation.*

DOI:10.3166/ISI.16.2.97-125 

## 1. Introduction

*Service-Oriented Computing* (SOC) est un nouveau paradigme pour le développement flexible, évolutif et à faible coût d'applications logicielles et distribuées à base de services web (Huhns *et al.,* 2005). SOC a été largement adopté car il offre la possibilité de construire efficacement des applications à valeur ajoutée en composant des services web déjà existants. L'idée de base du SOC est de modéliser les exigences des utilisateurs d'abord, puis de découvrir, sélectionner et coordonner les services web qui répondent le mieux aux exigences fonctionnelles et non fonctionnelles des utilisateurs. Les exigences fonctionnelles correspondent aux opérations fournies par les services web. Nous nommons les services qui répondent le mieux aux exigences fonctionnelles des services *pertinents*. Les exigences non fonctionnelles sont exprimées par le terme qualité de service (QdS) qui réfère à diverses propriétés telles que la disponibilité, le temps de réponse, la sécurité et le débit (Menascé, 2002). Modéliser les exigences des utilisateurs est réalisé en identifiant tout d'abord ces exigences, ensuite en les représentant dans un formalisme d'ingénierie des exigences. Découvrir les services web qui répondent aux exigences des utilisateurs est réalisé en interrogeant un moteur de recherche de services qui permet de parcourir les services web disponibles dans l'annuaire en considérant plusieurs critères de recherche (par exemple les opérations, la QdS, etc.). Parmi l'ensemble des services obtenus par la découverte, seuls les services qui répondent le mieux aux exigences fonctionnelles et non fonctionnelles des utilisateurs sont sélectionnés et coordonnés. Modéliser les exigences des utilisateurs est déjà atteint par des formalismes existants d'ingénierie des exigences qui sont étendus et adaptés pour répondre aux spécificités des applications à base de services. À cette fin, des formalismes comme TROPOS (Bresciani *et al.,* 2004), KAOS (Lamsweerde *et al.,* 2000) et la Carte (Rolland *et al.,* 2000) ont été utilisés pour modéliser des applications à base de services en termes des exigences des utilisateurs. Toutefois la découverte, la sélection et la coordination des services web sur la base des exigences des utilisateurs posent toujours problème et ceci pour trois causes principales :

– le nombre croissant et la diversité des services web en plus de leur publication sur plusieurs registres publics et privés font de la découverte de services pertinents une tâche difficile à accomplir ;

– le nombre souvent important de services web retournés par la découverte rend la sélection des services pertinents et de haute QdS une tâche coûteuse et fastidieuse ;

– la disparité des langages utilisés par les utilisateurs finaux et par les développeurs fait de la coordination des services à partir des exigences des utilisateurs une tâche délicate puisqu'elle peut entraîner des discordances conceptuelles.

Dans cet article, nous proposons une nouvelle approche centrée exigences qui permet : (i) la modélisation des applications à base de services en termes d'exigences fonctionnelles et non fonctionnelles; (ii) la découverte des services web pertinents qui répondent le mieux aux exigences fonctionnelles; (ii) la sélection des services pertinents et de haute QdS et (iv) la coordination des services sélectionnés en vue de

les orchestrer. Cette approche est l'extension du travail présenté dans (Driss *et al.*, 2010) dans lequel nous nous contentons de présenter la modélisation, la découverte et la sélection de services. Notre approche consiste donc en quatre étapes successives comme le montre la figure 1.

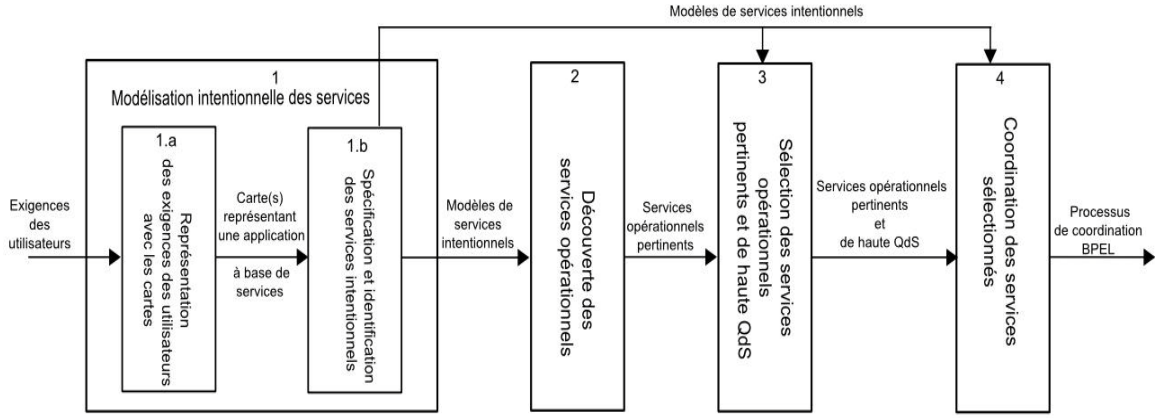


**Figure 1.** *Approche centrée exigences pour la composition de services web*

1) Étape 1. Modélisation intentionnelle des services : à cette étape, nous proposons d'utiliser le formalisme la Carte (Rolland *et al.*, 2000) afin d'assurer une modélisation intentionnelle des applications à base de services. La Carte permet la modélisation des exigences des utilisateurs dans une série de graphes composés d'*intentions* et de *stratégies*, appelés *cartes*. Dans des travaux antérieurs (Rolland *et al.*, 2007), un modèle appelé modèle intentionnel de services (MIS) a été proposé pour spécifier les services présentés par les cartes et qui sont nommés des services *intentionnels*. Dans cet article, le même modèle a été étendu afin d'inclure les aspects de QdS et sera utilisé pour spécifier les services intentionnels.

2) Étape 2. Découverte des services opérationnels : à cette étape, nous proposons de découvrir les services *opérationnels*, qui répondent aux intentions exprimées au niveau des services intentionnels, par l'interrogation du moteur de recherche de services *Service-Finder*[1]. L'interrogation se fait en utilisant des mots-clés qui sont extraits des modèles MIS des services intentionnels. Afin de sélectionner plus efficacement les services opérationnels pertinents et de haute QdS, nous proposons une filtration à trois niveaux des services découverts. Dans le premier niveau, nous éliminons les services redondants. Les services qui passent le premier niveau sont filtrés en considérant deux propriétés de QdS à savoir la validité (nous vérifions si les adresses URI des services

1. http ://www.service-finder.eu/

sont valides) et la disponibilité (nous vérifions si les services sont fonctionnels). Les services qui passent le deuxième niveau sont filtrés selon une mesure de similarité sémantique entre les spécifications intentionnelles fournies par MIS et les spécifications opérationnelles fournies par WSDL[2]. Seuls les services ayant une mesure de similarité supérieure ou égale au seuil empirique considéré sont retenus.

3) Étape 3. Sélection des services opérationnels pertinents et de haute QdS : à cette étape, l'ensemble des services retenus après l'étape de découverte est classé dans une structure ordonnée appelée *treillis de concepts* et ceci par l'application de l'analyse formelle de concepts (AFC) (Ganter *et al.*, 1999). L'AFC est un cadre formel qui permet de regrouper des individus qui partagent des propriétés communes et les organiser en des treillis de concepts. L'AFC permet d'automatiser la tâche de sélection et ceci en fournissant une vue claire et organisée des services afin de permettre aux utilisateurs d'identifier plus facilement les services pertinents et de haute QdS.

4) Étape 4. Coordination des services sélectionnés : à cette étape, nous proposons d'utiliser la technique de transformation de modèles pour la génération automatique des processus de coordination BPEL[3] à partir des modèles MIS des services intentionnels.

Notre approche est illustrée par un cas d'étude d'une application à base de services d'arrangement de conférences et elle est validée empiriquement en termes de rappel et de précision sur cette application.

Le reste de l'article est structuré comme suit. Dans la section 2, nous présentons la modélisation intentionnelle des applications à base de services. Dans la section 3, nous décrivons la découverte de services opérationnels effectuée à l'aide de *Service-Finder*. Dans la section 4, nous expliquons comment appliquer l'AFC pour sélectionner les services pertinents et de haute QdS. Dans la section 5, nous présentons les règles de transformation utilisées pour la génération automatique de processus BPEL à partir des modèles MIS des services intentionnels. Les résultats expérimentaux sont décrits dans la section 6. Dans la section 7, nous présentons un bref état de l'art portant sur les approches d'ingénierie des exigences qui traitent les applications à base de services et les approches qui appliquent l'AFC dans le domaine des services web. Finalement, dans la section 8, nous concluons en récapitulant les contributions apportées et en annonçant les perspectives de travaux de recherche futurs.

## 2. Modélisation intentionnelle des services

Un nombre considérable d'approches se rapportant à la modélisation des applications à base de services ont été proposées à la fois par des comités académiques et industriels. Ces approches visent à proposer des langages (par exemple BPEL, WS-CDL (Kavantzas *et al.*, 2005) et OWL-S[4] et des formalismes

2. http ://www.w3.org/TR/wsdl20/
3. http ://www.oasis-open.org/committees/tc_home.php ?wg_abbrev=wsbpel
4. http ://www.w3.org/Submission/OWL-S/

(par exemple les réseaux de Petri (Mecella *et al.,* 2002), les machines de Mealy (Berardi *et al.,* 2003) et les machines d'états (Foster *et al.,* 2004)) pour la modélisation des applications à base de services. Toutes ces approches adoptent une modélisation « centrée fonctions » fournissant une description de « bas niveau » qui se base sur des déclarations techniques (comme les messages de coordination, les paramètres d'entrée/sortie et le protocole web utilisé pour la communication). Toutefois, les utilisateurs ont besoin d'interagir avec les développeurs pour obtenir des applications à base de services qui répondent le mieux à leurs exigences. Pour cela, une application à base de services doit être modélisée en termes d'exigences des utilisateurs et non pas en termes de déclarations techniques. Dans cet article, nous adoptons une approche « centrée exigences » qui permet une modélisation de « haut niveau » d'abstraction des applications à base de services. Cette approche utilise le formalisme la Carte pour représenter les exigences des utilisateurs.

La Carte est un formalisme de représentation qui permet la modélisation de processus ou d'applications logiciels en termes intentionnels. Nous qualifions les services présentés par les cartes des services *intentionnels*. Nous avons choisi d'utiliser le formalisme la Carte, premièrement parce que ce formalisme a été déjà utilisé pour la modélisation des applications à base de services web (Rolland *et al.,* 2007 ; Mirbel *et al.,* 2010 ; Rolland *et al.,* 2010) donc nous pouvons réutiliser les connaissances antérieures, et deuxièmement parce que nous voulons bénéficier des atouts de ce formalisme à savoir :

– la modélisation explicite de l'intentionnalité : la Carte modélise explicitement l'aspect intentionnel des applications à base de services. Le point de vue intentionnel d'une application à base de services facilite la découverte et la sélection de services. En effet, à partir de ce qu'il souhaite, l'utilisateur peut identifier les services qui répondent le mieux à ses exigences ;

– la simplicité du langage : la Carte utilise un langage simple et facile à comprendre par des utilisateurs non experts du domaine des services web à la différence des langages de services qui exigent des connaissances techniques approfondies ;

– l'expression de la variabilité : la Carte donne une multitude de chemins entre deux intentions. Chaque chemin correspond à une variante d'usage de l'application à base services ;

– la notion d'affinement : la Carte représente une application à base de services à différents niveaux de granularité. Cette notion d'affinement offre une certaine modularité. En effet, chaque carte d'affinement peut être vue comme une composition à part qui peut être réutilisée par d'autres applications.

Dans la suite, nous donnons un aperçu du formalisme la Carte, nous présentons le modèle MIS permettant la spécification des services intentionnels et nous expliquons les directives à suivre pour identifier les services intentionnels et leur composition à partir des cartes.

### 2.1. *Représentation des exigences des utilisateurs avec les cartes*

La Carte est un formalisme de représentation des processus et des applications logiciels dans un mode intentionnel. Une carte[5] est un graphe étiqueté et orienté dans lequel les nœuds sont des *intentions* et les arcs sont des *stratégies*. Une intention est un but que nous désirons atteindre. Dans chaque carte, nous trouvons deux intentions particulières appelées *Démarrer* et *Arrêter*. L'intention *Démarrer* permet de commencer la navigation dans une carte alors que l'intention *Arrêter* permet de la terminer. Une stratégie est une manière permettant la réalisation d'une intention.

La figure 2 représente les exigences des utilisateurs pour une application à base de services d'arrangement de conférences avec le formalisme la Carte. Cette application permet l'arrangement du transport, du logement et du divertissement aux participants d'une conférence.

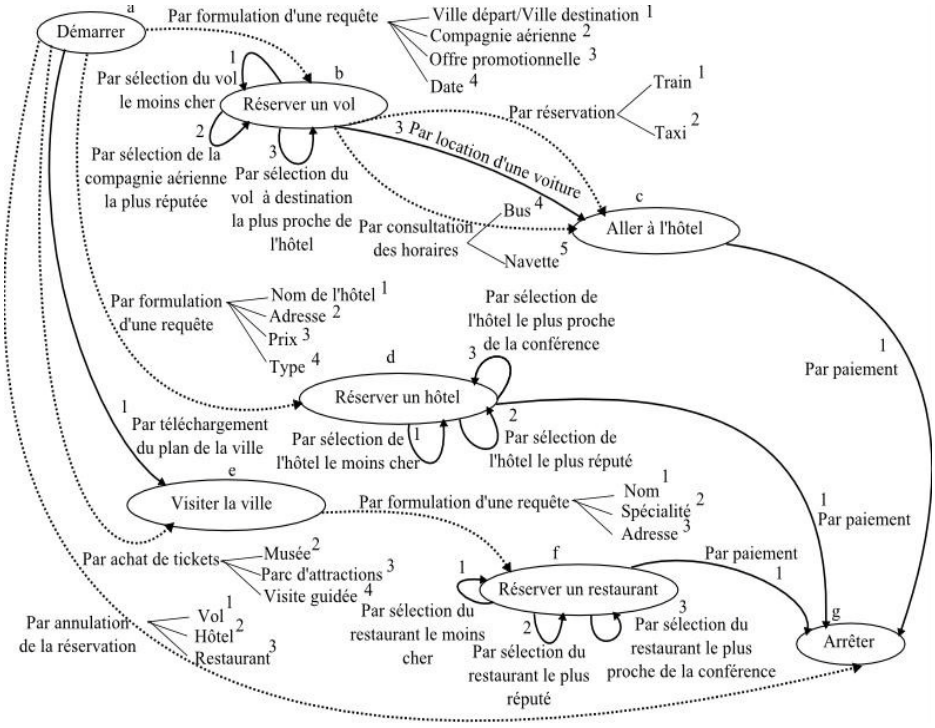


**Figure 2.** *La carte d'une application à base de services d'arrangement de conférences*

5. Dans la suite de cet article, nous faisons la différence entre Carte qui correspond au formalisme et carte qui correspond à une instance de la Carte.

La carte de cette application a cinq intentions, à savoir *Réserver un vol*, *Aller à l'hôtel*, *Réserver un hôtel*, *Visiter la ville* et *Réserver un restaurant*, et a plusieurs stratégies permettant de satisfaire chacune de ces intentions. Par exemple, pour satisfaire l'intention *Visiter la ville*, les utilisateurs peuvent suivre deux stratégies différentes : *Par téléchargement du plan de la ville* ou *Par achat de tickets*.

L'élément principal d'une carte est la *section*. En effet, une carte est composée d'une ou plusieurs sections. Une section est représentée sous la forme d'un triplet <Intention Source $I_i$, Intention Cible $I_j$, Stratégie $S_{ij}$>. Une section capture une manière spécifique d'atteindre une intention cible à partir d'une intention source. Par exemple, <*Démarrer*, *Réserver un vol*, *Par formulation d'une requête*> représente une manière d'atteindre l'intention cible *Réserver un vol* à partir de l'intention source *Démarrer* en suivant la stratégie *Par formulation d'une requête*. Il existe quatre relations entre les sections d'une carte : *paquet*, *multi-segment*, *chemin* et *multi-chemin*.

– Paquet : si deux sections ayant une même intention source et une même intention cible et sont liées par cette relation, alors l'exécution d'une section empêche l'exécution de l'autre. Les deux sections sont mutuellement exclusives ; elles sont liées par la relation logique $OU_{ex}$. Un paquet est représenté graphiquement par une flèche en pointillés. « $\otimes$ » est l'opérateur utilisé pour relier deux ou plusieurs sections exclusives. Dans la figure 2, <*Démarrer*, *Réserver un vol*, *Par formulation d'une requête*> est un paquet composé de quatre sections ayant chacune une stratégie différente : *Par ville départ/ville arrivée*, *Par compagnie aérienne*, *Par offre promotionnelle*, et enfin *Par date*. Ce paquet est noté $P_{ab}= \otimes (ab_1,ab_2,ab_3,ab_4)$[6].

– Multi-segment : si deux sections, ayant une même intention source et une même intention cible et sont liées par cette relation, alors elles peuvent être exécutées soit les deux ensemble soit l'une d'elles seulement. Les deux sections sont complémentaires ; elles sont liées par la relation logique ET/OU. « $\vee$ » est l'opérateur utilisé pour relier deux ou plusieurs sections complémentaires. Dans la figure 2, les trois stratégies *Par sélection de l'hôtel le moins cher*, *Par sélection de l'hôtel le plus réputé* et *Par sélection de l'hôtel le plus proche de la conférence* représentent trois façons différentes pour *Réserver un hôtel*. Les trois sections <*Réserver un hôtel*, *Réserver un hôtel*, *Par sélection de l'hôtel le moins cher*>, <*Réserver un hôtel*, *Réserver un hôtel*, *Par sélection de l'hôtel le plus réputé*> et <*Réserver un hôtel*, *Réserver un hôtel*, *Par sélection de l'hôtel le plus proche de la conférence*> forment un multi-segment. Ce multi-segment est noté $MS_{dd}= \vee (dd_1,dd_2,dd_3)$.

– Chemin : si deux sections, sont liées par cette relation, alors l'intention cible de la section qui précède est l'intention source de la section qui succède. Cette relation

6. Nous utilisons une codification locale pour désigner une section d'une carte. Les intentions sont codées par des lettres de l'alphabet. Les stratégies sont numérotées relativement à leurs intentions source et cible. Les niveaux d'affinement d'une carte vont de i à i+n. Par exemple, la section ab1i est une section de la carte de niveau i, son intention source est a, son intention cible est b et la stratégie permettant de réaliser l'intention b à partir de a est la stratégie 1.

introduit la notion de précédence/succession entre sections. Elle conduit à ordonner les sections dans un chemin. Une relation de type chemin repose sur le lien de précédence/succession noté par " • ". Dans la figure 2, les trois sections **<***Démarrer, Réserver un vol, Par formulation d'une requête : ville départ/ville arrivée***>**, **<***Réserver un vol, Réserver un vol, Par sélection du vol le moins cher***>** et **<***Réserver un vol, Aller à l'hôtel, Par location d'une voiture***>** constituent un chemin. Ce chemin est noté $C_{a,\{b\},c} = \bullet(ab_1, bb_1, bc_3)$.

– Multi-chemin : une intention cible peut être atteinte en suivant plusieurs chemins de la carte. Cette relation de type multi-chemin repose sur le lien " $\cup$ " entre les chemins alternatifs. Dans la figure 2, il y a trois chemins distincts pour atteindre l'intention *Arrêter* depuis l'intention *Démarrer*. Le premier est le chemin via les intentions *Réserver un vol* et *Aller à l'hôtel*. Le deuxième est le chemin via l'intention *Réserver un hôtel*. Et le troisième est le chemin via les intentions *Visiter la ville* et *Réserver un restaurant*. Ce multi-chemin est noté $MC_{a,\{b,c,d,e,f\},g} = \cup(C_{a,\{b,c\},g}, C_{a,\{d\},g}, C_{a,\{e,f\},g})$.

En général, une carte, de son intention *Démarrer* à son intention *Arrêter*, est un multi-chemin pouvant contenir des multi-segments.

Enfin, il est possible d'affiner une section par une carte entière. L'affinement est un mécanisme d'abstraction par lequel un assemblage complexe de sections au niveau i+1 est considéré comme une section unique au niveau i. La figure 3 montre l'affinement de la section <*Réserver un hôtel, Arrêter, Par paiement*> au niveau i par une carte au niveau i+1. Cette carte propose plusieurs chemins entre *Démarrer* et *Arrêter*. Chacun de ces chemins est équivalent à la section <*Réserver un hôtel, Arrêter, Par paiement*> du niveau i.

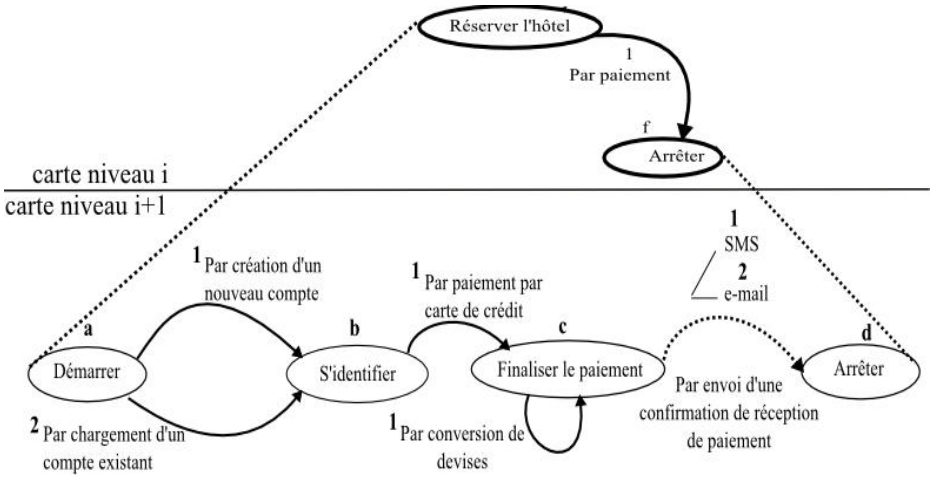


**Figure 3.** *L'affinement de la section* **<***Réserver un hôtel, Arrêter, Par paiement***>**

## 2.2. *Spécification et identification des services intentionnels*

### 2.2.1. *Spécification des services intentionnels*

Les services intentionnels sont des services présentés par les cartes. Ils permettent la réalisation des exigences des utilisateurs modélisées sous forme d'intentions à l'aide du formalisme la Carte. Les services intentionnels sont spécifiés par le modèle intentionnel de services MIS. La figure 4 présente le méta-modèle de MIS en utilisant les notations du diagramme de classes UML.

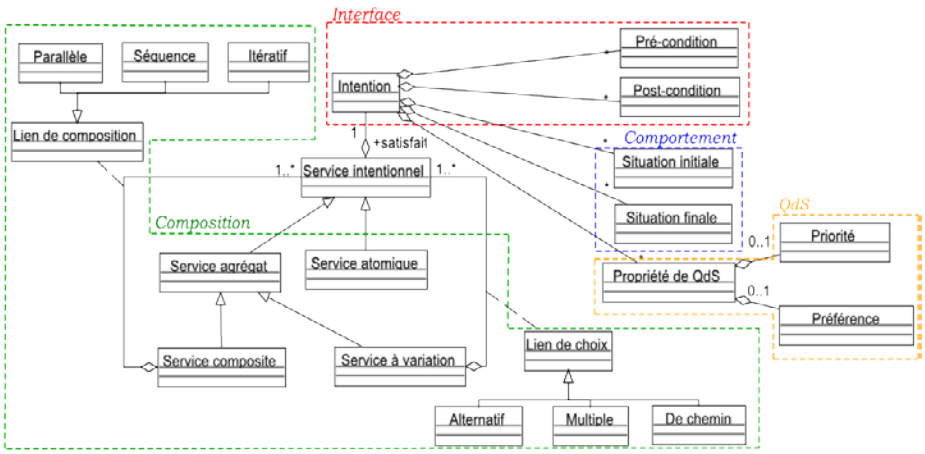


**Figure 4.** *Le métamodèle de MIS*

La figure 4 montre que la spécification d'un service intentionnel comporte quatre parties : l'interface, le comportement, la composition et la QdS. Nous décrivons chacune de ces parties dans les paragraphes suivants.

L'interface. Il y a trois éléments décrivant la partie interface qui sont : l'*Intention*, la *Situation initiale* et la *Situation finale*. L'*Intention* représente l'identité du service intentionnel, la *Situation initiale* définit les paramètres d'entrée et la *Situation finale* fournit les paramètres de sortie.

Le comportement. Il y a deux éléments décrivant la partie comportement d'un modèle intentionnel d'un service qui sont : la *Pré-condition* et la *Post-condition*. La *Pré-condition* et la *Post-condition* sont respectivement l'état initial et final, à savoir, l'état nécessitant la réalisation de l'intention et l'état résultant de sa réalisation.

La composition. Il y a deux types de services : un service *atomique* et un service *agrégat*. Un service *atomique* n'est pas décomposable en d'autres services intentionnels, alors qu'un service *agrégat* l'est. Un service *atomique* est associé à une intention dite « opérationnalisable », alors qu'un service *agrégat* est associé à

une intention de haut niveau qui doit être décomposable afin d'atteindre des sous-intentions opérationnalisables. Une intention opérationnalisable est une intention pour laquelle il est possible de définir une série d'opérations permettant sa réalisation. Dans ce sens, nous pouvons dire qu'un service *atomique* est directement exécutable, alors qu'un service *agrégat* ne l'est pas. Nous distinguons deux types de services *agrégats* : ceux qui sont opérationnalisables par une décomposition OU de l'intention, les *variants* et ceux qui sont opérationnalisables par une décomposition ET, les *composites*. Il existe trois types de variation : *choix alternatif*, *choix multiple* et *choix de chemin*. Le *choix alternatif* correspond à une relation $OU_{ex}$ entre services (nous associons au *choix alternatif* le symbole "$\otimes$"). Le *choix multiple* correspond à une relation OU entre services (nous associons au *choix multiple* le symbole "$\vee$"). Finalement, le *choix de chemin* correspond à des enchaînements alternatifs de services (nous associons au *choix de chemin* le symbole "$\cup$"). Il existe aussi trois types de composition : la *séquence*, le *parallélisme* et l'*itération*. Dans une *séquence*, les services composants sont exécutés d'une manière séquentielle (nous associons à la *séquence* le symbole "$\bullet$"). Dans un *parallélisme*, les services composants sont exécutés d'une manière parallèle (nous associons au *parallélisme* le symbole "//"). Dans une *itération*, les services composants sont exécutés à plusieurs reprises (nous associons à l'*itération* le symbole "$*$").

**La QdS.** Il y a trois éléments décrivant la partie QdS d'un modèle intentionnel d'un service qui sont : la *Propriété de QdS*, la *Priorité* et la *Préférence*. La *Propriété de QdS* exprime un critère de QdS comme le temps de réponse ou la disponibilité. Par la *Priorité*, nous associons une priorité à la *Propriété de QdS*, et par la *Préférence* nous exprimons une préférence.

### 2.2.2. *Identification des services intentionnels*

Afin d'identifier les services intentionnels et leur composition à partir d'une carte, nous proposons de suivre les trois directives suivantes (Rolland *et al.*, 2007) :

1) identification des services atomiques : un service atomique est associé à chaque section opérationnalisable d'une carte. Une section opérationnalisable est une section qui ne peut être affinée par une autre carte. En considérant les cartes de l'application de l'arrangement de conférences présentée par la figure 2 et la figure 3, nous identifions 38 services atomiques. Parmi ces services, nous citons : $ab_{1i} \rightarrow S_{\text{Formuler une requête: ville départ/ville d'arrivée}}$ et $bc_{1i+1} \rightarrow S_{\text{Payer avec carte de crédit}}$.

2) identification de tous les chemins d'une carte : pour identifier tous les chemins qui existent entre l'intention *Démarrer* et l'intention *Arrêter*, nous appliquons un algorithme permettant d'identifier tous les chemins dans un automate fini. Cet algorithme est une adaptation de l'algorithme de MacNaughton et Yamada (1960). Par application de cet algorithme sur la carte de la figure 2, nous obtenons des chemins comme : $C_{a,\{b,c\},g}=\bullet(P_{ab},MS_{bb},P_{bc},cg_1)$ avec $P_{ab}=\otimes(ab_1,ab_2,ab_3,ab_4)$, $MS_{bb}=\vee(bb_1,bb_2,bb_3)$,$P_{bc}=\otimes(bc_1,bc_2)$, et cg1=<*Aller à hôtel*, *Arrêter*, *Par paiement* >.

3) identification des services agrégats : nous utilisons les règles suivantes afin d'identifier les différents types de services agrégats à partir des sections d'une carte et les relations qui existent entre elles :

– R1 : Affecter à chaque paquet de la carte un service à choix alternatif.

– R2 : Affecter à chaque multi-segment de la carte un service à choix multiple.

– R3 : Affecter à chaque chemin de la carte un service composite séquentiel.

– R4 : Affecter à chaque multi-chemin de la carte un service à variation de chemin.

Dans le cas de la carte de la figure 2 et de la figure 3, nous identifions des services agrégats comme : $S_{\text{Visiter la ville}} = \vee (S_{\text{Télécharger le plan de la ville}}, S_{\text{Acheter des tickets}})$ et $S_{\text{Paiement}} = \bullet (S_{\text{S'identifier}}, S_{\text{Payer avec carte de crédit}}, S_{\text{Convertir la devise}}, S_{\text{Envoyer une confirmation}})$ qui sont respectivement un service à variation et un service composite.

## 3. Découverte des services opérationnels

Dans ce qui suit, nous allons montrer comment découvrir les services opérationnels correspondant aux services intentionnels de notre application d'arrangement de conférences. Nous considérons le service $S_{\text{Convertir la devise}}$ présenté par la section cc1 de la carte de niveau i+1. Ce service assure la conversion de devises. La figure 5 montre le modèle intentionnel du service $S_{\text{Convertir la devise}}$ qui est un fichier .xmi. Ce fichier est obtenu en instanciant le méta-modèle MIS.

Comme le montre la figure 5, $S_{\text{Convertir la devise}}$ est un service atomique identifié par l'intention *Convertir la devise*. $S_{\text{Convertir la devise}}$ prend deux paramètres en entrée qui sont *Devise* et *Reservation* et fournit en sortie *Conversion*. La pré-condition d'exécution de ce service est *Reservation.Etat=vrai* et sa post-condition est *Conversion.Valeur<>0*. $S_{\text{Convertir la devise}}$ doit être caractérisé par une disponibilité très élevée et un temps de réponse très faible. La priorité accordée au temps de réponse est 3, tandis que la priorité accordée à la disponibilité est 1. Les valeurs des préférences et des priorités des propriétés de QdS sont attribuées par l'utilisateur.

Pour découvrir les services opérationnels permettant l'exécution du service intentionnel $S_{\text{Convertir la devise}}$, nous interrogeons le moteur de recherche de services *Service-Finder*. *Service-Finder* est un environnement web 2.0 dédié à la découverte de services, il permet la recherche dans plus de 25 000 services web liés à plus de 200 000 pages web[7]. Nous avons choisi d'utiliser *Service-Finder* parce qu'il fournit pour chaque service découvert des informations sur sa QdS, plus précisément sur sa disponibilité et son temps de réponse. Ces informations sont obtenues par monitoring.

7. http ://demo.service-finder.eu/about

```
<?xml version="1.0" encoding="ASCII"?>
<mis:carte xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mis="http://ism/1.0"
xsi:schemaLocation="http://mis/1.0 MIS.ecore">
  <service xsi:type="ism:Atomique"
  intention="//@Intention.0"
  pre_condition="//@Pre-Condition.0"
  post_condition="//@Post-Condition.0"
  situation_initiale="//@Situation Initiale.0"
  situation_finale="//@Situation Finale.0"
  ressource.0="//@Ressource.0"
  ressource.1="//@Ressource.1"
  ressource.2="//@Ressource.2"
  propriete_qds.0="//@Propriete QdS.0"
  propriete_qds.1="//@Propriete QdS.1"
  priorite.0="//@Priorite.0"
  priorite.1="//@Priorite.1"
  preference.0="//@Preference.0"
  preference.1="//@Preference.1"
  id="Convertir la devise"/>
  <intention description="Convertir la devise"/>
  <pre_condition valeur="Reservation.Etat=vrai"/>
  <post_condition valeur="Conversion.Valeur<>0"/>
  <ressource.0 nom="Devise"/>
  <ressource.1 nom="Reservation"/>
  <ressource.2 nom="Conversion"/>
  <situation_initiale input="//@resource.0 //@resource.1"/>
  <situation_finale output="//@resource.2"/>
  <propriete_qds.0 id="temps de reponse">
    <priorite.0 valeur="3"/>
    <preference.0 valeur="Tres faible"/>
  </propriete_qds>
  <propriete_qds.1 id="Disponibilite">
    <priorite.1 valeur="1"/>
    <preference.1 valeur="Tres eleve"/>
  </propriete_qds.1>
</mis:carte>
```

**Figure 5.** *Le modèle intentionnel du service* $S_{Convertir\ la\ devise}$

L'interrogation de *Service-Finder* se fait par mots-clés et nous utilisons pour notre exemple « Convertir + Devise ». Ces mots-clés sont extraits à partir du modèle intentionnel du service $S_{Convertir\ la\ devise}$ présenté par la figure 5. L'extraction des mots-clés est faite en appliquant une méthode utilisée en recherche d'information qui est la méthode TF/IDF *(Term Frequency/Inverse Document Frequency)* (Salton *et al.,* 1988). Cette méthode permet de mesurer l'importance d'un terme contenu dans un document en calculant son poids qui équivaut au nombre de ses occurrences dans le document. Nous appliquons cette méthode sur le fichier .xmi présenté dans la figure 5 et plus précisément sur les valeurs des champs *intention id*, *intention description*, *pre_condition valeur*, *post_condition valeur*, *ressource.0 nom*, *ressource.1 nom*, et *ressource.2 nom*. Les deux mots ayant les pondérations les plus élevées sont « Convertir » et « Devise » et par conséquence sont les mots-clés du modèle intentionnel de $S_{Convertir\ la\ devise}$. En réponse à la requête « Convertir + Devise », *Service-Finder* retourne un ensemble de 76 services web[8] comme le montre la figure 6.

8. Ce résultat est obtenu le 20 janvier 2011.

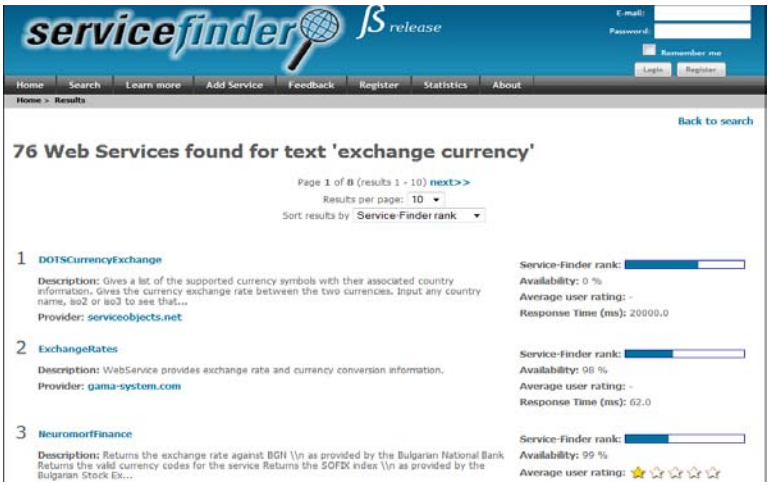


**Figure 6.** *Résultat d'interrogation de Service-Finder à la requête « Convertir + Devise »*

Pour réduire cet ensemble de services, nous procédons à une filtration à trois niveaux. Au premier niveau, nous éliminons les services redondants. Si un service apparaît plusieurs fois dans l'ensemble de services fourni par *Service-Finder* suite à son interrogation, alors nous laissons uniquement sa première occurrence. Pour le service $S_{\text{Convertir la devise}}$ et après ce premier niveau de filtration, nous avons un ensemble de 46 services parmi les 76 retournés par *Service-Finder*. Au deuxième niveau, nous examinons les services qui restent selon deux propriétés de QdS à savoir la validité et la disponibilité. Ces deux propriétés sont vérifiées comme suit :

– la validité : nous vérifions si l'adresse URI figurant au niveau de la spécification WSDL du service est valide. Un message du type "adresse invalide" est affiché si l'adresse URI d'un service est invalide ;

– la disponibilité : nous vérifions si le service est fonctionnel ; c'est-à-dire s'il fournit une réponse suite à son invocation. Un message du type « le service est momentanément indisponible » est affiché si un service ne répond pas suite à son invocation.

En appliquant ce deuxième niveau de filtration, nous obtenons un nouvel ensemble formé de 26 services seulement. Ces services sont passés à un troisième niveau de filtration qui est basé sur une mesure de similarité sémantique entre les modèles MIS des services intentionnels et les spécifications WSDL des services opérationnels. Pour ceci, nous avons implémenté une fonctionnalité qui sert à mesurer la similarité sémantique entre deux fichiers xml. Cette fonctionnalité se base sur la mesure de Pirrò *et al.* (2008) qui est déjà implémentée et disponible au niveau de la librairie JWSL *(Java WordNet Similarity Library)*. Le choix de cette librairie est justifié par son utilisation de la base très connue WordNet (Miller, 1995) qui offre l'avantage d'une grande couverture lexicale. Avec cette

fonctionnalité, un seuil empirique de 0.8 permet de discriminiser la similarité ou la dissimilarité sémantique entre deux fichiers xml. Ainsi, seuls les services opérationnels dont les spécifications WSDL ont une valeur de similarité sémantique normalisée supérieure ou égale à 0.8 avec les modèles MIS des services intentionnels auxquels ils correspondent sont maintenus. Ce troisième niveau de filtration génère un nouvel ensemble qui renferme 9 services web seulement. Ces 9 services sont les services pertinents qui répondent le mieux à l'intention *Convertir la devise*. Dans le tableau 1, nous donnons un résumé du nombre de services opérationnels obtenus après interrogation de *Service-Finder* et après chaque niveau de filtration.

| | ***Service-Finder*** | **Filtration** | | |
|---|---|---|---|---|
| **Nombre de services** | 76 | **1er niveau** | **2e niveau** | **3e niveau** |
| | | 46 | 26 | 9 |

**Tableau 1.** *Résultat de l'étape de découverte du service intentionnel* $S_{Convertir\ la\ devise}$

## 4. Sélection des services opérationnels pertinents et de haute QdS

Pour automatiser la sélection des services pertinents et de haute QdS, nous proposons d'utiliser l'analyse formelle de concepts (AFC) (Ganter *et al*., 1999).

### 4.1. *Introduction à l'AFC*

L'AFC est un cadre de travail formel qui permet de regrouper des individus qui partagent des propriétés communes. Les groupes d'individus et de propriétés mutuellement correspondants sont appelés des *concepts formels* (Willee, 1981). Un concept formel est constitué de deux parties : *l'extension* qui contient les individus appartenant au concept et *l'intension* qui contient les propriétés partagées par les individus. Ces concepts sont organisés en une hiérarchie, dite *treillis de concepts*, suivant une *relation d'incidence binaire* entre individus et propriétés. Dans l'AFC, les données sont représentées sous la forme d'un tableau booléen à deux dimensions, appelé *contexte binaire*, définissant la relation d'incidence binaire entre deux ensembles finis d'individus et de propriétés. Un contexte binaire se décrit formellement comme suit :

Un contexte binaire est un triplet $\mathcal{K} = (\mathcal{O},\mathcal{A}, \mathcal{I})$ où :

- $\mathcal{O}$ est l'ensemble des individus ;
- $\mathcal{A}$ est l'ensemble des propriétés ;
- $\mathcal{I}$ est une relation d'incidence binaire entre les éléments de $\mathcal{O}$ et les éléments de $\mathcal{A}$ telle que $\mathcal{I} \subseteq \mathcal{O} \times \mathcal{A}$ indiquant pour chaque individu les propriétés qui lui sont associées.

Nous avons choisi d'utiliser l'AFC parce qu'elle nous permet de sélectionner et de composer dynamiquement les services web. En effet, en réponse à des exigences de QdS, l'AFC fournit non pas un seul service présentant la solution optimale, mais un ensemble formé d'un ou de plusieurs services. Ces services partagent des propriétés communes de QdS et présentent ainsi des candidats de substitution qui peuvent être interchangés dynamiquement pour pallier des problèmes de pannes ou d'indisponibilités rencontrés lors d'une composition de services.

Pour notre problème de sélection de services, nous définissons un contexte $\mathcal{K}$ où les individus sont les services pertinents obtenus après l'étape de découverte et les propriétés représentent des propriétés de QdS. Nous considérons deux propriétés de QdS : la disponibilité et le temps de réponse. Les valeurs de la disponibilité et du temps de réponse des services sont fournies par *Service-Finder*[9] comme le montre la figure 6. La relation d'incidence binaire est : un service « est caractérisé par » une propriété de QdS. Par l'application de l'AFC, en considérant le contexte $\mathcal{K}$, nous voulons identifier les services pertinents et qui offrent le meilleur compromis entre disponibilité et temps de réponse.

Les valeurs de la disponibilité et du temps de réponse sont données en deux unités différentes (la disponibilité est en % et le temps de réponse est en ms). Pour cette raison, nous proposons d'échelonner les valeurs de ces deux propriétés de QdS afin de faire correspondre pour chaque échelon une valeur qualitative ordinale. Pour l'échelonnage, nous utilisons la technique statistique du boxplot (Chambers *et al.*, 1983). Un boxplot divise un ensemble de valeurs numériques en quatre quarts, ou quartiles. La figure 7 montre un exemple typique d'un boxplot.

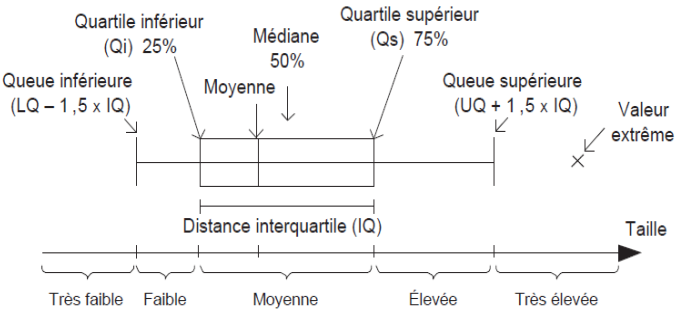


**Figure 7.** *Le boxplot*

Le quartile inférieur $Q_i$ d'un ensemble de valeurs est la valeur telle que 25 % des valeurs sont égales ou inférieures à cette valeur et il représente le 25$^e$ percentile. Le quartile supérieur $Q_s$ représente le 75$^e$ percentile. Les quartiles inférieur et supérieur de la distribution des valeurs représentent les limites du boxplot. La distance interquartile du boxplot, $IQ = Q_s - Q_i$, correspond à la distance entre le quartile

9. Ces valeurs sont obtenues le 20 janvier 2011.

inférieur et le quartile supérieur. La distance interquartile est utilisée pour calculer les queues de la distribution. Les valeurs des queues inférieure et supérieure sont calculées respectivement par $Q_i$ - 1,5 × IQ et $Q_s$ + 1,5 × IQ.

Nous associons des valeurs qualitatives ordinales définies avec une échelle de Likert à 5 points (Malhotra, 2007) aux quartiles du boxplot comme suit : très faible (très élevé, respectivement) correspond aux valeurs de la disponibilité et du temps de réponse en dessous de la queue inférieure (au-dessus de la queue supérieure, respectivement) ; faible (élevé, respectivement) correspond aux valeurs entre la queue inférieure et le quartile inférieur (entre la queue supérieure et le quartile supérieur, respectivement) ; moyen correspond aux valeurs comprises entre les quartiles inférieur et supérieur.

| | (D1) Disponibilité très faible | (D2) Disponibilité faible | (D3) Disponibilité moyenne | (D4) Disponibilité élevée | (D5) Disponibilité très élevée | (TR11) Temps de réponse très faible 1 | (TR12) Temps de réponse très faible 2 | (TR13) Temps de réponse très faible 3 | … | (TR51) Temps de réponse très élevé 1 | (TR52) Temps de réponse très élevé 2 | (TR53) Temps de réponse très élevé 3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ExchangeRates | × | × | × | × | | × | × | × | … | × | × | × |
| CurrencyServer | × | × | × | × | × | × | × | × | … | × | × | × |
| CurrencyService | × | × | × | × | × | × | × | × | … | × | × | × |
| ForeignExchangeRates | × | × | × | × | × | × | × | × | … | × | × | × |
| ExchangeRateWebService | × | × | × | | | | | | … | × | × | × |
| ForeignExchangeService | × | × | × | × | | | | | … | × | × | × |
| CurrencyRequest | × | × | × | × | | | | | … | × | × | × |
| Financial | × | × | × | | | × | × | × | … | × | × | × |
| IOSXMLReq | × | × | × | × | | | | | … | × | × | × |

**Tableau 2.** *Contexte $\mathcal{K}$ reliant les services aux propriétés de QdS*

Comme spécifié dans le modèle intentionnel du service $S_{\text{Convertir la devise}}$ présenté par la figure 5, les utilisateurs donnent trois fois plus d'importance au temps de réponse qu'à de la disponibilité (la priorité accordée au temps de réponse est 3, tandis que la priorité accordée à la disponibilité est 1). Pour tenir compte de cette priorité, nous définissons pour chaque valeur ordinale du temps de réponse trois sous-valeurs. Par exemple, nous considérons trois sous-valeurs : très faible 1, très faible 2 et très faible 3 pour la valeur ordinale très faible. Un service ayant un temps de réponse très faible devrait avoir une relation d'incidence avec les trois sous-valeurs de la valeur ordinale très faible. Aussi, nous considérons que si un service a une relation d'incidence avec une valeur ordinale du temps de réponse (avec une valeur ordinale de la disponibilité,

respectivement), alors il devrait avoir une relation d'incidence avec toutes les valeurs ordinales qui viennent juste après (qui viennent juste avant, respectivement). Par exemple, si un service a un temps de réponse très faible (une disponibilité très élevée, respectivement), alors il a également un temps de réponse faible, moyen, élevé et très élevé (une disponibilité élevée, moyenne, faible et très faible, respectivement). Les services pertinents et de haute QdS sont les services qui ont le plus de relations d'incidence avec les propriétés du contexte. Le tableau 2 donne une vue partielle du contexte $\mathcal{K}$. Il montre les 9 services pertinents obtenus après l'étape de découverte (les individus en lignes), et leur(s) relation(s) « est caractérisé par » avec la disponibilité et le temps de réponse (les propriétés en colonnes).

### 4.2. *Utilisation des treillis de concepts pour la sélection des services pertinents et de haute QdS*

L'AFC organise les concepts formels en une hiérarchie appelée *treillis de concepts*. L'organisation des concepts en treillis permet une navigation et une recherche faciles et une représentation optimale de l'information. La figure 8 représente le treillis de concepts associé à notre contexte $\mathcal{K}$. Ce treillis est construit en utilisant l'outil Galicia[10] *(Galois Lattice Interactive Constructor)* (Valtchev *et al.*, 2003). Galicia est une plate-forme open-source permettant la construction, la visualisation et le stockage des treillis de concepts. Le treillis présenté par la figure 8 est avec un étiquetage réduit : un concept de ce treillis est défini par l'ensemble des individus des concepts situés en-dessous et des propriétés des concepts situés au-dessus. En effet, l'étiquetage réduit consiste à afficher un individu (respectivement une propriété) une seule fois dans le treillis au niveau du concept où il apparaît pour la première fois. Ainsi, l'extension complète (respectivement l'intension) d'un concept est obtenue en cumulant sans répétition tous les individus (respectivement les propriétés) se trouvant en-dessous (respectivement au-dessus) de ce concept. La navigation dans le treillis de la figure 8 nous permet de conclure que les services pertinents qui offrent le meilleur compromis entre disponibilité et temps de réponse sont les services de l'extension du concept 1 se trouvant au bas du treillis. L'extension de ce concept contient 3 services : `CurrencyServer`, `CurrencyService` et `ForeignExchangeRates`. Ces services ont une disponibilité très élevée (D5) et un temps de réponse très faible (TR11, TR12 et TR13). Ces services sont interchangeables : en cas de panne ou d'indisponibilité de l'un de ces services, ce dernier peut être remplacé par l'un des deux services qui restent et ceci tout en gardant la même QdS.

10. http ://galicia.sourceforge.net/

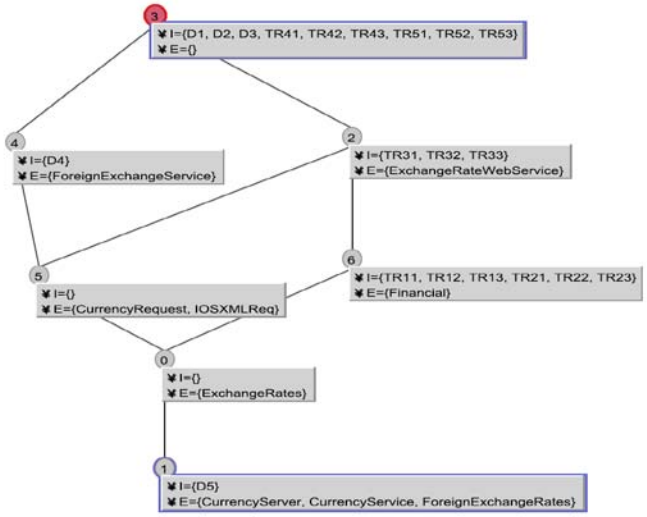


**Figure 8.** *Le treillis du contexte* $\mathcal{K}$

## 5. Génération de processus de coordination BPEL

Pour coordonner les services sélectionnés, nous proposons de générer automatiquement par transformations de modèles des processus d'orchestration BPEL à partir de modèles intentionnels de services MIS.

### 5.1. *Introduction aux transformations de modèles*

Les transformations de modèles sont au coeur de l'approche de l'ingénierie dirigée par les modèles. Hubert Kadima (2005) nous donne les définitions suivantes :

**Définition 1.** *Une transformation de modèle est une opération qui consiste à générer un ou de plusieurs modèles cibles à partir d'un ou de plusieurs modèles sources conformément à une définition de transformation.*

**Définition 2.** *Une définition de transformation est un ensemble de règles de transformation qui décrivent globalement comment un modèle décrit dans un langage source peut être transformé en un modèle décrit dans un langage cible.*

**Définition 3.** *Une règle de transformation est une description de la manière dont un ou plusieurs éléments du modèle source peuvent être transformés en un ou plusieurs éléments du modèle cible.*

### 5.2. *BPEL*

BPEL ou BPEL4WS *(Business Process Execution Language for Web Services)* est un langage exécutable standardisé par l'OASIS et basé sur XML. BPEL est issue de la fusion de XLANG[11] de Microsoft et de WSFL[12] d'IBM. Il combine les caractéristiques d'un langage de processus structuré par blocs (XLANG) avec ceux d'un langage de processus basé sur les processus métier (WSFL). Nous avons choisi d'utiliser BPEL parce qu'il s'agit du langage d'orchestration le plus utilisé actuellement par l'industrie. Une orchestration de services est un procédé de coordination de services qui peuvent être atomiques ou composés. De l'orchestration résulte un nouveau service composé qui contrôle la collaboration entre les services, tel un chef d'orchestre. BPEL repose sur un modèle constitué d'un workflow d'activités qui peuvent être de deux types : de base ou structurées. Les activités de base sont utilisées pour l'invocation d'une opération d'un service (`invoke`, `receive`, `reply`, `assign`, `wait`, `throw` et `terminate`). Elles manipulent un certain nombre de variables et elles sont liées à des `partnerLinks`. La conversation entre partenaires est assurée par des `correlationSets`. Les activités structurées sont basées sur l'approche des hiérarchies d'activités dans laquelle un processus est spécifié par le raffinement progressif d'une activité dans un arbre d'activités (`sequence`, `flow`, `while`, `switch`, `scope` et `pick`). Un modèle BPEL intègre aussi un mécanisme de gestion des exceptions `faultHandlers`, des erreurs `compensationHandlers` et des compensations `eventHandlers`.

### 5.3. *Règles de transformation vers BPEL*

La transformation de modèles MIS vers BPEL se déroule en deux étapes successives. Dans une première étape, un premier modèle BPEL dit *intentionnel* est élaboré. Ce modèle est non exécutable et constitué de services intentionnels extraits des cartes et spécifiés par les modèles MIS. Dans une deuxième étape, un second modèle BPEL dit *opérationnel* est fourni. Ce modèle est exécutable et constitué de services opérationnels obtenus suite aux étapes de découverte et de sélection de notre approche. La première étape de cette transformation est réalisée en considérant le méta-modèle de MIS présenté par la figure 4 et le méta-modèle BPEL 2.0 récupéré du CVS du projet Eclipse. La transformation est implémentée avec Kermeta[13] qui est un outil de transformations de modèles développé sous forme de plugin Eclipse. Un ensemble de règles ont été définies afin de mettre en œuvre cette transformation. Elles sont résumées dans le tableau 3.

11. http ://msdn.microsoft.com/en-us/library/aa577463%28v=bts.10%29.aspx
12. http ://www.ibm.com/developerworks/library/ws-ref4/
13. http ://www.kermeta.org/

| Élément MIS | Équivalent BPEL |
|---|---|
| *Service atomique* | `invoke` |
| *Situation initiale* | `receive` |
| *Situation finale* | `reply` |
| *Pré-condition, Post-condition* | `assign` |
| *Lien de composition Séquence* | `sequence` |
| *Lien de composition Parallèle* | `flow` |
| *Lien de composition itératif* | `while` |
| *Point de variation Alternatif* | `switch, if else` |
| *Point de variation De chemin* | `scope` |
| *Point de variation Multiple* | `pick` |

**Tableau 3.** *Règles de transformation de MIS vers BPEL*

Un *Service atomique* est la représentation intentionnelle du service opérationnel, il est donc transformé vers l'activité de base `invoke` qui permet l'invocation d'un service. Les paramètres d'entrée *Situation initiale* sont introduits par `receive` et les paramètres de sortie *Situation finale* sont fournis par `reply`. Les *Pré-condition* et *Post-condition* sont transformées vers `assign`. Pour les services agrégats, le lien de composition *Séquence*, *Parallèle* et *itératif* sont transformés respectivement vers `sequence`, `flow` et `while`. Les points de variation *Alternatif*, *De chemin*, et *Multiple* sont transformés respectivement vers (`switch` ou `if else`), `scope` et `pick`. Le résultat de cette première étape de transformation est une instance du méta-modèle BPEL 2.0 constitué de services intentionnels. Cette instance est un fichier .xmi incompréhensible par les moteurs d'orchestration. Pour pouvoir exécuter ce fichier, nous utilisons l'API *BPELWriter* disponible sous le plugin Eclipse BPEL Designer[14]. Cet API permet de générer des fichiers exécutables .bpel à partir de modèles BPEL. Pour illustrer cette transformation, nous présentons dans la figure 9 le modèle MIS de la carte d'affinement de la section **<***Réserver un hôtel, Arrêter, Par paiement***>** (voir figure 3) et le modèle BPEL qui lui correspond. Le modèle MIS décrit une séquence de services qui est transformée vers `sequence` en BPEL. Les deux services $S_{\text{Créer un compte}}$ et $S_{\text{Charger un compte}}$ forment un point de variation *Multiple* transformé vers `pick`. Ces deux services sont suivis par $S_{\text{Payer avec carte de crédit}}$ et $S_{\text{Convertir la devise}}$. Finalement, nous trouvons $S_{\text{Envoyer un sms}}$ et $S_{\text{Envoyer un e-mail}}$ qui forment un point de variation *Alternatif* transformé vers `if else`. Le modèle BPEL élaboré est *intentionnel*. Pour pouvoir l'exécuter, il faudrait faire

14. http ://www.eclipse.org/bpel/downloads.php

correspondre un service opérationnel à chaque service intentionnel. Par exemple, pour le service $S_{\text{Convertir la devise}}$, nous pouvons exécuter l'un des services `CurrencyServer`, `CurrencyService` et `ForeignExchangeRates` obtenus par application de la FCA lors de l'étape de sélection de notre approche.

```
...
<Sequence>
    <Multiple>
        <Service atomique nom="Créer un compte"/>
        <Service atomique nom="Charger un compte"/>
    </Multiple>
<Service atomique nom="Payer avec carte de crédit"/>
 <Service atomique nom="Convertir la devise"/>
<Alternatif>
<Service atomique nom="Envoyer un sms"/>
<Service atomique nom="Envoyer un mail"/>
</Alternatif>
</Sequence>
...
```

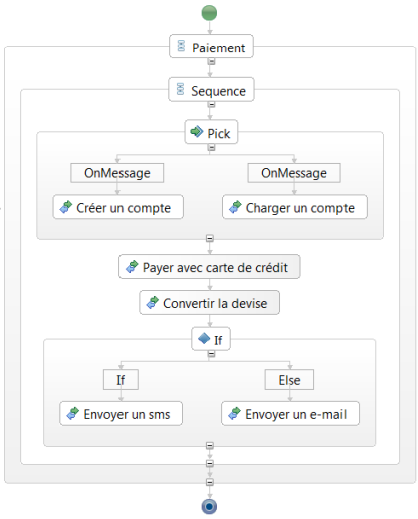


**Figure 9.** *Le modèle MIS de la carte d'affinement de la section <Réserver un hôtel, Arrêter, Par paiement> et le modèle BPEL qui lui correspond*

## 6. Expérimentations

Nous validons notre approche sur les 38 services intentionnels de l'application d'arrangement de conférences. Nous vérifions manuellement que les services retournés par notre approche correspondent à de réels services pertinents et de haute QdS. Nous projetons la validation de notre approche dans le domaine de l'extraction d'information et nous utilisons les mesures de précision et de rappel (Frakes *et al.*, 1992). La précision évalue le nombre de réels services pertinents et de haute QdS parmi les services sélectionnés par notre approche, tandis que le rappel évalue le nombre de services sélectionnés par notre approche parmi les réels services pertinents et de haute QdS, selon les équations suivantes :

$$\text{précision} = \frac{\left|\{\text{Réels services pertinents et de haute QdS}\} \cap \{\text{Services sélectionnés par notre approche}\}\right|}{\left|\{\text{Services sélectionnés par notre approche}\}\right|}$$

$$\text{Rappel} = \frac{\left|\{\text{Réels services pertinents et de haute QdS}\} \cap \{\text{Services sélectionnés par notre approche}\}\right|}{\left|\{\text{Réels services pertinents et de haute QdS}\}\right|}$$

Dans le tableau 4, nous donnons les résultats expérimentaux des 38 services intentionnels de notre application d'arrangement de conférences.

| Services intentionnels | *Service-Finder* | | | Notre approche | | |
|---|---|---|---|---|---|---|
| | Mots-clés | Services retournés | Réels services pertinents et de haute QdS | Découverte | Sélection | Réels services pertinents et de haute QdS |
| $ab1_i$ → S*Formuler une requête : ville départ/ville d'arrivée* | « réserver +vol +ville » | 11 | 2/11 | 6/11 | 2/6 | 2/2 |
| $ab2_i$ → S*Formuler une requête : compagnie aérienne* | « réserver +vol +compagnie +aérienne » | 6 | 1/6 | 2/6 | 1/2 | 1/1 |
| $ab3_i$ → S*Formuler une requête : offre promotionnelle de vols* | « réserver +vol +promotion » | 3 | 1/3 | 1/3 | 1/1 | 1/1 |
| $ab4_i$ → S*Formuler une requête : date du vol* | « réserver +vol +date » | 12 | 2/12 | 5/12 | 2/5 | 2/2 |
| $bb1_i$ → S*Sélectionner le vol le moins cher* | « trier +vol +prix » | 6 | 1/6 | 3/6 | 1/3 | 1/1 |
| $bb2_i$ → S*Sélectionner la compagnie aérienne la plus réputée* | « trier +compagnie +aérienne » | 6 | 1/6 | 2/6 | 1/2 | 1/1 |
| $bb3_i$ → S*Sélectionner la destination la plus proche de l'hôtel* | « trier +vol +localisation » | 6 | 1/6 | 1/6 | 1/1 | 1/1 |
| $bc1_i$ → S*Réserver un train* | « train » | 20 | 3/20 | 12/20 | 4/12 | 2/4 |
| $bc2_i$ → S*Réserver un taxi* | « taxi » | 9 | 2/9 | 4/9 | 2/4 | 2/2 |
| $bc3_i$ → S*Louer une voiture* | « louer +voiture » | 6 | 1/6 | 4/6 | 1/4 | 1/1 |
| $bc4_i$ → S*Consulter les horaires des bus* | « horaires +bus » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $bc5_i$→ S*Consulter les horaires de la navette* | « horaires +navette » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $ad1_i$ → S*Formuler une requête : nom de l'hôtel* | « réserver +hôtel +nom » | 19 | 2/19 | 7/19 | 3/7 | 1/3 |
| $ad2_i$→ S*Formuler une requête : adresse de l'hôtel* | « réserver +hôtel +adresse » | 18 | 2/18 | 6/18 | 2/6 | 2/2 |
| $ad3_i$→ S*Formuler une requête : prix de l'hôtel* | « réserver +hôtel +prix » | 18 | 2/18 | 7/18 | 2/7 | 2/2 |
| $ad4_i$→ S*Formuler une requête : type de l'hôtel* | « réserver +hôtel +type » | 20 | 2/18 | 6/18 | 2/6 | 2/6 |
| $dd1_i$→ S*Sélectionner l'hôtel le moins cher* | « trier +hôtel +prix » | 11 | 1/11 | 6/11 | 1/6 | 1/1 |

| | | | | | | |
|---|---|---|---|---|---|---|
| $dd2_i$ → S*Sélectionner l'hôtel le plus réputé* | « trier +hôtel +réputation » | 12 | 1/12 | 4/12 | 1/4 | 1/1 |
| $dd3_i$ → S*Sélectionner l'hôtel le plus proche* | « trier +hôtel +localisation » | 2 | 1/2 | 1/1 | 1/1 | 1/1 |
| $ae1_i$ → S*Télécharger le plan de la ville* | « télécharger +plan +ville » | 14 | 2/14 | 7/14 | 2/7 | 1/2 |
| $ae2_i$ → S*Acheter un ticket de musée* | « musée » | 3 | 1/3 | 2/3 | 1/2 | 1/1 |
| $ae3_i$ → S*Acheter un ticket de parc d'attractions* | « parc d'attractions » | 0 | - | - | - | - |
| $ae4_i$ → S*Acheter un ticket de visite guidée* | « visite guidée » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $ef1_i$ → S*Formuler une requête : nom du restaurant* | « restaurant +nom » | 6 | 1/6 | 4/6 | 2/4 | 1/2 |
| $ef2_i$ → S*Formuler une requête : spécialité du restaurant* | « restaurant +spécialité » | 0 | - | - | - | - |
| $ef3_i$ → S*Formuler une requête : adresse du restaurant* | « restaurant +adresse » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $ff1_i$ → S*Sélectionner le restaurant le moins cher* | « trier +restaurant +prix » | 2 | 1/2 | 2/2 | 1/2 | 1/1 |
| $ff2_i$ → S*Sélectionner le restaurant le plus réputé* | « trier +restaurant +réputation » | 2 | 1/2 | 2/2 | 1/2 | 1/1 |
| $ff3_i$ → S*Sélectionner le restaurant le plus proche* | « trier+ Restaurant+ localisation » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $ag1_i$ → S*Annuler la réservation d'un vol* | « annuler +réservation +vol » | 10 | 1/10 | 7/10 | 1/7 | 1/1 |
| $ag2_i$ → S*Annuler la réservation d'un hôtel* | « annuler +réservation +hôtel » | 15 | 3/15 | 5/15 | 4/5 | 3/4 |
| $ag3_i$ → S*Annuler la réservation d'un restaurant* | « annuler +réservation +restaurant » | 1 | 1/1 | 1/1 | 1/1 | 1/1 |
| $ab1_{i+1}$ → S*Créer un compte* | « créer +compte » | 330 | 26/330 | 203/330 | 25/203 | 22/25 |
| $ab2_{i+1}$ → S*Charger un compte* | « charger +compte » | 29 | 4/29 | 20/29 | 5/20 | 4/5 |
| $bc1_{i+1}$ → S*Payer avec carte de crédit* | « payer +carte +crédit » | 77 | 4/77 | 16/77 | 4/16 | 4/4 |
| $cc1_{i+1}$ → S*Convertir la devise* | « convertir +devise » | 76 | 3/76 | 9/76 | 3/9 | 3/3 |
| $cd1_{i+1}$ → S*Envoyer un sms* | « envoyer +sms » | 162 | 17/162 | 105/162 | 18/105 | 15/18 |
| $cd2_{i+1}$ → S*Envoyer un e-mail* | « envoyer +e-mail » | 58 | 7/58 | 30/58 | 9/30 | 6/9 |

**Tableau 4.** *Résultats des expérimentations réalisées sur l'application d'arrangement de conférences*

La première colonne de ce tableau correspond aux services intentionnels. Dans la seconde colonne, nous avons, premièrement, la liste des mots-clés utilisés pour interroger *Service-Finder*, puis, le nombre de services opérationnels retournés par *Service-Finder* et enfin le nombre de services identifiés manuellement comme réels services pertinents et de haute QdS. Dans la dernière colonne, nous avons, premièrement, le nombre de services obtenus après l'étape de découverte de notre approche, deuxièmement, le nombre de services obtenus après l'étape de sélection et enfin le nombre de réels services pertinents et de haute QdS parmi ceux qui sont sélectionnés.

Par exemple, deux mots-clés sont utilisés pour rechercher des services opérationnels permettant de satisfaire le service intentionnel $S_{\text{Envoyer un sms}}$ qui sont : "Envoyer + SMS". La requête d'interrogation de *Service-Finder* retourne un ensemble de 162 services opérationnels. Parmi cet ensemble, seulement 17 services sont vérifiés manuellement comme de réels services pertinents et de haute QdS. L'étape de découverte réduit l'ensemble retourné par *Service-Finder* en un autre ensemble renfermant seulement 105 services pertinents. Parmi ces 105 services, seulement 18 services sont maintenus après l'étape de sélection. Nous avons vérifié manuellement que seulement 15 services parmi les 18 sont de réels services pertinents et de haute QdS.

| **Services intentionnels** | **Précision** | **Rappel** |
|---|---|---|
| $ab1_i \rightarrow S_{\textit{Formuler une requête : ville départ/ville d'arrivée}}$ | 2/2 (100 %) | 2/2 (100 %) |
| $ab2_i \rightarrow S_{\textit{Formuler une requête : compagnie aérienne}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $ab3_i \rightarrow S_{\textit{Formuler une requête : offre promotionnelle de vols}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $ab4_i \rightarrow S_{\textit{Formuler une requête : date du vol}}$ | 2/2 (100 %) | 2/2 (100 %) |
| $bb1_i \rightarrow S_{\textit{Sélectionner le vol le moins cher}}$ | 1/1 (100 %) | (1/1) (100 %) |
| $bb2_i \rightarrow S_{\textit{Sélectionner la compagnie aérienne la plus réputée}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $bb3_i \rightarrow S_{\textit{Sélectionner la destination la plus proche de l'hôtel}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $bc1_i \rightarrow S_{\textit{Réserver un train}}$ | 2/4 (50 %) | 2/3 (66,66 %) |
| $bc2_i \rightarrow S_{\textit{Réserver un taxi}}$ | 2/2 (100 %) | 2/2 (100 %) |
| $bc3_i \rightarrow S_{\textit{Louer une voiture}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $bc4_i \rightarrow S_{\textit{Consulter les horaires des bus}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $bc5_i \rightarrow S_{\textit{Consulter les horaires de la navette}}$ | 1/1 (100 %) | 1/1 (100 %) |
| $ad1_i \rightarrow S_{\textit{Formuler une requête : nom de l'hôtel}}$ | 1/ 3 (33,33 %) | 1/2 (50 %) |

| | | |
|---|---|---|
| $ad2_i \rightarrow S$ *Formuler une requête : adresse de l'hôtel* | 2/2 (100 %) | 2/2 (100 %) |
| $ad3_i \rightarrow S$ *Formuler une requête : prix de l'hôtel* | 2/2 (100 %) | 2/2 (100 %) |
| $ad4_i \rightarrow S$ *Formuler une requête : type de l'hôtel* | 2/2 (100 %) | 2/2 (100 %) |
| $dd1_i \rightarrow S$ *Sélectionner l'hôtel le moins cher* | 1/1 (100 %) | 1/1 (100 %) |
| $dd2_i \rightarrow S$ *Sélectionner l'hôtel le plus réputé* | 1/1 (100 %) | 1/1 (100 %) |
| $dd3_i \rightarrow S$ *Sélectionner l'hôtel le plus proche* | 1/1 (100 %) | 1/1 (100 %) |
| $ae1_i \rightarrow S$ *Télécharger le plan de la ville* | 1/2 (50 %) | 1/2 (50 %) |
| $ae2_i \rightarrow S$ *Acheter un ticket de musée* | 1/1 (100 %) | 1/1 (100 %) |
| $ae4_i \rightarrow S$ *Acheter un ticket de visite guidée* | 1/1 (100 %) | 1/1 (100 %) |
| $ef1_i \rightarrow S$ *Formuler une requête : nom du restaurant* | 1/2 (50 %) | 1/1 (100 %) |
| $ef3_i \rightarrow S$ *Formuler une requête : adresse du restaurant* | 1/1 (100 %) | 1/1 (100 %) |
| $ff1_i \rightarrow S$ *Sélectionner le restaurant le moins cher* | 1/1 (100 %) | 1/1 (100 %) |
| $ff2_i \rightarrow S$ *Sélectionner le restaurant le plus réputé* | 1/1 (100 %) | 1/1 (100 %) |
| $ff3_i \rightarrow S$ *Sélectionner le restaurant le plus proche* | 1/1 (100 %) | 1/1 (100 %) |
| $ag1_i \rightarrow S$ *Annuler la réservation d'un vol* | 1/1 (100 %) | 1/1 (100 %) |
| $ag2_i \rightarrow S$ *Annuler la réservation d'un hôtel* | 3/4 (75 %) | 3/3 (100 %) |
| $ag3_i \rightarrow S$ *Annuler la réservation d'un restaurant* | 1/1 (100 %) | 1/1 (100 %) |
| $ab1_{i+1} \rightarrow S$ *Créer un compte* | 22/25 (88 %) | 22/26 (84,62 %) |
| $ab2_{i+1} \rightarrow S$ *Charger un compte* | 4/5 (80 %) | 4/4 (100 %) |
| $bc1_{i+1} \rightarrow S$ *Payer avec carte de crédit* | 4/4 (100 %) | 4/4 (100 %) |
| $cc1_{i+1} \rightarrow S$ *Convertir la devise* | 3/3 (100 %) | 3/3 (100 %) |
| $cd1_{i+1} \rightarrow S$ *Envoyer un sms* | 15/18 (83.33 %) | 15/17 (88,24 %) |
| $cd2_{i+1} \rightarrow S$ *Envoyer un e-mail* | 6/9 (66,67 %) | 6/7 (85,71 %) |
| **Moyenne** | **91 %** | **95,14 %** |

**Tableau 5.** *Précision et rappel de notre approche appliquée sur le cas d'étude de l'application d'arrangement de conférences*

Dans le tableau 5, nous donnons la précision et le rappel correspondant aux 36 services intentionnels de l'application d'arrangement de conférences (nous ne considérons pas $S_{\text{Acheter un ticket de parc d'attractions}}$ et $S_{\text{Formuler une requête: spécialité du restaurant}}$ puisque nous n'avons pas trouvé de services qui leur correspondent). La précision de sélection de notre approche pour le service intentionnel $S_{\text{Envoyer un sms}}$ est de 83,33% et le rappel est de 88,24 %. Le tableau 4 montre que la précision et le

rappel de notre approche sont très élevés. La moyenne de la précision est de 91 % et la moyenne du rappel est de 95,14 % pour le cas d'étude de l'application d'arrangement de conférences. Le rappel est légèrement supérieur à la précision, cela signifie que notre approche permet de retourner, dans la plupart du temps, tous les services pertinents et de haute QdS. La précision est faible lorsqu'il s'agit de sélectionner à partir d'un ensemble restreint de services. C'est le cas par exemple de $S_{\text{Formuler une requête: nom de l'hôtel}}$ où il fallait sélectionner à partir d'un ensemble formé de 7 services seulement (voir tableau 4). Notre approche a retourné 3 services pertinents et de haute QdS alors qu'il en ait réellement qu'un seul service qui répond aux exigences fonctionnelles des utilisateurs et qui offre une haute QdS. La pertinence pour cet exemple est de 33,33 % seulement. Ceci est dû essentiellement au mécanisme d'échelonnage de QdS qui donne de meilleurs résultats que lorsqu'il s'agit d'échelonner un très grand nombre de valeurs.

## 7. Etat de l'art

Récemment, plusieurs approches centrées exigences se rapportant aux applications à base de services web ont été proposées. Le travail de Zachos *et al.* (2007) a comme objectif d'aligner les exigences des utilisateurs aux services opérationnels disponibles. L'idée-clé de ce travail est de créer un cahier des charges spécifiant les exigences des utilisateurs, transformer ces exigences en requêtes pour l'interrogation de l'annuaire de services et modifier le cahier des charges élaboré initialement en fonction des services retournés. Ce travail souffre de plusieurs problèmes : (i) la description textuelle et incomplète des exigences ; (ii) la non-considération des exigences non fonctionnelles et enfin (iii) la difficulté d'aligner les exigences aux services disponibles dans l'annuaire. Le travail présenté par Pistore *et al.* (2004) propose de dériver des compositions de services par affinement de modèles d'exigences TROPOS. L'approche proposée permet d'enrichir les modèles formels TROPOS avec du code BPEL et d'exploiter des techniques de vérification formelles sur les modèles TROPOS pour assurer la validation des compositions de services. Plus récemment, le formalisme la Carte a été utilisé dans (Rolland *et al.*, 2007) pour décrire les applications à base de services. La contribution principale de ce travail est la proposition du modèle intentionnel de services MIS qui permet de spécifier les services et leur composition d'une manière intentionnelle. Le modèle proposé dans le cadre de ce travail omet les aspects liés à la QdS. Les travaux présentés dans (Pistore *et al.*, 2004) et (Rolland *et al.*, 2007) permettent la modélisation des applications à base de services en termes d'exigences des utilisateurs mais, ils ne traitent les problèmes de découverte, sélection et coordination de services. (Rolland *et al.*, 2010) est la continuité du travail (Rolland *et al.*, 2007). Dans le cadre de ce travail, une extension du modèle MIS est proposée afin de décrire des aspects de QdS liés aux services intentionnels. Aussi, propose-t-il une méthodologie basée sur un modèle opérationnel de services et un ensemble de règles de transformation pour l'opérationnalisation des services intentionnels. Contrairement à notre travail, Rolland *et al.* (2010) proposent d'implémenter des

services et non pas de réutiliser des services déjà existants pour opérationnaliser les exigences des utilisateurs. Mirbel *et al.* (2010) utilisent aussi le formalisme la Carte pour la modélisation des exigences des utilisateurs. Ce travail propose une approche sémantique guidée par les intentions pour la modélisation et la recherche de services. Mirbel *et al.* exploitent les capacités de raisonnement des modèles et langages du web sémantique pour dériver les caractéristiques des services web recherchés des besoins des utilisateurs. Un autre travail permettant la découverte de services à partir des exigences des utilisateurs est présenté par Pérez *et al.* (2010). Ces derniers utilisent le formalisme *i** pour la modélisation des exigences des utilisateurs et proposent un mécanisme semi-automatique permettant la découverte des services web. L'approche que nous présentons dans cet article est complémentaire à (Mirbel *et al.,* 2010) et (Pérez *et al.,* 2010) puisqu'elle permet non seulement la modélisation des exigences des utilisateurs et la découverte des services, mais aussi la sélection et la coordination des services pertinents et de haute QdS.

## 8. Conclusion

Dans cet article, nous proposons une approche centrée exigences pour la composition de services web qui permet : (i) la modélisation des applications à base de services en termes d'exigences fonctionnelles et non fonctionnelles à l'aide du formalisme la Carte ; (ii) la découverte des services web pertinents qui répondent le mieux aux exigences fonctionnelles par l'interrogation du moteur de recherche de services *Service-Finder* ; (ii) la sélection des services pertinents et de haute QdS par l'application de l'AFC et (iv) la génération automatique de processus de coordination BPEL par transformation de modèles. Nous avons validé notre approche par des expérimentations réalisées sur un cas d'étude d'une application d'arrangement de conférences. Les résultats expérimentaux montrent que notre approche permet de découvrir, sélectionner et coordonner des services pertinents et de haute QdS avec une haute précision et une grande efficacité.

Les travaux futurs seront consacrés à identifier les services composites pertinents et de haute QdS à l'aide de d'une variante de l'AFC qui est l'analyse relationnelle de concepts (ARC) et à définir et exploiter des règles d'association pour automatiser la navigation dans les treillis générés par AFC ou par ARC.

## 9. Bibliographie

Berardi D., Calvanese D., Giacomo G. D., Lenzerini M., Mecella M., "Automatic composition of e-services that export their behavior", *ICSOC '03*, 2003, p. 43-58.

Bresciani P., Perini A., Giorgini P., Giunchiglia F., Mylopoulos J., "Tropos: an agent-oriented software development methodology", *Autonomous Agents and Multi-Agent Systems*, vol. 8, n° 3, 2004, p. 203-236.

Chambers J. M., Cleveland W. S., Kleiner B., Tukey P. A., *Graphical methods for data analysis*, Wadsworth & Brooks/Cole, 1983.

Driss M., Moha N., Jamoussi Y., Jézéquel J-M., Hajjami Ben Ghézala H., "A requirement-centric approach to Web service modeling, discovery, and selection", *ICSOC'10*, 2010, p. 258-272.

Foster H., Uchitel S., Magee J., Kramer J., "Compatibility verification for Web service choreography", *ICWS'04*, 2004, p. 738-741.

Frakes W. B., Baeza-Yates R., *Information retrieval: data structures and algorithms*, Prentice-Hall, 1992.

Ganter B., Wille R., *Formal concept analysis: mathematical foundations*, Springer-Verlag New York, Inc., 1999.

Huhns M. N., Singh M. P., "Service-oriented computing: key concepts and principles", *IEEE Internet Computing*, vol. 9, n° 1, 2005, p. 75-81.

Kadima H., *Conception orientée objet guidée par les modèles*, Dunod, 2005.

Kavantzas N., Burdett D., Ritzinger G., Lafon Y., "Web services choreography description Language (WS-CDL) Version 1.0", *W3C Candidate Recommendation*, 2005.

Lamsweerde A.V., Letier E., "Handling obstacles in goal oriented requirements engineering", *IEEE Transactions on Software Engineering*, vol. 26, n° 10, 2000, p. 978-1005.

MacNaughton R., Yamada H., "Regular expressions and state graphs for automata", *IEEE Transactions on Electronic Computers*, vol. 9, n° 1, 1960, p. 39-47.

Malhotra N., *Etudes marketing avec SPSS*, Pearson France, 2007.

Mecella M., Presicce F.P., Pernici B., "Modeling e-service orchestration through petri nets", *TES '02*, 2002, p. 38-47.

Menascé D. A., "QoS issues in Web services", *IEEE Internet Computing*, vol. 6, n° 6, 2002, p. 72-75.

Miller G., "WordNet: a lexical database for English", *Communications of the ACM*, vol. 38, n° 11, 1995, p. 39-41.

Mirbel I., Crescenzo P., « Des besoins des utilisateurs à la recherche de services web : une approche sémantique guidée par les intentions », *Ingénierie des systèmes d'information*, vol. 15, n° 4, 2010, p. 89-112.

Pérez M., Sanz I., Berlanga R., Aramburu M. J., "Semi-automatic discovery of Web services driven by user requirements", *DEXA '10*, 2010, p. 62-75.

Pirró G., Seco N., "Design, Implementation and Evaluation of a New Semantic Similarity metric combining features and intrinsic information content", *ODBASE '08*, 2008, p. 1271-1288.

Pistore M., Roveri M., Busetta P., "Requirements-driven verification of Web service", *WSFM'04*, 2004, p. 95-108.

Rolland C., Kaabi R. S., Kraeim N., "On ISOA: intentional services oriented architecture", *CAISE '07*, 2007, p. 158-172.

Rolland C., Kirsch-Pinheiro M., Souveyet C., “An intentional approach to service engineering”, *IEEE Transactions on Services Computing*, vol. 3, n° 4, 2010, p. 292-305.

Rolland C., Prakash N., “Bridging the gap between organizational needs and ERP functionality”, *Requirements Engineering*, vol. 5, n° 3, 2000, p. 180-193.

Salton G., Buckley C., “Term-weighting approaches in automatic text retrieval”, *Information Processing and Management*, vol. 24, n° 5, 1988, p. 513-523.

Valtchev P., Grosser D., Roume C., Hacene. M. R., “GALICIA: an open platform for lattices”, *ICCS ’03*, 2003, p. 241-254.

Willee R., *Restructuring lattice theory: An approach based on hierarchies of concepts*, ordered sets, Ivan Rival Ed., NATO Advanced Study Institute, 1981.

Zachos K., Maiden N., Zhu X., Jones S., “Discovering Web services to specify more complete system requirements”, *CAiSE ’07*, 2007, p. 142-157.