Une nouvelle approche de Découverte et de Composition de Services Web à base de médiation sémantique et de raisonnement déductif : application au domaine informatique

Nadia Yacoubi Ayadi

CONSERVATOIRE NATIONAL DES ARTS ET MÉTIERS

THESE EN COTUTELLE

Préparée au sein des laboratoires RIADI-GDL (ENSI) et CEDRIC (Equipe Sempia)

présentée par :

Nadia YACOUBI AYADI

Date de la soutenance : 1^{er} Juillet 2010

pour obtenir le diplôme de Doctorat en Informatique du CNAM (Paris) et de l'ENSI (Université de la Manouba)

Discipline/ Spécialité : Informatique

Une Nouvelle Approche de Découverte et de Composition de Services Web à base de Médiation Sémantique et de Raisonnement déductif

Application au domaine bioinformatique

Thèse dirigée par :

Prof. MohamedBen Ahmed	Professeur Emérite à l'ENSI
Prof. Yann Pollet	Professeur au CNAM, Paris

RAPPORTEURS :

Prof. Michel Léonard	Université de Genève
Prof. Hervé Pingaud	Ecole des Mines d'Albi

JURY :

Rapporteurs	Michel Léonard	Professeur à l'Université de genève
	Hervé Pingaud	Professeur à l'Ecole des Mines d'Albi
Examinateur	Jean François Zagury	Professeur au CNAM
Directeurs de Thèse	Mohamed Ben Ahmed	Professeur Emérite à l'ENSI
	Yann Pollet	Professeur au CNAM, Paris

Dédicace

A mes très chers parents
A mon très cher mari
A mes chers frères
A mon enfant Khadija Ayadi
A tous ceux que j'aime

Remerciements

C'est avec une grande émotion et beaucoup de sincérité que je voudrais exprimer ma gratitude à toutes les personnes ayant participé, soutenu et apprécié mon travail.

Tout d'abord, je tiens à remercier et exprimer toute ma reconnaissance auprès de mon directeur de thèse Pr. émérite **Mohamed Ben Ahmed** qui m'a initiée à la recherche et m'a toujours motivé, soutenu et encouragé. Méticuleux et perfectionniste, il m'a prodigué des conseils inestimables, dans tous les domaines, tout au long de ma thèse. Ses idées, son expérience et ses précieux conseils m'ont énormément aidé dans ce travail. Je le remercie pour sa disponibilité, son soutien et ses conseils nombreux et importants. Il a beaucoup contribué à la mise en valeur de mon travail, n'a cessé de m'encourager à avancer et m'a aidée à progresser à travers les difficultés et les doutes inhérents à tout travail de recherche. Je suis très fière de la formation de chercheuse acquise sous son encadrement.

Je tiens à exprimer ma profonde gratitude à mon co-directeur de thèse, Pr. **Yann Pollet** qui m'a accueilli pendant de longs séjours au Conservatoire National des Arts et Métiers, au cours desquels il s'est montré très disponible et accueillant pour discuter de mes travaux de thèse pendant de longues heures.

Je remercie mon rapporteur Pr. **Michel Léonard** pour l'honneur qu'il m'a accordé en acceptant d'évaluer ma thèse. Je suis très heureuse que mon travail ait trouvé grâce à ses yeux. Je remercie également mon rapporteur Pr. **Hervé Pingaud** pour l'honneur qu'il m'a accordé en acceptant d'évaluer mon travail et pour les commentaires très constructifs qu'il a formulés qui m'ont permis d'avoir plus de recul sur certains aspects que j'ai abordés.

Je souhaite remercier chaleureusement mon examinateur **Jean-François Zagury** pour l'enthousiasme qu'il a manifesté à l'égard de ma thèse et pour avoir accepté de faire partie de mon jury de thèse.

Je souhaite également remercier chaleureusement Pr. **Zoé Lacroix** qui m'a accueilli dans son Laboratoire à l'université d'Arizona aux Etats-Unis au sein duquel j'ai effectué un stage de six mois en bioinformatique. Mes remerciements s'adressent aussi au Pr. **Maria-Esther Vidal** à l'université Simon Bolivar au Venezuela avec qui j'ai eu l'honneur de collaborer pendant de longues années (depuis 2007) et cela malgré la distance (seulement géographique) qui nous sépare. Je remercie également ses doctorants Edna Ruckhaus, Marlene Goncalves et Alfredo Regalado. *Muchas gracias* !!!

Enfin, mes sentiments les plus chaleureux sont pour ma famille et mes amis. Je remercie mes parents qui m'ont toujours soutenu dans mes choix et qui m'ont toujours encouragé à aller de l'avant. Avec une infinie patience et tendresse, mon très cher mari m'a toujours

aidé à surpasser les moments de stress et de doute qui m'ont accablé, je lui dis cette thèse est aussi la tienne ! ! ! Une pensée chaleureuse à mes deux frères pour leur complicité, humour et amour. Enfin, à ma petite fille qui a partagé pendant sept mois cette drôle d'aventure qu'est la thèse.

Parmi mes amis, j'ai une pensée particulière pour Soumaya D., Lilia, Imen et Soumaya C. avec qui j'ai partagé les quatre premières années de l'université, mais aussi Olfa C. et Olfa D. avec qui j'ai partagé mes longues années de thèse. Sans leur amitié, la thèse n'aurait jamais été l'expérience qu'elle a été. J'ai également une pensée spéciale pour mes collègues et amis à l'école Supérieure de Commerce de Tunis, Yousr Karoui ElMekki et Khaled Boughzou qui comptent tellement pour moi.

Résumé

L'avènement du Web sémantique a permis l'apparition d'une nouvelle génération de services Web, dénommés Services Web Sémantiques (SWS) qui intègrent dans leurs descriptions une dimension sémantique décrivant les aspects fonctionnels et/ou non fonctionnels d'un service Web. L'objectif principal de cette nouvelle génération de services est d'améliorer l'interopérabilité des systèmes SOA (Service-Oriented architectures) hétérogènes et dynamiques. La problématique de cette thèse s'inscrit dans le cadre de la médiation de SWS visant à la fois la réconciliation des représentations de SWS et la réconciliation des conflits d'hétérogénéité pouvant émerger lors d'une découverte ou d'une composition de services. Ainsi, nous proposons dans cette thèse une méthodologie de sémantisation de services Web qui se base sur un modèle de descriptions canoniques permettant d'expliciter la sémantique d'un service Web.

Dans ce travail, la médiation de services repose sur deux grandes classes de processus : la découverte et la composition de SWS permettrant de satisfaire à la fois un ensemble de contraintes fonctionnelles et non fonctionnelles. Pour ce faire, nous proposons un moteur inférentiel *Datalog*-like permettant d'effectuer un matching sémantique approximatif entre les requêtes utilisateurs et les descriptions des services Web. Le moteur inférentiel s'appuie sur une axiomatique sémantique ainsi qu'une ou plusieurs ontologies du domaine afin d'inférer de nouvelles propriétés implicites décrivant les services Web.

Les expérimentations menées ont ciblé le domaine de la bioinformatique et ont permis de montrer l'impact des techniques de raisonnement sur la taille de l'espace de recherche des services découverts et la cardinalité des plans des services composés. Enfin, nous explorons les techniques de classement (ranking) permettant de classer un skyline de services (i.e., un ensemble de services fonctionnellement similaires mais incomparables d'un point de vue attributs QdS) ce qui nous permettra de déceler les meilleurs services atomiques et/ou plans de composition.

Mots-clés : services Web, services Web sémantiques, moteur inférentiel, moteur Datalog-like, découverte et composition de services, propriétés fonctionnelles et non fonctionnelles, bioinformatique, skyline, top-k.

Abstract

Semantic Web services (SWS) technology have been proposed and investigated by the Semantic Web community to support effective and efficient service discovery, composition and invocation by machines. In order to deal with semantic heterogeneity of emerging frameworks, we propose a semantic model to describe Web services semantics in an uniform way. The existence of a domain ontology to capture domain knowledge in an explicit and formal way is crucial to describe Web service semantics.

Therefore, two main classes of processes are addressed in this thesis : service discovery and composition. We consider both functional and non functional users constraints in discovery queries. Thus, we propose a Datalog inference engine that enables approximate matching of discovery queries. Different reasoning tasks are performed in order to select services that satisfy the requested functionality with different degrees of matching. Indeed, reasoning tasks allow to infer new implicit services properties based on the domain knowledge which augment the search space of the query. Reasoning tasks rely on explicit taxonomic and non-taxonomic relationships defined between concepts and properties to infer new knowledge and match a discovery query with different degrees of approximate matching.

The empirical study was conducted in the bioinformatics domain, we report the size of the search space of candidate services and composable services when reasoning tasks are performed in comparison without any reasoning tasks. Finally, we propose a ranking technique that enables users to identify the best services of a skyline of services in terms of a set of non-functional properties.

Keywords : Web services, Semantic Web services, Datalog inference engine, discovery and composition, functional and non functional properties, bioinformatics domain, skyline, top-k.

Table des matières

Liste des tableaux

Table des figures

Intoduction Générale

Le contexte général de cette thèse est celui de l'*interopérabilité des systèmes d'information*. La problématique de l'interopérabilité est cruciale surtout avec l'émergence d'un besoin accru en termes de partage, d'interopération et de réutilisation de ressources au sein de plusieurs communautés d'utilisateurs qui sont elles-mêmes hétérogènes. Nous nous sommes particulièrement intéressés au problème de l'*interopérabilité sémantique* [APB06, ABP06a, ABP06b]. En l'occurrence, l'*Architecture Orientée Services* (AOS) est une plate-forme d'interopérabiltié qui se base sur la notion de *services Web* pour assurer une communication inter et intra-communautaire faiblement couplée. Les technologies développées autour des services Web permettent de masquer les problèmes d'hétérogénéité liés à la disparité des plate-formes, des protocoles de communication ou de mode de représentation des données et de leur sémantique. La **médiation** représente le pré-requis essentiel à l'interopérabilité des systèmes ayant pour objectif de garantir que toutes les parties interprètent correctement les informations échangées. Les ontologies conçues comme des *spécifications formelles et explicites d'une conceptualisation partagée* représentent l'outil incontournable assurant une médiation sémantique et par conséquent une interopérabilité sémantique. La médiation sémantique de services Web est donc au centre de nos travaux de thèse [AL07].

La médiation sémantique de Services Web (SW) a pour objectif d'exploiter la sémantique explicite des services afin de faciliter leur découverte, leur sélection, leur invocation et leur composition en *workflows* par des utilisateurs finals. La sémantisation des descriptions de services est une phase nécessaire ayant pour objectif d'éliciter la sémantique des services Web. Une description sémantisée de services Web comprend un ensemble de *primitives* descriptives (déclaratives) permettant de décrire les propriétés pertinentes caractérisant le service et d'un ensemble d'annotations sémantiques décrivant le *lien* entre les primitives descriptives et un ensemble de concepts définis formellement au niveau d'une ontologie du domaine. A ce niveau, nous distinguons deux types d'hétérogénéité sémantique qui peuvent exister au niveau des descriptions sémantisées de services Web, à savoir : l'hétérogénéité des primitives descriptives liée au framework adopté pour déclarer la sémantique du service pouvant être par exemple l'ontologie OWL-S (Ontology Web Language for Web Service) [MBM+07] ou le framework WSMO (Web service Modeling Service) [RKL+05a] et l'hétérogénéité des annotations sémantiques dans le cas où elles sont définies en se basant sur différentes ontologies du domaine.

Pour répondre à cette problématique, nous proposons dans cette thèse un méta-framework stratifié de médiation sémantique de services Web que nous nommons BioMed (*BioInformatic Mediation Meta-Framework*) permet-

tant une découverte et une composition dynamiques de services Web dans le domaine bioinformatique. Pour ce faire, le méta-framework intègre une triple dimension sémantique à la fois : descriptive, ontologique et inférentielle déductive.

La première dimension permet de déclarer explicitement les propriétés sémantiques pertinentes d'un service Web. C'est dans ce sens que nous proposons un modèle d'alignement sémantique pour pallier l'hétérogénéité des primitives descriptives des trois principaux frameworks de descriptions sémantisées de services Web à savoir : OWL-S [MBM+07], WSMO [RKL+05a] et SAWSDL[KVBF07]. Nous proposons un modèle d'unification vers lequel les modèles de descriptions cités sont mappés. Nous retenons un ensemble de primitives descriptives permettant d'expliciter un ensemble de propriétés fonctionnelles (PF) et non fonctionnelles (PNF) d'un service. Les PF considérées dans le modèle d'unification constituent la signature fonctionnelle du service courant, à savoir l'ensemble des entrées, des sorties, des pré-conditions (les conditions devant être vérifiées par les entrées) et des post-conditions (les conditions devant être vérifiées par les sorties du service). Au niveau du méta-framework BioMed, les PNF qui permettent de décrire la qualité d'un service (QdS) se déclinent en deux catégories : celles qui sont indépendantes du domaine du service comme le temps de réponse, le débit, la disponibilité ou la fiabilité et celles qui sont spécifiques au domaine du service en l'occurrence la complétude et le degré de confiance d'un service Web dans le domaine bioinformatique. Nous argumentons le choix de ces descriptions au niveau de cette thèse. Une des particularités du modèle proposé est qu'il permet d'intégrer dans une même description les aspects fonctionnels et non fonctionnels d'un service, ce qui n'est pas le cas de la plupart des frameworks de description de SW. L'intérêt de ce choix est de couvrir par cette approche de découverte et de composition les deux types de propriétés.

La dimension ontologique dans BioMed est représentée par une **méta-carte ontologique** du domaine bioinformatique modélisant un réseau sémantique des concepts clés du domaine visé et réconciliant les concepts et leurs instances des différentes ontologies du domaine. La méta-carte se présente comme une *ontologie régionale* [Bac00] modélisant la sémantique partagée du domaine visé et valable seulement localement. La définition de la méta-carte est une tâche de modélisation menée à partir de l'expression linguistique des connaissances. La modélisation est effectuée en trois étapes, correspondant à trois engagements : un engagement sémantique, normalisant le sens linguistique des concepts, un engagement ontologique fixant leur sens formel et offrant une sémantique formelle pour les primitives non logiques et enfin un engagement computationnel déterminant leur exploitation effective. La méta-carte conçue représente le soubassement de l'annotation sémantique de services Web et du raisonnement déductif.

Enfin, la dimension inférentielle déductive du framework est décrite par le biais d'une sémantique axiomatique formalisant les différents types de raisonnement déductif permettant d'exploiter les *connaissances descriptives* émanant du modèle d'unification, les *connaissances ontologiques* émanant de la méta-carte ontologique et des annotations sémantiques générées sur la base de la méta-carte. En se basant sur les trois types de connaissances, le raisonnement déductif permet de relaxer les propriétés des services afin d'élargir

leur couverture sémantique.

L'approche de découverte proposée au niveau du framework BioMed comporte deux phases :

- une **phase de sélection** de services atomiques satisfaisant les contraintes fonctionnelles d'une requête de découverte étendue (i.e. couvrant les propriétés fonctionnelles que les propriétés non fonctionnelles, notamment les attributs de la QdS). Cette phase repose sur le moteur inférentiel pour relaxer les propriétés des services et matcher au mieux les contraintes fonctionnelles de la requête. Différents degrés d'appariement approximatifs sont identifiés selon l'opérateur de relaxation utilisé.
- une **phase de classement** permettant de classer l'ensemble des services sélectionnés sur la base de leurs propriétés non fonctionnelles. La phase de classement permet de trier l'espace des solutions possibles en combinant deux techniques : celle de l'opérateur *skyline* et celle du *top-k*. L'algorithme WSTKF proposé dans [GVRA09, GVRA10] permet d'identifier les top-k services de l'opérateur skyline de services satisfaisant les contraintes non fonctionnelles d'une requête de découverte.

Les expérimentations menées dans le cadre de l'évaluation de l'approche de découverte montrent l'impact du modèle référentiel sur l'espace des solutions possibles d'une requête de découverte, cet espace de solutions est nettement plus large en terme de nombre de services atomiques que celui du mode sans raisonnement. Les expérimentations montrent aussi l'intérêt des techniques de classement pour déceler les meilleures solutions de l'espace sur la base des PNF des services.

Dans le cas où le moteur de découverte de BioMed ne retourne pas de services atomiques répondant à la requête, c'est la composante de composition de services qui est sollicitée afin de déduire un ou plusieurs plan(s) de composition satisfaisant une requête étendue. Le moteur de composition permet de sélectionner les services Web vérifiant les critères de composabilité exacte ou approchée. L'axiomatique proposée a été enrichie par deux nouveaux axiomes permettant d'inférer une composabilité fonctionnelle entre services. De plus, nous proposons d'étendre les techniques de relaxation basées sur les raisonnement déductif par l'utilisation de techniques de similarité sémantique entre concepts, l'objectif étant de garantir la réussite de l'exécution du plan de composition. Un graphe de composition est généré pouvant contenir plusieurs plans possibles de composition de services ayant chacun une *valeur agrégée des attributs des propriétés non fonctionnelles*, i.e., où chaque service atomique composant est affecté de son vecteur QdS. Cette étape est suivie d'un processus de sélection des services composites dits admissibles (i.e., vérifiant les contraintes de qualité de la requête du client) qui sera elle-même suivie d'une étape de classement. Deux alternatives de sélection et de classement des plans de composition admissibles sont proposées dans cette thèse :

- La première consiste à appliquer la même technique utilisée dans la découverte à savoir une combinaison de l'opérateur skyline et de la technique de classement top-k.
- La seconde consiste à sélectionner les plans de composition dont les services atomiques (composant le plan) vérifient toutes les contraintes QdS de la requête de l'utilisateur. Cette sélection s'effectue après que les valeurs des attributs de qualité de ces services atomiques aient été normalisées. Une fois les plans de composition

admissibles sélectionnés, une étape de classement de ces derniers est effectuée sur la base de la valeur normalisée de la fonction d'utilité globale de ces plans.

Organisation du rapport

Le présent mémoire de thèse est organisé en deux parties comme suit :

La première partie de cette thèse est consacrée à la revue bibliographique des principaux travaux de recherche portant sur la découverte et la composition des services Web dans le contexte du *Web sémantique*. La première partie comprend 4 chapitres répartis comme suit :

Dans le premier chapitre (chapitre introductif), nous présentons le contexte scientifique de nos travaux de thèse. Nous présentons une vue d'ensemble des différents domaines de recherche couverts par notre problématique pour, enfin, se pencher sur les questions scientifiques abordées.

Dans un le deuxième chapitre, nous présentons les services Web sémantiques comme l'adaptation des services Web au contexte du Web sémantique. Différents approches de descriptions sont proposées autour de ce nouveau concept. Nous présentons un état de l'art assez exhaustif de ces approches et nous concluons par un comparatif des différentes approches présentées.

Dans le troisième chapitre, nous présentons un certain nombre de travaux portant sur la découverte de services Web en couvrant celles qui adoptent des techniques de recherche d'information, celles qui se basent sur le raisonnement logique et celles qui combinent les deux. Nous dressons également un comparatif des différents approches de découverte présentées dans le chapitre.

Dans le quatrième chapitre, nous présentons une large panoplie de travaux développés autour de la composition de services. Nous détaillons les différentes approches proposées dans plusieurs domaines principalement celui de l'Intelligence Artificielle exploitant les approches de planification ou les systèmes multi-agents.

La seconde partie de cette thèse est consacrée à la présentation du méta-framework BioMed et ses différentes composantes. Le premier chapitre de la seconde partie (chapitre cinq) est consacré à une présentation du méta-framework BioMed en détaillant les différents niveaux de stratification. Nous présentons au niveau de ce chapitre trois de nos propositions à savoir un modèle sémantique d'unification permettant d'exprimer des connaissances descriptives de services Web, une *méta-carte ontologique* du domaine bioinformatique et un modèle inférentiel permettant d'exploiter deux types de connaissances : les connaissances descriptives et les annotations sémantiques définies sur la base de l'ontologie noyau.

Le chapitre six est consacré à la description de l'approche de découverte proposée et reposant sur les propositions présentées dans le chapitre précédent, nous présentons également une opérationnalisation du framework BioMed que nous nommons méta-service déductif [ALVR07, ALV08b]. Nous décrivons aussi les résultats des expérimentations menées dans ce contexte.

Au niveau du chapitre sept, nous présentons une approche de composition permettant

de déduire un ou plusieurs plans de services composables étant donné le besoin d'un client exprimé en termes de contraintes fonctionnelles et non fonctionnelles et nous étudions deux approches possibles permettant de sélectionner et classer les plans de composition entre eux en se basant sur leur critères QdS.

Une conclusion générale fera une synthèse du présent travail, exposera ses limites et proposera de nouvelles perspectives de recherche.

Après la bibliographie, nous présentons deux annexes : l'annexe A est consacrée aux ontologies fondationnelles alors que l'annexe B traite de quelques méthodes de la décision multicritère. Enfin, le document se conclut avec deux glossaires l'un consacré aux services Web et l'autre à la génétique moléculaire.

Chapitre 1

Position du Problème

Ce chapitre est une introduction à nos travaux de thèse. Nous y précisons le contexte scientifique dans lequel nous avons mené nos recherches, puis nous décrivons notre problématique en précisant notre contribution ainsi que les domaines de recherche abordés.

Sommaire

1.1 Contexte de la thèse

Depuis l'invention du World Wide Web par Tim Berners-Lee et Robert Cailliau [1] et le développement des technologies associées, l'Internet et le Web sont devenus bien plus qu'un simple instrument de partage d'informations. La manière dont les systèmes d'information interagissent au travers les réseaux, et la façon dont les applications sont développées avant l'avènement du Web ont été complètement remises en cause. En effet, l'Internet fournit aux organisations un moyen universel pour partager leurs ressources et savoir-faire, pour combiner, intégrer, agréger leurs applications afin d'offrir de nouvelles applications à valeur ajoutée, sans pour autant perdre de leur autonomie. Le concept de service Web, basé sur les standards de l'Internet et du Web notamment la boîte à outils XML [2], vise à promouvoir la réutilisation de certains composants logiciels développés au sein des organisations. Dans l'objectif de réutiliser un service, ce dernier devrait être recherché et sélectionné [3] par

1. Au sein du CERN à Genève, en 1989.
2. Extensible Markup Language, http ://www.w3.org/XML/
3. Il s'agit du processus de découverte d'un service Web.

des clients ayant des besoins divers et également interagir avec d'autres services provenant d'autres organisations dans le contexte d'une composition pour créer un service composite/composé correspondant à la chorégraphie ou l'orchestration d'un ensemble de services atomiques (élémentaires) et/ou composites (composés). Ayant été créés pour favoriser le partage, l'interoperation et l'interaction faiblement couplée entre plusieurs organisations, les services Web sont conçus et implantés indépendamment les uns des autres. Cela a donné lieu à une hétérogénéité qui pose divers problèmes au moment de la découverte du service et de sa mise en interaction avec d'autres services, i.e., lors d'une composition de services. Nous identifions dans cette thèse les différents types d'hétérogénéité existants dans le contexte des services Web et proposons des solutions pour les pallier.

Un des domaines d'application concernés par les problèmes de découverte et de composition de services Web est le domaine de la bioinformatique. Dans ce domaine, de nombreux services sont conçus et créés pour uniformiser l'accès à des sources d'informations gigantesques et à de nombreux outils scientifiques. Cependant, ces services (et par conséquent les ressources qu'ils encapsulent) sont caractérisés par une forte hétérogénéité qui s'illustre à différents niveaux, nous retenons dans cette thèse les deux niveaux : *syntactique* et *sémantique*.

C'est en nous intéressant aux problématiques de découverte et de composition de services que nous avons conçu un méta-framework de médiation de services Web dont le principal objectif est de faciliter la découverte des services pertinents aux besoins de la communauté de la bioinformatique et de guider ses membres dans la composition de ces services en *workflows* lorsqu'aucun service atomique ne satisfait leurs besoins.

La thématique traitée dans cette thèse se situe à la croisée de quatre domaines de recherche majeurs : l'interopérabilité des systèmes d'information, l'ingénierie ontologique, la bioinformatique et l'analyse multicritère. Comme approche d'interopérabilité de systèmes d'information, nous nous sommes particulièrement intéressés aux technologies de services Web en étudiant leur utilisation dans un domaine scientifique qu'est la bioinformatique. Nous nous sommes également intéressés à une branche de l'ingénierie de connaissances [SBF98], celle de l'ingénierie ontologique, le tout intéressant la communauté bioinformatique qui connaît depuis le séquençage du génome un regain d'intérêt incomparable.

1.2 Domaines de recherche

Dans cette section, nous décrivons les différents domaines de recherche abordés et qui sont principalement ceux de l'intéropérabilité des Systèmes d'Informations, de l'ingénierie ontologique, la bioinformatique et de la décision multi-critère multidimensionnelle.

1.2.1 L'interopérabilité des systèmes d'information

Les dernières décennies ont été marquées par l'émergence de nouvelles applications nécessitant un besoin accru de partage et d'échange d'informations distribuées entre différentes applications et ressources sur le Web. C'est le cas des applications du e-commerce, du e-gouvernement, du e-learning, de la bioinformatique et d'autres. Faciliter le partage et la collaboration entre différents systèmes d'information ne peut se réaliser que par le biais

d'une *plate-forme d'intéropérabilité*. L'interopérabilité des SI consiste en leur capacité à échanger des données et des fonctionnalités entre eux sans ambiguïté dans le but d'opérer ensemble [4].

Ouksel and Sheth, dans [OS99], distinguent trois niveaux d'interopérabilité : *syntactique* et *sémantique* qui correspondent aux différents types d'hétérogénéité retrouvés dans les systèmes d'information. L'hétérogénéité technique est celle liée à l'hétérogénéité des plate-formes logicielles et matérielles ainsi qu'aux protocoles de communication, l'hétérogénéité syntactique concerne les conflits des formats et des modèles de représentation de données, l'hétérogénéité sémantique résulte des discordances d'interprétation d'une valeur, d'un terme, d'un concept ou d'une conceptualisation du monde réel. La mise en oeuvre d'une plate-forme d'interopérabilité repose avant tout sur le fait que plusieurs organisations s'accordent sur un certain nombre de choix techniques, syntactiques et structurels afin de garantir un échange réussi des données et services, par exemple, un accord sur le protocole de transmission des messages, sur le format et type de représentation et données échangées. L'interopérabilité sémantique garantit que les données et les services échangés sont interprétés correctement par les différents parties impliquées dans l'échange.

Cette évolution des SI a entraîné le développement de nouveaux standards et paradigmes facilitant l'interopérabilité des systèmes. Des standards comme CORBA [5][GGM99], DCOM [6][HK97] et RMI [7][Fro07] ont été créés dans le but de faciliter l'interconnexion entre applications distantes, indépendamment des plates-formes et des langages utilisés et donc garantir une interopérabilité aux niveaux technique et syntactique. Cependant, ces standards, de par leur complexité et leur aspect fortement couplé, sont en fait incapables de passer à l'échelle, ils restent donc, le plus souvent, confinés à l'intérieur des organisations et leurs réseaux. Plus récemment, l'Architecture Orientée Service (Service-Oriented Architecture, SOA) se présente comme un paradigme architectural qui permet aux concepteurs de systèmes d'information d'organiser un ensemble de logiciels isolés en un ensemble de services Web interconnectés, accessibles par le biais d'une interface et de protocoles standards [Pap03]. Dans cette thèse, nous situons la notion de médiation au centre des approches d'interopérabilité. Nous discutons cette notion dans ce qui suit.

1.2.2 Notion de médiation

Le travail de **Wiederhold** [Wie92] est considéré comme le travail fondateur autour de la notion de médiation, bien que des logiciels ad hoc existaient préalablement. Le besoin de médiation est né de l'explosion de la quantité de données accessibles sur les réseaux informatiques. La multiplication des sources de données a rendu impossible la gestion de l'ensemble des informations par un système monolithique. Pour pallier ce problème, **Wiederhold** avait proposé un modèle architectural où un module logiciel est chargé d'accéder à un ensemble de sources de données, tout en offrant aux clients l'illusion d'utiliser un unique système d'information. C'est dans ce sens que la médiation est considérée comme

4. Interoperability What is it and why should I want it.
http ://www.ariadne.ac.uk/issue24/interoperability/intro.html
5. CORBA : Common Object Request Broker Architecture.
6. DCOM : Distributed Component Object Model.
7. RMI : Remote Method Invocation.

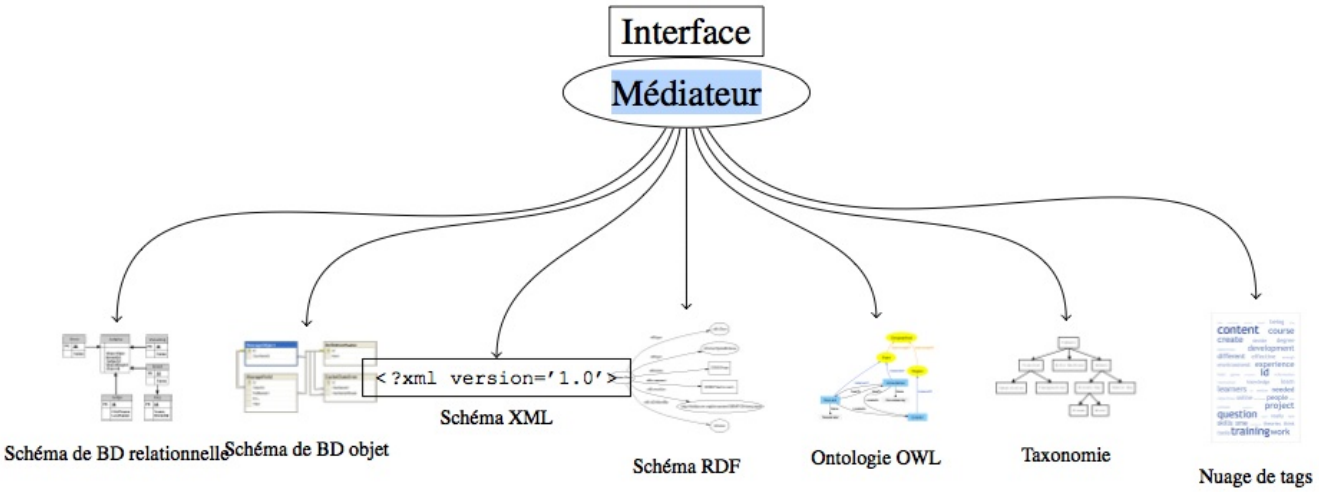


FIGURE 1.1 – Médiation de ressources sur le Web

une approche possible d'interopérabilité des systèmes.

Ce module logiciel est appelé médiateur ou intergiciel (*middleware*) et son existence est justifiée par deux caractéristiques des systèmes d'information.

- Le volume de données, empêche de traiter, d'exploiter ou de manipuler l'ensemble des informations simultanément. Il faut donc pouvoir appliquer une sélection sur celles-ci.
- La disparité empêche de regrouper l'ensemble des données pour offrir un système unique de sélection et de manipulation des informations. Il faut donc fournir des moyens de rendre les diverses sources de données interopérables.

La problématique reste la même lorsque les données et les sources de connaissances sont accessibles via des services Web. La médiation de services est censée permettre aux utilisateurs de découvrir et de sélectionner le service Web le plus adéquat à leur besoin et de composer un ensemble de services dans le cas où aucun service atomique ne répond au besoin de l'utilisateur.

1.2.3 L'ingénierie Ontologique

L'ingénierie des connaissances a connu depuis une quinzaine d'années un saut paradigmatique avec l'émergence du concept '*ingénierie ontologique*' et l'adoption de ce nouveau paradigme. C'est le professeur Riichiro MIZOGUCHI [8] qui a suggéré le premier l'appel-

8. MIZOGUCHI Riichiro est professeur à l'Université d'Osaka et directeur du "Knowledge Systems Lab" de cette institution. Pionnier dans le domaine de l'ingénierie ontologique, il est celui qui l'a introduit dans le monde des systèmes tutoriels intelligents et des EIAH. Sa présentation au colloque Intelligent Tutoring Systems : ITS '96, intitulée "Task Ontology Design for IntelligentEducational/Training Systems" fait date à ce sujet. Il est l'auteur de nombreux articles et communications sur le thème de l'ingénierie ontologique pour les environnements de formation. Président de la "International AI in Education Society"

lation Ingénierie Ontologique dès 1996 en ces termes ≪*Une ontologie est un système de concepts fondamentaux, c'est-à-dire un système de connaissances qui constitue les connaissances d'arrière-plan d'une base de connaissances ; une ontologie offre une conceptualisation du monde-cible ainsi qu'une base solide sur laquelle construire des bases de connaissances partageables, et plus largement utilisables qu'une base de connaissances conventionnelle.*"

L'Ingénierie Ontologique (I.O.) a été précisément conçue en vue de pallier les difficultés rencontrées par les systèmes à base de connaissances (SBC) en leur apportant une élégante solution car les connaissances considérées par l'I.O. sont cumulables, partageables et révisables en raison des modifications qui éventuellement surviennent au niveau de l'environnement socio-éco-scientifico-technologique. En effet, l'I.O. se fonde sur une méta-modélisation formelle dans laquelle les seules connaissances dont on dispose sont de nature ontologique ou cognitive ; l'objectif étant de générer à la fois des modèles de connaissances pour spécifier des problèmes dans des domaines spécifiques et des modèles inférentiels pour la résolution de ces problèmes.

Nous signalons aussi que la définition communément admise du concept "*ontologie*" dans le domaine de l'Intelligence Artificielle est celle énoncée par T. Gruber définissant l'ontologie comme une spécification explicite d'une conceptualisation [Gru93]. De ce point de vue, la construction d'une ontologie interviendrait après un travail de conceptualisation [9]. Cette démarche de conceptualisation consiste à identifier les connaissances spécifiques au domaine à représenter. Les travaux menés par N. Guarino permettent de définir cette notion de conceptualisation [Gua95]. Il s'agit, d'une part, de préciser la notion de domaine d'application ou de discours en distinguant parmi les entités peuplant un domaine, les individus et les propriétés, et, d'autre part, de préciser quelles sont leurs propriétés essentielles d'unicité et d'identité [Gua99].

La définition de Gruber a fait l'objet d'une précision de la part de B. Studer [SBF98] qui voit une ontologie comme la spécification formelle et explicite d'une conceptualisation partagée. Dans cette définition, il convient de mesurer la portée de chaque terme utilisé. Ainsi, le terme "*spécification explicite*" indique qu'une ontologie est un ensemble de concepts, de propriétés [10], d'axiomes [11], de fonctions et de contraintes explicitement définis. Le terme "*formel*" précise que cette conceptualisation doit pouvoir être comprise et interprétée par une machine à travers un langage formel et une sémantique associée. Le terme "*partagé*" précise l'aspect consensuel de la conceptualisation reflétant une certaine vision partagée du domaine. Ce qui suppose que l'on doit assurer une réutilisation de la formalisation choisie. Enfin, le terme "*conceptualisation*" implique également l'aspect intentionnel ou téléologique, lié à un objectif de réalisation.

De l'adoption des ontologies comme un "*outil*" de spécification des connaissances "*locales*" au sein d'une organisation, émane le besoin d'une médiation ontologique ayant pour objectif de réconcilier les sémantiques exprimées à travers différentes ontologies existantes

de 2001 à 2003, il est président élu de la "Japanese Society for Artificial Intelligence".

9. Une conceptualisation fait référence à un modèle abstrait qui sert à identifier les concepts pertinents au sein d'un univers de discours.

10. Les propriétés sont des restrictions des concepts ou des relations sémantiques entre concepts.

11. Les axiomes d'une ontologie permettent de définir la sémantique des termes, leurs propriétés et toutes les contraintes quant à leur interprétation.

sur le Web et donc de garantir l'interopérabilité sémantique. Un des domaines les plus concernés par l'ingénierie ontologique est le domaine bioinformatique. Dans ce domaine, plusieurs ontologies cohabitent, chacune offrant une vision partielle et subjective du domaine ou d'une portion du domaine (séquences, gènes, protéines, maladies, etc.). Elles ont été développées par des biologistes pour des fins spécifiques et consistent dans le cas le plus général, soit en une hiérarchie informelle, une taxinomie voire un thésaurus de concepts. Par ailleurs, nous notons que le développement d'ontologies dans ce domaine ne repose sur aucune méthodologie de développement d'ontologies et manque d'outils de formalisation et dans certains cas de clarté conceptuelle [SK05].

1.2.4 La bioinformatique

La *bioinformatique* est un champ de recherche multi-disciplinaire où travaillent de concert biologistes, informaticiens, mathématiciens et physiciens, dans le but de résoudre un problème scientifique posé par la biologie. Le terme bioinformatique peut également décrire (par abus de langage) toutes les applications informatiques résultantes de ces recherches. La bioinformatique est constituée par l'ensemble des concepts et des techniques nécessaires à l'interprétation de l'information génétique (*séquences*) et structurale (*repliement* 3D). La bioinformatique consiste à analyser, modéliser ou prédire les informations issues d'activités de recherche. Dans un sens encore plus étendu, on peut aussi inclure sous le concept de bioinformatique le développement d'outils de traitement de l'information basés sur des systèmes biologiques comme, par exemple, l'utilisation des propriétés combinatoires du code génétique pour la conception d'ordinateurs à ADN permettant de résoudre des problèmes algorithmiques complexes. Les principaux axes de recherche dans cette discipline est l'analyse de séquences, la modélisation moléculaire (en l'occurrence tridimensionnelle) et la construction d'arbres phylogénétiques.

1.2.5 Décision multi-critère multidimensionnelle

La problématique de cette thèse étant une approche de découverte et de composition de services Web sur la base d'un ensemble de besoins couvrant à la fois des propriétés fonctionnelles et des propriétés non fonctionnelles, la réponse aux préférences de l'usager en termes de qualité de service fait appel aux théories de la décision multicritère multidimensionnelle.

Les systèmes décisionnels comme dans la majorité des problèmes d'intelligence artificielle, des systèmes de recommandations, des outils de configuration etc. nécessitent des approches grâce auxquelles une capacité de prise décision est offerte comme solution en vue de permettre au décideur, au final, d'assumer des actions potentielles.

Le but de la prise de décision est d'entreprendre l'action qui implique le meilleur résultat (i.e. le résultat le plus satisfaisant, le plus préférable). Les actions et les préférences sont représentées souvent par un ensemble de contraintes sur un ensemble de variables (attributs) décisionnelles. Dans de nombreux domaines d'application, l'ensemble des actions possibles et des décisions potentielles est fixe et dépend d'une dynamique bien établie. Les seuls composants variables dans le processus de décision sont les préférences de l'utilisateur qui doivent être prises en compte lors de la prise de décision. Les fonctions d'utilité

constituent un outil idéal pour la représentation et le raisonnement sur les préférences quantitatives de l'utilisateur. La représentation des préférences par une fonction d'utilité est primordiale pour le succès de nombreuses applications en Intelligence Artificielle. Une bonne fonction de préférence doit permettre de capturer des énoncés qui sont naturels, simples et intuitifs pour l'utilisateur. Cependant, les fonctions d'utilité peuvent être très difficiles à formuler et un effort considérable est requis de l'utilisateur.

L'annexe B de ce mémoire traite des méthodes d'analyse multicritère selon les deux classes les méthodes exactes et les méthodes approchées. Dans le cadre des processus de découverte et de composition de sevices , nous nous proposons d'utiliser des heuristiques (des solutions approchées) pour contourner le caractère NP-difficile des problèmes induits par la formalisation des méthodes exactes. Ces méthodes seront testées sur les critères (attributs) non fonctionnels et notamment ceux relatifs à la Qualité de Service (QdS) des services Web. Nous privilégierons pour la découverte de Services Web répondant à des besoins en PNF, parmi les méthodes à base de la frontière de Pareto, celle basée sur l'opérateur Skyline. Pour la partie composition, tout en adoptant l'opérateur Skyline, nous proposons une alternative à cet opérateur soit une méthode non Pareto parmi de nombreuses alternatives possibles.

1.3 Problématique

L'objectif de cette thèse est d'étudier les mécanismes nécessaires à la mise en oeuvre d'un *cadre de référence* générique de *médiation sémantique de services Web* permettant la découverte et la composition dynamiques de services dans **un environnement ouvert** dans lequel le nombre de services et leurs descriptions sont en évolution constante. Cette médiation aura pour objectif de faciliter l'interopérabilité au sein des Architectures Orientée Services (AOS) en respectant les deux principes suivants :

- Des interactions faiblement couplées entre les composants d'une architecture orientée services, en respectant l'autonomie de chaque participant qui offrent et requièrent (réutilisent) des services afin de satisfaire des besoins divers.
- Une forte médiation permettant aux participants d'invoquer différents services en sélectionnant ceux qui sont les plus appropriés à leurs besoins et en les composant dans des workflows.

Dans le contexte de cette thèse, nous nous situons dans le contexte des services Web sémantiques. Ce concept se situe au croisement du paradigme du Web sémantique et des technologies de services Web, visant à intégrer la sémantique du domaine [12] dans la description d'un service. Plusieurs facettes de la sémantique des services peuvent être identifiées, nous distinguons les quatre facettes suivantes :

1. La facette *fonctionnelle* permettant de décrire l'interface du service en termes d'entrées, de sorties, préconditions et effets (IOPE) ;
2. La facette *téléologique* décrivant le but ou la tâche du service ;
3. La facette *comportement* permettant de décrire l'aspect comportemental du service en terme d'opérations élémentaires ;

12. modélisée par le biais d'une ontologie du domaine

4. La facette *non fonctionnelle* décrivant des aspects liés à la Qualité Du Service (QdS) ou la sécurité.

Les approches existantes reposent sur les ontologies pour expliciter une ou plusieurs facettes [13] de sa sémantique. A cet effet, nous distinguons différents types d'ontologies décrivant chacune des facettes : une *ontologie de domaine* pour décrire la facette fonctionnelle, une *ontologie de tâche* pour représenter la facette téléologique du service, une *ontologie de processus* pour décrire le comportement d'un service et une *ontologie de propriétés non fonctionnelles* pour la description des aspects non fonctionnels (notamment la qualité de service). L'utilisation de différents types d'ontologies ayant différents niveaux de granularité sémantique, différents champs de couverture sémantique et différents niveaux de formalisation donnent certainement lieu à des descriptions sémantiquement hétérogènes de services Web.

Ainsi, les principales questions auxquelles nous tentons de répondre dans cette thèse sont résumées comme suit :

- Comment exprimer la sémantique d'un service Web atomique ? A cet effet, nous nous intéressons seulement aux facettes fonctionnelle, téléologique et non fonctionnelle. Nous n'évoquerons pas la facette comportement visant à décrire le comportement interne d'un service composite. Cette facette ne concerne donc pas la sémantique d'un service atomique.
- Comment exploiter la sémantique explicite d'un service Web en vue de raisonner dessus dans le but de guider les processus de découverte et/ou de composition ? Il s'agit de réaliser une médiation sémantique entre les besoins des clients exprimés par le biais de requêtes et les descriptions "*sémantisées*" [14] de services Web.
- Généralement, un processus de découverte permet de rechercher les services sur la base de leurs aspects fonctionnels, il est courant que de nombreux services répondent à un même ensemble de besoins fonctionnels. Comment distinguer parmi un ensemble de services ayant des profils fonctionnels sémantiquement similaires les *meilleurs* services et sur la base de quels critères ?
- Comment peut-on déduire (d'une manière automatique ou semi-automatique) un plan de composition comprenant un ensemble de services composables (ou approxiamtivement composables) au cas où le besoin de l'utilisateur ne peut pas être satisfait par un service atomique ? Comment évaluer globalement la qualité d'un plan de composition et enfin comment classer plusieurs plans entre eux au cas où plusieurs plans sont identifiés ?

La Figure 1.2 résume les questions abordées dans cette thèse.

Dans le cadre de cette thèse, nous proposons une approche permettant de représenter une triple sémantique : *descriptive*, *ontologique* et *inférentielle*. En effet, nous proposons d'intégrer une dimension inférentielle au niveau des descriptions de services Web visant à faciliter les tâches de découverte et de composition aux utilisateurs finals par le biais d'un

13. aucune approche existante ne possède une couverture complète dans le sens où elle permet la représentation des quatre facettes dans une même description sémantique d'un service Web

14. Nous ne conformons pas à la traduction établie dans la littérature française du terme composé de "Semantic Web service" en service Web sémantique, nous proposons plutôt la notion de service Web sémantisé, nous argumentons le pourquoi de cette appellation dans la suite de ce mémoire de thèse

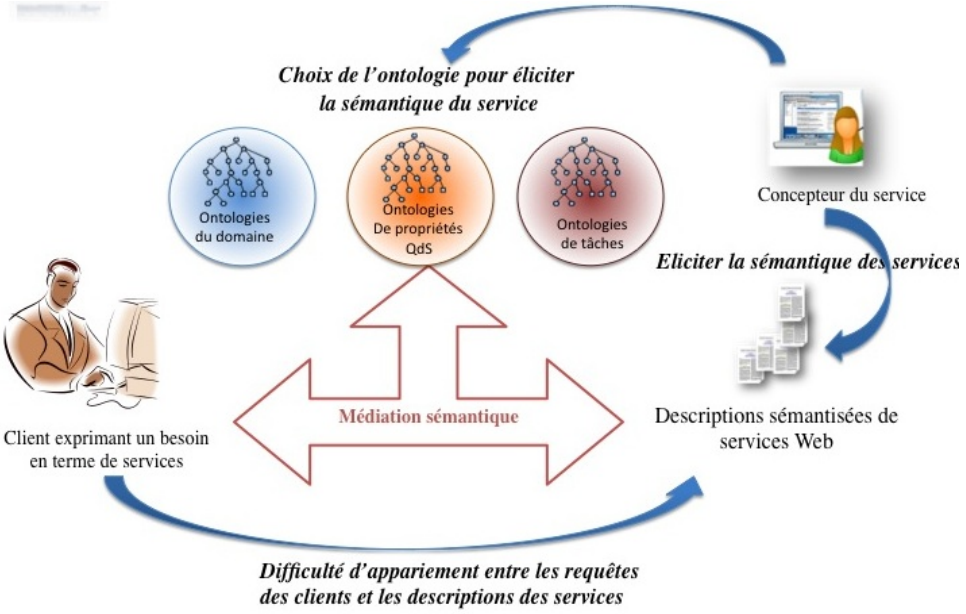


FIGURE 1.2 – Problématique de la médiation sémantique de services Web

moteur inférentiel dédié s'appuyant sur une sémantique axiomatique tout en exploitant les connaissances descriptives et ontologiques du domaine des services.

Ce travail de thèse tire ses motivations des limites suivantes identifiées au niveau de nombreux travaux étudiés dans l'état de l'art :

- Il est pratiquement impossible d'imaginer qu'une ontologie générique puisse être utilisée par tous les utilisateurs de n'importe quel domaine pour annoter un service Web parce que ceci suppose que chaque personne voulant créer un service adopte la même conceptualisation. De plus, dans certains domaines tels que le domaine bioinformatique, les ontologies existantes manquent de formalisation et de clarté conceptuelle. Pour ce faire, nous proposons une *méta-carte ontologique* conçue en se basant sur l'ontologie formelle comme DOLCE(*Descriptive Ontology for Linguistic and Cognitive Engineering*) [15].
- Les frameworks de description de services proposés dans le cadre du Web sémantique présentent l'inconvénient majeur qu'ils ne fournissent pas de techniques intégrées permettant d'exploiter les descriptions générées et ainsi automatiser le processus de découverte et de composition. D'un autre côté, les approches proposées dans le cadre de la découverte et de composition dépendent du framework sémantique adopté pour la description du service et donc il existe un besoin pressant d'un cadre de référence de médiation de services intégrant les différents frameworks de description sémantique de services.
- Le besoin d'une *sémantique formelle* est indispensable pour garantir l'automatisation des processus de découverte et de composition. Or, la plupart de frameworks pro-

15. Le choix de DOLCE sera justifié plus loin et notamment dans l'annexe A.

posés, à l'exception de WSMO [RKL+05b], ne possèdent pas de sémantique formelle au niveau des descriptions de services Web. Il est important de disposer d'un formalisme formel avec des primitives sémantiques permettant de préciser la sémantique du service et de concevoir des mécanismes d'inférence valables partout dans le domaine du discours.

En réponse aux limites et besoins identifiés et à la problématique générique de l'interopérabilité des systèmes, nous proposons un méta-framework stratifié de médiation sémantique de services Web. Nous présentons dans ce qui suit les différentes contributions du méta-framework proposé.

1.4 Contributions de la thèse

Les recherches que nous menons dans cette thèse visent à proposer une approche générique de médiation sémantique se basant sur l'ingénierie ontologique pour permettre la découverte et la composition dynamiques de services Web. La généricité recherchée sera réifiée par un méta-framework multi-strate multi-facette pour la médiation de services Web, nommé BioMed (BIOinformatic MEDiation Framework) qui repose sur :

1. **un modèle d'alignement sémantique** de descriptions de services Web hétérogènes provenant de différents frameworks permettant d'expliciter une sémantique unifiée en offrant un ensemble de primitives canoniques descriptives. Il s'agit à ce niveau d'une médiation sémantique entre les frameworks de SWS suivants : OWL-S, WSMO et SAWSDL, en proposant un **modèle pivot** vers lequel tous les modèles de descriptions sous-jacents aux frameworks de SWS sont *mappés*. La particularité de ce modèle d'unification est qu'il permet de modéliser à la fois les propriétés fonctionnelles (entrées, sorties, préconditions et postconditions) que nous nommons signature fonctionnelle du service et les propriétés non fonctionnelles liées à la qualité du service. La présence de ces deux types de propriétés au sein d'une même description n'est pas toujours réalisée dans les frameworks existants à l'exception de WSMO ;
2. **Une méta-carte ontologique** normalisant la sémantique des concepts et des sous-concepts des différentes ontologies du domaine conçue comme une ontologie intermédiaire entre l'ontologie de haut niveau de type DOLCE et les ontologies du domaine bioinformatique. L'engagement ontologique élaboré lors de la conception de la méta-carte émane du soubassement de l'ontologie fondationnelle DOLCE qui permet de qualifier la nature intrinsèque des concepts modélisés en les rattachant à des méta-catégories tels que les endurants ou les perdurants. Ainsi, chaque concept de la méta-carte synthétise dans son hyper-concept les concepts instances des ontologies de domaine relevant de cet hyper-concept (i. e., l'hyper-concept de la méta-carte '*gene*' correspond aux instances du concept Gene de la *Gene Ontology*) ;
3. **Un modèle inférentiel** permettant d'exploiter les différents types de connaissances modélisées au niveau du framework BioMEd : les connaissances descriptives explicitant la sémantique des services Web, les connaissances ontologiques explicitant les connaissances du domaine et les annotations sémantiques de la signature fonctionnelle décrivant le *mapping* entre connaissances descriptives et ontologiques. Le

modèle inférentiel repose sur un sémantique axiomatique [16] comprenant un ensemble de schémas d'axiomes [17] permettant de raisonner à trois niveaux différents : *le niveau ontologique*, *le niveau service* et *le niveau instance*.

4. **Une nouvelle approche de découverte et de composition** pour la sélection des services répondant fonctionnellement et non fonctionnellement à une requête spécifiant les contraintes du client dans un environnement ouvert. Cette nouvelle approche comprend deux phases : une *phase de sélection* et une *phase de ranking* (classement). La phase de sélection repose sur des techniques de relaxation des propriétés des services afin d'élargir leur portée et matcher au mieux les contraintes d'une requête de découverte. Cette phase permet de déduire différents degrés d'appariement. La phase de classement exploite les propriétés non fonctionnelles des services dans le but de classer l'ensemble des solutions déduites durant la phase de sélection. L'approche de classement proposée considère le *skyline* de services et leurs propriétés QdS. Un skyline (frontière de dominance) de services est un ensemble de services incomparables en terme de leurs valeurs QdS (aucun service n'est dominé par un autre). L'approche Top-k permet de sélectionner les meilleurs services ayant les meilleures valeurs maximisant une fonction de score. Au niveau de l'approche de composition, nous proposons deux alternatives de classement des plans de compositions obtenus : la première est la même que celle utilisée dans la découverte et la seconde est basée sur une technique de clusterisation.
5. **Un méta-service Web déductif** est proposé comme une opérationnalisation du méta-framework BioMed. Le méta-service repose sur une base déductive de données dont la composante extensionnelle comprend l'ensemble des connaissances descriptives et ontologiques et la composante intensionnelle comprend les connaissances inférentielles implémentant les différents schémas d'axiomes proposés par le modèle inférentiel de BioMed. Un moteur de raisonnement déductif implémenté en Prolog permet de montrer l'intérêt des techniques de raisonnement proposées pour un processus de découverte et de composition de services Web.

En résumé, le méta-framework BioMed se propose d'assurer une triple médiation : une médiation de services Web, une médiation ontologique et une médiation de données. La strate inférentielle de BioMed permet d'exploiter les connaissances ontologiques afin d'offrir des mécanismes flexibles de découverte et de composition de services Web (médiation de services Web). La médiation ontologique est garantie par la méta-carte ontologique permettant de représenter le double engagement sémantique et ontologique [18]. La médiation de données se base sur le modèle d'alignement sémantique afin d'offrir un modèle pivot pour la description d'un service.

16. Une sémantique axiomatique est une représentation formelle d'un ensemble de règles destinées à raisonner sur une connaissance. Elle est constituée de définitions, d'axiomes, de postulats, et de théorèmes.
17. Un axiome est une règle admise dans un système sans démonstration car une telle démonstration sera soit impossible, soit inutile comme étant évidente en soi.
18. Il s'agit de préciser quelles primitives (logiques ou non) munies de leur signification sont nécessaires pour la représentation des connaissances dans un contexte donné.

Première partie

Etat de l'art

Chapitre 2

Services Web, Sémantique et Descriptions sémantiques

Nous présentons dans ce chapitre un état de l'art des travaux portant sur la description et l'annotation sémantique de services Web développés sous la bannière du Web sémantique. Les différents frameworks présentés dans ce chapitre sont : OWL-S (Ontology Web Language for Services), WSMO (Web Service Modeling Ontology), SAWSDL (Semantic Annotation for WSDL), METEOR-S (Managing End-To-End OpeRations for Semantic Web service) et IRS (Internet Reasoning Service). En conclusion de ce chapitre, nous présentons une synthèse et un comparatif des différents frameworks.

Sommaire

2.1 Introduction

L'un des nombreux objectifs de l'Architecture Orientée Services (AOS) consiste en la réutilisation de ses briques de base d'implémentation, à savoir les services. Dans le cadre d'une architecture basée sur la technologie des services Web, la réutilisation de services Web repose sur le fait que ces derniers soient accessibles sur le Web et utilisables par le

plus grand nombre de clients possibles via un ensemble de protocoles standards [1]. [Bal08]

Afin de garantir leur réutilisation au sens large, les services Web doivent être décrits dans un langage interopérable. Un langage de description de services Web fournit un ensemble de primitives descriptives permettant de décrire les propriétés pertinentes d'un service. Nous distinguons une description syntactique telle que celles fournies par le langage WSDL (Web Service Description Language) d'une description sémantique intégrant une dimension sémantique au sein de la description du service Web. Une description syntactique d'un service Web est une description purement fonctionnelle comprenant un ensemble d'opérations, de messages et de ports de communications, en l'occurrence une description WSDL se base aussi sur un système de typage élémentaire comprenant les types de base du langage XML (type entier, chaîne de caractères). Nous nous penchons dans ce chapitre sur les différentes approches de description sémantique de services Web.

2.2 Du Web au Web Sémantique aux Services Web Sémantiques

Le Web sémantique est un domaine en pleine évolution depuis son lancement en 2001 [BHL01] par Tim Berners-Lee, l'inventeur du Web en 1989. L'originalité du concept du Web sémantique consiste à doter l'information des pages du Web d'une représentation de la sémantique de leur contenu pour exprimer du sens. Cette représentation sémantique sert à indexer ces pages sous la forme de *métadonnées* (données sur les données) que les machines pourraient interpréter ; les métadonnées constituent, en définitive, une médiation sémantique possible entre le contenu des pages du Web et les applications computationnelles ; en offrant une sémantique partageable par les humains et les machines. Il serait alors possible dans ce contexte d'instaurer des services automatiques supportant l'utilisation du sens de l'information contenue dans les pages Web.

Ainsi, le but du Web sémantique est d'ajouter une couche '*sémantique*' dont le sens devient exploitable et compréhensible par les machines. Tim Berners-Lee explicitait ce but en ces termes : "*L'enjeu étant que les logiciels acquièrent la capacité d'établir entre eux un véritable dialogue, c'est-à-dire un échange pertinent d'informations immédiatement exploitables sans intervention humaine*"[BHL01]. A ce jour, le standard de description de services Web WSDL ne supporte pas la description de services Web comme une ressource utilisable dans le contexte du Web sémantique.

Les premiers travaux qui s'intéressent à l'intégration fonctionnelle évitent le problème fondamental de l'automatisation des différentes étapes liées à la fourniture d'un service Web (par exemple, découverte et composition) puisqu'ils se limitent à l'usage des services Web par des utilisateurs humains plutôt que par des agents logiciels. En effet, de

1. L'avantage des services Web, par rapport aux autres approches de systèmes distribués tels que RMI (pour Remote Method Invocation), réside dans leur articulation autour du bouquet XML (standard promulgué par le W3C), ce qui leur procure l'avantage d'être non propriétaire et ainsi multi-plateforme. Comme protocoles standards développés autour de la technologie de services Web, nous distinguons WSDL (acronyme de Web Service Description Language) pour la description de l'interface du service Web servant à communiquer avec le service, UDDI (acronyme de Universal Description Discovery and Integration) qui est une technologie d'annuaire basée sur XML permettant de localiser sur le réseau un service Web recherché et SOAP (acronyme de Simple Object Access Protocol) qui est un protocole permettant l'échange d'informations structurées dans un environnement décentralisé et distribué.

nombreuses connaissances, indispensables à l'automatisation des processus liés aux services, sont soit absentes, soit décrites pour être interprétées et exploitées par des humains (sous forme de description textuelle). Il semble donc nécessaire de tendre vers des services intelligibles par des machines : c'est le concept de *Service Web sémantique*.

Ainsi, le besoin d'automatisation du processus de conception et de mise en œuvre des services Web rejoint les préoccupations à l'origine du Web sémantique, à savoir ***comment décrire formellement les connaissances de manière à les rendre exploitables par des machines***. C'est dans ce sens que les technologies et les outils développés dans le contexte du Web sémantique peuvent certainement compléter la technologie des Service Web en vue d'apporter des réponses crédibles au problème de l'automatisation. Par exemple, la notion d'ontologie peut jouer un rôle prépondérant pour permettre d'expliciter la sémantique des services facilitant ainsi les communications hommes-machines, d'une part, et les communications machines-machines, d'autre part.

De manière générale, l'objectif visé par la notion de Service Web sémantique est de créer ***un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines et ce, en utilisant les couches techniques sans pour autant en être conceptuellement dépendants (couplage faible, loose coupling).*** La sémantique ainsi exprimée permet l'automatisation des fonctionnalités suivantes qui sont nécessaires pour une collaboration inter-entreprises efficace :

- Le processus de description et de publication des services. La publication de services Web est l'action de rendre accessible un service Web ;
- La découverte des services. La découverte de services est l'action de localiser des services Web satisfaisant une tâche donnée ;
- La sélection des services. La sélection est l'action de choisir le service le plus approprié pour satisfaire une tâche donnée ;
- La composition des services. La composition de services est la mise en interaction de plusieurs services en vue de satisfaire un but qu'un service ne peut pas satisfaire.
- La fourniture et administration des services concerne l'action d'invocation et d'interaction au plus niveau (niveau protocole de transport de données).
- La négociation des contrats.

Mcllraith et Zeng [MSZ01] définissent les *services Web sémantiques* comme la combinaison de deux technologies : celle des services Web en tant que ressources logicielles réutilisables et interopérables et celle du Web sémantique ayant pour objectif de rendre les données interprétables par machine. La figure 2.1 situe les services Web sémantiques selon deux axes orthogonaux (syntaxique/sémantique) et (statique, dynamique).

Chris Preist [Pre04] introduit un ensemble de concepts-clés inhérents à cette technologie hybride que sont les services Web sémantiques. Nous n'évoquons ici que les concepts de notion de service et de représentation de service qui aident à la compréhension des travaux portant sur la description de services Web sémantiques.

La notion de service. Selon Preist [Pre04], la notion de service intègre trois idées principales. Les deux premières respectent des notions sous-jacentes à la description d'un service Web classique : un service est une interaction entre deux parties (communément

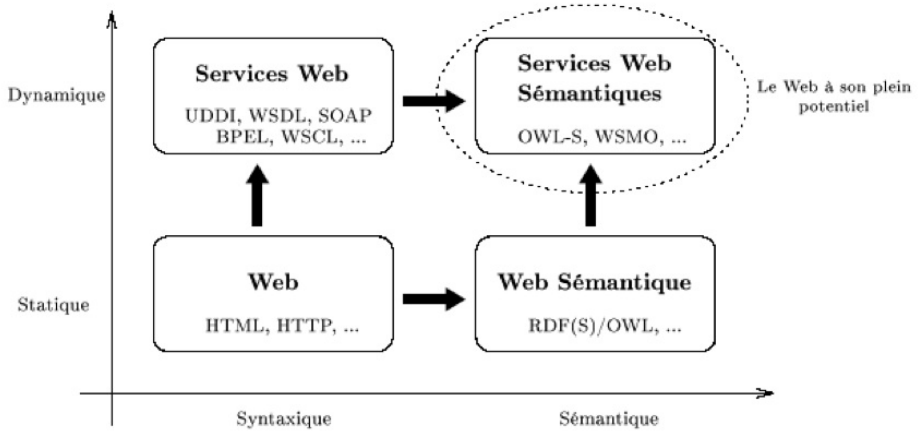


FIGURE 2.1 – Les services Web sémantiques se situent entre deux problématiques liées au Web : L'interopérabilité et l'annotation sémantique des ressources

appelées dans le domaine des services Web fournisseur et client). Un service doit être considéré à différents niveaux d'abstraction : le service concret (suite spécifique d'actions) et le service abstrait (ensemble de services concrets sans les précisions techniques utiles lors de la sélection de services). La troisième idée sous-jacente à la notion de service est que cette dernière a un sens dans la mesure où le service est inclus dans un domaine d'application (par exemple, le domaine du tourisme, de la bioinformatique, du e-commerce). [Pre04] définit ce domaine comme étant le *domaine de la valeur du service*.

La représentation du service. L'intérêt de combiner les principes du Web sémantique et la technologie de services Web est de proposer une représentation de services, compréhensible et interprétable automatiquement par les machines. Dans ce contexte, la description des services repose sur des techniques issues de l'ingénierie de connaissances et précisément celles de l'ingénierie ontologique. Dans ce domaine, de nombreux langages et techniques formels pour décrire les connaissances et raisonner sur ces dernières ont été développés autour du concept clé d'ontologie. La description des services Web sémantiques repose sur trois choix. Tout d'abord, le choix du méta-modèle qui précise les aspects du service à décrire (sa capacité, son interface, son fournisseur, ses opérations, son coût, sa qualité, etc.). Ensuite, le choix des concepts et relations à prendre en compte dans la description et leur(s) signification(s). Et enfin, le choix du langage de représentation de connaissances à utiliser pour opérationnaliser le méta-modèle de description de services Web. Des ontologies ont été développés comme méta-modèle de descriptions sémantiques de services Web, elles sont connues sous le terme d'*ontologies de service*. Les ontologies du domaine fournissent elles un support pour la description du domaine de la valeur du service.

Nous détaillons dans ce qui suit les différentes propositions existantes dans le contexte des services Web sémantiques. Chaque proposition/approche propose un méta-modèle de descriptions sémantiques de services Web.

2.3 Frameworks de Description de Services Web Sémantiques

Nous nous proposons de détailler les approches orientées services Web sémantiques en présentant les approches les plus représentatives et qui sont, à notre sens : OWL-S [BHL+04, MBM+07], WSMF [FB02], WSMO [RKL+05b], SAWSDL [KVBF07], METEOR-S [POSV04, SGR08], IRS-II [MDCG03] et son extension IRS III [CDG+06, DCG+08].

Chacune de ces approches est motivée par un certain nombre d'objectifs. Néanmoins, elles ont en commun les deux objectifs suivants :

- La découverte automatique de services Web permettant de localiser, d'une manière automatique, des services Web qui satisfont un besoin spécifié par un client.
- L'invocation automatique des services Web : En offrant une description déclarative d'un service Web et la sémantique de son domaine, un agent logiciel doit être capable d'invoquer automatiquement un service Web après l'avoir découvert.

Le reste de cette section est organisé de la façon suivante. Dans la section 2.3.1, nous présentons le framework OWL-S. La section 2.3.2 présente brièvement le framework WSMF avant d'évoquer le cadre conceptuel WSMO dans la section 2.3.3. Dans la section 2.3.4 nous étudions l'approche METEOR-S. La section 2.3.5 présente la recommendation SAWSDL pour l'annotation sémantique du standard WSDL (Web Service Description Language). Enfin, la section 2.3.6 est consacrée au cadre IRS-II et son extension IRS-III. Enfin la section 2.4 synthétise et compare les six approches présentées avant de conclure en introduisant nos motivations pour la conception d'un méta-framework générique de description pour harmoniser les principales approches existantes et de s'en servir pour développer une approche générique de découverte et de composition.

2.3.1 OWL-S (Ontology Web Language for Services)

OWL-S, anciennement DAML-S est une ontologie pour la description sémantique des services Web qui a été développée dans le cadre du projet DAML [2]. OWL-S se base sur le langage standard OWL [MvH04] qui permet de représenter des ontologies de façon standardisées sur le Web. L'objectif de OWL-S est de formaliser de façon non ambiguë les services Web de manière à ce qu'un agent logiciel puisse exploiter automatiquement les informations concernant ces services. Les bénéfices de l'emploi d'une ontologie de services peuvent être multiples, mais le plus intuitif concerne sans doute l'amélioration et la pertinence de la recherche de services et également leur composition.

OWL-S distingue trois éléments principaux qui interviennent dans la description d'un service :

- *ServiceProfile* : exprime ce que le service propose ainsi que les pré-requis exigés pour son emploi ;
- *ServiceModel* : exprime le fonctionnement du service ;
- *ServiceGrounding* : exprime comment le service peut être utilisé.

La figure 2.2 représente une vue globale de l'ontologie de service OWL-S.

2. DAML (acronyme de DARPA Agent Markup Language) est un langage de définition d'ontologies, prédécesseur du langage OWL (Ontology Web Language) et défini comme une extension de XML et de RDF (Resource Description Framework).

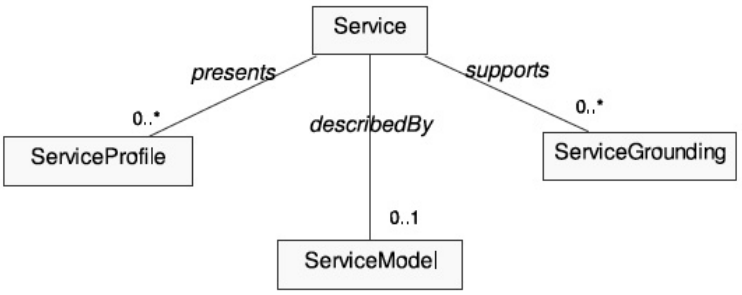


FIGURE 2.2 – Vue globale de l'ontologie de service OWL-S

La sous-ontologie *ServiceProfile* dont le modèle conceptuel est illustré par la figure 2.3 fournit les informations nécessaires à un agent pour publier un service ou le découvrir. La sous-ontologie décrit le profil d'un service par un ensemble de propriétés qui inclut entre autre le nom du service (*ServiceName*), sa description informelle (TextDescription) et des informations sur son fournisseur (*ContactInformation*). La description fonctionnelle d'un service Web spécifie :

- les entrées et les sorties du service par le biais des classes *Input* et *Output* définies comme étant des sous-classes de la classe *Parameter*,
- l'impact de l'exécution du service sur l'état du monde réel en termes de pré-conditions et d'effets par le biais des classes *Condition* et *Result*.

Le profil d'un service est associé à une liste de ses paramètres (*ServiceParameter*). Enfin, l'ontologie *ServiceProfile* permet de définir la classe du service (ServiceClassification) et le produit du service (ServiceProduct). Ceci est réalisé par le biais de correspondances vers des systèmes de classification tels que NAICS [3](North American Industry Classification System) ou de produits telles que UNSPSC [4](United Nations Standard Products and Services Code).

Dans le but de fournir une vue détaillée de la façon dont le service fonctionne, le framework OWL-S définit une ontologie de processus dans laquelle le service est perçu comme un processus (*Process*) ayant un ensemble d'entrées, de sorties, des pré-conditions et des effets (IOPE). Les pré-conditions et les effets permettent de rendre compte de l'interaction d'un processus avec son environnement de sorte que les pré-conditions doivent toutes être vérifiées pour que le processus puisse être invoqué et que les effets doivent spécifier les modifications de l'environnement suite à l'exécution du service.

Tel qu'illustrés dans la figure 2.4, les processus peuvent être de trois types différents :

- les processus atomiques (*AtomicProcess*) correspondent à des services simples directement invocables et s'exécutant en une seule fois sans intervention de l'utilisateur ;
- les processus composites (*CompositeProcess*) sont décomposables en processus atomiques et composites. Ils permettent de spécifier l'enchaînement de leurs différents composants à l'aide de structures de contrôle (*ControlConstruct*) comme *Sequence* ou *If-Then-Else* ;

3. http ://www.census.gov/eos/www/naics/
4. http ://www.unspsc.org/

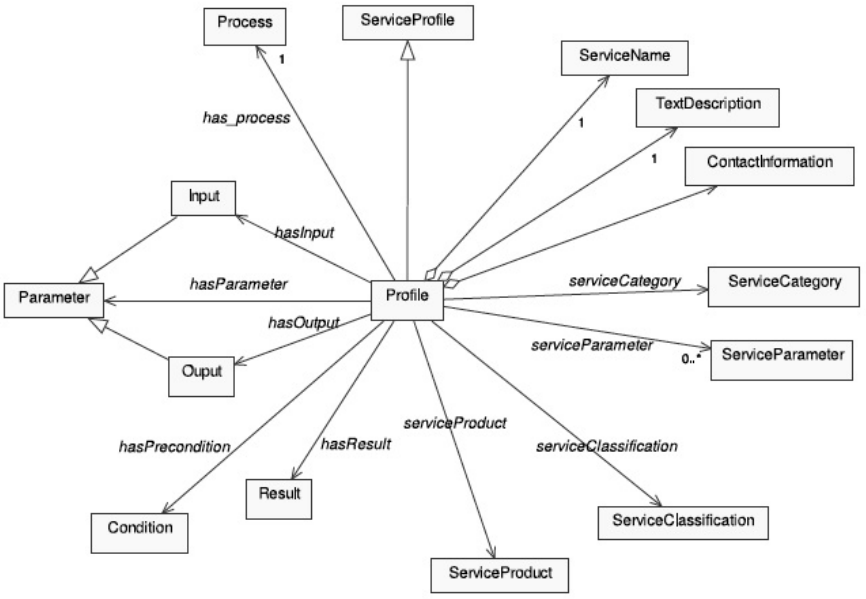


FIGURE 2.3 – Vue conceptuelle de la classe *ServiceProfile*

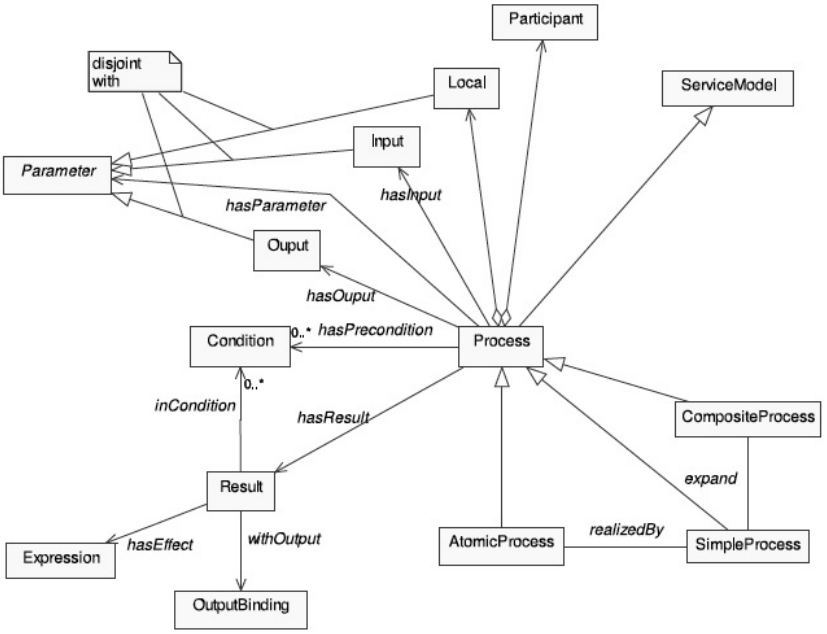


FIGURE 2.4 – Vue conceptuelle de la classe *ServiceModel*

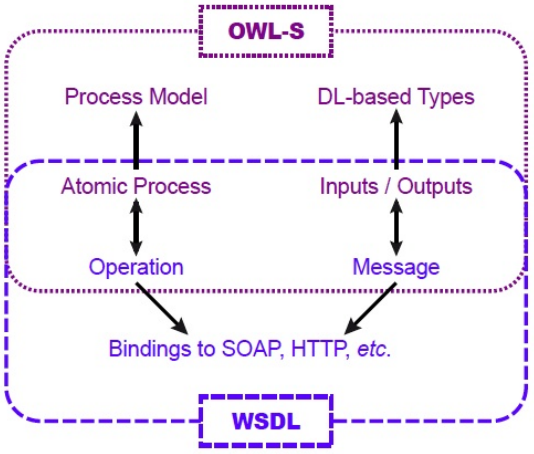


FIGURE 2.5 – Relation entre OWL-S et WSDL

– les processus simples (*SimpleProcess*) qui ne sont pas invocables et qui correspondent uniquement à des abstractions de processus atomiques ou composites. Un processus simple est « réalisé par » (*RealizedBy*) un processus atomique ou « se décrit par » (*ExpandTo*) un processus composite.

Enfin, la classe *ServiceGrounding* spécifie la manière dont les agents peuvent accèder et invoquer un service. Les sous-ontologies *ServiceProfile* et *ServiceModel* offrent une représenta-tion abstraite d'un service, alors que la classe *ServiceGrounding* offre une spécification concrète du service. Le passage d'une description OWL-S vers une description WSDL1.1. se base sur les trois correspondances suivantes : (a) chaque processus atomique OWL-S correspond à une opération WSDL, (b) les entrées et les sorties au niveau d'une description OWL-S correspondent aux entrées et sorties d'un message d'une opération WSDL et (c) les types des entrées et sorties OWL-S correspondent à des notions extensibles des types abstraits de WSDL. La figure 2.5 illustre les relations entre une description OWL-S et une description WSDL.

2.3.2 WSMF (Web Service Modeling Framework)

WSMF [FB02] est un framework complet de description de services proposé dans le cadre du projet européen SWWS (*Semantic Web enabled Web service*). Il constitue une extension du langage UPML (*Unified Problem-Solving Method Development Language*) [FMH+99] et il est centré autour de deux objectifs complémentaires, à savoir :

– Un découplage fort entre différents composants servant à réaliser une application de e-commerce à base de services Web ;

– Un service de médiation fort qui autorise tous les partenaires à communiquer entre eux et ce de manière évolutive (en terme du nombre de partenaires).

L'architecture de WSMF est constituée de quatre éléments principaux comme le montre la figure 2.6 : les *objectifs*, les *ontologies*, les *descriptions des services Web* et les *médiateurs*.

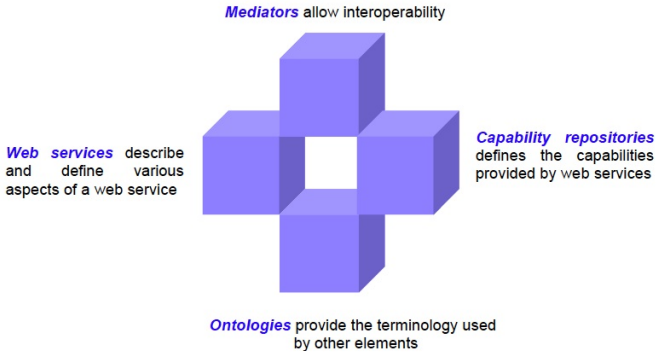


FIGURE 2.6 – Architecture du framework WSMF

Les objectifs permettent de définir les problèmes qui doivent être résolus par les Web services. Ils incluent des pré-conditions et des post-conditions. Les ontologies fournissent la terminologie et la sémantique utilisées par les autres éléments. Les descriptions des services Web définissent les différents aspects liés aux services Web (entrées, sorties, opérations, etc.). Les médiateurs permettent de résoudre les problèmes liés à l'intégration entre les services Web. Il est à noter que WSMF est un modèle conceptuel et par conséquent ne définit pas de sémantique concrète particulière pour la représentation des services. Selon la spécification publique de WSMF, ceci peut être réalisé à travers l'utilisation de WSFL ou de OWL-S, et de PSL [5].

2.3.3 WSMO (Web Service Modeling Ontology)

WSMO (acronyme pour *Web Service Modeling Ontology*) [RKL+05b] est une extension et un raffinement du framework WSMF. Elle fournit à la fois un langage formel et une ontologie permettant de décrire des aspects variés des services Web sémantiques. La spécification de WSMO repose sur un modèle stratifié qui inclut la définition de trois différentes ontologies : WSMO Lite (ontologie de base), WSMO Standard (ontologie intermédiaire) et WSMO Full (qui contient tous les concepts définis dans WSMO).

L'architecture Web Service Modeling Ontology (WSMO) qui a été proposée initialement par le laboratoire DERI devenu ensuite le Semantic Technology Institute (STI) de l'université d'Innsbruck (Autriche), est une architecture conceptuelle visant à expliciter la sémantique des services Web. Une vue globale des composants du framework WSMO est illustrée dans la figure 2.7. WSMO est organisée autour de quatre éléments principaux :

- Les **services Web** (*Web services*) sont définis comme des entités computationnelles qui fournissent une ou plusieurs fonctionnalités. Une description est associée à chaque service, dans le but de décrire sa fonctionnalités, son interface, et ses détails internes.
- Les **Objectifs** (*Goals*) servent à décrire les besoins des utilisateurs en termes de

5. L'objectif de PSL (acronyme de Process Specification Language) est de définir une ontologie et un format standard pour l'échange des processus industriels. La spécification de l'ontologie PSL est accessible à l'adresse http ://www.mel.nist.gov/psl/

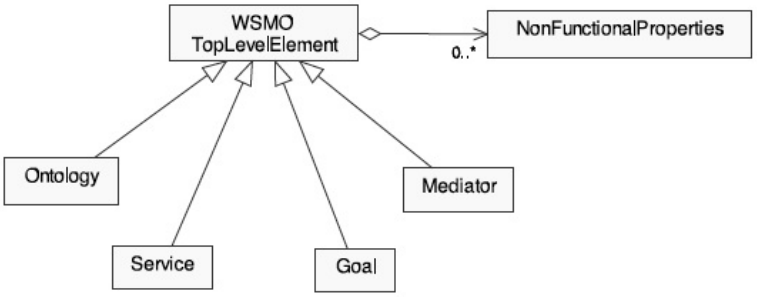


FIGURE 2.7 – Vue globale de l'ontologie WSMO

fonctionnalités requises. Les objectifs offrent une vue orientée utilisateur du processus d'utilisation des services Web, ils sont définis comme une entité à part entière dans le modèle WSMO. Un objectif décrit la fonctionnalité, les entrées/sorties, les pré-conditions et post-conditions du service Web requis.

– Les **Médiateurs** (*Mediators*) sont utilisés pour résoudre de nombreux problèmes d'incompatibilité.

– Les **Ontologies** (*Ontologies*) fournissent la terminologie de référence aux autres éléments de WSMO (services Web, Objectifs et médiateurs) afin de spécifier le vocabulaire du domaine de connaissance d'une manière interprétable par les machines.

Contrairement à OWL-S, WSMO inclut les médiateurs comme composants clés de son architecture. A ce niveau, les hétérogénéités rencontrées lors de l'utilisation de services Web sont gérées par les différents types de médiation :

– *La médiation de données* qui résout les incompatibilités de représentation des données,

– *La médiation de processus* est relative à la logique applicative de la composition,

– *La médiation de protocoles* adapte les différents protocoles de communication utilisés.

Ces différents niveaux de médiation sont pris en charge au niveau du framework WSMO par quatre familles de médiateurs :

– Les *GG-médiateurs* permettent d'effectuer la médiation entre deux objectifs, GG signifiant <*goal-goal*>. Cela signifie qu'ils permettent d'établir des correspondances entre objectifs, en se servant des ontologies d'objectifs disponibles dans le cadre de WSMO.

– Les *WG-médiateurs*, avec WG pour <*Web service-Goal*>, établissent les correspondances entre les fonctionnalités offertes par les services Web et les requêtes des utilisateurs, qui sont toutes les deux définies comme des objectifs. L'intérêt de ce type de médiateurs est d'aider la découverte et la sélection de services Web.

– Les *WW-médiateurs*, avec WW pour <*Web service-Web service*>, établissent les correspondances entre services Web. Leur tâche est de résoudre les conflits au niveau des données, du protocole, et du processus de composition. Ils sont mis en oeuvre lors de l'orchestration des services Web au sein d'une composition.

– Les OO-médiateurs, avec OO pour <*Ontology-Ontology*>, sont destinés à résoudre les conflits entre ontologies en créant des correspondances inter-ontologies.

Les autres types de médiateurs énoncés ci-dessus, ainsi que n'importe quel élément de

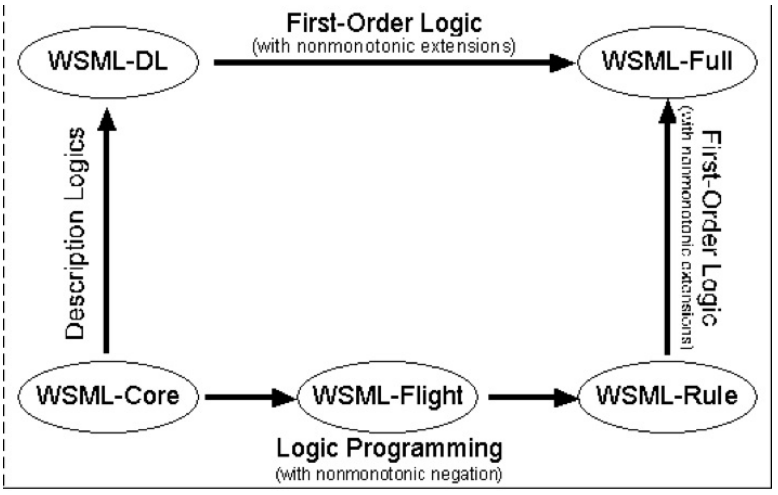


FIGURE 2.8 – Différentes variantes du langage WSML

l'architecture WSMO pourrait utiliser un OO-médiateur pour résoudre un conflit sémantique. Le travail des OO-médiateurs consiste à établir des correspondances entre les terminologies contenues dans les différentes ontologies pour les intégrer en une représentation homogène des données. Cette représentation homogène permet de résoudre les hétérogénéités sémantiques et de répondre aux requêtes soumises par les composants de l'architecture WSMO.

Dans l'approche WSMO, on distingue le langage de modélisation WSML (*Web Service Modeling Language*) [BLPF06] et l'environnement d'exécution WSMX (*Web Service Execution Environment*) [HCM+05]. WSML fournit un langage formel pour la description des éléments définis dans l'architecture WSMO. WSML est basé sur la combinaison de différents langages logiques qui sont la logique de description, la logique de premier ordre et la logique de programmation. Le langage WSML est structuré en cinq sous-ensembles : WSML-Core, WSML-DL, WSML-Light, WSML-Rule et WSML-Full. Chacun de ces sous-ensembles fournit un niveau d'expressivité croissante. La figure 2.8 résume les différentes variantes du langages WSML.

WSMX est un environnement d'exécution qui permet d'offrir un certain nombre de modules utilisables permettant de prendre en charge la découverte, la sélection, la médiation et l'invocation des services sémantiques décrits en WSMO. Nous listons ci-après huit outils développés par le groupe ESSI (European Semantic Systems Initiative) [6] pour la version 2005 de WSMO :

– *WSML2Reasoner* [7] propose une architecture fortement modulaire combinant de nombreuses fonctionnalités de validation, de normalisation, et de transformation comme la traduction d'ontologies en WSML qui est la syntaxe appropriée du moteur d'inférence du framework.

6. http ://www.essi-cluster.org/
7. Disponible à cette adresse : http ://tools.deri.org/wsml2reasoner/

– *WSMO Studio*[8] est un éditeur de services Web sémantiques conforme au framework WSMO. WSMO studio est disponible comme un ensemble de plugins d'Eclipse.
– *WSML Rule Reasoner*[9] est une implémentation d'un raisonneur pour WSML.
– *WSML DL Reasoner* une implémentation d'un raisonneur à base de la Logique de Descriptions pour le langage WSML.
– *WSML Validator* est une variante de validateur de WSML implémenté comme partie de WSMO4J.
– *WSMO4J*[10] est une API et une implémentation de référence pour la construction d'applications de SWS conforme aux spécifications de WSMO dans ses versions WSMO v1.2 et WSML 0.2.
– *Web Service Modeling Toolkit (WSMT)*[11] est une boîte à outils pour un déploiement rapide d'outils graphiques utilisables avec WSMO, WSML et WSMX.
– *Web Services Execution Environment (WSMX)*[12] est un environnement d'exécution pour les tâches d'appariement, de sélection, de médiation et d'invocation de SWS basés sur WSMO.

Enfin, signalons que la dernière version de WSMO qui est la version1.3. dont le draft final date du 21 Octobre 2006 est disponible à l'adresse suivante :

http ://www.wsmo.org/TR/d2/v1.3/

2.3.4 METEOR-S

METEOR-S[13] (METEOR for Semantic Web Services) est une initiative du laboratoire LSDIS[14] de l'Université de Georgia (Etats-Unis d'Amérique) ayant pour objectif de fournir des services Web sémantiques dans le cadre du projet METEOR (acronyme pour *Managing End-To-End OpeRations*). Alors que les frameworks OWL-S et WSMO ont pour objectif de proposer des ontologies de services, METEOR-S adopte une approche différente en proposant d'intégrer les standards des services Web (BPEL4WS, WSDL et UDDI) aux technologies du Web sémantique pour l'annotation, la découverte, la composition, la qualité de service et l'exécution de services Web. C'est ainsi que METEOR-S prend en compte la totalité du cycle de vie des services Web sémantiques comme l'illustre la figure 2.9.

La figure 2.10 montre que METEOR-S s'articule autour de trois étapes : la première étape propose de définir des mécanismes d'annotation sémantique, la deuxième étape propose de se baser sur les annotations sémantique pour offrir une découverte de services flexible et dynamique et une médiation de données, enfin, la troisième étape suggère des techniques d'adaptation afin de gérer les événements au cours de l'exécution d'un processus (i.e., l'invocation d'un service ou d'une composition). En terme d'annotation sémantique, le mérite de METEOR-S est d'avoir proposé une extension de WSDL nommée WSDL-

8. Disponible à cette adresse : http ://www.wsmostudio.org/
9. Disponible à cette adresse : http ://dev1.deri.at/wsml2reasoner/
10. Disponible à cette adresse : http ://wsmo4j.sourceforge.net
11. Téléchargeable à partir de http ://sourceforge.net/
12. Disponible à cette adresse : http ://www.wsmx.org/, http ://sourceforge.net/projects/wsmx/
13. http ://lsdis.cs.uga.edu/projects/meteor-s/
14. LSDIS (acronyme pour Large Scale Distributed Information Systems) est un laboratoire de recherche de l'Université de Georgia (USA) qui a été lancé en 1994 et a été dirigé par Professeur Amit SHETH jusqu'en 2007 avec le soutien du Professeur John MILLER qui en pris la suite.

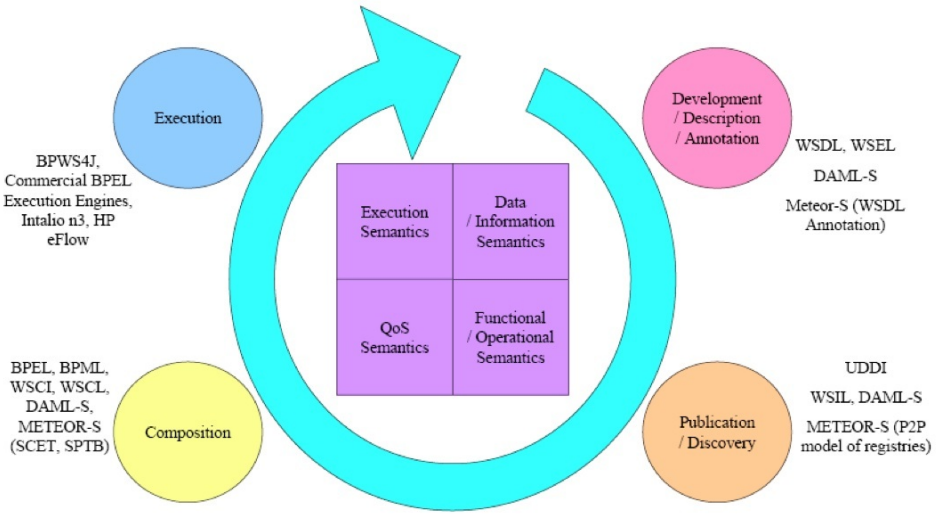


FIGURE 2.9 – Cycle de vie d'un service Web sémantique et les différentes technologies proposées dans le cadre de METEOR-S

S [SVSM03], qui a été plus tard adoptée par le consortium W3C pour être intégrée dans SAWSDL. WSDL-S propose de définir quatre types de sémantiques : (a) la *sémantique des interfaces* décrivant la sémantique des entrées et des sorties du service, (b) la *sémantique fonctionnelle* décrivant la sémantique des opérations et des fonctionnalités offertes par le service, (c) la *sémantique non fonctionnelle* décrivant les propriétés non fonctionnelles (qualité, sécurité) du service et (d) la *sémantique d'exécution* décrivant la sémantique de la gestion des exceptions.

Ainsi, le projet METEOR-S s'est développé selon les trois phases principales qui ont permis d'introduire les trois composants essentiels de cette initiative :

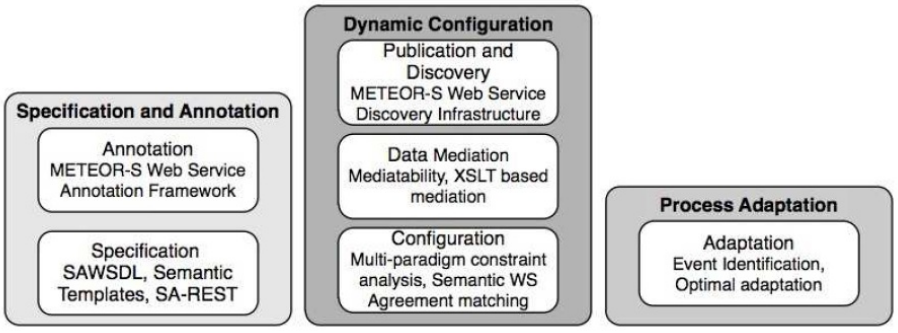


FIGURE 2.10 – Les différents composants de METEOR-S

1. **la mise en place d'une infrastructure de découverte sémantique** définie au dessus d'un registre UDDI, appelée MWSDI (METEOR-S Web Service Discovery Infrastructure) [VSS+05]. MWSDI adopte une approche basée sur les ontologies pour classer les services Web dans les registres en considérant les domaines des services. Enfin, MWSDI adopte une architecture pair-à-pair pour la découverte de services qui dispose de plusieurs registres de services et d'ontologies, de plusieurs points d'entrées et des clients.
2. **l'annotation sémantique** basée sur l'outil MWSAF (METEOR-S Web Service Annotation Framework) [POSV04] permet d'enrichir sémantiquement les services Web en utilisant une WSDL-S (WSDL Semantics) et plus tard celle de SAWSDL. Selon [AFJ+05], le rôle de WSDL-S est d'ajouter un niveau sémantique aux services Web à travers l'enrichissement des fichiers WSDL mais également du registre enrichi UDDI ;
3. **la composition et l'exécution de services** ayant pour rôle de créer des services composites en se basant sur BPEL4WS. L'outil se chargeant de cette tâche se nomme MWSCF (METEOR-S Web Service Composition Framework).

La figure 2.11 illustre l'architecture de METEOR-S avec ses deux modules : le module front-end et le module back-end. Le module front-end est utilisé pour créer des services Web sémantiques en procédant à l'annotation du code source avec des ontologies. La source ainsi enrichie est convertie en descriptions sémantiques en utilisant soit des fichiers WSDL annotés soit des fichiers WSDL-S. Ces derniers sont ensuite publiés dans un registre UDDI enrichi. Le module de back-end offre principalement des fonctionnalités liées à la découverte de services, à la composition de services et à l'exécution et l'orchestration des services composés.

Les nouvelles versions de METEOR-S comme le précise l'article [SGR08] proposent de nouvelles techniques en vue de renforcer les annotations sémantiques dans le framework. Nous citons les outils suivants qui ont été intégrés : SAWSDL4J [15] et Woden4SAWSDL [16], l'annotateur sémantique RADIANT [17]et pour la découverte et la publication l'outil LUMINA [18].

2.3.5 SAWSDL (Semantic Annotation for WSDL)

À l'initiative du groupe de travail d'annotations sémantiques pour WSDL (*Semantic Annotations for WSDL Working Group*) du W3C, SAWSDL (*Semantic Annotations for WSDL and XML Schema*) a été proposé avant d'être adopté comme recommendation par le consortium W3C à partir du mois d'août 2007 [KVBF07] . L'objectif de SAWSDL est d'insérer de la sémantique dans les descriptions WSDL des services et des schémas XML [TBM+04]. Dans la suite, nous n'abordons pas l'enrichissement sémantique des schémas XML, bien que ce concept reste proche de l'enrichissement de fichiers WSDL.

15. http ://lsdis.cs.uga.edu/projects/meteor-s/opensource/sawsdl4j/
16. http ://lsdis.cs.uga.edu/projects/meteor-s/opensource/woden4sawsdl/
17. http ://lsdis.cs.uga.edu/projects/meteor-s/downloads/index.php ?page=1
18. http ://lsdis.cs.uga.edu/projects/meteor-s/downloads/Lumina/

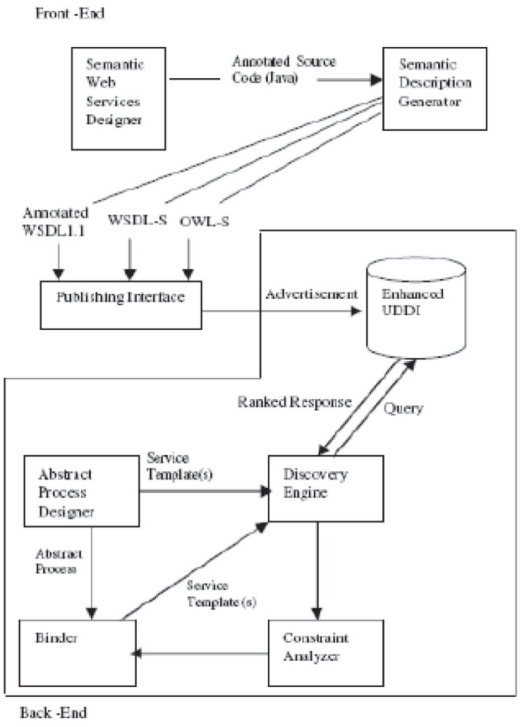


FIGURE 2.11 – L'architecture de METEOR-S

2.3.5.1 Motivations de SAWSDL

En plus de la découverte et l'invocation automatiques des services Web, SAWSDL vise la réalisation des objectifs suivants :

– *Un langage au dessus des standards des services Web existants* : Les standards de services Web sont devenus rapidement une technologie préférée pour l'intégration des applications. Les entreprises investissent dans des projets d'intégration basés sur des services Web. Par conséquent, les concepteurs de SAWSDL considèrent que n'importe quelle approche pour la description sémantique des services Web doit être compatible avec l'existant. C'est dans ce sens que SAWSDL a été conçu comme une extension sémantique de WSDL.

– *Concevoir un langage incrémental* : il est relativement facile de modifier les outils existants autour de WSDL afin qu'ils s'adaptent à SAWSDL. Ce qui fait de SAWSDL une approche incrémentale,

– *Le mécanisme d'annotation doit être indépendant du langage de représentation de la sémantique* : En l'occurrence, OWL-S considère les ontologies définies en OWL, WSMO repose sur des ontologies définies en WSML, chaque langage offrant différents niveaux d'expressivité. Les concepteurs de SAWSDL proposent de créer des annotations sémantiques indépendamment du langage de représentation d'ontologies.

SAWSDL est donc la suite de WSDL-S (*Web Service Description Language – Semantic*) [AFJ+05]. SAWSDL définit un mécanisme d'annotation en vue de spécifier les éléments

de WSDL à l'aide d'ontologie(s). Cette annotation repose sur la définition d'attributs étendant le standard de description WSDL. Les annotations sémantiques référencent des ontologies pré-existantes. L'avantage majeur de SAWSDL est de fournir un mécanisme d'annotation indépendant de tout langage de représentation d'ontologies.

2.3.5.2 Annotations SAWSDL

SAWSDL propose deux types d'annotations sémantiques : un premier pour identifier le concept ontologique, représentée par l'attribut *modelReference* et un second pour établir le lien entre le concept et le document WSDL, représentée par les attributs *liftingSchemaMapping* et *loweringSchemaMapping*.

L'attribut *modelReference* permet d'annoter tous les éléments de WSDL 2.0 (figure 2.12). Cependant, la recommandation de SAWSDL considère que l'annotation des éléments suivants apporte une signification particulière :

– L'élément *interface*. Cet élément décrit l'ensemble des méthodes disponibles d'un service Web. L'annotation de cet élément apporte une définition sémantique de l'ensemble des méthodes proposées.
– L'élément *operation*. Cet élément décrit une méthode particulière. Cette annotation permet d'expliciter les effets des méthodes proposées par le service.
– L'élément *fault*. Cet élément représente un message d'erreur lors de l'appel d'une méthode de l'interface. L'annotation de ce type d'élément donne un sens au message d'erreur.
– L'élément *type*. Cet élément représente les structures de données des paramètres dans des messages échangés. Si cet élément requiert des spécifications exprimées en XML Schema pour définir des types particuliers (simple ou complexe), l'annotation sémantique de SAWSDL peut être aussi utilisée dans cet élément afin d'annoter sémantiquement ces structures de données.

Un attribut *modelReference* a pour valeur zéro, une ou plusieurs URI localisant des ontologies qui apportent de la sémantique aux éléments annotés. Ce type d'annotation est utilisé lors des processus de recherche et de sélection. A titre d'exemple, nous avons choisi d'annoter sémantiquement le service Web *GlobalWeather*. La figure 2.13 donne un extrait de cette ontologie, représentant la classe *WeatherObservation* utilisée dans l'exemple d'annotation SAWSDL décrite dans la figure 2.14. La classe *WeatherObservation* possède quatre propriétés : *hasTemperature* (ligne 7), *hasHumidity* (ligne 8), *hasWindSpeed* (ligne 13) et *hasPrecipitation* (ligne 16). Dans l'illustration de l'ontologie nous remarquons que la propriété *HasWindSpeed* est représentée par le type float (ligne 22) et a pour unité le miles par heure (ligne 21).

Ainsi, la figure 2.14 illustre l'annotation sémantique de la structure de données *GetWeatherResponse* (élément GetWeatherResponse, ligne 8), avec SAWSDL (attribut *sawsdl :modelReference*, ligne 9), par l'intermédiaire de l'itème *WeatherObservation* de l'ontologie weather-ont (cf. ligne 9). Deux autres attributs, nommés *liftingSchemaMapping* et *loweringSchemaMapping*, étendent le fichier WSDL afin de spécifier l'association entre la donnée sémantique (le concept) et l'élément WSDL à annoter et réciproquement. Ces mises en correspondance sont effectuées à l'aide de schémas XML (dont la localisation est

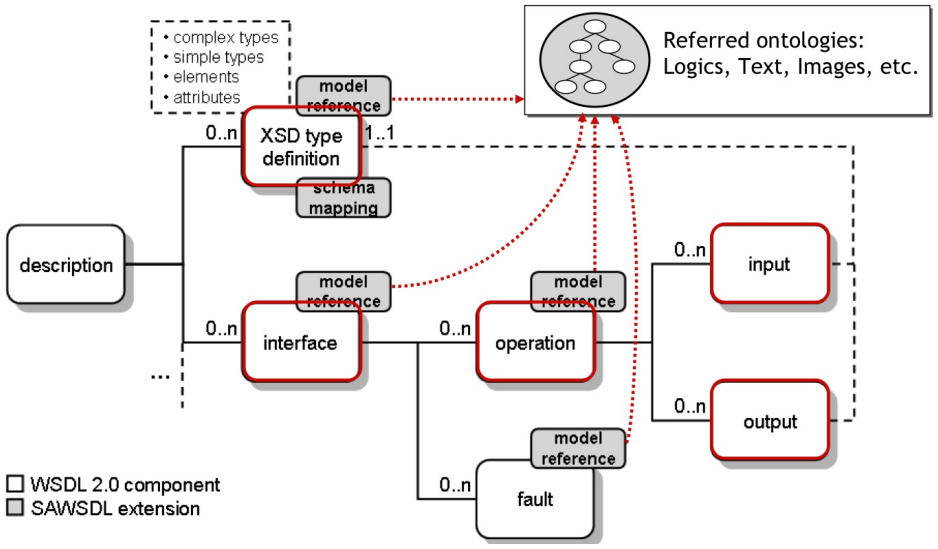


FIGURE 2.12 – Les annotations SAWSDL

```
<rdf:RDF (...)
    xmlns="http://localhost/weather/ont/weather-ont.owl#"
    xml:base="http://localhost/weather/ont/weather-ont.owl">
    <!-- Weather Observation Class -->
   <owl:Class rdf:ID="WeatherObservation"> (...)
     <rdfs:subClassOf>
       <owl:Restriction> <owl:onProperty rdf:resource="#hasTemperature"/> (...) </owl:Restriction>
     </rdfs:subClassOf>
     <rdfs:subClassOf>
       <owl:Restriction> <owl:onProperty rdf:resource="#hasHumidity"/> (...) </owl:Restriction>
     </rdfs:subClassOf>
     <rdfs:subClassOf>
       <owl:Restriction> <owl:onProperty rdf:resource="#hasWindSpeed"/> (...) </owl:Restriction>
     </rdfs:subClassOf>
     <rdfs:subClassOf>
       <owl:Restriction> <owl:onProperty rdf:resource="#hasPrecipitation"/> (...) </owl:Restriction>
     </rdfs:subClassOf>
   </owl:Class>
   <!-- Properties -->
   <owl:ObjectProperty rdf:ID="hasWindSpeed">
     <rdfs:label>Wind speed, mph.</rdfs:label>
     <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
   </owl:ObjectProperty>
</rdf:RDF>
```

FIGURE 2.13 – Extrait de l'ontologie weather-ont, illustrant la classe WeatherObservation.

```
<description ... >
    <rdf:RDF
        xmlns:base="http://refapp.semwebcentral.org/weather/ont/weather-ont.owl">
    </rdf:RDF>
    <types>
      <s:schema ... >
        <s:element name="GetWeather"> ... </s:element>
        <s:element name="GetWeatherResponse"
        sawsdl:modelReference="http://refapp.semwebcentral.org/weather/ont/weather-ont.owl#WeatherObservation">
          <s:complexType>
            <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name='GetWeatherResult" type="s:string"/>
            </s:sequence>
          </s:complexType>
        </s:element>
        <s:element name="GetCitiesByCountry"> ... </s:element>
        <s:element name="GetCitiesByCountryResponse"> ... </s:element>
        <s:element name="string" nillable="true" type="s:string"/>
      </s:schema>
      ...
   </types>
   ...
</description>
```

FIGURE 2.14 – Annotation sémantique de l'élément *type* du service GlobalWeather.

donnée par la valeur de l'attribut). L'attribut *liftingSchemaMapping* associe aux éléments XML un modèle sémantique. Inversement, l'attribut *loweringSchemaMapping* associe au modèle sémantique une structure XML. Ces mises en correspondance sont utiles, lors de l'invocation du service, lorsque la structure de données ne correspond pas de manière intuitive à l'organisation du modèle sémantique.

SAWSDL permet d'ajouter facilement une description sémantique aux descriptions WSDL déjà existantes. L'avantage de cette proposition est qu'elle permet de modifier des descriptions WSDL existantes sans trop alourdir le fichier de description. Cependant, SAWSDL ne permet pas d'annoter tous les éléments XML du document WSDL (par exemple, la description du fournisseur de service ne peut pas être annotée) et ne permet pas d'apporter de la sémantique aux éléments non décrits par WSDL (tels que la qualité de service), ce qui constitue un inconvénient majeur du fait que la couverture des annotations n'est pas totale.

2.3.6 IRS II et IRS III

IRS-II (*Internet Reasoning Service*) est un framework et une infrastcucture de services Web sémantiques, supportée par le projet **AKT**[19] (*Advanced Knowledge Technologies*) dans le cadre d'une collaboration interdisciplinaire de recherche regroupant plusieurs universités européennes dont l'université d'Aberdeen et d'Edinburgh. IRS-II est conçu comme un moyen de résolution de problèmes en ligne réutilisables [MDCG03]. Le but principal d'IRS-II est de supporter la découverte et la récupération de services de bibliothèques distribuées sur Internet et de leur configuration semi-automatique, dans le but de réaliser des tâches spécifiques en fonction des exigences des utilisateurs. IRS-II tente d'apporter

19. http ://www.aktors.org/

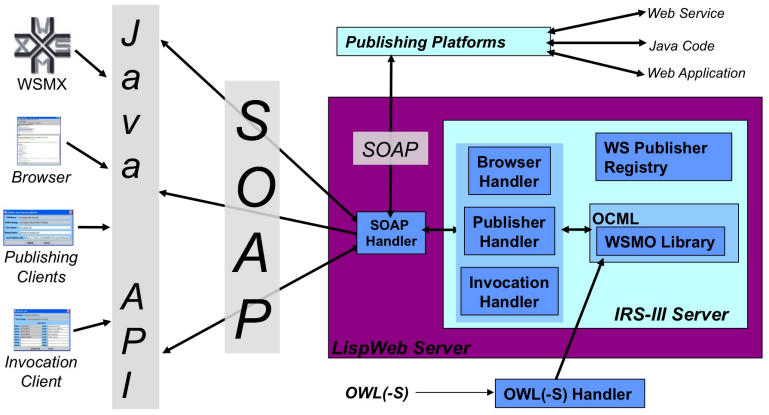


FIGURE 2.15 – Architecture de IRS III

une adaptabilité et une médiation flexibles entre problèmes et services [CMML03]. IRS-II possède deux caractéristiques majeures : Premièrement, il est construit sur un modèle de connaissance, et deuxièmement il est basé sur le langage UPML (Unified Problem-Solving Method-Development Language)[OCF+03] pour l'annotation des services.

La version IRS-III qui a été présentée dans [CDG+06] dispose d'une architecture générique qui est illustrée dans la figure 2.15. IRS III comme extension de l'architecture IRSII adopte le framework WSMO comme modèle de description de services Web sémantiques.

Par l'intermédiaire d'une plate-forme de publication propre à IRS, le fournisseur peut décrire l'ensemble des ontologies nécessaires à la description de son service en utilisant WSMO. Les ontologies sont ensuite associées au service Web déployé par le fournisseur. Lors du processus de publication, ce dernier doit aussi associer au service une méthode de résolution de problème.

La recherche et la sélection dans IRS-III a la particularité de se baser sur les méthodes de résolution de problème. Cette résolution est faite à l'aide du langage UPML (Unified Problem-Solving Method-Development Language) comme pour IRS-II.

UPML distingue quatre notions : les modèles du domaine, les modèles de tâche, les méthodes de résolution de problème et les ponts.

– *Modèles du domaine.* Ces modèles décrivent le domaine d'application (par exemple, celui du tourisme). Ces modèles sont associés à l'élément ontologie du domaine du framework WSMO.
– *Modèles de tâche.* Ces modèles fournissent une description générique de la tâche à résoudre. Ces modèles sont associés à l'élément objectif du framework WSMO.
– *Méthodes de résolution de problème.* Cette notion fournit une description abstraite des processus de raisonnement qui permettent de résoudre des tâches dans des domaines spécifiques. Ces méthodes sont associées à l'élément service Web du framework WSMO.
– *Ponts.* Les ponts spécifient les relations entre les différents modèles d'une application. Cette notion est associée à l'élément médiateur du framework WSMO.

Le cadre de travail IRS-III est intéressant puisqu'il prend en compte de manière trans-

versale quatre processus sous-jacents à l'architecture des services Web (la description, la recherche, la sélection et l'invocation) via l'utilisation d'ontologies WSMO. De notre point de vue, l'inconvénient de ce travail réside dans leur processus de publication. Lors de cette étape, les fournisseurs doivent associer à leurs services une méthode de résolution de problème. Or, nous pensons que les services Web doivent être publiés dans un registre indépendamment d'une résolution particulière afin d'être facilement réutilisés dans le plus grand nombre de cas possibles.

L'outil graphique IRS-III guide l'utilisateur étape par étape pour la composition de services décrits en WSMO (Web Service Modeling Ontology) présenté plus haut. Le framework exploite les connaissances du domaine pour guider l'utilisateur voulant construire un workflow. L'outil fournit à cet utilisateur des conseils pour la sélection et l'instanciation des services et lui permet de mémoriser les workflows en vue de faciliter la réutilisation.

C'est ainsi qu'avec l'architecture IRS-III, Cabral et Dominigue [CDG+06] ont réussi à modéliser les contraintes de communication par un ensemble de règles logiques. Un médiateur de processus s'occupe de résoudre les conflits de communication en utilisant ces règles, tout en se positionnant entre le client et le service Web afin d'intercepter les communications et de les adapter. Ceci repose sur un composant appelé *interpréteur de chorégraphie*, qui permet d'exécuter un service Web en créant les messages requis par ce dernier sur la base de sa description. Il garde en mémoire l'état d'avancement des communications entamées avec le service Web, en mémorisant les appels exécutés par un composant appelé invocateur.

2.4 Synthèse et Discussion

Nous avons choisi de résumer, dans le tableau 2.1, les principales caractéristiques des approches de services sémantiques présentées dans ce chapitre. Les caractéristiques de ces initiatives sont synthétisées sur la base de cinq critères considérés, à notre sens, comme les plus pertinents. Il s'agit, en l'occurrence, de deux critères génériques couramment utilisés pour l'analyse des approches syntactiques auxquels nous avons décidé de rajouter trois critères spécifiques aux approches sémantiques.

Les deux critères génériques retenus sont comme suit :

– Le critère *Point de vue* qui permet de préciser le niveau d'abstraction (conceptuel, technique) associé à l'approche considérée ;

– Le critère *ouverture* qui permet de représenter le degré de standardisation permis par l'approche

Les trois critères supplémentaires spécifiques que nous avons choisi pour l'analyse des approches sémantiques sont :

– le critère *description sémantique* qui permet de préciser la manière avec laquelle les services sont décrits sémantiquement par l'approche étudiée ;

– le critère *médiation de services* qui permet de préciser la prise en compte des mécanismes de médiation par l'approche étudiée ;

– et le critère *maturité* qui désigne en fait le degré de maturité de l'approche vis-à-vis de la problématique d'intégration des applications industrielles.

Critères d'analyse	OWL-S	WSMF	WSMO	METEOR-S	IRS-II	IRS-III
Point de vue (niveau d'abstraction)	conceptuel (basé sur une ontologie générique)	conceptuel (basé sur un cadre conceptuel)	conceptuel (basé sur le cadre WSMF)	technique (basé sur l'enrichissement sémantique de WSDL)	conceptuel (basé sur des ontologies de tâches et PSM)	conceptuel (basé sur le cadre conceptuel WSMO)
Ouverture	très forte (basée sur WSDL et OWL)	faible (basée sur UMPL)	faible (basée sur WSML)	forte (basée sur WSDL, mais WSDL-S n'est pas standard)	faible (basée sur UPML)	faible (basé sur OCML)
Description sémantique	Ontologie générique OWL-S	pas de description mais recommandation d'utilisation d'utilisation de OWL-S et de PSL	Ontologie WSMO	Annotation des fichiers WSDL	Ontologie PSM et Ontologie de tâche	Ontologie WSMO
Médiation de services	non définie	non définie	définie mais en cours de développement	non définie	non définie	définie mais en cours de développement
Maturité	forte	très faible	moyenne	moyenne	moyenne	faible

TABLE 2.1 – Comparatif des frameworks de description sémantique de services Web

L'analyse du tableau 2.1 permet de retenir un certain nombre de points importants qui sont :

– **OWL-S constitue une ontologie générique de services permettant principalement de décrire les services Web dans un cadre général**. Une des caractéristiques majeures de OWL-S est le fait qu'elle soit compatible avec les standards industriels notamment WSDL, et ce grâce au ServiceGrounding qui permet de définir une correspondance entre OWL-S et WSDL. Certains travaux existent quant à la découverte et la composition de services en utilisant OWL-S, même si certains prédisent la disparition de ce framework au profit de SAWSDL.

– **WSMF est un cadre générique pour la réalisation des services sémantiques**. Il constitue donc un framework offrant un cadre méthodologique plutôt qu'une infrastructure opérationnelle.

– **WSMO** constitue une approche basée sur des fondements conceptuels issus de WSMF pour la réalisation des services sémantiques. Il est beaucoup **plus orienté utilisateur** dans le sens ou l'approche permet d'offrir un langage très proche des utilisateurs.

– Mais elle demeure toutefois **immature du fait qu'il s'agit d'une initiative en cours de développement**. De plus, l'une de ses limites les plus importantes est le fait que cette approche ignore les standards industriels existants, ce qui suppose que des mappings doivent être définis afin de garantir l'intégration avec les standards existants au sein des entreprises.

– **METEOR-S** constitue une approche basée sur l'extension de standards industriels. Le principe d'enrichissement sémantique est basé principalement sur l'utilisation des fichiers WSDL-S qui sont le résultat de l'annotation sémantique de fichiers WSDL. **Bien que cette approche repose sur des standards, il n'en demeure pas moins qu'elle manque d'abstraction dans le sens où elle constitue une approche plus proche du niveau d'implémentation que du niveau conceptuel.**

– **IRS-II** est une approche qui permet de réaliser les services sémantiques à travers le framework UPML. Ceci constitue à la fois son point fort et son point faible. De part sa formalisation UPML est considéré comme le point fort de l'approche IRS II. Mais il constitue aussi l'un des griefs les plus cités à l'encontre de cette approche du fait qu'elle est fortement liée à ce framework (UPML) négligeant ainsi d'autres préoccupations importantes liées aux services sémantiques. Il s'agit en particulier des besoins, qui nous intéressent beaucoup dans le cadre de nos travaux, à savoir l'intégration des services.

– **IRS-III** se base sur le cadre conceptuel WSMO offrant un environnement d'exécution pour le déploiement, la sélection, l'invocation, la composition de services Web décrits en WSMO. L'approche IRS-III adopte les médiateurs pour la résolution des requêtes clients et cela de manière similaire que WSMO. Il est difficile d'identifier les discordances entre les deux approches. IRS-III apparaît comme une approche immature vu le manque d'outils et de plate-formes d'exécution. Les auteurs se contentent d'une étude de cas dans le domaine du e-government [CDG+06].

En résumé, il est possible de retenir que OWL-S est une approche générique, basée sur le standards OWL et WSDL et que WSMF et WSMO constitue beaucoup plus un cadre méthodologique. Toutefois, notons que WSMO est basée sur un langage propriétaire combinant plusieurs logiques. Quant à METEORS-S il s'agit d'une approche pragmatique mais qui manque d'abstraction alors que IRS-II est fortement liée au framework UPML.

Il est à remarquer que OWL-S et METEOR-S sont compatibles, en ce sens que ces deux approches peuvent être combinées. Mais, par contre, elles sont plutôt contradictoires avec WSMO et avec IRS-II dans la mesure où ces dernières utilisent des langages et des outils incompatibles avec les standards industriels.

L'état de l'art présenté dans ce chapitre permet de révéler à notre sens trois points essentiels :

– Le premier point porte sur l'inadéquation des architectures actuelles et des ontologies de services à capturer de façon flexible et efficace la sémantique des services Web.

– Le deuxième point porte sur le manque de méthodologie à mettre en oeuvre pour définir des services sémantiques.

– Le troisième point concerne la limite des approches actuelles en matière de description et de médiation de services dans le contexte d'une variété d'utilisateurs. Enfin, le dernier point porte sur la complexité des techniques sémantiques d'où la nécessité, en vue de leur adoption, de mettre en œuvre des outils appropriés permettant de simplifier, autant que faire se peut, toute la complexité inhérente à l'utilisation de ces technologies.

2.5 Conclusion

Nous avons présenté dans ce chapitre un état de l'art des différents frameworks des services Web sémantiques suivi d'une synthèse et d'un comparatif. Notre finalité n'étant pas d'identifier le meilleur framework en matière de sémantisation, même si à travers les critères de comparaison la supériorité d'un ou plus de ces frameworks semble évidente, notre objectif consiste à montrer qu'il existe plusieurs points de vue de sémantisation selon l'approche adoptée. Les approches de description sémantique des services Web suivent un objectif commun : décrire de manière explicite et gérer la sémantique associée aux services Web, car cette sémantique reste absente de leur pile standard de protocoles. Cependant, nous pouvons distinguer deux approches différentes de sémantisation de services Web : (a) des ontologies de service telles que OWL-S et WSMO offrant des primitives conceptuelles riches sémantiquement tentent de remplacer le standard WSDL (Web Service Description Language), (b) des initiatives (essentiellement celle de METEOR-S avec WSDL-S et SAWSDL) exploitant les éléments d'extensibilité des langages existants (notamment WSDL 2.0.) pour insérer des annotations sémantiques tout en restant compatibles avec la pile standard des services.

Par ailleurs, notre ambition est de constituer un référentiel aussi large que possible de services Web destiné à la communauté de la bioinformatique ; comprenant notamment des services générés par l'un des trois principaux frameworks présentés dans cet état de l'art, à savoir : OWL-S, WSMO et SAWSDL. Pour ce faire, nous proposerons une couche médiatrice permettant la description, la découverte et la composition afin qu'elle soit le modèle de sémantisation adopté. Notre proposition BioMed fera l'objet de la partie II de ce mémoire. Dans le chapitre suivant, nous présentons un état de l'art des approches de découverte de services Web.

Chapitre 3

Approches de Découverte de Services Web

Nous présentons dans ce chapitre une revue de la littérature de la problématique de découverte de services Web. Nous distinguons les approches syntactiques qui sont basées sur une découverte par mots-clés dans des annuaires UDDI ou des catalogues de services tels que Seekda ! et des approches sémantiques exploitant des techniques de recherche d'informations (mesures de similiarité), des techniques de raisonnement ou combinant ces deux techniques.

Sommaire

3.1 Introduction

Afin d'exploiter les nombreux avantages des services Web, les organisations ont été amenées à se doter d'outils leur offrant la possibilité de rechercher, de découvrir, de sélectionner voire d'invoquer le service adéquat satisfaisant leurs besoins parmi une collection de services Web. Une fois décrit, un service Web doit être publié dans un annuaire pour être utilisé (réutilisé) soit tel quel dans le cas où il satisfait à lui seul le besoin ou en interaction avec d'autres services pour construire un service composite (à valeur ajoutée). La découverte de services désigne *l'action de localiser la description d'un service*

Web non connue à l'avance et qui satisfait certaines contraintes[1]. Nous identifions deux étapes essentielles dans le processus de découverte : la première étape est celle de la recherche d'un service en précisant les propriétés désirées tant au niveau fonctionnel que non fonctionnel et la seconde étape est celle de la sélection du service considéré le plus satisfaisant parmi un ensemble de services retenus durant la phase de recherche. Longtemps, le processus de découverte avait été basé sur l'utilisation des annuaires UDDI[2] [JC02]. Cependant, ce type de découverte a très rapidement montré ses limites puisqu'il offre une recherche purement syntactique basée sur des mots-clés. La solution proposée dans le cadre des services Web sémantiques est soit d'étendre les standards existants avec des descriptions sémantiques afin d'améliorer le processus de découverte ou d'adopter des techniques développées en Recherche d'Information (RI) ou en Intelligence Articifielle (IA). L'état de l'art de la découverte de services présenté dans ce mémoire ne se propose pas d'être exhaustif mais plutôt synthétique des principaux travaux de recherche développés autour de la problématique de découverte de services Web tout en distinguant les approches syntactiques pour la découverte de services Web des approches sémantiques.

3.2 Approches syntactiques pour la découverte de services Web

S'inspirant du modèle classique de recherche d'informations, les approches syntactiques de découverte de services Web permettent d'interroger un catalogue de services Web par le biais de requêtes constituées de mots-clés. Comme approches syntactiques de découverte, nous présentons dans ce qui suit les annuaires UDDI et deux exemples de portails de services Web accessibles via le Web à savoir QWSdataset[3] développé à l'université de Guelph (Ontario, Canada) depuis 2007 et Seekda ! développé par la société Seekda GmbH en 2007 en Allemagne. Le premier a été développé dans le cadre d'un projet académique alors que le second a été développé dans le domaine industriel. Chacun de ces portails propose une interface de recherche basée sur des mots-clés.

3.2.1 Les annuaires UDDI

Comme le montre la figure 3.1, UDDI est proposé comme le standard de publication et de découverte de services Web. UDDI [udd01] (Universal Description Discovery and Integration) propose une technologie d'annuaire basée sur XML permettant de manipuler différents types d'informations concernant un service Web, à savoir : son fournisseur (*Business Entity*), le service lui-même (*Business Service*), les accès au service (*Binding Template*), le type de service (*tModel*) et des relations entre deux parties (*Publisher Assertion*) (figure 3.2).

– ***L'entité fournisseur (Business Entity)*** comprenant les informations concernant l'organisation hébergeant le service, le fournisseur et d'autres informations nécessaires à l'identification de l'entreprise qui sont répertoriées dans ce document XML. Ce document

1. *D'après, Web Service Glossary http ://www.w3.org/TR/ws-gloss/*
2. Universal Description Discovery and Integration
3. http ://www.uoguelph.ca/ qmahmoud/qws/index.html

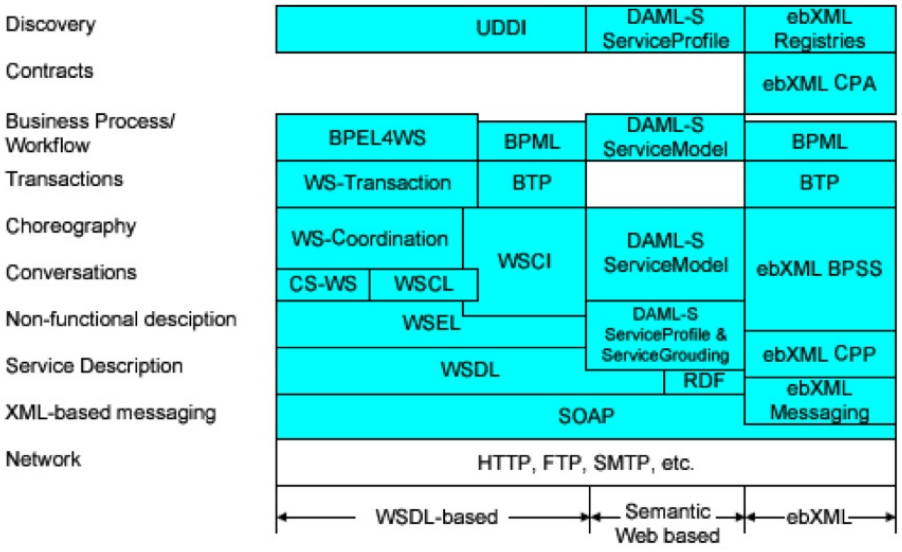


FIGURE 3.1 – Survol des langages standards de services Web

contient de l'information descriptive sur l'entreprise ou le fournisseur et sur les services proposés (lien vers l'entité Business Service dans la figure 3.2).

– ***Le service (Business Service)*** représente les services proposés par une organisation. La description des services contenue dans l'entité Business Service est de haut niveau, i.e., aucune information technique n'est décrite ici. Les informations à propos du nom du service et de son objectif sont représentées dans ce composant. Le fournisseur peut rassembler dans cette entité un ensemble de services répondant aux mêmes objectifs dans une même catégorie. Par exemple, une catégorie tourisme peut contenir un service de météo et un service pour la localisation des sites touristiques. Les catégories de service (contenant un ou plusieurs services) sont liées (selon le nombre de services) à un ou plusieurs points d'accès (*Binding Template* dans la figure 3.2).

– ***Les accès au service (Binding Template)*** décrivent les points d'accès aux services Web (URL) et le moyen d'y accéder (les différents protocoles à utiliser) afin d'invoquer les services.

– ***Le type de service (tModel)*** permet d'associer un service à sa description décrite en WSDL (Web Service Description Language). Le client peut ainsi avoir connaissance des conventions d'utilisation du service. La liaison entre les entités *Binding Template* et *tModel* est nécessaire pour l'invocation du service (figure 3.2).

La recherche et la sélection dans l'annuaire UDDI reposent sur la publication préalablement décrite du service et de son fournisseur. Le futur client peut connaître par l'intermédiaire de UDDI : les fournisseurs d'un service, les services proposés par un fournisseur donné, les moyens d'invoquer un service. Pour apporter aux clients la réponse à ces questions, UDDI organise l'ensemble des informations qu'il contient en trois catégories, spécifiées en XML. Chacune d'elles peut être utilisée pour effectuer une recherche via

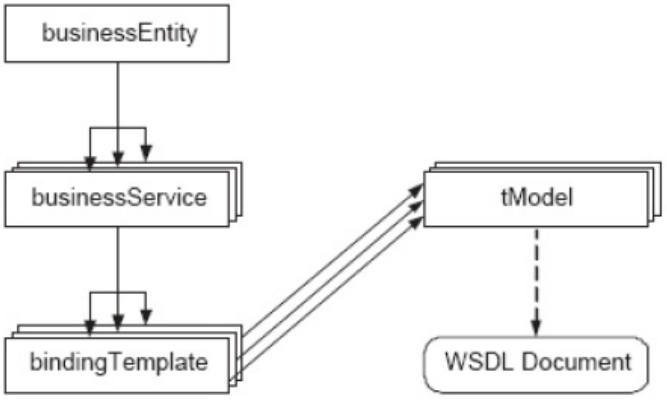


FIGURE 3.2 – Les entités composant un annuaire UDDI

UDDI. Ces parties sont les suivantes :

– *Les pages blanches (White Paper).* Ce composant permet de connaître les informations à propos de l'organisation proposant le service. Cette description contient toutes les informations jugées pertinentes pour identifier l'organisation (telles que son nom, son adresse physique). Le futur client du service retrouve dans les pages blanches les informations que le fournisseur a renseigné dans l'élément *Business Entity* lors de la publication.
– *Les pages jaunes (Yellow Paper).* Les pages jaunes de UDDI détaillent la description de l'organisation contenue dans les pages blanches en répertoriant les services proposés. Dans cette partie, sont décrits : la catégorie de l'entreprise, le secteur d'activité dans lequel exerce l'entreprise, les services offerts par cette organisation, le type de services et les conventions d'utilisation – prix, qualité de service, etc. Cette description repose sur les classifications standard de l'industrie nord américaine (NAICS [4]et UNSPSC [5]). La description des services contenue dans les pages jaunes est non technique et est renseignée par les fournisseurs eux mêmes.
– *Les pages vertes (Green Paper).* Les pages vertes comportent les informations techniques liées aux services Web et basées sur leur description WSDL.

À l'origine, il existait des annuaires UDDI dits publics (tels que ceux de Microsoft ou d'IBM) pour lesquels n'importe qui pouvait devenir, soit fournisseur, soit client de services Web. L'universalité de ces annuaires avait amené UDDI à devenir le standard de publication des services Web. En 2006, le nombre de services Web publiés avait atteint le nombre de 50000. Malgré ce nombre, UDDI n'a jamais atteint le but visé : devenir le registre standard des services Web. Par conséquent, la maintenance des annuaires publics UDDI (tels que ceux de Microsoft et d'IBM) a été suspendue. Le réel succès de UDDI se

4. North American Industry Classification System, http ://www.census.gov/eos/www/naics/
5. United Nations Standards Products and Services, http ://www.unspsc.org/

situe au niveau des annuaires privés. En effet, de nombreuses organisations utilisent les spécifications de UDDI afin d'implémenter leur propre annuaire de services Web.

Cependant, les spécifications UDDI souffrent de plusieurs limitations :

1. Les APIs de publication de UDDI sont insuffisantes pour développer efficacement des méthodes de publication et de recherche ; certes ces API requièrent une grande expertise de la part du concepteur du service.
2. le modèle de recherche de UDDI est pauvre et limité se contentant d'une recherche par mots-clés portant sur l'identifiant de la catégorie du service, son nom ou certains éléments de son document WSDL. Ainsi, UDDI en offrant une recherche purement syntaxique basée sur mots-clés ne garantit pas des résultats pertinents.
3. UDDI ne supporte aucun mécanisme d'inférence, de ce fait, la recherche repose sur l'existence d'un matching exact entre la requête de l'utilisateur et les catégories des services. Un service associé à la catégorie "concessionnaires de nouvelles voitures" ne sera jamais sélectionné pour une recherche sur les "concessionnaires d'automobile", bien que la première catégorie n'est qu'une sous catégorie de la dernière.

Certains travaux proposent des extensions de UDDI pour y intégrer des descriptions sémantiques et ainsi améliorer le processus de découverte. Nous présenterons ces travaux dans la section 3.3.2.

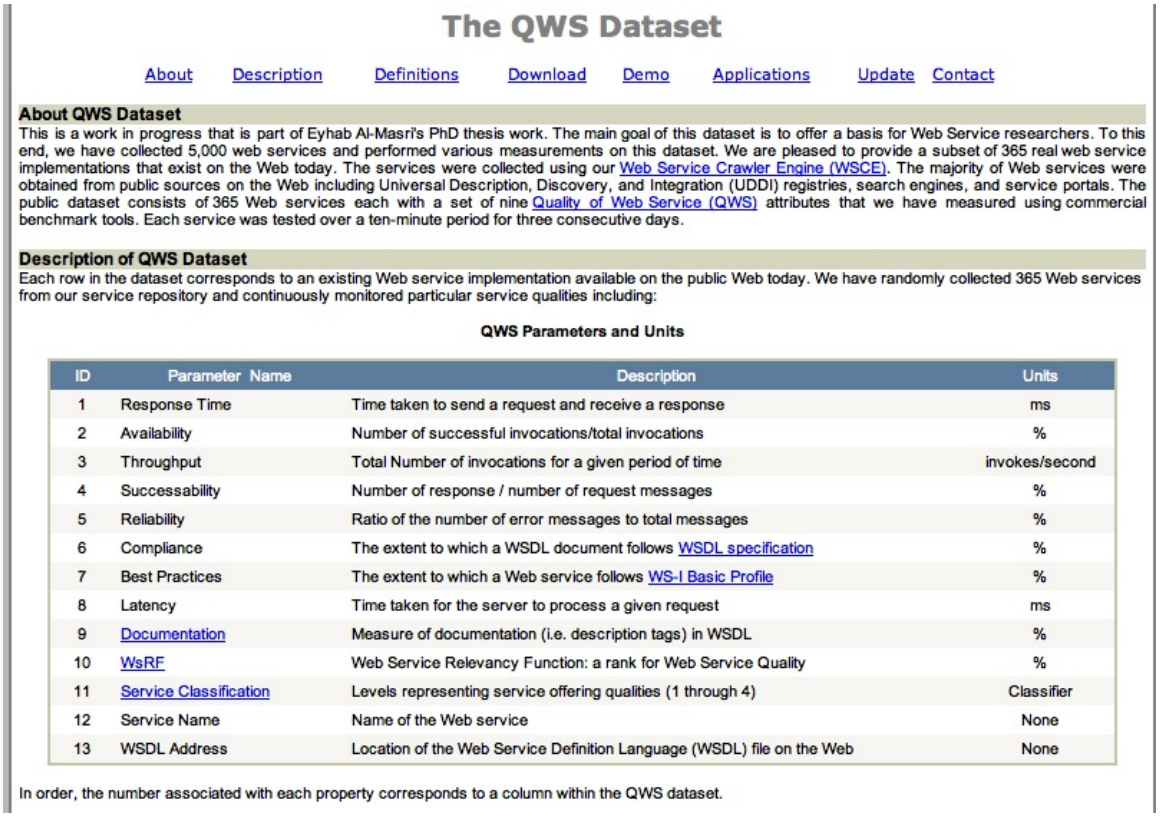
3.2.2 QWS dataset

L'objectif du projet QWS Dataset [AMM07a, AMM07b] est d'offrir la possibilité aux chercheurs dans le domaine des services Web de trouver des services Web élémentaires en vue d'implémenter un système à base d'Architecture Orientée Service. QWS Dataset contient 365 services Web [6] qui ont été récupérés grâce à un moteur d'aspiration de services Web (*Web Service Crawler Engine*) [AMM07a] à partir de différentes sources telles que des registres UDDI, des moteurs de recherche ou des portails spécifiques aux services Web. Les services Web aspirés sont ensuite centralisés dans le QWS Dataset (figure 3.3). La recherche de services dans QWS Dataset consiste en une simple recherche par mots-clés [AMM07b]. La figure 3.4 illustre le résultat d'une recherche par mots-clés en utilisant le terme "*weather*".

L'ensemble des services Web disponibles dans QWS Dataset a été testé pendant une période de dix minutes durant trois jours consécutifs, selon neuf attributs de qualité de service. Ces tests ont permis d'attribuer un ensemble de caractéristiques portant sur la qualité de service [AMM07b]. Parmi cet ensemble de caractéristiques, six sont présentées à la suite d'une recherche pour discriminer les services Web retrouvés : le temps de réponse, le débit, la fiabilité, les meilleures pratiques, la documentation et le niveau de classification (comme illustré dans la figure 3.4).

– Le temps de réponse (*Response Time*). Le temps de réponse est le temps mis entre l'envoi de la requête par le client et la réception du résultat envoyé par le service Web. L'unité de mesure du temps de réponse est la milliseconde (ms).

6. Date du dernier accès 26 Mars 2010



The QWS Dataset

About Description Definitions Download Demo Applications Update Contact

About QWS Dataset

This is a work in progress that is part of Eyhab Al-Masri's PhD thesis work. The main goal of this dataset is to offer a basis for Web Service researchers. To this end, we have collected 5,000 web services and performed various measurements on this dataset. We are pleased to provide a subset of 365 real web service implementations that exist on the Web today. The services were collected using our Web Service Crawler Engine (WSCE). The majority of Web services were obtained from public sources on the Web including Universal Description, Discovery, and Integration (UDDI) registries, search engines, and service portals. The public dataset consists of 365 Web services each with a set of nine Quality of Web Service (QWS) attributes that we have measured using commercial benchmark tools. Each service was tested over a ten-minute period for three consecutive days.

Description of QWS Dataset

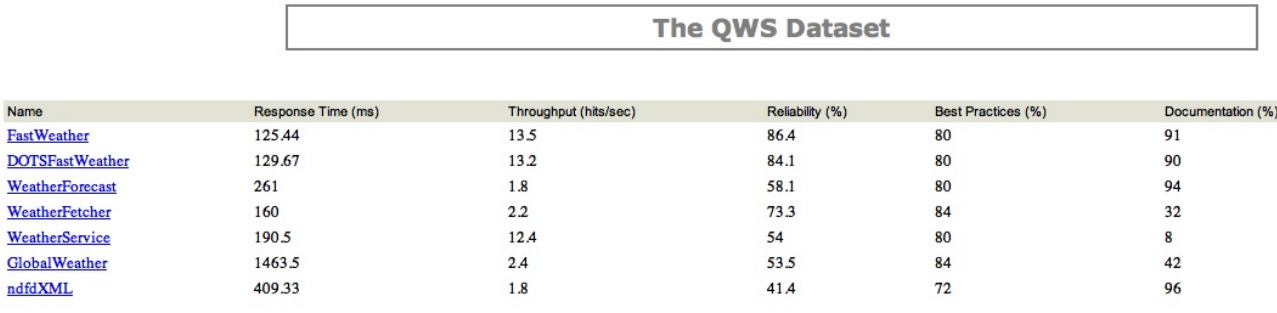
Each row in the dataset corresponds to an existing Web service implementation available on the public Web today. We have randomly collected 365 Web services from our service repository and continuously monitored particular service qualities including:

QWS Parameters and Units

ID	Parameter Name	Description	Units
1	Response Time	Time taken to send a request and receive a response	ms
2	Availability	Number of successful invocations/total invocations	%
3	Throughput	Total Number of invocations for a given period of time	invokes/second
4	Successability	Number of response / number of request messages	%
5	Reliability	Ratio of the number of error messages to total messages	%
6	Compliance	The extent to which a WSDL document follows WSDL specification	%
7	Best Practices	The extent to which a Web service follows WS-I Basic Profile	%
8	Latency	Time taken for the server to process a given request	ms
9	Documentation	Measure of documentation (i.e. description tags) in WSDL	%
10	WsRF	Web Service Relevancy Function: a rank for Web Service Quality	%
11	Service Classification	Levels representing service offering qualities (1 through 4)	Classifier
12	Service Name	Name of the Web service	None
13	WSDL Address	Location of the Web Service Definition Language (WSDL) file on the Web	None

In order, the number associated with each property corresponds to a column within the QWS dataset.

FIGURE 3.3 – L'inteface d'accueil du Projet QWSdataset



The QWS Dataset

Name	Response Time (ms)	Throughput (hits/sec)	Reliability (%)	Best Practices (%)	Documentation (%)
FastWeather	125.44	13.5	86.4	80	91
DOTSFastWeather	129.67	13.2	84.1	80	90
WeatherForecast	261	1.8	58.1	80	94
WeatherFetcher	160	2.2	73.3	84	32
WeatherService	190.5	12.4	54	80	8
GlobalWeather	1463.5	2.4	53.5	84	42
ndfdXML	409.33	1.8	41.4	72	96

FIGURE 3.4 – Exemple de recherche dans QWSdataset

– Le débit (*Throughput*). Le débit est mesuré par le nombre total d'invocations possibles d'un service Web dans un laps de temps donné. L'unité de mesure du débit est le nombre d'appels réussis par seconde.
– La fiabilité (*Reliability*). La fiabilité est le rapport du nombre de messages d'erreur sur le nombre total de messages. Ce critère est mesuré en pourcentage.
– Les meilleures pratiques (*Best Practices*). Le critère de meilleures pratiques évalue dans quelle mesure la description du service Web suit le WS-I BP (WS-I Basic Profile). Le WS-I BP [7] est une spécification proposée par le consortium d'industriels WS Interoperability [8] (WS-I). L'objectif de WS-I BP est de fournir un guide d'interopérabilité pour le noyau des spécifications des services Web (telles que SOAP, WSDL et UDDI), basé sur le format XML. WS-I BP est principalement utilisé dans le monde de l'industrie. Le critère des meilleurs pratiques est mesuré en pourcentage.
– La documentation (*Documentation*). La propriété QWS de documentation mesure pour le service Web sa capacité auto-descriptive selon l'examen du document WSDL. Cet examen consiste à vérifier s'il existe un contenu dans les balises décrivant le nom du service, les opérations du service, les messages échangés avec le client. L'unité de mesure de la documentation est un pourcentage.
– Le niveau de classification (*class*). Le niveau de classification du service décrit la qualité offerte par le service (de la plus basse à la plus haute qualité représentée par zéro à quatre étoiles). Cette classification est basée sur l'ensemble des niveaux de qualité fourni par la méthode WsRF (*Web service Relevancy Function*) [AMM07b] réalisée pour le projet.

L'avantage de ce travail est que tous les services Web proposés sont des services fonctionnels, soumis à un test rigoureux, ce qui permet d'éviter de sélectionner des services qui ne sont plus disponibles et/ou ne satisfaisant pas les contraintes minimales de qualité. De plus, le système d'aspiration de services Web (*Web Service Crawling Engine*, WSCE), permettant de retrouver automatiquement des fichiers de type WSDL sur le Web, est très intéressant. En effet, il permet de rendre visible les services Web disponibles sans que les fournisseurs n'activent le processus de publication. Cependant, Le QWSdataset présente à notre avis trois inconvénients majeurs : (1) les critères de description sur lesquels repose le processus de découverte sont exclusivement liés à la qualité de service, il manque une description (ne serait ce qu'informelle) de ce que propose le service comme tâche, (2) les fournisseurs n'ont pas le moyen de publier leurs services et (3) enfin, la recherche se base uniquement les mots-clés spécifiées par l'utilisateur.

3.2.3 Seekda !

Seekda [9] est un portail Web offrant un accès à un catalogue de services comprenant 28534 services Web décrits en WSDL [10]. le portail offre la possibilité aux fournisseurs de publier leurs services, il compte aujourd'hui près de 7690 fournisseurs de services Web.

7. http ://www.ws-i.org/Profiles/BasicProfile-2_0(WGD).html
8. http ://www.ws-i.org/
9. http ://Webservices.seekda.com/
10. Dernier accès le 12 juin 2010.

FIGURE 3.5 – L'inteface d'accueil du portail Seekda

Seekda propose deux types de recherche : une recherche par mots-clés et une recherche par navigation :

1. la *recherche par mots-clés* qui peut être basique consistant en une recherche simple ou avancée en considérant d'autres critères de recherche tels que le pays où le service est hébergé, le fournisseur du service, ou par tag [11],

2. *une recherche par navigation* basée sur la catégorisation des services Web par pays (hébergeant le service), mais aussi par l'ensemble des services les plus utilisés ou les plus récemment utilisés. La navigation représente un aspect avantageux de Seekda

Seekda offre également aux utilisateurs la possibilité de tester si un service est opérationnel ou non grâce à l'outil *Web service Tester*. Cependant, ce moteur de recherche présente les mêmes inconvénients de la recherche par mots-clés, d'autant plus que, de point de vue qualité de services, le seul critère QoS considéré est celui de la disponibilité du service. Nous présentons dans ce qui un aperçu des différentes approches développées pour la découverte de services. Nous les classons en trois grandes catégories : celles basées sur les techniques de Recherche d'informations (RI), celles basées sur le raisonnement logique et celles combinant le raisonnement logique et les techniques de RI.

L'avènement des services Web sémantiques a révolutionné les approches de découverte de services qui doivent intégrer la dimension sémantique des services pour guider le processus de découverte et améliorer la qualité des résultats de ce processus.

11. Un tag est un attribut fournissant une présentation informelle d'un service Web reflétant par exemple le domaine ou l'usage du service telle que finance, spelling and grammar, bioinformatic, etc.

3.3 Vers des approches sémantiques de découverte de services Web

Certaines approches sémantiques de services Web proposent d'étendre les standards existants pour la découverte de services Web, à savoir UDDI, en insérant une couche sémantique. D'autres approches adoptent des techniques de Recherche d'Informations (RI) pour calculer des similitudes entre descriptions de services Web, des techniques de raisonnement exploitant les descriptions sémantiques ou des approches hybrides combinant des techniques de RI et de raisonnement.

3.3.1 Approches basées sur les techniques de RI

Les premiers travaux sur la découverte ont utilisé des techniques de RI pour résoudre le problème de découverte de services Web.

Dong et al. proposent dans [ADH+04] un moteur de recherche de services Web nommé Woogle [12], qui permet une recherche simple par mots-clés et se base sur des mesures de similarité pour calculer la similitude entre des services Web en créant des clusters d'opérations (actions élémentaires d'un service Web) fonctionnellement similaires en tenant de la similarité entre les noms de leurs paramètres et de leurs descriptions textuelles.

Pour cela, les auteurs proposent un algorithme dénommé *agglomerative clustering* (clustérisation agglomérative) basé sur une approche ascendante de *classification hiérarchique agglomérative* pour déduire des clusters comprenant des entrées et des sorties d'opérations en se basant sur des règles d'association du type :

$$t_1 \rightarrow t_2 \ (s, c)$$

où s est le *support* mesurant la probabilité que le terme t_1 apparaît dans une entrée ou sortie d'une opération et c est la confiance mesurant la probabilité que les termes t_1 et t_2 apparaissent dans la même entrée ou sortie d'une opération. Un terme t_1 est fortement associé à un terme t_2 si la confiance de la règle d'association $t_1 \rightarrow t_2$ est plus grande qu'un seuil t_c choisi manuellement comme étant la meilleure valeur permettant d'évaluer la corrélation entre deux termes. Les règles d'association permettent de déduire des clusters de termes considérés comme fortement associés. L'une des heuristiques sur laquelle repose l'algorithme proposé est d'augmenter la cohésion au sein d'un cluster de termes tout en minimisant la corrélation entre les termes de deux clusters différents. L'algorithme propose de fusionner deux clusters s'ils respectent une condition de cohésion selon laquelle tous les termes du nouveau cluster (résultant d'une fusion) sont fortement liés au *terme noyau* [13]. D'un autre côté, l'algorithme permet de diviser un cluster en deux clusters si cette division permet d'obtenir deux clusters avec une meilleure cohésion/corrélation. Une opération d'un service Web est représentée sous la forme d'un vecteur op=(w, f, i, o), où w est la description textuelle du service auquel appartient l'opération, f est la description textuelle

12. http ://db.cs.washington.edu/WebService/

13. Un terme noyau est un terme fortement lié au moins à la moitié des termes du cluster auquel ils appartiennent.

d'une opération, i et o désignent les entrées et les sorties de l'opération. L'approche se propose de déterminer les opérations similaires en combinant les similarités des différents composants du vecteur. La mesure de similarité utilisée est la mesure classique de la RI TF/IDF (Term Frequency/Inverse Document Frequency) [Sal88]. Les expérimentations menées ont été effectuées avec une collection de services comprenant 1500 opérations de services Web. Etant donné un service Web, le prototype génère cinq listes différentes : une liste des opérations similaires (aux opérations du service donné), une liste des opérations avec des entrées similaires, une liste des opérations avec des sorties similaires et une liste des opérations composables. Les auteurs évaluent la précision des top-2, top-5 et top-10 réponses de chaque liste. Les valeurs de la précision sont respectivement 98 % (top-2), 83% (top-5) et 68 % (top-10). Ainsi, les expérimentations montrent que la technique de clustérisation adoptée est capable d'améliorer la précision du processus de découverte.

Bernstein et Kiefer de l'université de Zurich proposent dans [BK06] la solution *i*Matcher [14] permettant d'interroger des services Web décrits en OWL-S service profile et sérialisés en graphes RDF par le biais d'une extension du langage de requêtes RDQL, nommé iRDQL (*imprecise RDQL*). iRDQL est une extension du langage RDQL intégrant des mesures de similarité dans la structure de la requête afin de sélectionner non seulement les réponses exactes mais aussi des réponse rapprochées ou similaires. Un aperçu de la syntaxe iRDQL est illustré dans la figure 3.6. Les mesures retenues par les auteurs sont TF/IDF et la métrique *Levenshtein* [15]. Dans *i*Matcher, les services sélectionnés seront classés en fonction de la valeur de la mesure de similarité utilisée dans la requête d'interrogation et le seuil spécifié par l'utilisateur. Cependant, il est à noter que l'inconvénient de l'extension iRDQL est qu'elle permet aux utilisateurs de considérer une seule mesure de similarité par requête. Les expérimentations ont considéré la collection de tests OWLS Service Retreival Test Collection [16]. Les résultats rapportés dans [BK06] montrent que l'approche proposée est capable d'améliorer le rappel tout en ayant une bonne valeur de précision. Les auteurs ont comparé les résultats de l'approche proposée avec celle de OWLS-MX (qui est une approche hybride de découverte de services combinant raisonnement logique et mesures de similarité), cette comparaison montre que OWLS-MX est meilleur que iMatcher en termes des mesures classiques de recherche d'information, à savoir la précision, le rappel et la F-mesure.

3.3.2 Approches basées sur les techniques de raisonnement

Les approches utilisant les techniques de raisonnement se basent uniquement sur des inférences logiques pour déduire un appariement possible entre une requête de découverte et une description de service Web. Les descriptions sémantiques de services Web sont traduites sous la forme d'une expression formelle exprimée dans un langage logique du premier ordre, très souvent une logique de description [BLM+05] telles que OWL-DL ou WSML-DL. Le problème de découverte est réduit à un service d'inférence classique en

14. OWL-S Matcher Project Page *http ://www.ifi.uzh.ch/ddis/research/semWeb/imatcher/*

15. Définition et algorithme d'implémentation de la distance Levenshtein sont présentés à http ://fr.wikipedia.org/wiki/Distance_de_Levenshtein

16. http ://projects.semWebcentral.org/projects/owls-tc/

```
SELECT      [selectClauseVariables]
FROM        [fromClause]
WHERE       [whereClause]
AND         [filterClause]
USING       [usingClause]

IMPRECISE   [impreciseClauseVariables]
SIMMEASURE  [similarityMeasureClause]
OPTIONS     [similarityMeasureOptionsClause]
```

FIGURE 3.6 – La syntaxe d'une requête iRDQL

Logique de description tels que la satisfiabilité (la cohérence) d'un service ou la déduction de relations subsomption entre concepts par rapport à une base de connaissances donnée (une *TBox* au sens la logique de description). Quatre degrés d'appariement sont déduits par la plupart des approches basées sur le raisonnement et qui sont les suivants : *exact*, *plug-in*, *subsumes* et *échec* (fail). Nous illustrons les quatre types d'appariement dans l'exemple 1 et nous les définissons formellement ensuite.

Exemple 1 *Pour illustrer les différentes degrés d'appariement, nous considérons trois exemples de requêtes et les services candidats. Nous reposons sur une ontologie illustrée dans la figure 3.7. L'ontologie est décrite en logique de description*[17]*. Le premier cas est l'exemple d'un appariement en plugin étant que* Patient *(entrée du service) est un sous-concept de* Person *(entrée de la requête) et que* EmergencyPhysician *(sortie du service) est un sous-concept de* Physician*. Il est possible de déduire pour le cas A que* $S \sqsubseteq R$ *qui signifie que l'appariement est en plugin. Le cas B illustre un degré d'appariement en subsomption puisque les deux sorties du service S* Physician *et* Expertise *sont respectivement des super-concepts des sorties de la requête, i.e.,* HospitalPhysician *et* MedicalExpertise*. Enfin, le cas C illustre le cas d'un échec puisque les deux concepts* HospitalPhysician *et* EmergencyPhysician *sont deux concepts disjoints.*

Un matching en plugin entre une description de service S et une requête de découverte R se base sur la notion de subsomption[18] entre les concepts de S et ceux de R, étant

17. Les logiques de description aussi appelées logiques descriptives (LDs) sont une famille de langages de représentation de connaissances qui peuvent être utilisées pour représenter la connaissance terminologique d'un domaine d'application d'une manière formelle et structurée. Une base de connaissances en logique de description est un couple Σ=<Tbox, Abox>, où Tbox est l'ensemble de connaissances terminologiques et ABox est l'ensemble de connaissances assertionnelles (instances). Ce que nous illustrons dans la figure 3.7 est la composante terminologique TBox qui comprend la déclaration d'un ensemble de primitives non logiques explicitant les connaissances et les contraintes de l'univers de discours. Les opérateurs LD permettent d'exprimer l'inclusion entre concepts, e.g., MedicalExpertise $\sqsubseteq$ Expertise, d'exprimer des rôles, e.g., Physician $\equiv$ Person $\sqcap \exists$ TrainedAt.MedicalExpertise, d'exprimer des relations inverses, e.g., Works $\equiv$ $Employed^{-}$, ou aussi d'exprimer des restrictions de cardinalité, e.g., Patient $\equiv$ Person $\sqcap$=1HospitalizedAt.Hospital.

18. Soit C et D deux concepts, D subsumes C, noté $C \sqsubseteq D$, par rapport à une TBox si et seulement si $C^{\mathcal{I}} \sqsubseteq C^{\mathcal{I}}$ pour tout modèle $\mathcal{I}$ de la *TBox*

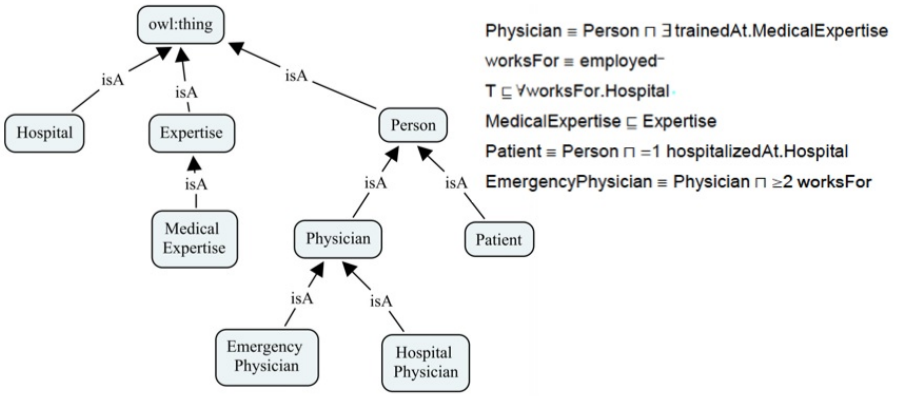


FIGURE 3.7 – Exemple d'une ontologie du domaine biomédical

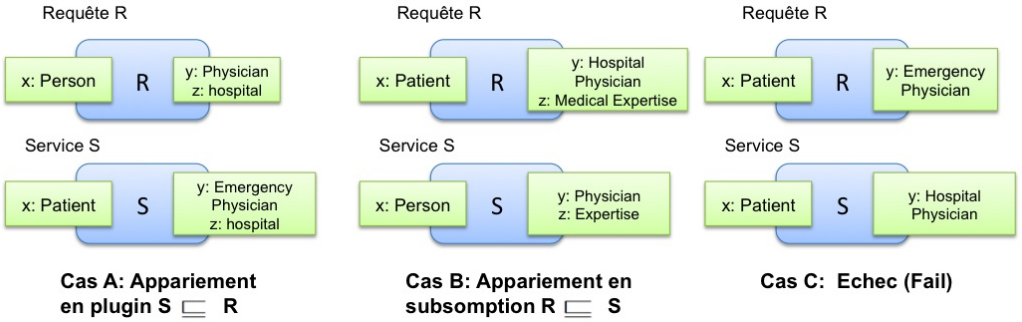


FIGURE 3.8 – Exemples illustrant les différents degrés d'appariement dans une approche de découverte basée sur la logique

donné une base de connaissances kb. Ce matching s'exprime formellement

$$kb \cup S \cup R \models \mathrm{S} \sqsubseteq \mathrm{R}$$

signifiant que pour chaque interprétation $\mathcal{I}$ (dans un monde possible) de la base de connaissances kb, l'ensemble des instances de services $S^{\mathcal{I}}$ sont contenues dans l'ensemble $R^{\mathcal{I}}$ des instances de services acceptables par la requête : $S^{\mathcal{I}} \sqsubseteq R^{\mathcal{I}}$. En d'autres termes, la description du service S est plus spécifique que la requête R, au mieux sémantiquement équivalent.

Un matching en subsomption entre requête de découverte R et une description d'un service S est noté formellement

$$kb \cup S \cup R \models R \sqsubseteq S$$

impliquant que les instances des services $R^{\mathcal{I}}$ sont contenues dans les instances des services $R^{\mathcal{I}}$: $R^{\mathcal{I}} \sqsubseteq S^{\mathcal{I}}$. En d'autres termes, la description du service S est plus générale que celle de la requête R.

Grimm dans [Gri07] propose de rajouter un autre degré de matching dit *matching partiel* défini comme suit : une description de service S matche partiellement une requête R par rapport à une base de connaissance kb si et seulement si l'intersection entre les concepts de S et R est non vide OU la non disjonction entre S et R est vérifiée. Dans le premier cas, la description d'un service S matche partiellement une requête R si la conjonction $S \cap R$ est non vide par rapport à une base de connaissances kb **dans un monde possible** $\mathcal{I}$, tel que $S^I \cap R^I \neq \varnothing$ est vérifié. Dans le second cas, il s'agit de vérifier que l'intersection des extensions des deux concepts S et R est non vide **pour chaque monde possible** $\mathcal{I}$.

Comme critères de matching, certaines approches considèrent uniquement les entrées et les sorties d'un service (*IO matching*), d'autres se basent uniquement la description du service en terme de préconditions et effets (*PE matching*) et il y a aussi des approches qui se basent sur la signature fonctionnelle complète en termes d'entrées, sorties, préconditions et effets (*IOPE matching*). La définition explicite des différents degrés de matching dans le cadre d'une approche considérant uniquement les entrées et les sorties est présentée dans la Table 3.1.

Paolucci et al. [PKP+02a, PKP+02b] furent les premiers à présenter un travail pour la découverte de services Web sémantiques décrits en DAML-S. L'architecture proposée repose sur un raisonneur DAML+OIL, un ensemble d'ontologies DAML et un registre UDDI pour stocker les descriptions DAML-S des services (un *wrapper* permet de traduire DAML-S en une spécification enrichie UDDI).

De manière similaire, **Srinivasan et al.** [SPS04] proposent un algorithme efficace pour le matching de la signature fonctionnelle (en terme d'entrées et de sorties) de services Web décrits en OWL-S et stockés dans des annuaires UDDI. Les auteurs présentent

Degré de matching	*Définition*
exact(S,R)	Un mapping exact entre S et R traduit un ensemble de mappings un-à-un entre les entrées de S et de R d'une part et les sorties de S et de R d'autre part : $\forall\ IN_S\ \exists\ IN_R : IN_S \equiv IN_R \wedge \forall\ OUT_R\ \exists\ OUT_S : OUT_R \equiv OUT_S$.
plug-in(S,R)	Un mapping plugin entre S et R permet de relaxer le mapping exact en autorisant que l'ensemble des entrées de R soit un sous-ensemble des entrées de S et que les sorties de S soient des sous-concepts directs des sorties de R. $\forall\ IN_S\ \exists\ IN_R : IN_S \supseteq IN_R \wedge \forall\ OUT_R\ \exists\ OUT_S : OUT_S \in \mathrm{LSC}(OUT_R)$.
subsumes(R,S)	Le mapping subsumes relaxe le mapping plugin en autorisant que les sorties de S soient des concepts spécifiques de R et pas seulement des sous-concepts directs comme le cas du plug-in : $\forall\ IN_S\ \exists\ IN_R : IN_S \supseteq IN_R \wedge \forall\ OUT_R\ \exists\ OUT_S : OUT_R \supseteq OUT_S$.
fail(S,R)	pas de matching

TABLE 3.1 – Définition des degrés d'appariement

un mécanisme de *mapping* permettant d'encapsuler les descriptions OWL-S des services dans un annuaire UDDI. L'algorithme de matching permet déduire les différents degrés de matching du tableau 3.1. Pour optimiser le temps d'exécution de l'algorithme (une amélioration par rapport à [PKP+02b]), les auteurs proposent qu'au moment de la publication du service, les concepts de l'ontologie du composant de découverte seront annotés par le degré de matching qu'ils peuvent avoir avec les concepts du service à publier.

Jaeger et al. [JRL+05] proposent également un algorithme de matching de services Web décrits en OWL-S à un niveau de granularité fin en considérant les entrées, les sorties et la catégorie du service. L'approche se base également sur la déduction d'une relation de subsomption ou d'équivalence entre concepts. L'algorithme comporte quatre étapes : (a) matching des entrées, (b) matching des sorties, (c) matching de la catégorie du service et (d) aggrégation des résultats avec les contraintes de l'utilisateur qui peut spécifier ses préférences en pondérant les différents types de matchings (par exemple, un utilisateur peut pondérer le matching des sorties avec un poids de 0.8,). Bien que les auteurs proposent un prototype basé le raisonneur OWLJessKB [19], ils n'ont pas rapporté aucun résultats de leurs expérimentations.

Pathak et al. [PKCH05] proposent un framework de découverte de services Web qui repose sur l'ensemble des *mappings* définis par l'utilisateur pour mettre en correspondances les concepts de l'ontologie utilisateur aux différents concepts de différentes ontologies ayant servi à la description du service. L'algorithme d'appariement exploite ces mappings pour sélectionner les services potentiels à une requête. L'approche propose de classer les services sélectionnés en terme de leur QdS, l'utilisateur pourra exprimer ses préférences en terme de QdS en pondérant certains critères par rapport à d'autres. L'approche est intéressante à notre avis pour deux principales raisons : (1) la première est que les services peuvent

19. OWLJessKB est un outil de raisonnement http ://edge.cs.drexel.edu/assemblies/software/owljesskb/

être décrits en se basant sur plusieurs ontologies du domaine, et (2) la seconde est que cette approche se base sur les propriétés QdS des services pour classer des services fonctionnellement équivalents.

PCEM [SHS08] est un moteur de découverte implémenté dans le cadre du projet CASCOM [20]. Le moteur PCEM traduit les descriptions OWL-S et les requêtes du client en PDDL (*Planning Domain Definition Language*)[MGK+98] et calcule une implication logique entre pré-conditions et les post-conditions. Il est possible de déduire différents niveaux de degrés de matching par exemple, un matching en plug-in entre un service S et une requête R exprime que, pour chaque modèle de la base de connaissances kb, l'effet du service S est plus spécifique que celui de R et la précondition de S soit plus générale que celle de R.

WSMX [RKL+05b, FKT09] est une architecture et un environnement d'exécution de découverte de services Web. Dans WSMX, les services Web et les buts (requêtes de découverte) sont décrits selon le framework conceptuel WSMO par le biais d'un ensemble d'expressions logiques déclarées dans le langage WSML. Une ontologie commune permet de définir l'ensemble des axiomes d'un domaine est créé et puis importé (par le biais de la primitive *importsOntology* pour décrire à la fois les services Web et les buts. La figure 3.9 illustre l'exemple d'un axiome définissant la relation isShippedDef spécifié par une instance du concept shipmentOrderReq qui restreint les adresses de livraisons (seulement en Europe, en Amérique du nord, en Asie et en Afrique) et le poids du colis (qui doit être inférieur à 50). Ce type d'axiome peut être utilisé pour décrire un service Web en WSMO. Les buts peuvent également spécifier des contraintes QoS comme l'indique la figure 3.10 définissant une postcondition devant être vérifiée par le service et qui porte sur le paramètre QoS *prix*. Un raisonneur WSML permet de matcher la requête de découverte à une description de services en vérifiant sa satisfiabilité. L'algorithme de matching crée une base de connaissances temporaire comprenant la spécification du but, la spécification d'un service Web W_i du catalogue de services et l'ontologie commune O ayant été utilisée pour la description des services et du but. Un test de satisfiabilité permet de déterminer si le service W_i satisfait le but en vérifiant à la fois les préconditions et les postconditions. L'ensemble des services sélectionnés sont classés selon les préférences des utilisateurs (e.g., *prix*).

Della Valle et al. [VC05] présentent COCOON GLUE, un prototype pour la découverte de services décrits en WSMO dans le domaine médical (figure 3.11). Le prototype est construit au dessus d'un moteur de raisonnement à base de F-logic [21] combinant de la programmation logique en Prolog et des techniques de bases de données déductives [KLW95].

20. CASCOM (acronyme de Context-Aware business application Service CoOrdination in Mobile computing) est un projet européen lancé en septembre 2004 par le centre de recherche allemand DFKI (German Research Center for Artificial Intelligence), l'école polytechnique de Lausanne et plusieurs partenaires industriels tels que TeliaSonera et EMA group (Finlande) et FRAMeTech (Italie). Ce projet a pour objectif de concevoir, implémenter et valider une infrastructure de services Web sémantiques pour les réseaux mobiles.

21. F-logic (Frame logic) est un langage de représentation de connaissances combinant les avantages de la modélisation orienté objet et les langages à base de frame, offrant ainsi un langage avec une syntaxe déclarative et simple et muni d'une sémantique formelle des langages à base de logique.

```
1  ...
2  axiom isShippedDef
3      definedBy
4        ?shipmentOrderReq[sop#to hasValue ?temp, sop#package hasValue ?package] memberOf sop#
               ShipmentOrderReq and
5        ?temp[so#address hasValue ?to] and
6        ?to[so#city hasValue ?city] and
7        isShippedContinents(?city, so#Europe, so#Asia, so#NorthAmerica, so#Africa) and
8        ( (?package [so#weight hasValue ?weight] memberOf so#Package) and (?weight =< 50) )
9      implies
10       sop#isShipped(?shipmentOrderReq).
11 ...
```

FIGURE 3.9 – La définition de l'axiome isShipped en WSML

```
1  Goal
2    nfp
3      _"preference" hasValue "?price"
4    ...
5    endnfp
6  ...
7  capability goal1aCapability
8     postcondition
9        definedBy
10   ?x[sop#price hasValue ?price] memberOf sop#PriceQuoteResp
11   and ws#isShipped(gl#shipmentOrderReq)
12 ...
```

FIGURE 3.10 – Exemple de postcondition en WSML

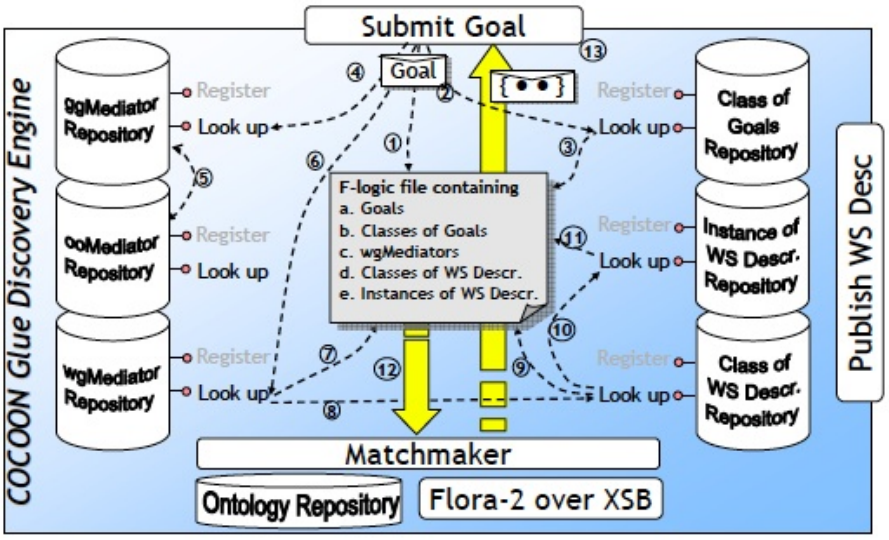


FIGURE 3.11 – Architecture du framework COCOON GLUE

Le prototype offre d'autres composants permettant la publication de services Web WSMO. L'approche proposée prend en considération l'aspect communautaire en introduisant le concept de communautés de pratique [22], chacune pouvant avoir sa propre conceptualisation (ontologie). L'algorithme d'appariement se base sur l'implémentation de médiateurs de type *wgmediator* (médiateur Web service-goal en WSMO). Etant donné une requête utilisateur, l'administrateur COCOON Glue implémente un médiateur *wgmediator* en spécifiant un ensemble de règles f-logic définissant les appariement entre une classe de description de services Web et une classe de buts. L'avantage principal de cette approche est qu'elle permet d'apparier la signature fonctionnelle complète en terme d'entrées, sorties, préconditions et postconditions (IOPE).

Enfin, **Dominigue et al.** [DCG+08] présente **IRS III** (présenté précédemment dans la section 2.2.6), un framework de découverte de services décrits en **WSMO**. IRS III est une solution permettant : (a) de découvrir les services potentiels à une requête, (b) de sélectionner les services les plus appropriés, (c) résoudre les conflits d'hétérogénéité au niveau conceptuel, (d) d'invoquer les services Web sélectionnés en respectant les contraintes d'invocation de chaque service. IRS III se base sur son propre langage de définition d'ontologies OCML. L'avantage de OCML par rapport à d'autres langages de définition d'ontologies tels que OWL est qu'il offre des primitives pour la définition de modèles ontologiques, règles et expressions logiques. IRS III sépare le contexte des utilisateurs du contexte des services Web et se propose de réconcilier les deux contextes en se basant sur des techniques

22. La notion de communauté de pratique, en anglais Community of Practice (ou CoP) désigne le processus d'apprentissage social émergeant lorsque des personnes ayant un centre intérêt commun collaborent mutuellement. Cette collaboration, qui doit se dérouler sur une période de temps notable consiste à partager des idées, trouver des solutions, construire des objets nouveaux, etc. On parle aussi de communauté de pratique pour désigner le groupe de personnes qui participe à ces interactions.

de médiation. La médiation dans IRS III se base sur la définition (de manière exhaustive) d'un ensemble de mappings pour résoudre tout type de conflit conceptuel, ces mappings sont représentés sous forme de règles logiques.

3.3.3 Approches hybrides combinant techniques de RI et raisonnement

Les approches hybrides pour la découverte de services Web proposent de combiner les approches se basant sur les techniques de raisonnement avec des techniques de RI pour améliorer la qualité des résultats du processus de découverte.

Dans ce contexte, **LARKS** [SWK+02] fut la première approche hybride considérant des signatures fonctionnelles en termes de IOPE décrites dans un langage à base de frames, dénommé LARKS (Language for Advertisement and Request for Knowledge Sharing). Une signature (spécification) définie en LARKS permet d'expliciter le contexte (domaine) du service, les types des variables utilisées, la déclaration des variables en entrée et en sortie, les contraintes logiques exprimées en clause de Horn (préconditions et postconditions), une description des contraintes basée sur des concepts ontologiques et une description textuelle des contraintes et du service. Le processus de découverte au niveau de l'approche LARKS se base sur cinq types d'appariement : appariement de contexte (context matching), comparaison de profil (profile comparison), appariement de similarité (Similarity matching), appariement de signature (signature matching) et appariement de contrainte (constraint matching). Notons que l'appariement de similarité repose sur des mesures de similarité syntactiques permettant de mesurer la similitude entre les descriptions textuelles et les noms des paramètres en entrée et en sortie. L'appariement de signature et de contraintes repose sur des règles d'inférence logique permettant de déduire un appariement de type logique (ou exact ou plugin). Bien que l'approche LARKS est a priori intéressante mais le manque d'expérimentations et de résultats constitue un inconvénient majeur de cette approche.

FC-match est une approche hybride proposée par [BAM08] combinant du raisonnement déductif basé sur les logiques de description et les mesures de similarité. En plus de la signature fonctionnelle, l'approche considère également la catégorie du service comme critères de *matching*. Les techniques proposées reposent sur une ontologie du domaine mais également sur l'ontologie linguistique WordNet afin de déduire des relations sémantiques tels que la synonymie, la méronymie, l'hyperonomie, etc. Utiliser simultanément une ontologie du domaine et une ontologie linguistique permet d'identifier une relation subsomption entre des concepts n'existant pas nécessairement dans l'ontologie du domaine. Bien que les auteurs de [BAM08] présentent une étude empirique de l'approche proposée, ils n'offrent pas de comparatif avec d'autres approches telles que OWLS-MX qui sera présentée dans ce qui suit.

Trois principales approches hybrides adaptées chacune à un framework de description sémantique ont été proposées, à savoir : **OWLS-MX** [KFS06, KFS09], **SAWSDL-MX** [KK08] et **WSMO-MX** [KK09].

Klusch et al. [KFS06, KFS09] ont proposé OWLS-MX alliant raisonnement déductif (basée sur LD) et des techniques de RI pour matcher des signatures de services Web décrites en OWL-S profile. Chaque variante de l'approche **OWLS-MX1** à **OWLS-MX4**

se propose d'appliquer une mesure de similarité spécifique [23] à un couple de signatures de services pour déterminer le degré de similarité syntaxique. Les auteurs proposent deux nouveaux degrés d'appariement hybrides qui définissent le degré en considérant la valeur de la mesure de similarité, à savoir :

– *Matching Subsumed-By.* Cet appariement reflète le cas où les sorties d'un service sont des concepts "plus généraux" (super-concepts) que les concepts de la requête et se définit comme suit :

$$\forall\, IN_S\ \exists\, IN_R : IN_S \supseteq IN_R \wedge \forall\, OUT_R\ \exists\, OUT_S : (OUT_S \doteq OUT_R \vee OUT_S \in \text{LGC}\,[24](OUT_R)) \wedge SIM_{IR}\,[25](\text{S,R}) \geqslant \alpha.$$

– *Matching Voisin-le-plus-proche.* Un service S est le voisin le plus proche d'une requête Q si et seulement si

$$\forall\, INT_S\ \exists\, INT_R : INT_S \supseteq INT_R \wedge \forall\, OUT_R\ \exists\, OUT_S : OUT_R \supseteq OUT_S \vee SIM_{IR} \geqslant \alpha.$$

C'est dans ce sens que OWLS-MX permet d'inclure dans l'ensemble des solutions d'une requête, des services qui sont *syntactiquement similaires* (par rapport à un seuil minimal) mais qui sont *logiquement disjoints*. Les expérimentations rapportées montre l'intérêt de combiner des méthodes statistiques (mesures de similarité) aux techniques de raisonnement. Cependant, le fait que l'approche n'a pas l'objet de comparaison avec d'autres approches hybrides de découverte ne nous permet pas de conclure sur sa pertinence.

Klusch and Kapahnke [KK08] proposent une approche hybride de découverte **SAWS-DL-MX** adaptée aux services annotés en SAWSDL exploitant du raisonnement déductif et des techniques de RI (les mêmes mesures de similarité que OWLS-MX). L'approche montre que l'hybridation donne des résultats plus significatifs que chacune des approches de son côté.

Enfin, **Kaufer et Klusch** [KK09] proposent l'approche **WSMO-MX** adaptée à des services décrits dans une variante du langage WSML-rule, nommé WSML-MX. L'approche consiste en la combinaison de trois approches : (1) l'approche hybride OWLS-MX en exploitant les mêmes mesures de similarités syntactiques évaluées par l'approche OWLS-MX, (2) l'approche de DSD-MM qui se base sur un matching de graphes orienté-objet pour déduire un appariement entre une requête et un service représentés dans le format DSD, et (3) une approche proposée par Keller et al. dans [KLL+05] qui se base sur la sémantique intensionnelle d'un service Web en termes d'hypothèses et effets [26]. Ainsi, WSMO-MX combine la programmation logique et le raisonnement approximatif pour déduire un matching entre une requête et une description de services Web, le degré de matching est le résultat de l'agrégation d'un triple matching : (a) un matching au niveau types sémantiques des paramètres [27] (pour déduire un matching logique d'équivalence, de subsomption ou de disjonction entre types), (b) un matching entre contraintes logiques (hypothèses et effets)

23. à titre d'exemple, cosinus/TFIDF, Jaccard étendu, Jensen-Shannon.

24. LGC(C), Least Generic Concept(C), désigne les super-concepts (parents) directs d'un concept C.

25. SIM_{IR} désigne une valeur numérique comprise entre 0 et 1 qui indique la similarité syntaxique entre les deux chaînes A et B en considérant une mesure de similarité syntaxique.

26. Dans l'approche présentée par Keller et al. [KLL+05], les auteurs attribuent une représentation concrète de ce que fait le service et sous quelles conditions. Le service est perçu comme une transition d'un état initial vers un état final.

27. Au niveau de cette approche, un type sémantique est soit un concept ou une relation ontologique

définies en Logique du premier ordre et (c) un matching syntactique en se basant sur des mesures de similarités. Une première étude expérimentale reportée dans [KK09] avait pris en considération une collection de 97 services (seulement !), 22 requêtes avec 325 concepts. Les expérimentations montrent que l'approche hybride donne de meilleurs résultats que l'approche basée seulement sur des techniques de raisonnement et que le matching syntactique en améliorant ainsi la pertinence des résultats. Seulement la taille des tests est trop petite à notre avis, les auteurs promettent d'autres résultats plus élaborés dans des prochains papiers.

D'un autre côté, **Kiefer and Bernstein** propose iMatcher2 [KB08], une approche hybride pour la découverte de services Web basée sur iSPARQL (imprecise SPARQL) qui consiste en une extension de SPARQL incluant différentes mesures de similairté [28]. Les services sont décrits en OWL-S profile qui seront transformé en un vecteur pondéré de mots-clés. iMatcher2 applique différentes stratégies (combinaisons de mesures) pour évaluer la similarité entre descriptions de services et requête. Les expérimentations ont été menées sur la base d'une collection de 576 services Web ont montré l'intérêt d'une telle hybridation. De plus, les auteurs montrent également l'apport de certains techniques d'apprentissage (basées sur des modèles de régression) à améliorer l'efficacité et la pertinence des résultats du processus de découverte. Enfin, les expérimentations ont montré qu'en terme de performance (précision et rappel), l'approche iMatcher2 est meilleure que celle de OWLS-MX4.

Enfin, **Meditskos et Bassiliades** [MB10] proposent un framework de découverte de services Web décrits en OWL-S Profile, nommé **OWLS-SLR** (SLR pour Structural and Logic-based Reasoning). Le framework repose sur une approche hybride combinant du raisonnement à base de cas structurel (Structurel Cased-based Reasoning- SBRC) et du raisonnement en logique de description. L'approche se base sur des descriptions de services dites légères (*lightweight*) comprenant les entrées, les sorties et le domaine du service. Les auteurs argumentent ce choix afin un temps minimal de recherche des services candidats à une requête. L'idée de base du raisonnement à base de cas structurel est de représenter la sémantique des services en se basant sur un modèle du domaine structuré selon une approche orienté-objet. Ainsi, les services et les requêtes sont modélisés comme des objet avec des connaissances additionnelles qui proviennent du modèle du domaine. La similarité entre objets est déterminée par le biais de deux métriques : la métrique *interclass* et celle de *intraclass*. La première détermine la similarité entre objets en terme de leur position hiérarchique dans une taxinomie alors que la dernière est définie en terme des attributs en commun entre les deux objets. Ces deux métriques ont été adaptées par les auteurs au contexte de la découverte de services Web. Les expérimentations menées pour évaluer l'approche ont considéré une collection de test de 1007 services Web décrits en OWL-S profile et 29 requêtes. En comparaison avec l'approche OWLS-MX, l'approche OWLS-SLR montre une meilleure performance au niveau du temps d'exécution de la requête, en précision et rappel au niveau des solutions.

28. Bi-Gram, Levenshtein, Monge-Elkan et Jaro.

3.4 Synthèse et Conclusion

L'état de l'art présenté dans ce chapitre montre la multitude d'approches développées afin de pallier les insuffisances du modèle (classique) de découverte de services Web basé sur les annuaires UDDI et adoptant une recherche par mots-clés. Afin de comparer les approches de découverte présentées tout au long de ce chapitre, nous avons proposé les trois critères suivants :

Critère 1 : Le framework de description de services. Comme cela a été présenté dans le chapitre 2 de cette thèse, différents frameworks de descriptions sémantiques de services Web ont été proposés. Nous distinguons les ontologies de service telles que OWL-S ou WSMO et l'extension (annotations) du standard WSDL (Web Service Description Langage) dénommé SAWSDL. Chacune des approches de découverte adopte un framework spécifique de description sémantique de services Web.

Critère 2 : La méthode/approche de la sélection du service. Il s'agit de préciser l'algorithme d'appariement sur lequel repose le moteur de découverte pour sélectionner les services candidats à une requête.

Critère 3 : Les critères de sélection retenus par l'approche de découverte . La phase de sélection de services Web consistant à apparier la requête de l'utilisateur à un certain nombre de propriétés du service. Certaines approches considèrent seulement les entrées et les sorties (IO) durant la phase de matching, d'autres considèrent les préconditions et les postconditions (PE) et enfin d'autres approches considèrent toute la signature fonctionnelle d'un service (IOPE).

Le critère 1 permet de classer les approches proposées selon le framework de description de services Web.

- *Approches de découverte de services décrits en OWL-S* : Des approches basées sur le raisonnement telles que celles OWLS-UDDI [PKP+02a, PKP+02b] et OWLSM [JRL+05] considérent des descriptions OWL-S, PCEM traduit les descriptions OWL-S en actions PDDL pour ensuite pouvoir raisonner sur les préconditions et les post-conditions (en se basant sur PROLOG). Plus récemment, des approches hybrides ont été développées telles que OWLS-MX [KFS06] visant à pallier les insuffisances des approches basées sur le raisonnement et améliorer leur performance. OWLS-iMatcher [BK06] est une approche basée sur les techniques de RI permet de matcher la signature fonctionnelle complète (en terme d'entrées, sorties, préconditions et postconditions) en exploitant différentes techniques de similarité. Tout récemment, les auteurs de [MB10] proposent l'approche hybride OWLS-SLR se basant sur du raisonnement de cas structurel, les auteurs reposent sur une modélisation orienté objet des services et des buts et se proposent d'utiliser deux métriques de similarité structurelle.
- *Approches de découverte de services Web décrits en WSML* : WSMX et COCOON Glue sont toutes les deux des architectures de découverte de services WSML, basées sur le raisonnement. WSMO-MX est une approche hybride pour la découverte de services décrits en WSML-MX, une variante de WSML-rule. Enfin, IRS III [CDG+06, DCG+08] est également un framework pour la découverte de services décrits dans

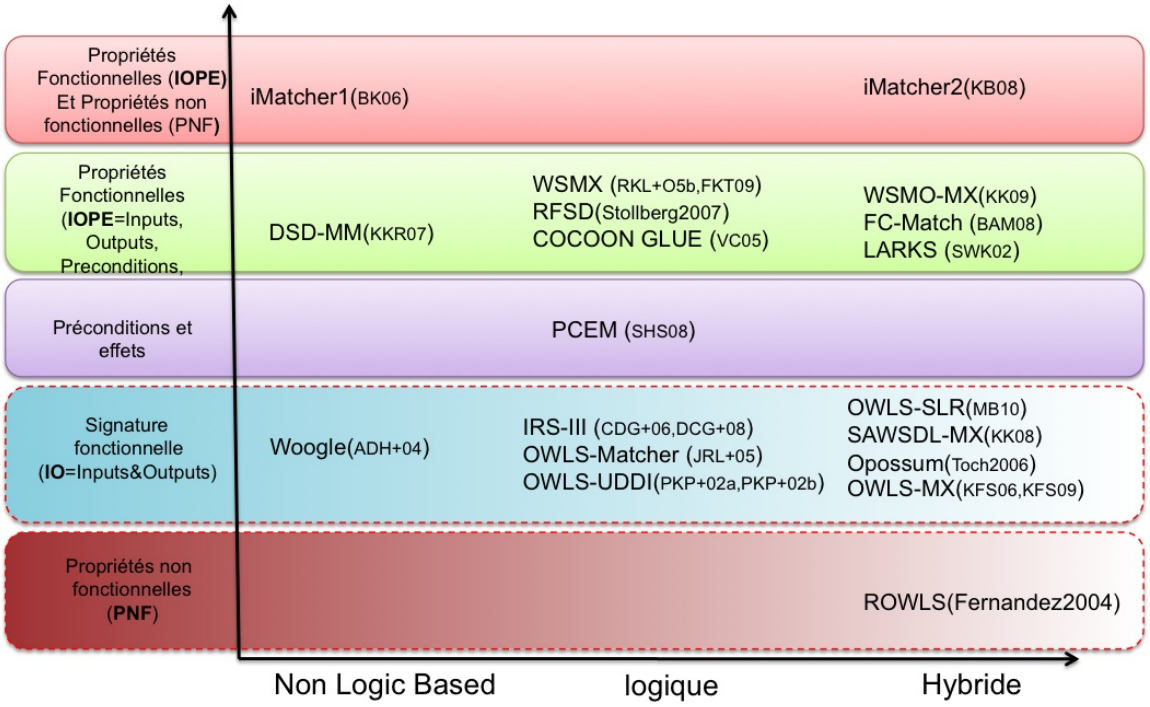


FIGURE 3.12 – Classification des approches de découverte

le framework WSMO, cependant IRS III traduit les descriptions WSMO dans son propre langage OCML pour pouvoir raisonner dessus.

– *Approches de découverte de services Web annotés en SAWSDL* : METEOR-SWSDI [VSS+05] est une infrastructure pair-à-pair pour la publication et la découverte de services Web SAWSDL reposant sur des registres décentralisés. Lumina est un moteur de découverte reposant sur UDDI pour stocker des services Web décrits en SAWSDL. Enfin, SAWSDL-MX [KK08] est une approche hybride développée à l'instar de OWLS-MX et WSMO-MX.
– *Autres frameworks* : LARKS [SWK+02] est une approche hybride pour la découverte de services Web décrits en LARKS (Language for Advertisement and Request for Knowledge Sharing). Le moteur de découverte effectue un matching au niveau du contexte (domaine du service), au niveau de la fréquence des termes spécifiés dans le profil, matching de signature et le matching de contraintes. DSD-MM [KKR07] est une approche de découverte pour les services décrits en DIANE qui se base sur des techniques de matching de graphes.

La figure 3.12 présente une classification des approches de découverte de services Web présentées, le premier axe de classification est la méthode de découverte utilisée (pas de raisonnement, raisonnement logique, hybride) alors que le second axe est l'ensemble des propriétés du service utilisé pour sa découverte. Nous remarquons qu'un nombre important d'approches adoptent le raisonnement logique. En général, ces approches s'appuient sur un raisonneur logique afin de déduire une relation d'équivalence ou de subsomption entre

les concepts d'un service et ceux d'une requête et cela en se exploitant les connaissances du domaine (explicitées généralement dans une ontologie). D'une part, ces approches offrent une bonne précision et rappel reflétant la qualité des réponses, mais d'autre part elles ont une flexibilité et une scalabilité limitée. Par ailleurs, nous remarquons que, ces dernières années, plusieurs approches hybrides combinant raisonnement déductif et mesures de similarité ont été proposées, bien que la première approche de ce type fut proposée en 2002 (à savoir l'approche LARKS). Ces approches offrent une grande flexibilité mais dépendent étroitement des mesures adoptées et donc les résultats sont variables.

Nous notons également que la majorité des approches considèrent comme critères de découverte les entrées et les sorties des services. Les tableaux 3.2 et 3.3 résument toutes les approches présentées dans ce chapitre en précisant le framework de description adopté par l'approche (critère 1), les l'approche adoptée pour l'appariement (critère 2) et enfin les propriétés du services utilisées pour la découverte en précisant comment les solutions sont classées (critère 3). Par ailleurs, nous notons l'inexistence d'approches se basant sur le raisonnement déductif et intégrant comme critères de découverte les propriétés non fonctionnelles d'un service. Ainsi, l'approche de découverte proposée vise à combiner les avantages du raisonnement déductif en termes de pertinence des résultats et ceux d'une approche d'analyse multi-critère pour plus de flexibilité. Notre approche peut être qualifiée d'originale dés lors qu'elle se propose de : (a) élargir la typologie des mécanismes inférentiels afin d'offrir un appariement plus flexible (b) considérer les critères non fonctionnels dans le processus de sélection et de découverte en plus des critères fonctionnelles d'un service, (c) prévoir un mécanisme de classement pour déceler les services Web ayant les meilleurs attributs QdS.

Ainsi, au niveau du méta-framework BioMed, nous formalisons les mécanismes de raisonnement grâce à une axiomatique formelle exploitant tout à la fois les descriptions unifiées de service Web, les annotations sémantiques et les connaissances ontologiques. Nous proposons une plus large typologie de techniques de raisonnement que celles fournies par les approches de découverte basées sur le raisonnement qui exploitent seulement la relation de subsomption.

Notre approche de découverte est une approche hybride dans le sens qu'elle combine des techniques de raisonnement déductif et une méthode d'analyse multi-critère. Les propriétés sémantiques considérées dans la phase de découverte correspondent aux entrées, sorties et propriétés non fonctionnelles reliées à la qualité du service. Dans ce contexte, la seule approche comparable à l'approche BioMed est le framework WSMX qui repose également sur une analyse multi-critère où les paramètres non fonctionnels sont pondérés selon les préférences des utilisateurs. Notre approche considère tous les critères d'égale importance et applique une analyse multi-critère à un ensemble de services dominants.

L'approche de découverte proposée dans cette thèse est présentée dans le détail dans le chapitre 6. Dans l'ensemble, notre approche se présente comme un méta-framework stratifié couvrant les phases de description, de découverte et de composition de services Web. La problématique de composition de services est à la fois importante et cruciale, elle se pose lorsqu'aucun service atomique ne satisfait la requête de l'utilisateur. Nous présentons dans le prochain chapitre un état de l'art des approches de composition de services Web développées dans divers domaines telles que l'Intelligence Artificielle ou la

conception des flots de données (ou *Workflow* en anglais).

Nom de l'approche de découverte	Woogle	OWLS-UDDI	OWLS-Matcher	OWLS-MX	iMatcher	OWLS-SLR	LARKS
Framework de description des services	WSDL	OWL-S profile	OWL-S profile	OWL-S profile	OWL-S profile	OWL-S profile	LARKS
Critères de sélection	IO (noms des opérations WSDL et des paramètres)	IO	IO	IO	IOPE	IO	IOPE
Approche Adoptée pour l'appariement	Clustérisation hiérarchique agglomérative	Raisonnement	Raisonnement	Hybride	Mesures de similarité	hybride	Hybride
classement des solutions (ranking)	-	Degré d'appariement (exact, plugin, subsomption et fail)	Degré d'appariement (exact, plugin, subsomption et fail) en prenant en considération les préférences de l'utilisateur	Degré d'appariement (exact, plugin, subsomption, subsumed-By, Nearest-Neighbour et fail)	Valeur de similarité	-	-

TABLE 3.2 – Comparaison des approches de découverte de services Web (1)

	PCEM	IRS III	COCOON GLUE	WSMX	WSMO-MX	METEORS-SWDI	SAWSDL-MX
Format de descriptions des services	Traduction de OWL-S profile en PDDL	Traduction de WSMO en OCML	WSMO	WSML	variante WSML	SAWSDL	SAWSDL
Critères de découverte	PE	IOPE	IOPE	IOPE	IOPE	IO	IO
Technique d'appariement	Raisonnement	Raisonnement	Raisonnement	Raisonnement	Hybride	Techniques de similarité	Hybride
classement des solutions	degré de matching (exact, plugin, subsomption et fail)		degré de matching (exact, plugin, subsomption et fail)	degré de matching (exact, plugin, subsomption, subsumed-By, Nearest-Neighbour et fail)		-	degré de matching (exact, plugin, subsomption, subsumed-By, Nearest-Neighbour et fail)

TABLE 3.3 – Comparaison des approches de découverte de services Web

Chapitre 4

Composition de Services Web

Sommaire

Dans les deux chapitres précédents, nous avons tenté de couvrir les principales contributions en matière de description et de découverte de services Web sémantiques en considérant les services comme des entités élémentaires (atomiques). Cependant, il est possible que le besoin de l'utilisateur ne soit pas satisfait par un service atomique et requiert la combinaison d'un ensemble de services. Ce processus est appelé *composition de services Web* [ACKM04, BDDM05]. La composition de services vise à faire interopérer, interagir et coordonner plusieurs services pour la réalisation d'un même but. Les services Web invoqués lors d'une composition sont appelés *services Web composants* et le service résultant de la composition est appelé *service composite*. La mise en oeuvre d'une composition de services engendre un certain nombre de problèmes liés principalement à la sélection des services Web composants et l'implémentation des interactions entre ces services.

Dans ce chapitre, nous présentons dans un premier temps le processus de composition de services de services Web pour nous focaliser ensuite sur l'ensemble de travaux propo-

sant des plates-formes d'exécution de composition de services développés dans plusieurs domaines.

4.1 Définitions et types de composition de Services Web

La composition de services est une problématique abordée par un nombre considérable de travaux, aussi bien dans la recherche académique que dans l'industrie. En dépit de tous les efforts déployés, la composition de services reste un problème très complexe. Cette complexité tient au fait que les solutions de composition de services doivent tenir compte du nombre croissant de services déployés sur le Web, de leur mise à jour continuelle et de leur multiple hétérogénéité. En conséquence, ces solutions nécessitent de traiter les problématiques de la coordination des services, leurs transactions, leur exécution et leurs modèles de conversation (ou d'interaction).

4.1.1 Définitions

La composition de services peut être vue comme un mécanisme permettant l'intégration des services pour réaliser une application. Il est à noter que pour passer d'un ensemble de services à une composition de services correctement structurée, il est nécessaire de suivre les cinq étapes suivantes allant de la spécification à la composition concrète exécutable :

1. *La définition de l'architecture fonctionnelle* : cette phase est effectuée pour identifier les fonctionnalités attendues de l'application résultante de la composition de services.
2. *l'identification des services* : selon les fonctionnalités attendues, il s'agit de déterminer les services nécessaires à la composition.
3. *la sélection des services et leur implantation* : à partir des services identifiés à l'étape de découverte, il faut sélectionner les services qui répondent correctement aux besoins ainsi que les implantations adaptées ; les phases d'identification et de sélection correspondent à la phase de découverte.
4. *la médiation entre services* : bien qu'à l'étape précédente, les services les plus adaptés aient été sélectionnés, en général, il n'est pas possible de les assembler tels qu'ils sont. Il est nécessaire de disposer de techniques de médiation afin que les interactions entre services fonctionnent correctement, i.e., afin que les services interprètent correctement les données échangées.
5. *le déploiement et l'invocation des services* : une fois la composition correctement réalisée, il faut déployer les services sur les plates-formes d'exécution. Il est ainsi possible d'invoquer les services pour obtenir la composition concrète.

Ces étapes montrent la difficulté et la complexité de la tâche de composition. Cette complexité est double : d'une part, la complexité de la logique métier inhérente aux applications qui nécessite une expertise métier et d'autre part, la complexité de la mise en oeuvre de l'approche à services.

Dans la littérature, il y a une distinction entre deux types de composition : l'*orchestration* [TP04] et la *chorégraphie* [ABPR04]. Dans le cas d'une orchestration, un processus principal (service Web) prend le contrôle du déroulement de la composition et coordonne donc les différentes opérations des différents services Web. À aucun moment les autres services contribuant à la composition n'ont connaissance de cette composition : ils remplissent leur rôle de service sans se soucier si un client humain ou un applicatif interagit avec eux. Le processus d'orchestration est connu a priori et mis en oeuvre à partir d'un coordinateur central. L'orchestration est une méthode très utilisée, et ce pour deux raisons :

1. Il est relativement simple d'écrire un processus centralisé gérant l'invocation de sous-services ;
2. dans le cas d'une réutilisation de services Web basiques, seule la partie centrale est à développer.

Dans le cas d'une chorégraphie, il n'existe pas de coordinateur central. Chacun des services intervenant dans la composition sait exactement ce qu'il doit faire, quand il doit le faire et avec qui le faire. Ainsi, ils ont tous une connaissance plus ou moins globale du processus métier dans lequel ils se retrouvent. Contrairement à la méthode basée sur l'orchestration, la chorégraphie demande nettement plus de développement et de test : il faut développer chaque service dans le but de la composition auquelle il peut participer. L'intérêt de la mise en place d'une chorégraphie de services est principalement lié à la non-centralisation du traitement. C'est la raison pour laquelle, nous placerons le présent travail dans le cas d'une chorégraphie de services Web atomiques.

4.1.2 La composition de services et ses catégories

Ces dernières années, plusieurs communautés se sont intéressées à la problématique de la composition de services. Des solutions ont été proposées par les communautés des bases de données, du Web sémantique, et plus récemment par la communauté de l'Intelligence Artificielle (IA).

Les solutions proposées peuvent être classées selon deux axes orthogonaux (figure 4.1) :

1. En fonction du degré de participation de l'utilisateur dans la définition du schéma de composition, ces propositions peuvent être *manuelles*, *semi-automatiques* ou *automatiques*.
2. Selon que la sélection des services et la gestion du flot soient faites ou non a priori, une approche sera dite *statique* ou *dynamique*.

Une composition est dite *statique* lorsqu'elle prend place à l'étape de conception, au moment où l'architecture et la conception du système logiciel sont planifiées. Les composants (ou services) qui seront utilisés sont préalablement choisis et reliés, et la gestion du flot est effectuée a priori. Les approches statiques de composition de services sont celles adoptées par l'industrie. Elles s'inspirent de la gestion de processus métiers quant à la description des données et des flots de contrôle pour le processus de composition.

Par opposition, une composition de services est dite *dynamique* si les services sont sélectionnés et composés à la volée en fonction des besoins formulés par l'utilisateur. Une approche dynamique pour la composition de services offre le potentiel de réaliser des

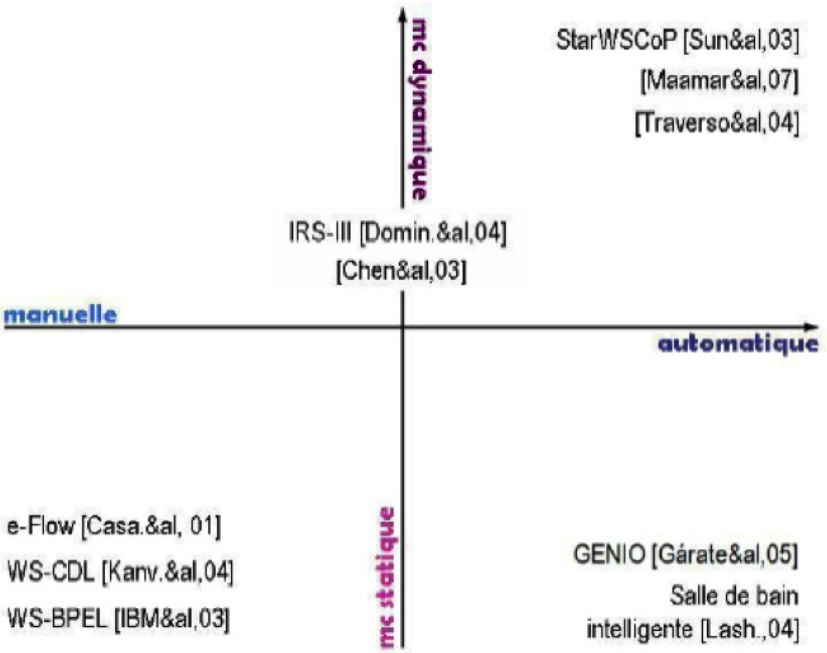


FIGURE 4.1 – Classification des principaux travaux de composition de services [Cha07]

applications flexibles et adaptables en sélectionnant et en combinant les services de manière appropriée sur la base de la requête et du contexte de l'utilisateur. Ce type de composition peut engendrer de nombreuses applications utiles qui n'ont pas été prévues à l'étape de conception. Notons que la composition dynamique de services est tout à fait intéressante dans un environnement ouvert tel que le Web ou l'informatique pervasive (diffuse ou ambiante) où les composants disponibles sont dynamiques et les attentes des utilisateurs variables et personnalisées ce qui est le .

Nous présentons dans ce qui suit les approches de composition selon le premier axe de la classification, soit les approches manuelles/semi-automatiques et automatiques. Lors de notre présentation, nous catégoriserons chacune des approches selon le deuxième axe en précisant si la sélection ou le moteur de composition y sont statiques ou dynamiques. Nous présentons, dans ce qui suit, les principales approches de composition selon la catégorisation de la figure 4.1.

4.1.2.1 Composition manuelle/semi-automatique

Les techniques de composition manuelle supposent qu'un expert se charge de définir et générer, graphiquement ou moyennant un éditeur de texte, des scripts de workflow [1] qui

1. Un workflow est une représentation d'un procédé métier dans un format interprétable par la machine. Il est constitué d'un réseau d'activités et de dépendances entre elles, des critères pour spécifier le démarrage et la terminaison d'un procédé et des informations sur les activités individuelles (participants, applications,

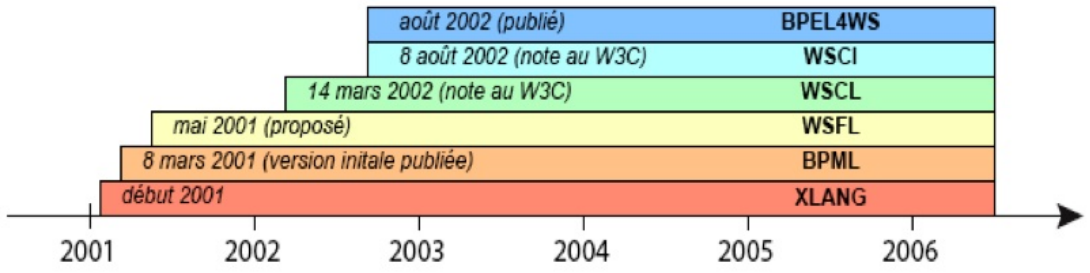


FIGURE 4.2 – Langages de définition d'orchestration et de chorégraphie par dates de publication

sont ensuite soumis à un moteur d'exécution de workflows. La plupart d'entre elles sont entièrement statiques puisque les services à composer sont préalablement sélectionnés, et le flot de composition est défini a priori. BPEL4WS [2] (*Business Process Execution Language for Web Services*) [KMW03, ACD+03] qui est un exemple de langage d'orchestration, permet de décrire les interactions des services composants à travers un plan en spécifiant les flots de contrôle quant à l'invocation des services, ainsi que les dépendances des données existantes entre plusieurs processus métiers. Les approches manuelles de composition reposent toutes sur un langage de description de workflows. La figure 4.2 résume les différents langages de définition de workflows développés essentiellement dans l'industrie.

D'autres approches manuelles comportent en revanche des moteurs de composition dynamiques. C'est par exemple le cas de *eFlow* [CIJ+00] qui est un système spécifiant, ordonnant et surveillant des e-services [3] composés.

Du côté des approches semi-automatiques, des outils sont développées afin de guider l'utilisateur dans la sélection des services à composer (par exemple, en fonction de leurs entrées/sorties, annotations, etc.) au cours du processus de composition. Par exemple, l'outil graphique IRS-III développé par Dominigue et Gualizia [GD04] guide l'utilisateur étape par étape pour la composition de services décrits en WSMO. Durant le processus de composition, le framework propose à l'utilisateur de sélectionner des buts, des médiateurs et des opérateurs de flot de contrôle. Une fois que le service composé est défini, l'outil instancie les opérateurs de contrôle ainsi que le workflow. Un framework de composition de services basé sur les connaissances (Knowledge-based Semantic Service Composition Framework) est décrit dans [CSG+03] permet d'exploiter les connaissances du domaine pour guider l'utilisateur voulant construire un workflow. L'outil fournit à cet utilisateur des suggestions de sélection et d'instanciation des services et lui permet de mémoriser les workflows pour en faciliter la réutilisation.

données informatiques associées, etc.) [WfM05].

2. http ://www.ibm.com/developerworks/library/specification/ws-bpel/

3. Un e-service est le moyen par lequel une entreprise offre ses produits, ses services et ses ressources via Internet.

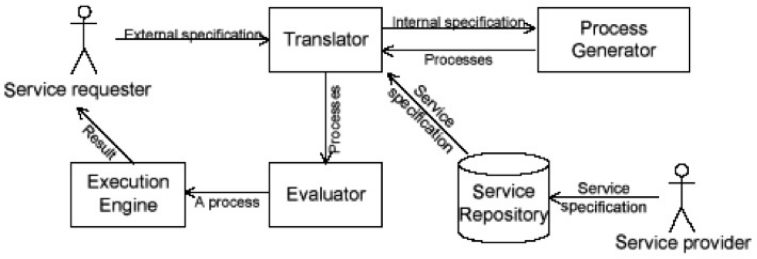


FIGURE 4.3 – Cadre général d'un système de composition automatique de services Web [RS03]

4.1.2.2 Composition automatique

Les approches automatiques de composition de services Web permettent d'automatiser entièrement le processus de composition en le traitant généralement comme un problème de planification [Pee05]. Dans ce cas, le moteur de composition y est toujours dynamique.

Nous présentons un cadre générique de composition automatique de services tel que proposé par Jinghai Rao en 2003 [RS03] puis développés dans [Rao04, RS04]. Il constitue une abstraction de haut niveau sans aucune considération de langage particulier, ou de plate-forme ou méthode particulières utilisées au cours du processus de composition. L'objectif de ce cadre est de constituer le fondement sur lequel sont identifiées les similarités et les différences entre les diverses méthodes de composition disponibles. Le cadre général d'un système de composition de SW est illustré par la figure 4.3.

Le système de composition est constitué de deux composants essentiels : l*'approvisionneur de services* (*service provider*) et le *requêteur de services* (*service requester*). L'approvisionneur de services propose des services Web pour usage en publiant leurs descriptions dans un entrepôt de services. Le requêteur de services consomme les informations et/ou les services offerts par l'approvisionneur de services. En outre, le système comprend les composants suivants : *traducteur*, *générateur de processus*, *évaluateur*, *moteur d'exécution* et *entrepôt de services*. Le *traducteur* traduit les langages externes utilisés par les usagers vers les langages internes utilisés par le générateur de processus. A chaque requête, le *générateur de processus* tente de générer un plan qui compose les services disponibles dans l'entrepôt de services en vue de satisfaire au mieux la requête. Si plus d'un plan est proposé, l'évaluateur compare les plans proposés et décide lequel est le meilleur pour son exécution. Le moteur d'exécution exécute le plan et retourne les résultats au client.

Les méthodes de composition dynamique sont multiples. La plupart de ces approches rencontrées dans l'état de l'art utilisent soit les **techniques d'Intelligence Artificielle de planification**, soit l'**approche déductive utilisée pour la preuve de théorèmes.** L'hypothèse la plus courante sur laquelle reposent ces méthodes est que chaque service est spécifié en ses pré-conditions et ses effets. Ces approches considèrent un service Web comme un composant logiciel qui prend des données en entrée pour produire des données en sortie, de sorte que les pré-conditions et les effets ne sont que les paramètres d'entrées et de sorties du service. Les pré-conditions représentent l'état pré-requis du monde pour l'exécution

du service et les nouveaux états du monde générés suite à l'exécution représentent les effets du service. Ainsi donc les modèles de composition considèrent comme hypothèses la connaissances des spécifications de l'usager relatives aux pré-conditions et aux effets (PE) attendus du service composite. Ces modèles génèrent un plan ou un processus soit par un prouveur logique de théorèmes soit par un planificateur d'Intelligence Artificielle (IA). Dans la prochaine section, nous nous focalisons sur les approches de composition dynamiques de services Web qu'elles soient automatiques et/ou semi-automatiques dans le domaine de l'Intelligence Artificielle.

4.2 Composition de Services Web et les techniques de l'IA

La composition dynamique de services Web par des techniques d'intelligence artificielle (IA), et plus particulièrement par des techniques de planification, a été considérée depuis des années comme la voie la plus prometteuse. En effet, les concepts de OWL-S sont très fortement inspirés de ceux de la planification : les pré-conditions présentent les conditions qui doivent être satisfaites pour garantir la bonne exécution d'un service Web. Les effets sont le résultat du succès de l'exécution d'un service.

Dans ce qui suit, nous présentons quelques axes de recherche actuels concernant la composition par planification et par d'autres techniques d'intelligence artificielle. C'est ainsi que les méthodes de composition de services Web que nous allons présenter seront classées en cinq classes, comme suit :

1. Composition de Services à base de logique (calcul situationnel : situation calculus),
2. Composition de Services à base de planification avec PDDLP,
3. Composition de Services à base de règles,
4. Composition de Services à base de planification hiérarchique,
5. Composition de Services à base de la synthèse de programmes
6. Composition orientée QdS (en anglais QoS).

4.2.1 Composition à base de la logique de premier ordre ou du calcul situationnel (Situation Calculus)

McILRAITH et. al. [MSZ01, NM02] adaptent et étendent le langage Golog [4][LRL+97] pour la composition automatique de services Web. Les auteurs de [MSZ01, NM02] utilisent une version étendue de Golog intégrant la sémantique des actions pour résoudre les

4. Le langage GOLOG (alGOl in LOGic) a été proposé en 1997 par l'équipe de Hector LEVESQUE à l'Université de Toronto (Canada) comme langage de programmation logique pour des domaines dynamiques basé sur le calcul de situation qui permet de raisonner sur le monde et sur ses évolutions (tous les mondes pouvant résulter d'une ou de plusieurs actions de l'agent). Les programmes en GOLOG spécifiant le comportement d'un agent peuvent être écrits avec un niveau élevé d'abstraction : les actions complexes d'un agent se décomposent en actions primitives qui font référence normalement à des actions externes aux agents comme par exemple ramasser un objet, par opposition à des actions qui n'affectent que leur état interne. Le modèle de l'environnement est mis à jour dynamiquement en fonction des règles définies par le programmeur.

problèmes de composition automatique de services Web. L'idée générale de cette méthode est que les agents logiciels peuvent raisonner sur les services Web en vue de réaliser automatiquement les opérations de découverte, d'exécution et de composition. La procédure générique ainsi que les contraintes peuvent être formulées dans une logique du premier ordre (LPO) du calcul situationnel, i. e. une logique permettant de raisonner à propos des actions et des changements. Les auteurs, à cet égard, conçoivent les services Web soit comme des actions primitives (PrimitiveAction) soit comme des actions complexes (ComplexAction). Les actions primitives sont conçues soit comme des actions modifiant l'état du monde, soit comme des actions modifiant l'état de connaissance de l'agent Les actions complexes sont des combinaisons d'actions primitives.

La connaissance de base de l'agent permet d'encoder logiquement les pré-conditions et les effets du service Web dans un langage de calcul situationnel. Les agents utilisent les constructeurs d'un langage de programmation procédurale avec des concepts définis pour les services et des contraintes utilisant une machine déductive. Un service composite est un ensemble de service atomiques reliés par des constructeurs d'un langage de programmation procédurale du genre : (if-then-else, while et so forth). Les auteurs ont, aussi, proposé un moyen de personnaliser les programmes en Golog en incorporant les contraintes formulées par le requêteur, de sorte qu'il puisse soit effectuer un choix non déterministe dans une situation donnée soit renforcer un ordre d'exécution entre deux actions. La génération du plan de composition est effectuée en respectant les contraintes prédéfinies.

4.2.2 Composition à base du langage PDDL (Planning Domain Definition Language)

Le grand intérêt accordé par la communauté de la composition de services aux méthodes de planification IA peut être expliqué par la grande similarité entre le langage de description OWL-S (et son prédesseur DAML-S) et les représentations de PDDL [5] (Planning Domain Definition Language) [MGK+98] qui est considéré comme un standard de fait de l'état de l'art des planificateurs. En effet, la communauté du Web sémantique s'est fortement inspirée du langage PDDL lors de la conception du langage de description OWL-S. Ainsi une description OWL-S peut être facilement traduite en une représentation PDDL : les services Web sont représentés par des opérateurs ayant des pré-conditions et produisant des effets. Puis un planificateur peut alors être utilisé pour produire un plan qui correspond à la composition des services Web. Pour disposer d'un état de l'art complet de la composition dynamique par planification, nous pouvons nous reporter aux travaux de J. PEER publiés en 2005 [Pee05].

De nombreux travaux ont été produits ces dernières années. Les travaux de **SHESHAGIRI et al.** [SdF03] proposent l'utilisation d'ontologies de domaine afin d'ajouter des contraintes pour optimiser la recherche dans un espace de plans. Ceux de **McDERMOTT** [Mcd02] proposent d'utiliser la planification par régression estimée (Estimated-Regression

5. Le formalisme de planification classique étant très restreint, des extensions ont été nécessaires afin de décrire des domaines plus complexes. Ces principales extensions sont le typage des variables, les opérateurs de planification conditionnelle, les expressions de quantification, les pré-conditions disjonctives ou encore les axiomes d'inférences. Le langage de planification PDDL (Planning Domain Description Language) permet d'exprimer ces différentes extensions.

Planning) qui permet d'utiliser des heuristiques pour guider la recherche dans un espace de situations. Le principe est le suivant : un agent qui désire interagir avec un service Web va construire un plan puis l'exécuter. Si l'exécution échoue, l'agent aura appris de nouvelles informations de cette interaction, qui seront utilisées dans le prochain cycle de planification et exécution.

SIRIN et al. [SHP03] présentent une méthode semi-automatique pour la composition de services Web. A chaque fois qu'un utilisateur doit sélectionner un service Web, tous les services disponibles qui correspondent au service sélectionné, sont présentés à l'utilisateur qui fait alors le choix à la place du système. L'idée de la composition semi-automatique est assez intéressante car il est très difficile de capturer le comportement des services Web dans leurs moindres détails, puis de les composer automatiquement. Bien que cette méthode soit simple, elle permet de voir comment un planificateur automatique et un humain peuvent travailler ensemble pour répondre au mieux à la requête de l'utilisateur.

Enfin, **PEER** [Pee04] propose une approche basée sur le langage PDDL dans laquelle, après la création du domaine de planification à partir de la description sémantique, un planificateur est choisi parmi plusieurs en fonction des instructions PDDL utilisées dans le domaine ou de la complexité du but à atteindre. Dans la composition de services à base du langage PDDL, **McDERMOTT** [Mcd02] a introduit un nouveau type de connaissance appelée valeur d'une action (value of an action), qui est persistante et qui n'est pas traitée comme une valeur littérale. Du point de vue de la construction d'un service Web, cette caractéristique permet de distinguer la transformation de l'information de l'état de changement provoqué par l'exécution du service. L'information est représentée par les paramètres d'entrée/sortie lesquels sont réutilisables dans l'exécution de plusieurs services. Au contraire, les états du monde ne sont modifiés que par l'exécution du service. Ainsi le changement peut être interprété comme la consommation des anciens états du monde pour en produire de nouveaux.

4.2.2.1 Composition avec une planification à base de règles (Rule-based Planning)

MEDJAHED et al. [MBE03] présentent une technique pour générer des services composites partant du haut niveau jusqu'au niveau de la description déclarative. La méthode utilise les règles de composition afin de déterminer si deux services sont composables ou non. Cette approche de composition consiste en quatre phases :

1. La phase de spécification permet une description de haut niveau des compositions désirées par l'usage du langage CSSL pour (Composite Service Specification Language).
2. La phase de matching (makematcher) qui utilise les règles de composition afin de générer des plans de composition conformes aux spécifications du requêteur.
3. Si plus d'un plan est généré durant la phase de sélection, le service requêteur choisit un plan basé sur les critères de qualité de composition (Quality of Composition (QoC) comme le rang, le coût etc..).
4. La phase finale est une phase de génération. Une description détaillée du service composite est générée automatiquement et présentée au service requêteur.

Les règles de composabilité constituent le coeur de l'opération de composition, en ce sens qu'elles définissent comment le plan doit être généré. Ces règles considèrent à la fois les propriétés syntactiques et celles sémantiques des services Web. Les règles syntactiques couvrent les règles des modes opératoires ainsi que les règles d'interaction des services à travers les liaisons protocolaires (binding protocols). Les règles sémantiques incluent ce sous-ensemble de règles :

1. La composabilité indique si deux services Web sont composables et ce, si le message de sortie d'un service est compatible avec le message d'entrée d'un autre ; nous proposons un extension de la notion de composabilité dans le chapitre 7.
2. l'opération de la composabilité sémantique définit la compatibilité entre le domaine, les catégories et les objectifs de deux services ;
3. La composabilité qualitative définit les préférences du requêteur en matière de qualité des opérations du service composite ;

La principale contribution de cette approche est la notion de règles de composabilité qui considèrent les propriétés syntactiques et sémantiques des services Web. Les règles syntactiques incluent des règles pour les types d'opérations possibles et pour les liaisons protocolaires entre les services (i.e., bindings). Les règles sémantiques incluent des règles concernant la compatibilité des messages échangés et la compatibilité des domaines sémantiques des services, mais également des règles de qualité de la composition. Cette notion de règles de composabilité met en avant les attributs possibles des services Web qui peuvent être utilisés dans la composition de services et peut ainsi jouer le rôle de directive pour d'autres méthodes de composition par planification.

Un autre outil de composition de services Web générant des plans à base de règles est **SWORD** [PF02]. SWORD n'utilise pas les standards des services Web comme WSDL ou DAML-S, il utilise le modèle entité-relation pour spécifier les services Web. Il modélise un service Web par ses pré- et post-conditions. Plus précisément, SWORD représente un service sous la forme d'une règle de HORN pour dénoter les post-conditions que le composeur doit réaliser. L'outil SWORD automatise la tâche de composition en utilisant des descriptions de services basées sur les règles. L'utilisateur spécifie les faits des états initial et final. Le planificateur tente alors de construire une chaîne de services pouvant satisfaire ces besoins.

4.2.3 Planification à base de réseaux HTN

D'autres planificateurs IA ont été proposés pour la composition automatique de services Web. Le planificateur SHOP2 [WPS+03, WSH+03] a été proposé par **WU et al.** en utilisant des descriptions de services de type DAML-S. SHOP2 est un planificateur à base d'un réseau hiérarchique de tâches [6] (*Hierarchical Task Network* (HTN). Les auteurs

6. La planification hiérarchique (Hierarchical Task Network, HTN) utilise un langage similaire au langage de la planification classique. En effet, chaque état du monde est représenté par un ensemble de prédicats instanciés et chaque action correspond à une transition déterministe entre deux états. Cependant la planification hiérarchique diffère de la planification classique sur la façon de planifier [NIK+03]. Dans un planificateur HTN, l'objectif n'est pas d'atteindre un ensemble de buts (exemple : porte ouverte) mais de résoudre un ensemble de tâches (exemple : ouvrir porte). Les entrées du planificateur sont, comme en

affirment que le planificateur hiérarchique HTN est plus efficace que d'autres langages de planification comme Golog qui a été précédemment cité.

D'après les auteurs, le principe de décomposition d'une tâche en sous-tâches dans la planification hiérarchique est très similaire au concept de décomposition de processus composites dans OWL-S. Afin de rendre leur proposition réalisable, les auteurs font deux hypothèses concernant le modèle de processus :

1. Les processus atomiques ont, soit des sorties, soit des effets, mais pas les deux en même temps.
2. Les structures de contrôle Split et Split et Join ne sont pas utilisées dans les processus composites

En 2002, **SIRIN et al.** ont présenté une première méthode de composition semi-automatique. En 2004, **SIRIN et PARSIA** [SP04] sont allés plus loin en intégrant un raisonneur sur les langages de descriptions dans le planificateur SHOP2. La planification HTN est très puissante pour les domaines où la connaissance est complète et détaillée, ce qui n'est pas le cas des services Web selon [KG05].

4.2.4 Composition de Services à base de la synthèse de programmes

La synthèse de programmes est une méthode de génie logiciel qui génère automatiquement des programmes. Il existe trois différentes approches pour la synthèse de programmes :

- transformationnelle,
- inductive et
- déductive.

Nous nous focalisons sur la facette déductive de la synthèse qui est basée sur l'observation que les preuves sont équivalentes au programme en raison du fait que chaque preuve peut être interprétée comme un pas de calcul. L'idée centrale est la transformation de la synthèse de programmes en une approche de preuve de théorèmes. L'idée d'une homologie entre les théorèmes et leurs spécifications, d'une part et les preuves et programmes constructifs, d'autre part a été détaillée dans le travail de de **MANNA et WALDINGER** datant de 1992 [MW92]. Tenant compte de certaines analogies, il est possible de voir les méthodes de composition de services Web utilisant des programmes de déduction de la synthèse de programmes comme des cas particuliers de l'usage de l'approche planification de l'IA. Mais la génération du processus de composition est fortement reliée aux théories de la preuve. Le processus général de composition à base de synthèse de programmes est schématisé par la figure 4.4.

Le processus général de composition de services Web utilisant la synthèse de programmes est décrit comme suit : premièrement, les descriptions des services Web existants sont traduites en axiomes logiques et les exigences en matière de composition spécifiées

planification classique, un ensemble d'opérateurs, mais également, un ensemble de méthodes. Une méthode décrit la manière de décomposer une tâche en un ensemble de sous-tâches. Le principe de la planification HTN est de décomposer les tâches non primitives en sous-tâches de plus en plus petites, jusqu'à obtenir des tâches primitives qui peuvent être résolues en utilisant les opérateurs de planification.

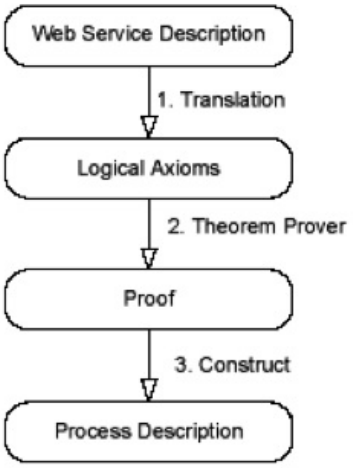


FIGURE 4.4 – Processus général de composition à base de synthèse de programmes

sous la forme de séquents logiques à prouver. Deuxièmement, un prouveur de théorèmes est utilisé pour déterminer quels séquents peuvent être prouvés à l'aide des axiomes disponibles. Si la réponse est positive, la dernière étape consiste alors à construire la description du processus de composition à partir de la preuve complète.

WALDINGER dans [Wal01] élabore une idée selon laquelle la synthèse de services peut être réalisée comme une preuve de théorème. L'approche est basée sur la déduction automatique et la synthèse de programmes dont les bases théoriques ont été jetées dans un travail antérieur présenté dans [MW92].

Initialement les services disponibles et les exigences de l'usager sont décrits dans une logique du premier ordre en relation avec la logique classique avant que les preuves constructives soient générées par le prouveur de théorèmes pour que finalement, les descriptions du service composite soient extraites d'un certain nombre de preuves particulières. **LÄMMERMANN** [Lam02] utilise le programme de synthèse structurale (Structural Synthesis of Program SSP) pour automatiser la composition de services. SSP est une approche déductive pour la synthèse des programmes à partir de spécifications. Les spécifications des services considèrent les propriétés structurales, i.e. l'information entrée/sortie. SSP utilise des variables propositionnelles comme identifieurs pour les paramètres d'entrée/sortie et une logique propositionnelle intuitionniste pour résoudre le problème de composition.

La composition est basée sur les propriétés des programmes de preuve de la logique intuitionniste ce qui permet de transformer un programme de composition de services en un programme de recherche de preuve. L'auteur utilise aussi l'avantage des disjonctions de la logique classique pour décrire les exceptions qui peuvent être rejetées durant l'invocation du service.

4.3 Approches de composition orientées Qualité de Service

Le concept de QdS (Qualité de service) recouvre un large domaine de propriétés inhérentes au domaine de la transmission de données. Plusieurs travaux pour définir la QdS sont à l'origine d'une proposition de standardisation de l'ISO.

La référence normative est le « *Quality of Service Framework* » publiée en 1998 par l'ISO dans [ISO98]. D'après cette référence, les concepts fondamentaux suivants structurent la QdS :

1. Caractéristiques de QdS : un aspect quantifiable de QdS, qui est défini indépendamment des moyens par lesquels il est représenté ou contrôlé.
2. Etablissement de la QdS : l'utilisation de mécanismes pour créer les conditions pour l'activation d'un système afin que les caractéristiques de QdS désirées soient atteintes.
3. Mécanisme de QdS : un mécanisme spécifique, pouvant utiliser des paramètres de QdS ou des informations de contexte, potentiellement en coordination avec d'autres mécanismes de QdS, dans le but de permettre l'établissement, le monitoring, la maintenance, le contrôle et la demande de QdS.

Dans la suite de cette section, nous étudions successivement ces concepts dans le cadre des services Web.

4.3.1 Caractéristiques de QdS des Services Web

Dans le cadre des services Web, le W3C a identifié un ensemble de caractéristiques de QdS pertinentes pour le domaine des services Web [W3C03] :

- **La performance** représente la vitesse avec laquelle un service Web répond à une requête. Cette qualité peut être mesurée en termes de débit, temps de réponse, latence, temps d'exécution, temps de transaction, etc.
- **La confiance** qualité d'un service à exécuter certaines fonctions sous certaines conditions et dans un intervalle de temps donné. Il s'agit d'une mesure de la capacité d'un service à maintenir sa qualité (généralement le nombre de coupures de service par jour, semaine, mois). La confiance est également associée à la garantie de l'ordre et de la bonne livraison des messages
- **Le passage à l'échelle (scalability)** permet de quantifier le nombre de requêtes auxquelles le service peut faire face dans un intervalle de temps donné.
- La **Capacité** indique le nombre de requêtes que le service est capable de traiter simultanément.
- **La robustesse** est la capacité à fonctionner alors que les données en entrée provoquent des conflits ou sont incomplètes.
- **Le traitement des exceptions** : de nombreuse situations ne peuvent être prédites à l'avance ce qui implique de supporter les exceptions de façon adaptée.
- La **Précision** indique le taux d'erreurs générées par le service.
- L'**intégrité** est une propriété garantissant que l'intégrité des données et des transactions est bien respectée, afin de ne pas aboutir à une situation inconsistante.
- L'**accessibilité** est la capacité à répondre à des requêtes.

- La **disponibilité** du service devrait être prêt pour une consommation immédiate lors d'une invocation. La disponibilité est souvent associée au temps de réparation (ou TTR pour Time to Repair).
- L'**interopérabilité** est capacité à pouvoir communiquer quels que soient la plateforme et les langages utilisés. En ce qui concerne les services Web, c'est la normalisation et l'extensibilité de SOAP et des métadonnées qu'il contient qui permet de garantir l'interopérabilité.
- La **sécurité** des communications entre services Web (Authentification, , Confidentialité, Intégrité, Non répudiation). De multiples normes sont à mettre en relation avec la sécurité, dont principalement WS-Security, WS-Trust, WS-SecureConv, WS-Federation, Contrats de service

Etablir et s'assurer de la QdS d'un composant comme le service Web constitue un enjeu crucial puisque ceci permet d'établir une relation de confiance entre le fournisseur d'un service et un client en attente d'une certaine fiabilité. Cependant, contrairement aux spécifications bien établies dans le domaine fonctionnel des services Web (telles que WSDL, SOAP ou UDDI), il n'existe pas de standards officiellement reconnus par la communauté en ce qui concerne la description et l'établissement de propriétés de QdS. Pour autant, plusieurs travaux visent à apporter des réponses à cette lacune par le biais des contrats de service (en anglais : SLA pour Service Level Agreement) dont l'objectif est de permettre la réalisation d'un accord entre consommateur et fournisseur de service portant sur le niveau de QdS que doit fournir le service. Plus spécifiquement, un SLA consiste en un document généré à l'issue d'un protocole de négociation (plus ou moins complexe) entre le consommateur et le fournisseur de service, et qui permet de définir les offres et exigences de ces deux rôles. Les spécifications de ce document doivent être vérifiées à l'exécution via un mécanisme de monitoring. Le cycle de vie du SLA prévoit ensuite différentes possibilités : soit une spécification du SLA est mise en défaut (violation d'un terme) et un mécanisme correctif (prévu ou non dans le SLA) doit être mis en oeuvre, soit le SLA arrive à son terme (dans le cas où un intervalle temporel est spécifié), soit une partie (consommateur ou fournisseur de service) décide de mettre fin à la relation client-service.

4.3.2 Framework à base de broker

Nous allons considérer comme référence aux frameworks basés sur un broker (Broker based Framework) le travail de YU Tao et LIN Kwei-Jay à l'Université de Californie Irvine (UCI) développé dans deux papiers publiés en 2005 [YL05a, YL05b], même si des travaux antérieurs avaient initiés de tels frameworks à l'instar de [PSL03]. Dans ce travail, les deux auteurs présentent une architecture à base d'un broker appelée *Qbroker* pour l'intégration et l'adaptation dynamique de services Web. A l'instar d'autres approches, ils proposent une modélisation de la QdS ainsi que des algorithmes de sélection de services basés sur les caractéristiques globales de la composition. Le broker exploite les informations contenues dans les contrats SLA des services et analyse les rapports de la QdS du service à l'exécution afin de vérifier si un contrat SLA est violé ou non. Dans ce cas, le broker est en charge de trouver un autre service qui satisfait la QdS globale. L'algorithme de sélection ne peut supporter qu'une seule contrainte de QdS et un seul point d'échec. La figure 4.5 illustre l'architecture globale du framework.

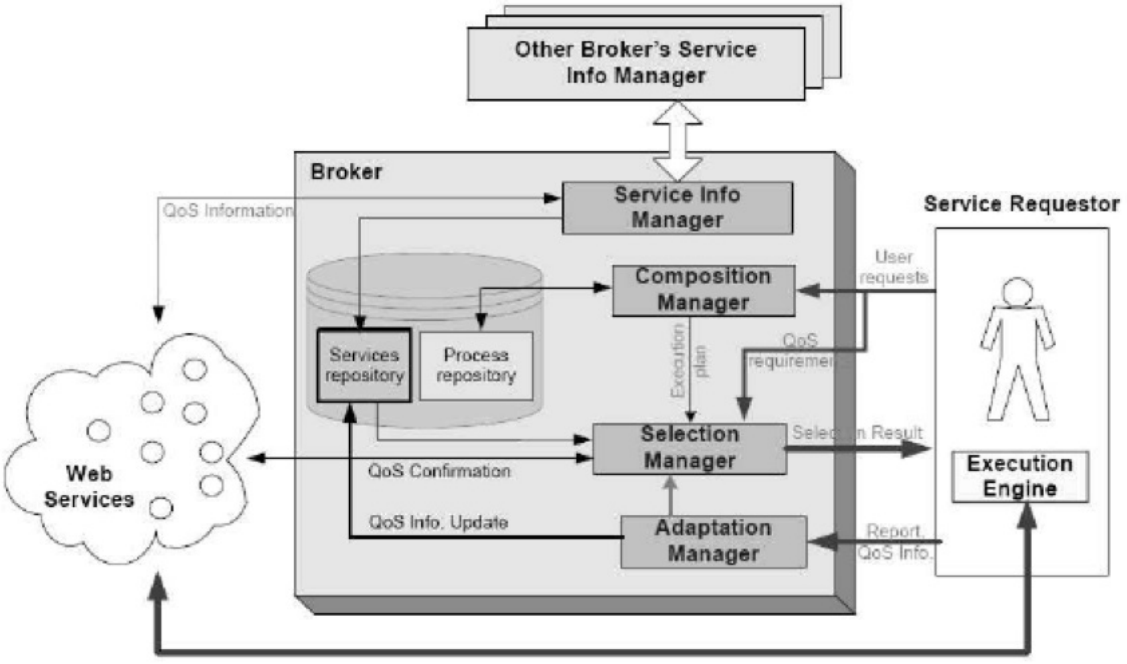


FIGURE 4.5 – Architecture globale du framework à base de broker

Dans cette architecture, la motivation principale pour la conception du broker est d'intégrer la logique de sélection des services. Ainsi, la logique de composition et celle de demandeur de service sont clairement séparées. Les brokers sont ensuite vus comme des services maintenus de façon autonome et gérés par un acteur indépendant (à la manière d'un intermédiaire comme une agence de voyage). Par la suite les brokers peuvent être amenés à travailler ensemble pour répondre à la requête d'un client. Après chaque transaction avec le client, l'utilisateur du service doit rapporter les performances de QdS du service pour que le broker puisse mettre à jour les données de QdS de chaque service. L'architecture du broker, exhibée dans la figure ci-dessus, est constituée de quatre composants :

1. **Le gestionnaire d'information** (*Service Information Manager*) garde en mémoire un ensemble de services Web candidats dans un annuaire constitué de deux tables (pour les informations fournies par le service et les données statistiques relatives à la QdS). La table service permet de stocker la description fonctionnelle des services (identifiant, URL, opération, données d'entrée/sortie, description) et les données du SLA (capacité, temps de réponse, disponibilité, fiabilité). La table des données statistiques enregistre les données transmises par les clients ayant utilisé les services (stabilité du service, disponibilité, etc.). En fonction de ces informations statistiques agrégées, les valeurs de QdS des services sont mises à jour dans la table service.
2. **Le gestionnaire de composition** (*Composition Manager*) maintient le registre de processus qui contient les plans de processus. Ces plans de processus correspondent aux services et à leurs connexions, formant un chemin d'exécution. Ce composant traite également les requêtes de recherche de services pour générer un plan d'exécution. Les plans d'exécution consistent en des processus abstraits. Ils sont formés à partir des plans de processus choisis par un utilisateur.
3. **Le gestionnaire de sélection** (*Selection Manager*) retrouve les services pour le plan d'exécution généré par le gestionnaire de composition (Composition Manager). La sélection s'effectue sur la base des exigences de QdS de l'utilisateur. L'algorithme utilise une technique de résolution par contraintes sur le graphe de la composition afin de vérifier que la QdS globale (après agrégation) respecte bien les contraintes exigées par l'utilisateur. Pour optimiser la QdS globale obtenue, une fonction-objectif (appelée aussi fonction d'utilité est définie. Cette fonction consiste en une somme pondérée de paramètres de QdS. Plus sa valeur est élevée, plus la QdS globale est élevée. Des algorithmes d'optimisation sont proposés (comme nous l'expliciterons plus loin) pour le problème de résolution par contraintes de sorte à générer un ensemble de chemins satisfaisants où un épisode de chemin peut comporter un opérateur OR-SPLIT portant sur deux services équivalents l'un pouvant servir de secours dans le cas où l'autre n'est pas accessible.
4. **Le gestionnaire d'adaptation** (*Adaptation Manager*) : Le résultat de la sélection étant renvoyé au moteur d'exécution de la composition, la composition peut être instanciée. Dans ce modèle d'architecture, c'est au moteur d'exécution d'observer la QdS des services impliqués dans la composition. Si les SLA ne sont pas violés, alors la QdS des services est renvoyée à l'Adaptation Manager. Au cas où un SLA est violé, alors le gestionnaire d'adaptation (Adaptation Manager) doit renvoyer un des nouveaux chemins d'exécution pré-calculés au moteur d'exécution et demande au gestionnaire de sélection (Selection Manager) de recalculer un ensemble de plans

Propriété QoS	*Sequence*	*And*	XOR	Loop
le temps de réponse	$\sum_{i=1}^{n} rt_i$	$max(rt_i)$	$max(rt_i)$	$rt \times k$
Le débit	$\min(th_i)$	$\min(th_i)$	$\min(th_i)$	th
La disponibilité	$\prod_{i=1}^{n} av_i$	$\prod_{i=1}^{n} av_i$	$\min(av_i)$	av^k
La fiabilité	$\prod_{i=1}^{n} re_i$	$\prod_{i=1}^{n} re_i$	$\min(\mathrm{re}_i)$	re^k

TABLE 4.1 – Agrégation des paramètres QoS dans un patron de composition

de rechange. La table des statistiques est également mise à jour pour tenir compte de la violation du SLA.

Bien que ce travail ait été limité sur de nombreux points (une seule propriété peut être contrainte globalement et une seule erreur est traitée à la fois), il propose une séparation très nette entre les rôles de concepteur de composition et de sélecteur de services. Le broker et le moteur BPEL (dans le cas des processus d'affaires) sont clairement découplés et aucune donnée confidentielle ne transite par le broker puisque celui-ci n'agit pas comme un proxy, contrairement à la plupart des architectures des plateformes concurrentes. Ceci permet au gestionnaire du broker de servir de multiples clients et même de travailler en partenariat avec d'autres gestionnaires de broker. Par ailleurs, ce travail réutilise certaines informations des SLA et maintient à jour une base de données statistiques portant sur la QdS des services, ce qui apporte plus de fiabilité qu'une simple relation client-fournisseur de service. Enfin, dans des conditions idéales, l'adaptation **dynamique** ne coûte rien puisque les chemins de rechange sont préparés à l'avance et donc il n'y a pas de surcharge lorsqu'un service viole son SLA. En revanche, du fait de l'indépendance des rôles, le broker n'a pas de perception de la QdS à l'exécution et c'est donc au moteur BPEL d'implémenter la logique de monitoring et de surveillance de la disponibilité. Il est donc de la responsabilité du client (la composition de services) de s'assurer de la QdS fournie. Cela nécessite en outre de modifier le moteur BPEL et donc de le rendre spécifique à l'utilisation du broker. Le point qui nous intéresse dans ce framework est celui relatif à la sélection de services atomiques en vue de constituer un ensemble de chemins possibles constituant des services composites vérifiant les exigences de QdS de bout en bout. Mais avant d'étudier les algorithmes et les heuristiques de sélection, nous allons préciser les patrons de composition possibles pour générer les graphes de composition, les classes de caractéristiques et attributs de la QdS et la constitution de la fonction d'utilité. Le passage des requêtes de l'usager à la sélection de services QdS au service composite complexe au niveau de **Qbroker** est illustré dans la figure 4.6.

Les auteurs considèrent les six patrons de composition (le patron *sequantiel*, l'opérateur *ANDSplit* et *ANDJoin*, l'opérateur *XOR Split* et *XORJoin*) et l'opérateur *Loop*) qui sont illustrés dans la figure 4.7. L'agrégation des attributs de la QdS pour quatre des six classes de patrons susmentionnés est explicitée dans la table 4.1. Nous allons étudier sur un exemple illustré dans la figure 4.8, la composition d'un service avec les patrons susmentionnés.

Considérons le service composite représenté par la figure 4.8 où le service S_2 est suivi soit par S_4 soit par S_5 avec, respectivement, les probabilités p_1 et p_2. Le service S_6 peut être exécuté au plus K fois. Supposons que t_i et c_i représentent, respectivement, le temps de réponse et le coût du service atomique S_i, le réponse totale T et le coût total C du

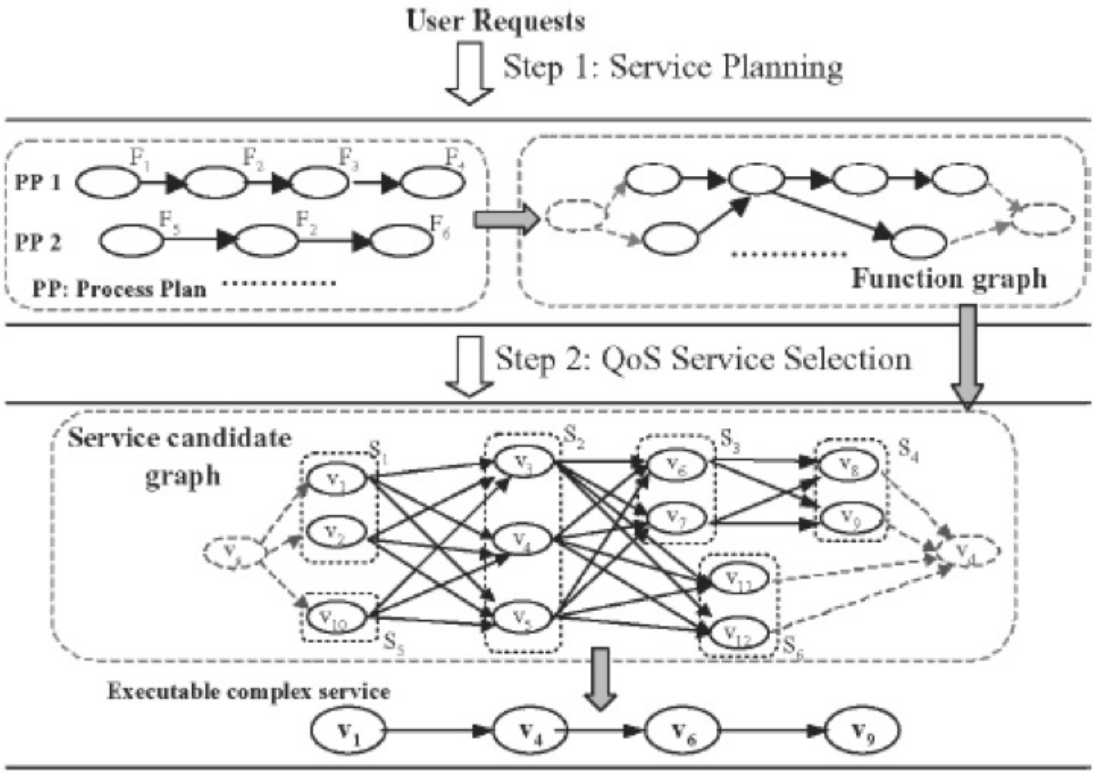


FIGURE 4.6 – Composition orientée QdS de services Web

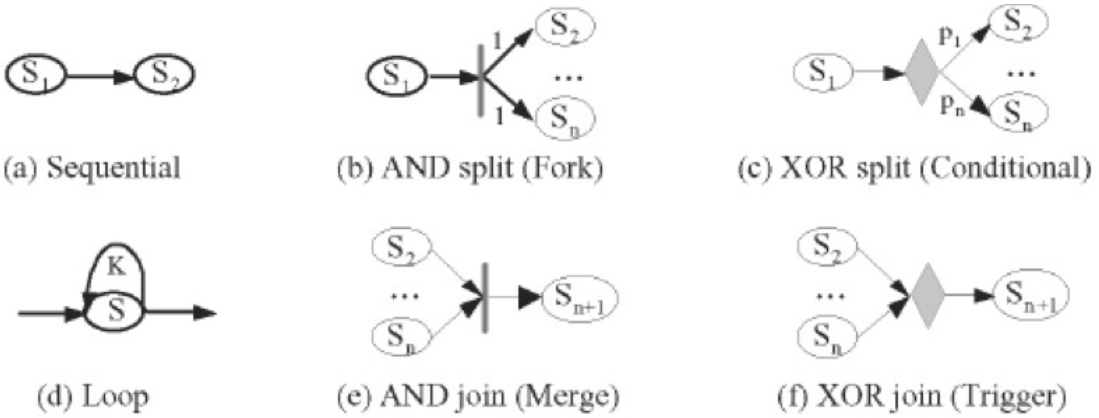


FIGURE 4.7 – Les six patrons de composition

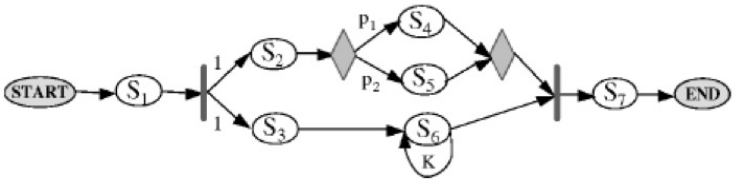


FIGURE 4.8 – Un exemple de service Composite

service composite sont données par les formules ci-après :

$$\mathrm{T} \;=\; t_1 + \max(t_2 + p_1 \times t_4 + p_2 \times t_5,\, t_3 + \mathrm{K} \times t_6) + t_7$$

$$\mathrm{C} \;=\; \mathrm{c} + c_2 + p_1 \times c_4 + p_2 \times c_5 + c_3 + K \times c_6 + c_7.$$

En général la valeur des paramètres QdS pour les critères Qds de chaque service atomique est soit une valeur moyenne (q_{avg}) soit une valeur du pire des cas (q_{max}). Afin de poser le problème de la sélection, nous allons adopter les notations suivantes. Nous supposons que l'on dispose d'un univers de services Web noté $\mathbb{S}$ défini comme l'union d'un nombre de classes abstraites de services : $\mathbb{S} = \cup_i \mathrm{S}_i$, i = 1,...,n.

Chaque classe abstraite de services S_i (comme la classe des agences de voyage, celle des compagnies aériennes, celle de l'alignement multiple de séquences, celle de la construction d'arbres phylogénétiques) peut être utilisée pour décrire un ensemble de services fonctionnellement équivalents comme par exemple *Agence Cook*, *Tunisair*, *BLAST*, *ClustalW*, *Phylip*, etc. Un service atomique de la classe S_i est noté s_{ij} et décrit par un vecteur d'attributs QdS ; le vecteur des m attributs QdS exigés pour un service composite est noté Q_c et s'écrit comme suit : $Q_c = [Q_{1c}... \; Q_{mc}]$.

A chaque service candidat v à la composition est affectée une fonction d'utilité notée $\mathcal{F}_v$ qui est définie comme un ensemble de paramètres QdS. La finalité de la fonction d'utilité est de choisir les services atomiques qui vérifient les contraintes mais qui fournissent la meilleure valeur à la valeur de la fonction d'utilité en réponse aux exigences en la matière de l'usager – client. Dans le cas où nous disposons de x attributs '*positifs*' (ceux dont la maximisation est recherchée comme le critère de la disponibilité) et y attributs '*négatifs*' (ceux dont la minimisation est recherchée comme le critère coût) les W_α et W_β sont les poids de chaque attribut QdS de deux classes d'attributs positifs et négatifs avec la somme de tous les poids est égale à 1. La fonction d'utilité d'un service s'écrit donc sous la forme suivante :

$$\mathcal{F}_{ij} = \sum_{\alpha=1}^{x} w_\alpha * \left(\frac{q_{ij}^{\alpha} - \mu^{\alpha}}{\sigma^{\alpha}}\right) + \sum_{\beta=1}^{y} w_\beta * \left(1 - \frac{q_{ij}^{\beta} - \mu^{\beta}}{\sigma^{\beta}}\right)$$

w_α et w_β désignent les poids des attributs QdS ($0 < w_\alpha$ et $w_\beta < 1$), $\sum_{\alpha=1}^{x} w_\alpha + \sum_{\beta=1}^{y} w_\beta = 1$. μ et σ désignent respectivement l'écart type et la variance des valeurs des attributs QdS pour chaque classe de services.

4.3.2.1 Les contraintes QdS globales

Les contraintes QdS représentent les exigences à vérifier par le service composite en termes de QdS de bout en bout. Cette vérification peut être exprimée en termes de maximisation ou de minimisation d'une fonction d'utilité. Pour la première classe les attributs positifs seront gardés tels quels alors que les attributs négatifs doivent être multipliés par -1. On procède inversement dans le cas d'une minimisation, les attributs positifs devant être multipliés par -1. Soit un service composite abstrait noté SC_{abs} défini par une requête de composition donnée et soit $C^{'} = [c'_1 \ldots c'_m]$ où m est le nombre d'attributs choisis dans la requête parmi les r solutions possibles, un vecteur de contraintes globales sur le service composite SC_{abs}.

Soit SC une instanciation de SC_{abs} dans laquelle un service Web concret est choisi comme candidat à la composition dans chaque classe de services S_i de l'univers $\mathbb{S}$. On dira que la sélection SC est admissible ssi :

$$\text{q'(SC)} \leq \text{C' pour} \quad \forall\, c'_k \in C^{'}\ k = 1,...,m.$$

Autrement dit, un SC est admissible si toutes les contraintes globales sont satisfaites.

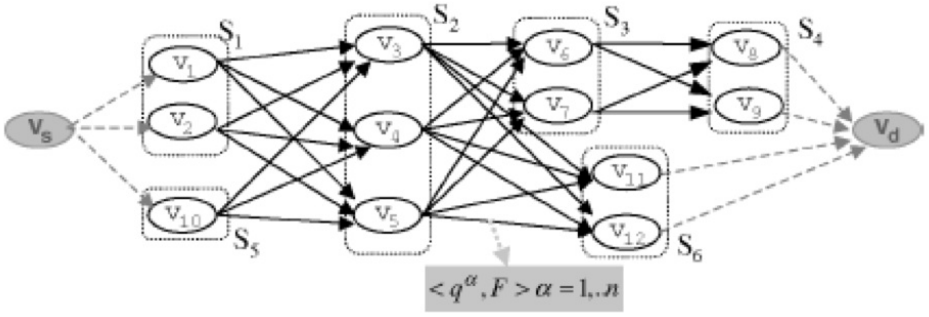
Une fonction d'utilité globale est considérée en vue de permettre de trier et de classer les services candidats. Une sélection est dite *optimale* (nous dirons plutôt *la plus satisfaisante*) pour une requête de composition abstraite de services Web atomiques est un vecteur de contraintes QdS globales si elle maximise (resp. minimise) la valeur de l'utilité globale $\mathcal{F}^{'}\ (SC_{opt})$ = maxi (resp. min) $\mathcal{F}(SC_i)$. Pour une requête de composition de n classes de services et l services atomiques par classe il y a l^n combinaisons possibles à examiner pour identifier le service composite le plus satisfaisant. C'est donc un problème d'optimisation qui est **NP-complet.**

4.3.3 Algorithmes de sélection de composition de services avec des contraintes QdS

Plusieurs algorithmes ont été proposés dans la littérature pour la sélection de composition de services avec différentes structures de composition et diverses contraintes de QdS. Une taxonomie de ces solutions a été produite dont la première classe de ces approches vise à déterminer une composition optimale (i. e. composition à QdS élevée) utilisant des algorithmes de force brute comme le GP (*Global Planning*) dans [ZBN+04] ou **BBLP**, **MCSP**, **WS-IP** dans [YZL07]. Dans [YZL07], deux classes de problèmes de sélection selon que le service composite est constitué d'une suite séquentielle de services atomiques ou qu'il est constitué par une suite de services atomiques reliés par un ou plusieurs patrons de composition parmi les six susmentionnés. Pour la classe de composition séquentielle, l'algorithme proposant des solutions exactes est le **MMKP** (pour **M**ultidimension **M**ultichoice **K**napsack **P**roblem) qui se base sur la méthode de résolution du problème du sac à dos multi-choix multicritère dans [0, 1]. Ce problème est réputé faire partie des problèmes NP-complets. La majorité des algorithmes pour identifier les solutions optimales pour le problème MMKP sont basés sur la méthode *Branch-and-Bound* qui est une méthode de programmation linéaire. **KHAN** [KKD+98] a proposé un tel algorithme BBLP pour MMKP qui cherche la solution optimale itérativement.

Le temps de calcul de **BBLP** croissant exponentiellement avec la taille du problème, ainsi, pour des problèmes de sélection il n'est pas du tout recommandé pour la sélection des services Web composites. YU et al. ont proposé en 2007 une heuristique dénommée **WS-HEU** (heuristique HEU pour les services Web) en vue de trouver une solution sous-optimale au problème MMKP [YZL07]. L'heuristique WS-HEU est, en fait, une adaptation de l'heuristique HEU proposée en 2001 par S. KHAN et al. [AMSK01]. Les tests rapportés dans [YZL07] indiquent que la meilleure solution est trouvée par WS-HEU dès le premier test dans 98% des cas en consommant la moitié du temps de calcul de l'heuristique HEU. Pour n classes de services avec l candidats à la composition dans chaque classe et m contraintes QdS, la complexité temporelle de WS-HEU est O(n^2 $(l-1)^2$ m) qui est donc polynomiale (tout comme HEU).

La seconde classe de composition concerne les structures générales de workflows (disposant de tâches séquentielles, conditionnelles et parallèles) modélisées sous la forme d'un graphe acyclique direct (DAG). Une fois le graphe de composition obtenu entre le service source v_s et le service destination v_d comme le montre la figure suivante.



Chaque noeud étant affecté d'un ensemble de valeurs d'attributs QdS, et une fonction d'utilité atomique. Le problème consiste à identifier le chemin optimisant la fonction utilité globale et respectant les attributs globaux de QdS spécifiés par l'usager-client et résolu exactement par le biais de l'algorithme **MCSP** (Maximal Constraint Satisfaction Problem). Là aussi une solution sous-optimale est obtenue grâce à MCSP-K qui est l'algorithme **MCSP** dans lequel la fonction de relaxation est modifiée. Pour le cas où les structures des flots ne sont exclusivement séquentielles, ce qui est le cas dans la réalité, les algorithmes proposés tant au niveau des modèles combinatoires que dans le modèle des graphes sont comme suit : **MMKP** (Multidimension Multichoice Knapsack Problem) pour le modèle combinatoire et **MCOP** (Multiconstraint Optimal Path : chemin optimal multicontraint) pour le modèle à base de graphe comme proposé dans [XB05].

L'algorithme **MCOP** étant comme **MMKP** NP-complet, les auteurs de [YZL07] ont proposé une heuristique dénommée *Wflow* qui donne une solution quasi-optimale en un temps polynomial. L'idée principale de l'heuristique *WFlow* est de décomposer les workflows en une multiplicité de chemins d'exécution. La complexité temporelle est polynomiale tout comme celle de **WS-HEU** soit en O(n^2 $(l-1)^2$ m) . Pour les modèles en graphe, MCOP étant NP-complet, les auteurs de [YZL07] ont proposé une amélioration en suggérant l'heuristique MCOP-K qui est plus rapide que l'algorithme MCOP et consommant moins d'espace. Les performances demeurent, selon les tests effectués, proches de

l'optimum dans 95% des cas. Il est à signaler que l'heuristique **MCOP-K** avait été proposée en 2006 par Tao YU dans sa thèse de doctorat à l'Université Irvine [Yu06].

D'autres travaux avaient adressé le problème de la sélection avant et après les travaux de [YZL07]. C'est ainsi que **ZENG et al** ont utilisé la méthode de planification globale (*Global Planning*) dans [ZBD+03] afin d'obtenir les meilleurs composants d'un service composite en utilisant des techniques de programmation linéaire mixte.

ZENG et al. dans [ZBN+04] adoptent une telle sélection par optimisation locale puisque la sélection du service Web devant exécuter une tâche donnée dans un service composite est effectuée au dernier moment et sans tenir compte des autres tâches impliquées dans le service composite. Quand une tâche doit être exécutée, le système récolte les QdS des services Web qui sont candidats pour l'exécution de ladite tâche. Après quoi, un vecteur de qualité est affecté à chaque service candidat et c'est sur la base de ce vecteur que le système choisit un des services candidats en faisant appliquant une méthode de décision multicritère (**MCDM** pour **M**ultiple **C**riteria **D**ecision **M**aking). Le processus de sélection est basé sur la pondération assignée par l'usager (le client) à chaque critère et un ensemble de contraintes sur les attributs qualitatifs. Ces contraintes sont exprimées sur les tâches individuelles et non sur une de leurs combinaisons.

ARDAGNA et PERNICI ont proposé en 2005 [AP05] et en 2007 [AP07] un algorithme de programmation linéaire étendu aux contraintes locales. D'autres approches présentent des heuristiques basées sur des méthodes évolutionnaires (Algorithmes génétiques) proposées dans [ZSC06, CLC07, GCC07, KZ07, VRB08].

Plus récemment **ALRIFAI et al**. ont présenté en 2008 dans [ARDN08] et en 2009 dans [AR09] une nouvelle approche combinant des techniques d'optimisation locales et globales. Cette approche démarre du niveau global pour résoudre le problème de la sélection au niveau local. Elle procède en décomposant les contraintes globales de QdS (celles imposées par le client-usager à la totalité de l compositions) en un ensemble de contraintes locales (celles imposées aux sous-tâches comme partie de la composition). Pour ce faire, les auteurs proposent des techniques mixant la programmation linéaire et la programmation en nombres entiers ; techniques dites **MILP** (Mixed Integer Linear Programming) afin de trouver la meilleure décomposition des contraintes de QdS.

4.3.3.1 Evaluation et Couverture de la QdS dans les plateformes existantes

Dans sa thèse, Fabien BALIGANDA [Bal08] a comparé les performances de 10 plateformes existantes pour la gestion de la QdS dans les compositions de services. Les principales plateformes sont les suivantes :

eFlow [CIJ+00] est une plateforme développée par la société HP pour spécifier, établir et observer la QdS de services composites, tout en fournissant également un certain nombre de fonctionnalités telles que la gestion des événements, des exceptions, des transactions ou de la sécurité [CIJ+00, CS01]. Etant une approche relativement ancienne, eFlow cherche à fournir aux utilisateurs un moyen de spécifier aisément des services composites à partir de services atomiques, à l'instar de l'approche BPEL. Avec eFlow, une composition de services est modélisée sous la forme d'un graphe qui définit l'ordre d'exécution de noeuds. Les arcs du graphe supportent des prédicats sur les variables du processus pour évaluer

les transitions entre nœuds du graphe.

AgFlow est le résultat des travaux de Zeng et al. [ZBD+03, ZBN+04] qui ont cherché à apporter des réponses à plusieurs challenges fondamentaux liés à la composition orientée QdS. Tout d'abord, à effectuer une étude de plusieurs caractéristiques de QdS chez les services Web. Ensuite, à poser le problème de la sélection des services Web appropriés pour permettre à un utilisateur d'obtenir certaines propriétés de QdS dans sa composition de services comme minimiser le prix et maximiser la réputation des services. Finalement, ce travail souligne l'intérêt d'une approche dynamique pour la prise en compte des changements de QdS inhérents à la variabilité du Web et des applications. Les contributions qui en découlent consistent en un modèle multi-dimensionnel capturant les propriétés de QdS et exhibant des méthodes pour attacher des valeurs à ces propriétés dans le cadre de services atomiques et composites. Deux approches (locale et globale) pour la sélection de services dans les compositions sont ensuite détaillées et un moteur d'exécution, ≪AgFlow≫, qui réagit aux changements en planifiant de nouveau les services à l'exécution est présenté.

ORBWork est fruit des travaux de [CMSA02, CMSA04] qui se sont fixent pour tâche de permettre de déduire et de gérer la QdS des workflows de services Web. Pour cela, un modèle prédictif de QdS basé sur la QdS de tâches atomiques a été élaboré, ainsi qu'une implémentation, ORBWork, basée sur un moteur d'exécution de workflow.

La plateforme **WS-Binder** est présentée dans [CPE+06, DPEV+06] s'intéresse aux problématiques de recherche statique et dynamique de services dans les compositions. Ce travail réutilise un certain nombre de contributions faites par les travaux précédents (notamment la modélisation de la QdS et les formules d'agrégation), améliore certaines propositions (découpage du workflow pour réduire l'espace de recherche à l'ensemble des services n'ayant pas encore été invoqués), et propose des contributions originales dont un algorithme modulable pour la sélection de service, un modèle de déclenchement de nouveau la planification et la gestion de métriques spécifiques à un domaine. Ce travail propose une plateforme (WS-Binder) qui contient l'implémentation de ces propositions et qui travaille en parallèle du moteur d'exécution BPEL.

La plateforme **Qbroker** est un basée sur un broker est le résultat des travaux de Tao YU [YL05a, YL05b], présente une architecture de brokering pour l'intégration et l'adaptation dynamique des services Web dans les compositions de services. A l'instar d'autres approches, elle propose une modélisation de la QdS ainsi que des algorithmes de sélection de services basés sur les caractéristiques globales de la composition.

Le tableau de la figure 4.10 montre des caractéristiques de QdS considérées par les différentes plateformes susmentionnées. Ce tableau synthétise les évaluations de différentes plateformes sous les 5 critères suivants : Réutilisation : (du BPEL, des contrats SLA (Service Level Agreement) et des WS-Policy), Séparation des préoccupations (SdP) : (isolation des spécifications, intrusivité des plateformes), Couverture : (plage de caractéristiques de QdS, mécanismes de QdS, granularité), Dynamicité et Expressivité (richesse de l'interface, déclarativité). SdP désigne la séparation des deux préoccupations qui sont la préoccupation de la gestion de la QdS et la préoccupation de la gestion de la composition. L'intrusivité désigne le degré d'intrusion les mécanismes de gestion de la QdS dans la plateforme mettant en exécution la composition. C'est **eFlow** qui offre une plus grande ouverture, à cet

	Réutilisation		SdP		Couverture			Dynam.	Expressivité	
	BPEL	SLA	Isol.	Intrus.	Chara.	Méca.	Gran.		Rich.	Decl.
AO4BPEL	+	−	+	±	−	±	±	±	−	+
DYNAMO	+	±	+	−	±	±	+	±	+	+
MASC	−	±	+	−	±	−	±	±	±	−
TRAP/BPEL	+	−	+	+	±	−	−	±	−	+
eFlow	−	−	−	−	+	±	−	+	−	+
AgFlow	−	+		−	+	±	+	+	±	
ORBWork	+	−		+	+	−	+	+	±	
WS-Binder	+	+	+	+	+	±	+	+	±	+
Optimizing QoS	±	±					+	±		
broker	+	+		±	±	±	+	±		

FIGURE 4.9 – synthèse des évaluations des plateformes [Bal08]. Dans ce tableau, les cases vides représentent des fonctionnalités n'ayant pas été prises en compte par le système. Dans les autres cas, les travaux ont été jugés selon une notation comportant trois niveaux croissants : "-", "±", "+".

égard, grâce à son langage de requête **XQL**. Très peu de plateformes considèrent les caractéristiques de sécurité. La gestion de la QdS recouvre un ensemble de domaines qui sont adressés de manière inégale par les différentes plateformes. Les plateformes **WS-Binder**, **AgFlow** et **Qbroker** permettent également de gérer les SLA (négociation, violation).

Dans un rapport technique plus récent datant de février 2010 et ayant pour auteurs **Moreno MARZOLLA** et **Raffaela MIRANDOLA** [MM10], il a été procédé à un comparatif des plateformes adressant la question de la QdS des services Web, nous en extrayons les tableaux 4.2 and 4.3 . De cette évaluation, il est possible de tirer quelques enseignements. Pour ce qui est des attributs QdS, trois plateformes utilisent de multiples QdS, les autres utilisent entre 1 et 6 attributs. Pour ce qui est du point de vue, la plupart des plateformes utilisent le point de vue du consommateur, quatre seulement utilisent le double point de vue Fournisseur/Consommateur, et une seule le point de vue fournisseur. Dans la colonne niveau de développement Les auteurs différencient les approches Design Time. Enfin, en ce qui les modèles d'enrichissement des QdS : cinq plateformes utilisent des méthodes d'optimisation, trois plateformes utilisent des modèles parmi lesquelles deux utilisent les algorithmes génétiques ; les trois autres sont basées broker et ont été présentées en détails dans ce chapitre (voir supra).

4.4 Conclusion

L'état de l'art présenté dans ce chapitre couvre toutes les dimensions de la composition de services Web que cette composition soit manuelle, semi-automatique ou automatiquue ; qu'elle soit statique ou dynamique ; qu'elle se base sur des wokflows ou sur des graphes de composition ; qu'elle soit basée sur des propriétés fonctionnelles ou non fonctionnelles.

Approche	**Critères QdS**	**Point de vue**	**Niveau de développement**	**modèles d'enrichissement des QdS**
Cardoso et al. [CMSA04]	Temps de réponse, coût et fiabilité	Consommateur	Design Time	Modèles
Maximilien et al. [MS04a, MS04b]	Réputation et Confiance	Consommateur	Design Time, Run Time	Ontologie
Vu et al. [VHA05]	Temps de réponse, Réputation et Confiance	Consommateur	Design Time, Run Time	Ontologie
Casati et al. [CCDS04]	Temps de réponse	Consommateur	Run Time	Fouille des données mesurées
Zeng et al. [ZBN+04]	Prix, Disponibilité, fiabilité et Réputation	Consommateur	Design Time	Modèle d'optimisation
Ardagna et al. [AP05]	Temps de réponse, coût et Réputation	Consommateur	Design Time, Run Time	Modèle d'optimisation
Cardellini et al. [CCGM06]	Temps de réponse, coût et Réputation	Consommateur	Design Time	Modèle d'optimisation
Canfora et al. [CPE+06]	Temps de réponse, coût, Disponibilité et fiabilité	Consommateur	Design Time	Modèles (Algorithmes génétiques)
Claro et al. [CAkH05]	Temps de réponse, coût, Disponibilité et fiabilité	Consommateur	Design Time	Modèles (Algorithmes génétiques)

TABLE 4.2 – Synthèse des évaluations des Approches de Composition des services orienté QdS selon [MM10] (1)

Approche	Critères QdS	Point de vue	Niveau de développement	modèles d'enrichissement des QdS
Yu and Lin [YL05b]	Multiple QdS	Consommateur	Design Time	Modèle d'optimisation
Yu and Lin [YL05a]	Temps de réponse	Consommateur	Run Time	Framework basé sur broker
Patel [PSL03]	Multiple QdS	Consommateur	Run Time	Découverte, Sélection et Monitring de services Web
Serhani et al. [SDHS05]	Temps de réponse, Temps de latence, Disponibilité, Réputation et Prix	Consommateur	Run Time	Framework basé sur broker
Dong and Yu [DY06]	Temps de réponse, Fiabilité et coût	Consommateur	Design Time	Modèle d'optimisation
Rosenberg et al. [RPD06]	Temps de latence, Temps de réponse, Disponibilité et Précision	Consommateur	Run Time	Mesures
Schmid et al. [Sch06]	Temps de latence	Consommateur	Run Time	Mesures
Shercliff et al. [SSGF06]	Multiple QdS	Fournisseur	Design Time et Run Time	Suivi et Modèles
Yu and Lin [YL04]	Multiple QdS	Fournisseur et Consommateur	Run Time	Framework basé sur Broker

TABLE 4.3 – Synthèse des évaluations des Approches de Composition des services orienté QdS selon [MM10] (2)

Nous avons examiné les classes de composition automatique, même si elles se fondent sur le formel, la logique, les preuves de théorèmes, la planification avec ses variantes, la logique linéaire ou le calcul situationnel, aucune classe ne semble disposer de plus d'avantages que les autres.

Pour notre part, nous allons nous inspirer des méthodes logiques et notamment déductives, qui fera l'objet du chapitre 7 pour proposer notre approche de composition de services Web.

Par ailleurs, Les méthodes de sélection de services composites à base de QdS qui ont été proposées au cours des cinq dernières années, sont nombreuses et diversifiées. Les solutions '*optimales*' ne peuvent être obtenues qu'en faisant appel à des algorithmes conçus pour des problèmes d'optimisation réputés NP-complets. En faisant appel à la programmation linéaire ou à la programmation en nombres entiers, à la programmation dynamique ou au problème du sac à dos, au problème de la recherche d'un chemin optimal ou à la CSP, la composition abstraite peut sembler séduisante mais la composition concrète est loin d'être implémentable en raison de la complexité temporelle de ces algorithmes qui est exponentielle. Il a fallu recourir à des heuristiques pour rendre ces approches praticables dans des cas réels où le nombre de classes de services est élevé et où la cardinalité de chaque classe est élevée. Pour avoir une complexité temporelle polynomiale, il avait fallu se contenter de solutions sous-optimales. Pour le méta-framework BioMed, nous allons considérer des requêtes étendues de composition en ce sens qu'elles portent sur deux volets : le premier centré sur les propriétés fonctionnelles alors que le second est centré sur des propriétés non fonctionnelles (attributs de QdS). Dans le cas où BioMed ne découvre pas de service atomique répondant au mieux à la requête, c'est le moteur de composition qui est sollicitée.

Cette seconde partie de ce mémoire aura pour finalité de présenter une nouvelle approche de médiation de ressources Web opérant à différents niveaux : une médiation de services Web, une médiation ontologique et une médiation de données. La médiation de services repose sur la strate inférentielle de BioMed permettant d'exploiter les connaissances ontologiques afin d'offrir des mécanismes flexibles de découverte et de composition. La médiation ontologique repose sur une méta-carte ontologique conçue afin de représenter le double engagement sémantique et ontologique reflétant une conceptualisation partagée du domaine ciblé. La médiation de données se base sur le modèle d'alignement sémantique afin d'offrir un modèle pivot de descriptions canoniques explicitant la sémantique d'un service Web.

La seconde partie comprend les trois chapitres suivants :

Dans le chapitre 5, nous présentons le méta-framework BioMed et ses différents niveaux de stratification. Nous présentons les trois composantes clés de ce méta-framework à savoir : le modèle d'alignement (normalisation) sémantique des descriptions de services Web, une méta-carte ontologique permettant d'offrir une sémantique unifiée et un modèle inférentiel permettant de formaliser une large typologie de raisonnement déductif.

Le chapitre 6 présente l'approche de découverte du méta-framework BioMed basée sur le raisonnement déductif afin d'inférer les solutions satisfaisant les contraintes fonctionnelles d'une requête. Un mécanisme de classement est mis en place en considérant l'ensemble des solutions sélectionnées par le moteur de raisonnement pour déceler les bonnes voire meilleures solutions en combinant deux techniques celle de l'opérateur skyline

[BKS01] et celle du top-k [IBS08].

Le chapitre 7 sera consacré à la présentation de l'approche de composition du méta-framework BioMed permettant de déduire dans un premier temps par le biais des techniques de raisonnement l'ensemble des plans de composition comprenant un ensemble de services vérifiant les critères de composabilité fonctionnelle des service composants (composabilité exacte ou rapprochée) et dans un second temps de sélectionner parmi les plans de composition identifiés lors de la première phase, les plans qui vérifient les critères QdS. La seconde phase sera résolue par deux approches possibles : la première identifique à celle adoptée dans le chapitre 6 et la seconde adoptant une heuristique différente *non Pareto*.

Deuxième partie

Proposition, Contributions et Expérimentations

Chapitre 5

Un Méta-Framework stratifié de médiation de Services Web sémantiquement hétérogènes

Sommaire

Le présent chapitre est consacré à la présentation du méta-framework de médiation sémantique proposé dans le cadre de cette thèse que nous nommons BioMed pour ***Bio****informatic* ***Med****iation Framework*. Le chapitre s'organise comme suit, la section 5.1 présente le cadre conceptuel et les objectifs du méta-framework proposé. Le modèle unifié de descriptions canoniques de services est présenté dans la section 5.2. La section 5.3 présente la méta-carte ontologique que nous proposons afin de pallier l'hétérogénéité des ontologies

existantes dans le domaine visé et pour assurer les engagements sémantique et ontologique. La section 5.4 discute l'annotation sémantique des descriptions de services Web. Enfin, dans la section 5.5, nous explicitons les mécanismes inférentiels que nous proposons afin d'exploiter les descriptions unifiées et les annotations sémantiques permettant ainsi de relaxer les propriétés des services et d'élargir leur portée et leur couverture sémantique.

5.1 Cadre Conceptuel du Méta-Framework

La médiation sémantique de services a pour objectif d'exploiter la sémantique explicite des services afin d'en faciliter la découverte, la sélection et la composition en *workflow* pour des utilisateurs finals. La sémantisation des descriptions de services représente une phase nécessaire dont l'objectif est d'expliciter la sémantique d'un service et de générer sa description sémantisée. Une description sémantisée comprend un ensemble de *primitives descriptives* permettant de décrire explicitement les propriétés pertinentes d'un service associées à un ensemble d'annotations sémantiques décrivant le *lien* entre les primitives descriptives et les concepts d'une ontologie du domaine. Ainsi, les annotations sémantiques permettent d'instancier les primitives descriptives à l'aide de concepts, relations sémantiques ou instances décrits dans une ontologie. Elles ont pour objectif d'exprimer formellement la sémantique du service afin d'améliorer sa découverte et sa composition et donc sa réutilisation.

Toutefois, différents conflits d'hétérogénéité sémantique peuvent surgir. Nous distinguons une double hétérogénéité sémantique au niveau des descriptions sémantisées de services :

1. L'hétérogénéité des primitives descriptives liée au framework adopté pour déclarer la sémantique du service ;
2. L'hétérogénéité des annotations sémantiques qui peuvent être définies par le biais d'ontologies diverses.

En effet, les services peuvent faire appel à différentes ontologies du domaine comme les descriptions peuvent être décrites à l'aide de primitives appartenant à différents frameworks de services Web sémantiques. Pour ce faire, nous proposons un méta-framework stratifié permettant de décrire, d'exploiter et de raisonner sur des services Web. Afin de pallier l'hétérogénéité des primitives descriptives, le méta-framework BioMed repose sur un modèle d'alignement sémantique visant à générer des descriptions canoniques de services Web, ce modèle offre un ensemble de primitives sémantiques résultant de l'alignement des méta-modèles sous-jacents aux ontologies OWL-S et WSMO et celui des annotations SAWSDL. Ces primitives permettent d'expliciter les propriétés fonctionnelles et non fonctionnelles d'un service Web.

Afin de réconcilier les ontologies du domaine bioinformatique, nous avons proposé une méta-carte ontologique reflétant une sémantique consensuelle du domaine spécifiant les objets du domaine qui peuvent être associés à chaque concept de la méta-carte, conformément à sa signification formelle. Cette signification est induite par d'une part une *sémantique référentielle* établissant le lien entre l'intension et les extensions du concept impliquant notamment des relations d'hyperonomie/hyponymie, d'holonymie/méronymie et d'autre

part une *sémantique différentielle* entre les extensions du concept impliquant un arc sémantique allant de la synonymie à l'antonymie. La méta-carte ontologique est utilisée pour la génération des annotations sémantiques de services Web comme elle représente le soubassement du modèle inférentiel de Biomed. Le méta-framework illustré dans la figure 5.2 comporte cinq niveaux de conceptualisation : le niveau ressource, le niveau description, le niveau ontologique, le niveau inférentiel et le niveau processus. Nous détaillons dans ce qui suit ces cinq niveaux de stratification.

5.1.1 Le niveau ressource

Le niveau ressource comprend un ensemble de ressources Web accessibles par le biais d'interfaces de services Web. Dans le domaine bioinformatique, les services Web sont conçus pour standardiser la façon d'accéder à divers types de ressources comprenant des sources de données génomiques telles que GenBank [1] ou protéiques telles que Swiss-Prot [2] ou de maladies génétiques telles que OMIM [3] mais également des outils tels que BLAST [4]. Les ressources bioinformatiques sont caractérisées par une forte hétérogénéité qui s'illustre à différents niveaux :

- **le niveau syntactique** : les modèles de données sont différents selon les sources : on retrouve le modèle relationnel (par exemple dans *Ensembl*[5]), le modèle XML (tel que dans la source UniProt [6]) ou encore les fichier balisés (tels que ceux de *SwissProt* et *GenBank*). La technologie de services Web permet de masquer le niveau d'hétérogénéité syntactique. Les services dans le domaine bioinformatique sont en grande partie des services orientés données (en opposition aux services orientés fonctionnalités) du fait qu'ils encapsulent de gigantesques sources de données afin de masquer la structure de ces sources et leur contenu informationnel en mettant en vigueur les politiques de sécurité associées.
- **le niveau conceptuel** : nous distinguons deux types d'hétérogénéité conceptuelle, la diversité des focus et de niveau de granularité. En bioinformatique, chaque source se focalise sur une entité conceptuelle du domaine. Certaines se focalisent sur la protéine comme entité centrale, d'autres sources se focalisent sur les gènes ou les maladies comme entité centrale, d'autres s'intéressent aux marqueurs de séquences exprimées alors que certains se focalisent sur les interactions entre protéines . Le niveau de granularité peut aussi varier selon les sources, celles qui se concentrent sur le génome humain et ceux qui s'intéressent à différents génomes de plusieurs espèces.
- le niveau sémantique : l'hétérogénéité sémantique s'illustre au niveau des instances des sources et des annotations sémantiques qu'elles contiennent. En l'occurrence,

1. http ://www.ncbi.nlm.nih.gov/Genbank/
2. http ://www.ebi.ac.uk/uniprot/
3. http ://www.ncbi.nlm.nih.gov/omim/
4. BLAST (acronyme de Basic Local Alignment Search Tool) est une méthode de recherche heuristique utilisée en bioinformatique permettant de trouver les régions similaires entre deux ou plusieurs séquences de nucléotides ou d'acides aminés. Ce service permet de retrouver rapidement dans des bases de données, les séquences ayant des zones de similitude avec une séquence donnée (spécifiée comme entrée par l'utilisateur). La méthode BLAST a été implémenté National Center for Biotechnology Information (NCBI), cette implémentation est accessible à l'adresse http ://blast.ncbi.nlm.nih.gov/Blast.cgi
5. http ://www.ensembl.org/index.html
6. http ://www.uniprot.org/

différents noms et symboles existent pour le même gène ou pour la même protéine. De plus, les sources décrivent une même entité du domaine selon des perspectives différentes, ceci peut être relative par exemple à l'expertise du bioinformaticien.

Durant les 20 dernières années, les ressources de données biologiques disponibles sur le Web se sont multipliées. Leur croissance est en forte progression depuis 10 ans. La *Databases Issue* de la revue Nucleic Acids Research (NAR), qui liste chaque année les sources les plus importantes du Web dans le domaine bioinformatique, recense plus de 1230 ressources en 2010, soit 58 ressources plus que l'année précédente et 168 ressources de plus qu'en 2009. Au total, le nombre de ressources a été multiplié par trois durant les dix dernières années. En pratique, ces ressources forment un réseau fortement connecté de noeuds inter-reliés par des références croisées qui permettent de faciliter la navigation à l'utilisateur. En effet, une entrée dans Swiss-Prot consistant en une protéine fera référence au gène lié par le biais d'un lien hypertexte pointant vers l'entrée relative au gène dans la ressource GenBank qui fera à son tour référence à la maladie associée à la mutation de ce gène dans la ressource OMIM. De plus, ces ressources sont sémantiquement annotées avec des concepts de différents vocabulaires contrôlés tels que Gene Ontology (GO) ou Mesh ou de nomenclatures telles que SNOMED.

L'avènement des services Web a révolutionné les modes d'accès à ces ressources et a permis de masquer les problèmes d'hétérogénéité syntactique inhérents aux ressources bioinformatiques et cela en offrant une plate-forme d'interopérabilité basée sur les services. Cependant, de nouveaux problèmes et défis ont vu le jour en termes de découverte et de sélection de services dans un environnement ouvert tel que le Web et face aux attentes diversifiés de scientifiques de plus en plus exigeants. D'autant plus que ces derniers ont constamment besoin de comparer les résultats générés à partir de plusieurs ressources et outils voire même les orchestrer afin de répondre à une requête complexe. Ainsi, le besoin en découverte et composition de services dans le domaine bioinformatique devient de plus en plus pressant afin de faire face à l'évolution du nombre de services Web et afin de guider les scientifiques à découvrir et composer les services Web les plus appropriés à leur besoin.

5.1.2 Le niveau description

La description d'un service correspond à la spécification de ses capacités en termes de fonctionnalités exprimée dans un langage formel et interopérable afin qu'elle soit exploitée par des agents humains et/ou artificiels. Une description peut être syntaxique telles qu'une description WSDL ou sémantisée si elle comprend des annotations faisant référence à une ontologie du domaine. Une description, quelle soit syntaxique ou sémantisée, est définie par le biais d'un ensemble de primitives que nous nommons *primitives descriptives*. A titre d'exemple, les primitives descriptives du langage WSDL constituent l'ensemble des mots-clés du langage tels que *operation*, *message*, *porttype*, etc. Une description sémantisée d'un service Web est définie par le biais d'un ensemble de primitives descriptives associées à un ensemble d'annotations sémantiques faisant référence à des concepts, relations ou instances ontologiques. Plusieurs frameworks de descriptions sémantiques ont été étudiées dans le chapitre 3 du présent mémoire et qui sont principalement : OWL-S, WSMO, SAWSDL. Ces frameworks offrent chacun de son côté des primitives descriptives permettant de générer

des descriptions sémantisées de SW qui sont hétérogènes entre elles. Cette hétérogénéité rend difficile l'élaboration d'une approche générique de découverte et de composition de services.

Pour ce faire, nous proposons au niveau description du méta-framework BioMed un modèle unifié de descriptions sémantisées de services Web. Ce modèle offre un ensemble de primitives descriptives qualifiées de canoniques ayant comme objectif d'unifier les descriptions des frameworks existants de services Web sémantiques et les aligner afin de converger vers une sémantique unifiée représentative des aspects fonctionnels et non fonctionnels d'un service atomique (élémentaire ou non décomposable). Le modèle unifié permet de déclarer la signature fonctionnelle d'un service atomique en termes d'entrées, de sorties, préconditions et postconditions et un ensemble de propriétés non fonctionnelles décrivant la qualité du service. Nous avons omis de considérer les primitives permettant de décrire le comportement interne d'un service pour deux raisons, la première est liée au fait que les services du domaine bioinformatique sont en grande partie des services atomiques avec des interfaces (signatures fonctionnelles) riches vu la complexité des schémas des sources qu'ils encapsulent, et la seconde est liée à notre besoin de faire émerger une **sémantique minimale et suffisante** pour décrire canoniquement un service, le découvrir et le composer éventuellement avec d'autres services.

5.1.3 Le niveau ontologique

Au niveau ontologique du méta-framework BioMed, un référentiel normalisé de services bioinformatiques est construit manuellement en adoptant le modèle pivot de descriptions canoniques du modèle unifié et en se basant sur une représentation sémantique permettant d'expliciter la sémantique des services Web.

La diversité des ontologies existantes dans le domaine bioinformatique comme le montrent des portails dédiés tels que OBO [7] (Open Biological and Biomedical Ontologies) foundary et le BioPortal [8] montre le besoin d'une représentation sémantique partagée consensuelle vérifiant un double engagement sémantique et ontologique [Bac00]. En effet, de la richesse et la complexité des connaissances du domaine cible et de la co-existence de plusieurs théories, facettes, points de vue dans ce même domaine émerge le besoin d'une sémantique partagée pourvue d'un ensemble de primitives sémantiques flexibles au sein d'une architecture modulaire reliant les composants d'une ontologie ou de différentes ontologies. D'autant plus que l'étude des ontologies proposées dans le domaine bioinformatique et biomédical montre leur inadéquation au besoin de médiation, de réconciliation et d'interopérabilité sémantique.

Ainsi, nous proposons de concevoir une méta-carte ontologique représentant la sémantique du réseau formé par les ontologies du domaine bioinformatique et nous la situons à un niveau intermédiaire entre une ontologique fondationnelle [9] telles que DOLCE, SUMO ou GFO et les ontologies du domaine. La conception de la méta-carte aura à répondre à un certain nombre de besoins qui sont décrit comme suit :

7. http ://www.obofoundry.org/
8. http ://bioportal.bioontology.org/
9. L'annexe A de ce mémoire présente un état de l'art des ontologies fondationnelles tout en les comparant.

- **Fondements des primitives :** Les primitives de classes et de relations utilisées dans une la méta-carte doivent être fondées sur un ensemble de principes reflétant la motivation cognitive (comprenant l'intersubjectivité et la plus large capacité de réutilisation). L'appui d'une ontologie fondationnelle DOLCE permet de modéliser les méta-propriétés cognitives qui seront héritées par les concepts définis au niveau de la méta-carte.
- **Couverture Sémantique** : La méta-carte ontologique devrait assurer la couverture de toutes les entités du domaine (couverture extensionnelle) tout en garantissant la couverture de tous les types d'entités du domaine (couverture intensionnelle).
- **Précision de la modélisation** : La méta-carte doit tenter de représenter tous les modèles possibles du domaine d'intérêt et couvrir les tâches devant être accomplis, c'est dans ce sens qu'elle est multi-facette.
- **Modularité légère et forte** : dans le but de moduler l'espace du domaine conceptuel autant que possible, la méta-carte doit permettre d'organiser les théories du domaine soit dans des ontologies différentes soit dans des descriptions et situations différentes.
- **Scalabilité** : Les langages expressifs devraient représenter les détails des significations en conformité avec le domaine et les tâches qui y sont accomplies. Les ontologies expressives peuvent être utilisées comme ontologies référentielles ou comme aide et support à des services sophistiqués. Comme ontologies référentielles elles doivent supporter la scalabilité en vue d'aider computationnellement des services ardus comme l'extraction d'information (ou de connaissances) à partir d'entrepôts assez larges ou à partir du Web comme dans le cas du Web Sémantique.
- **Consistance Logique** : Les ontologies doivent être exprimées dans un langage formel disposant d'une sémantique formelle explicite à l'instar de la logique de premier ordre (LPO) de KIF, d'une Logique de Description (DL), de OWL etc.

Le choix d'utiliser une ontologie fondationnnelle comme soubassement pour la conception de la méta-carte ontologique est justifié par les trois principaux arguments suivants, comme signalé par OBERLE et ses collègues dans [OAH+07] et qui font que l'ontologie fondationnelle :

- Peut servir comme base de modélisation en fournissant un point de départ pour construire de nouvelles ontologies. Au lieu de modéliser à partir de zéro, l'utilisation des ontologies fondationnelles fournit aux concepteurs un ensemble d'entités ontologiques prédéfinies qui sont réutilisables pour les ontologies qu'ils veulent construire ;
- Peut être choisie pour sa clarté conceptuelle car fournissant un point de référence pour une comparaison rigoureuse entre plusieurs approches ontologiques possibles, et un cadre de travail pour analyser, harmoniser, et intégrer des ontologies existantes ;
- Peut offrir des modèles de conception d'ontologie : idéalement, une ontologie fondationnelle définit et propose des modèles de conception d'ontologies pour des besoins de modélisation récurrents, tels que la localisation dans le temps et l'espace.

La méta-carte ainsi conçue comprend un ensemble de concepts "*génériques*" du domaine que nous nommons *hyper-concepts* interconnectés par un ensemble de relations sémantiques. Chaque hyper-concept est relié à une ou plusieurs ontologies décrivant l'ex-

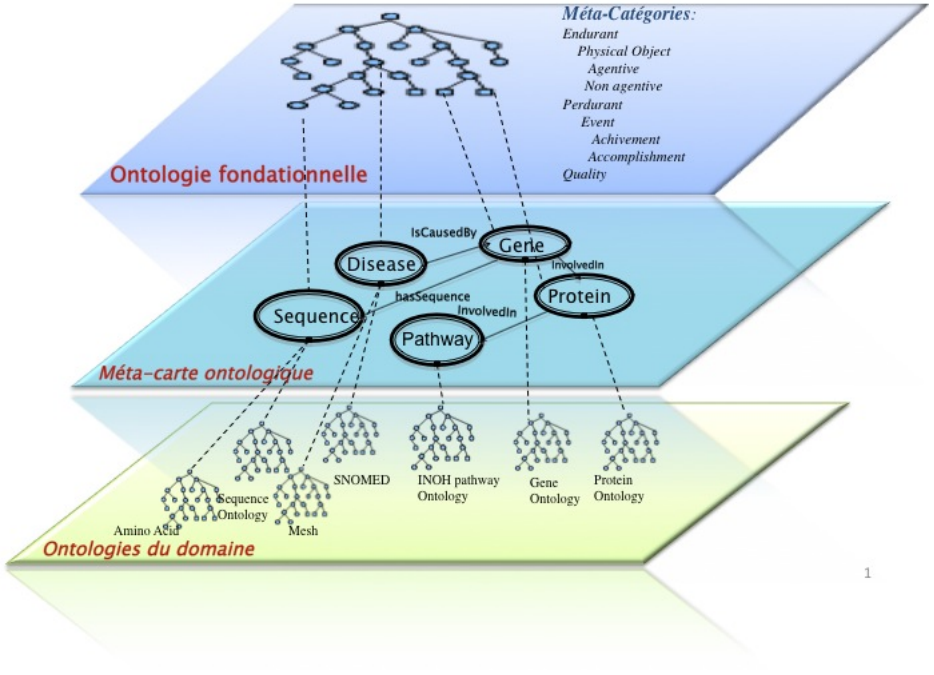


FIGURE 5.1 – Illustration du niveau ontologique du méta-Framework BioMed

tension de l'hyper-concept [10]ou son intension [11]. En l'occurrence, l'ontologie Gene Ontology (GO) décrit les fonctions d'un gène au niveau extensionnel, la PO (Protein Ontology) offre une description unifiée du concept Protéine donc l'intension du concept associé (protéine) comme l'illustre la figure 5.1. La Sequence Ontology offre une description unifiée du concept séquence en spécifiant ses propriétés selon plusieurs points de vue : la structure (exon, intron, promoteur), la longueur, le type (nucléotidique ou acides aminés), le génome, les variations chromosomiques [12].

Définition 1 (Méta-carte Ontologique) *Etant donné un ensemble d'ontologies du domaine $\mathcal{O}=\{o_1,...,o_n\}$, la méta-carte ontologique d'un domaine $\mathbb{D}$ notée $\mathbb{C}=(V,E)$ est un graphe direct orienté composé d'un ensemble de noeuds V désignant les concepts du domaine reliées par un ensemble d'arcs E désignant des relations sémantiques. Une carte sémantique $\mathbb{C}$ est munie d'une fonction de mapping $\mathcal{M} : V \rightarrow \mathcal{O}$ qui permet d'associer à chaque concept c de l'ensemble des concepts V, une ontologie o décrivant l'extension du concept c.*

10. l'ensemble d'instances du hyper-concept
11. les caractéristiques spécifiques du concept
12. Ces concepts sont définis dans le glossaire de la bioinformatique présenté en fin du rapport.

5.1.4 Le niveau inférentiel

Le niveau inférentiel permet de modéliser l'ensemble des connaissances inférentielles du domaine des services. A cet effet, nous proposons une axiomatique pour formaliser ce type de connaissances. Cette axiomatique comprend différents schémas d'axiomes formalisant différent types de raisonnement qui sont réalisés à différent niveaux de conceptualisation de BioMed. Nous distinguons trois niveaux de raisonnement : (a) au niveau ontologique ayant pour objectif d'exploiter les connaissances ontologiques, (b) au niveau description ayant pour objectif d'exploiter les descriptions sémantisées de SW et (c) au niveau assertionnel ayant pour objectif de raisonner sur les instances des services (instances de paramètres).

Le niveau inférentiel comprend un moteur d'inférence qui se base sur un raisonnement déductif. Ainsi, le moteur d'inférence implémente trois types de schémas d'axiomes qui sont décrits comme suit :

1. les axiomes du niveau **ontologique** permettent d'exploiter les connaissances de la carte ontologique et se déclinent en axiomes portant sur les concepts tels que le lien *is-A* ou la propriété de *disjonction de concepts* et des axiomes portant sur les relations sémantiques telles que les propriétés algébriques des relations (transitivité, réflexivité, anti-réflexivité, symétrie, anti-symétrie).
2. les axiomes du niveau **service** permettent de raisonner au niveau des descriptions unifiées des instances de services afin d'inférer de nouvelles descriptions sémantiques décrivant les services. Les axiomes du niveau service reposent sur les axiomes du niveau ontologique et du niveau assertionnel.
3. les axiomes du niveau **assertionnel** permettent de raisonner au niveau des instances et d'inférer par exemple le concept spécifique/général auquel une instance appartient.

Enfin, la sémantique axiomatique qui fait l'objet de la sous-section 5.5 permet de formaliser l'ensemble des connaissances inférentielles du domaine, les trois niveaux de raisonnement permettent d'enrichir la couverte sémantique des descriptions des services Web, ce qui permettra un appariement flexible entre les contraintes des clients et les descriptions de services.

5.1.5 Le niveau processus

Le niveau processus correspond aux différents processus pouvant être réalisés dans une architecture orienté service à savoir principalement ceux de la découverte et de la composition de services Web.

Le processus de découverte dans BioMed repose sur le moteur de raisonnement déductif pour sélectionner les services Web atomiques qui répondent fonctionnellement et non fonctionnellement à une requête de découverte. Nous montrons que le raisonnement déductif permet d'élargir considérablement l'espace de recherche (search space) des services candidats afin d'apparier au mieux les contraintes de la requête. Contrairement aux approches existantes de découverte de services qui proposent des solutions de découverte exclusivement basées soit sur les propriétés fonctionnelles soit sur les propriétés non fonctionnelles d'un service et non pas les deux à la fois, l'approche de découverte proposée ici sélectionne les services en se basant sur les deux familles de propriétés. Etant donné que le nombre

de services candidats à une requête peut être élevé dans certains cas, nous avons adopté une heuristique de sélection des meilleurs services combinant les techniques basées sur l'opérateur *skyline* et les techniques de ranking *top-k*, ce qui permet de déterminer les meilleurs (top-k) services en se basant sur les valeurs des propriétés non fonctionnelles. L'approche de découverte est originale du fait qu'elle permet de combiner des techniques de raisonnement et de classement (ranking). L'approche de découverte proposée est présentée en détails dans le chapitre 6.

D'un autre côté, l'approche de composition se base sur le moteur de raisonnement pour déduire des chemins (possibles) constitués d'un ensemble de services atomiques vérifiant les critères de composabilité. En effet, une requête utilisateur est interprétée en premier lieu en tant que requête de découverte, si aucun service atomique ne répond aux spécifications de la requête alors le moteur de composition la prend en charge pour définir un graphe de composition définissant au moins un plan de composition de services satisfaisant la requête. Le moteur de raisonnement déductif permet de déduire un ensemble de services fonctionnellement satisfaisant la requête. Dans une seconde phase, l'approche de composition permettra de classer les plans de composition entre eux afin d'identifier les *plus satisfaisants*. L'approche de composition proposée sera présentée dans le chapitre 7.

5.2 Un Modèle unifié pour la description de SW

Nous proposons dans cette section un modèle de descriptions canoniques pour la représentation de services Web sémantisés. Ce modèle consiste en une vue unifiée des descriptions sémantiques de SW afin de pallier l'hétérogénéité des frameworks existantes à savoir OWL-S (Ontology Web Service), WSMO (Web Service Modeling Ontology) et le langage SAWSDL (Semantic Annotations for WSDL). Il ne s'agit pas de proposer un nouveau modèle pour la description sémantique de services Web mais de faire converger les modèles des propositions citées dessus vers un ensemble de descriptions canoniques qui capture une sémantique minimale et suffisante à l'automatisation des processus de découverte et de composition de SW. Il est important de considérer les trois dimensions suivantes dans la conception d'une description d'un service Web :

- Le niveau d'expressivité des descriptions. Entre une description en langage naturel et une formalisation en logique du premier ordre pour décrire un service, il s'agit de trouver un compromis entre une représentation formelle et riche sémantiquement pour garantir l'automatisation des tâches de découverte et de composition de services Web.
- La portée ou la couverture des descriptions et leur complétude. Il s'agit de trouver l'ensemble des propriétés canoniques **nécessaires et suffisantes** à la découverte d'un service Web. S'agit-il du même ensemble de propriétés canoniques lors d'un processus de compositionfi
- L'usage de la description pour implémenter des scénarios d'expérimentation des Services Web sémantisées dans le monde réel.

En premier lieu, nous notons une hétérogénéité des labellisations au niveau des constructeurs ou primitives de description des frameworks. En effet, pour désigner l'interface du service, OWL-S prévoit une sous ontologie nommée *Ontology Profile*, le terme profil désigne

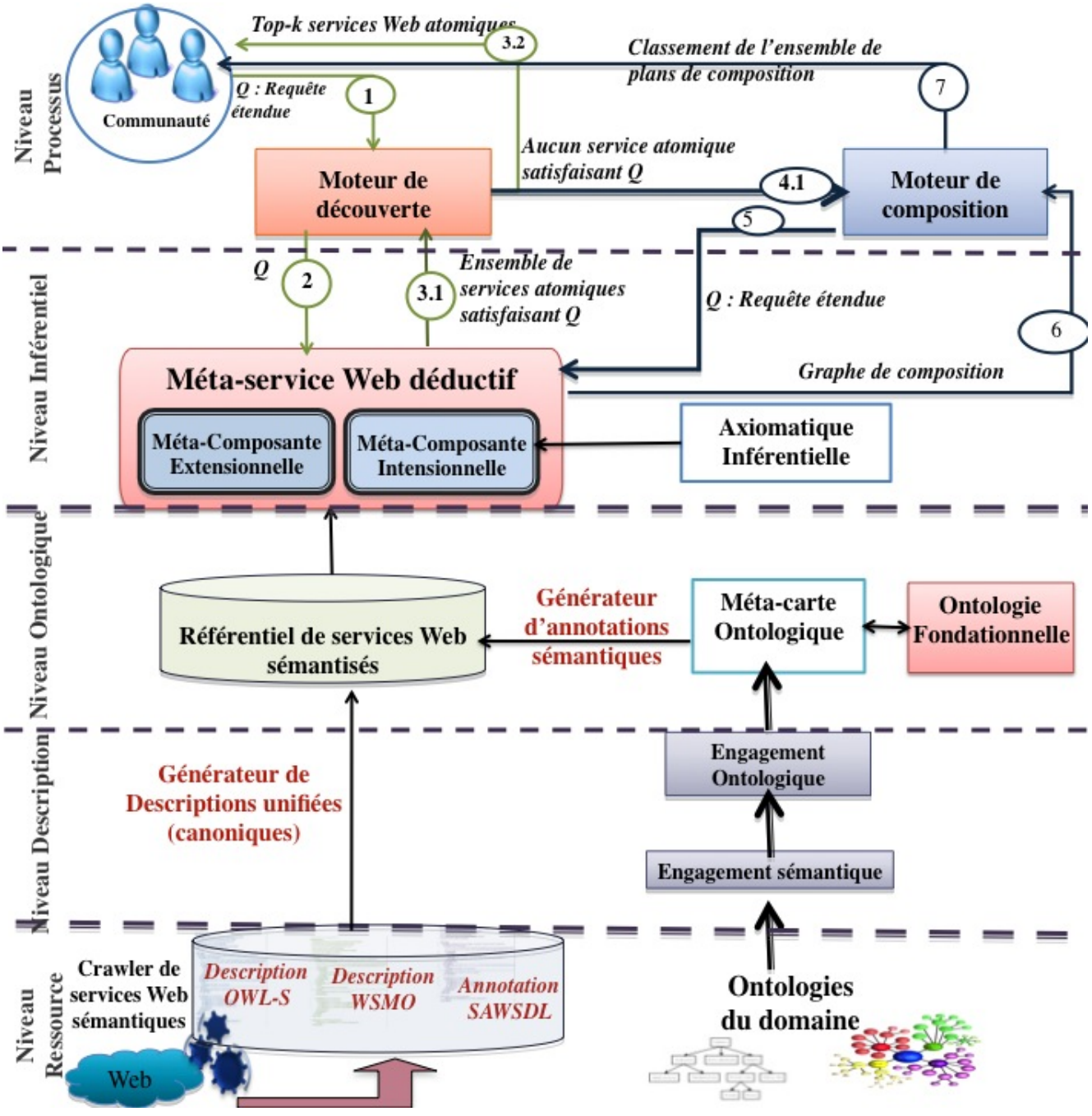


FIGURE 5.2 – Différents niveaux de stratification du méta-framework BIOMED

l'interface du service. WSMO explicite l'objectif (Goal) et la capacité (capability) du service comme connotation de l'interface d'un SW. SAWSDL se base sur les mêmes constructeurs que le langage WSDL en rajoutant des annotations faisant référence à des concepts ontologiques (*reference annotation*).

De plus, les langages de formalisation de ces descriptions sont eux-mêmes hétérogènes, OWL-S se base sur le langage OWL (Ontology Web Language) pour être compatible avec l'infrastructure du Web sémantique, WSMO se base sur un langage logique du premier ordre WSML (Web Service Modelling Language) pour pouvoir raisonner sur les descriptions et SAWSDL consiste en une simple extension du langage WSDL pour garantir une compatiblité avec l'infrastructure actuelle des SW. L'usage des frameworks existants dans le monde est très limité, peu de travaux présentent des scénarios d'expérimentation des trois frameworks dans le monde réel et leur usage reste purement académique.

Le modèle canonique de BioMed, illustré par la figure 5.5, s'appuie sur le schéma RDF-S (RDF Schema) pour décrire les différentes propriétés des services Web. RDF-S est le langage de description de ressources sur le Web permettant de déclarer des taxonomies de classes et de propriétés. Au niveau du modèle unifié des descriptions de services, ce dernier est décrit par un ensemble de primitives descriptives permettant d'expliciter sa signature fonctionnelle et un ensemble de propriétés non fonctionnelles décrivant sa qualité. Nous nommons les descriptions générées par le modèle unifié, **descriptions canoniques**.

En effet, un premier sous-ensemble de primitives descriptives permet de déclarer les propriétés fonctionnelles du service comprenant la liste des entrées, des sorties, de paramètres, de préconditions (les conditions devant être vérifiées par les entrées) et les postconditions (les conditions devant être vérifiées par les sorties du service). Ces propriétés fonctionnelles sont décrites d'une manière hétérogène au niveau des frameworks OWL-S et WSMO. Nous prévoyons l'existence de wrappers pour traduire les descriptions de services en OWL-S et WSMO vers notre modèle de description canonique. L'ensemble de mappings entre les primitives OWL-S, WSMO et le modèle de description BioMed est illustré par les figures 5.3 et 5.4.

Le second sous-ensemble de primitives permet de décrire les propriétés non fonctionnelles (PNF) décrivant la qualité d'un service. Nous distinguons deux types de propriétés QdS, celles qui sont *indépendantes du domaine du service* et qui peuvent être adoptées quel que soit le domaine et celles qui sont *spécifiques au domaine* et variant d'un domaine à l'autre. Par exemple, le critère débit exprimé en *bit par seconde* est significatif dans le contexte d'un service de *vidéo à la demande*, par contre, dans le contexte de services en e-commerce, il serait primordial de considérer le critère *sécurité*.

Les PNF indépendantes du domaine correspondent à celles fréquemment utilisées dans le domaine de la Qualité de Service (en anglais, Quality of Service QoS) [Ran03, KKL03] et incluent : le temps de réponse, le débit, la disponibilité, la fiabilité.

- Le temps de réponse (*Response Time*) est le temps mis entre l'envoi de la requête par le client et la réception du résultat envoyé par le service Web. L'unité de mesure du temps de réponse est la milliseconde (ms).
- Le débit (*Throughput*) est mesuré par le nombre total d'invocations possibles d'un service Web dans un laps de temps donné. L'unité de mesure du débit est le nombre d'appels réussis par seconde.

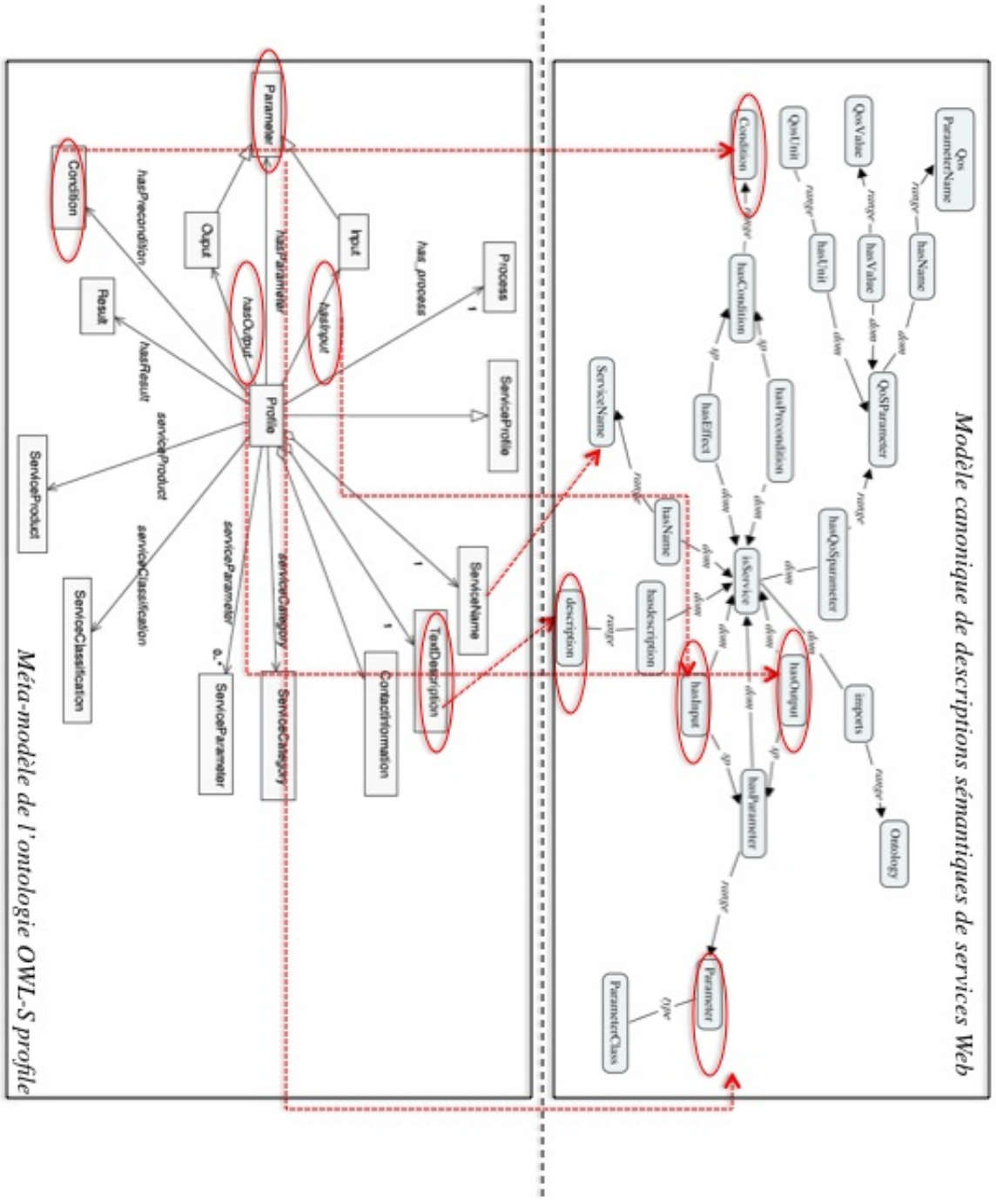


FIGURE 5.3 – Mappings entre le modèle unifié de BIOMED et le méta-modèle de l'ontologie de profil OWL-S

FIGURE 5.4 – Mappings entre le modèle unifié BIOMED et le méta-modèle du framework WSMO

- la disponibilité (*availability*) exprimé en pourcentage, permet de mesurer la probabilité que le service soit opérationnel à un instant t. Ce critère est le rapport du nombre de fois de disponibilité du service par rapport au nombre total d'invocations du service dans un laps de temps.
- La fiabilité (*reliability*) est le rapport du nombre de messages d'erreur sur le nombre total de messages. Ce critère est mesuré en pourcentage.

D'un autre côté, nous avons identifié deux critères QdS spécifiques au domaine bioinformatique. Les deux critères non fonctionnels sont : le *degré de confiance* et la *complétude*. Ces deux critères ont été déjà considérés dans les travaux de thèse de Sarah Cohen Boulakia intitulé "*Intégration de données biologiques : Sélection de sources centrée sur l'utilisateur*"[Bou05]. Le degré de confiance reflète le degré de précision (sûreté) des informations fournies par un service. Un biologiste aurait plus confiance dans le service *SwissProt* ayant un degré de confiance de dix sur dix du fait que la source contient des informations protéiques vérifiées à la main (curated) par des experts du domaine. La complétude reflète la capacité d'une service à fournir le maximum d'instances d'une entité spécifiques.

Le service eFetchGeneService [13] a un niveau de complétude maximum (10), alors que le service OMIM [14] n'a qu'un niveau de complétude de 4 car la ressource OMIM ne contient qu'une partie des gènes séquencés ceux pouvant être associés aux maladies humaines.

Pour ce qui est des propriétés non fonctionnelles, notons qu'elles sont représentées de façon hétérogène au niveau des frameworks OWL-S et WSMO et sont inexistantes dans SAWSDL. Le framework OWL-S se limite à une description textuelle du service, de la catégorie du service et des informations sur le fournisseur comme propriétés non fonctionnelles comme il n'intègre pas des descriptions sur la qualité du service. Au niveau WSMO, il existe deux types de propriétés non fonctionnelles, le premier type appelé *core properties* relève des méta-données Dublin Core [15], et le second type est spécifique à l'aspect qualité du service Web. Cependant, WSMO ne prévoit pas l'ajout de propriétés QdS relatives au domaine du service comme c'est le cas du dans le modèle unifié BioMed. Le modèle de description canonique est illustré dans la figure 5.5.

5.3 Méta-Carte Ontologique

Dans l'exercice qui est le nôtre, nous nous proposons d'utiliser les ontologies sous la forme d'artéfacts sémantiques construits à partir de plusieurs ontologies de domaine, et une ontologie fondationnelle et reposant sur l'explicitation d'une sémantique formelle. Une telle sémantique constitue le fondement des langages utilisés pour la spécification des tâches et des domaines d'intérêt. Ainsi, quand un langage dispose d'une sémantique formelle, la représentation du contenu exprimé dans ce langage est une théorie formelle. C'est dans ce sens que les ontologies, sont en dernier ressort, des théories formelles sous des formes appropriées.

13. eFetchGene est un service Web permettant d'interroger la base génomique Entrez Gene de NCBI, la description du service est accessible à l'adresse http ://www.ncbi.nlm.nih.gov/entrez/eutils/soap/v2.0/efetch_gene.wsdl
14. http ://xml.nig.ac.jp/wsdl/OMIM.wsdl
15. http ://dublincore.org/

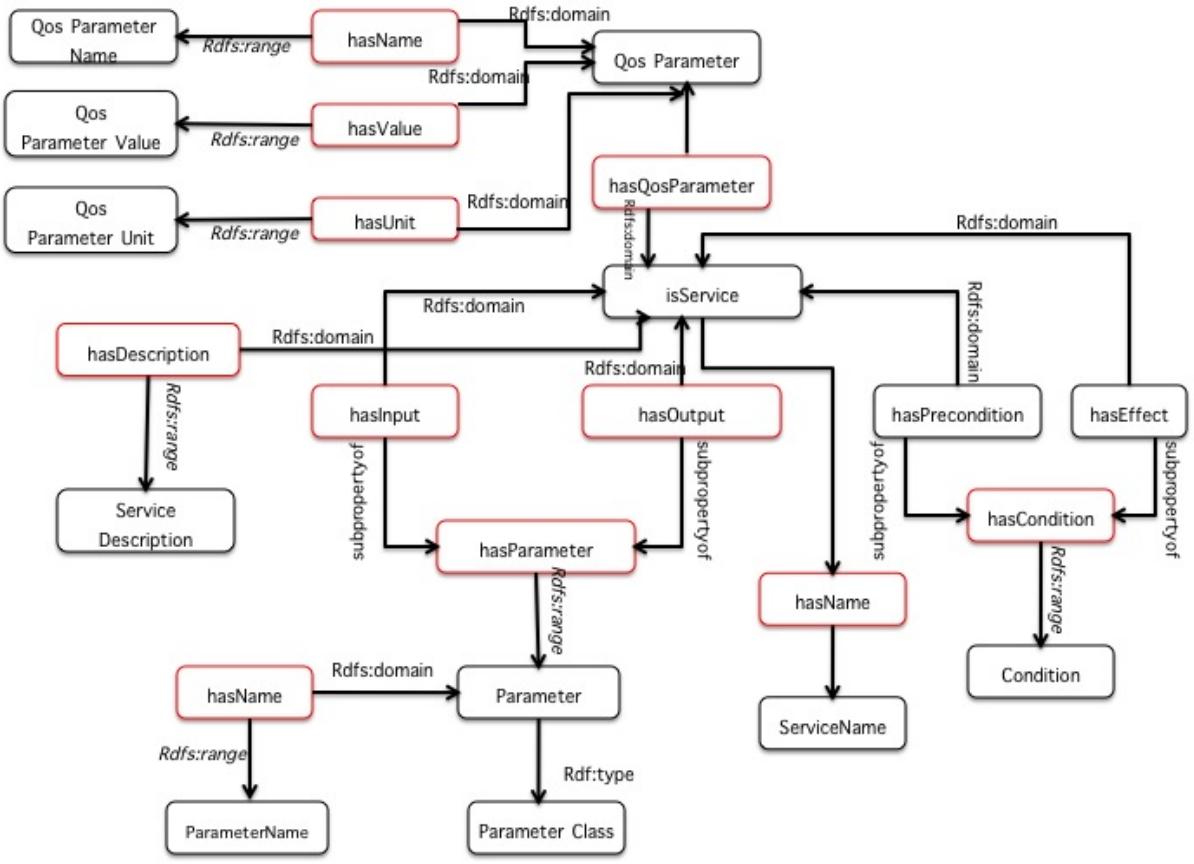


FIGURE 5.5 – Modèle RDF-S de descriptions canoniques

En utilisant différentes ontologies apparaissent les interrelations quand ces ontologies sont utilisées pour découvrir et/ou composer des services. Il en résulte une architecture conceptuelle intégrant les théories et leurs interliens. A cet égard, le Web Sémantique est l'exemple le plus illustratif. L'alignement d'ontologies est un problème très complexe et il le serait plus dans le contexte bioinformatique dans lequel les ontologies ont une couverture sémantique réduite et présentent des incohérences dans leur structure et leur définition des entités, notamment parce qu'elles n'ont pas été fondées sur des principes ontologiques clairs. Nous pensons que l'alignement de ces ontologies sera d'autant plus compréhensible si les ontologies s'adossaient sur des ontologies fondationnelles. En effet, les ontologies fondationnelles ont pour objectif de faciliter et de favoriser l'interopérabilité entre des ontologies, de domaine ou d'application. Elles fournissent une axiomatisation logique rigoureuse explicitant les engagements ontologiques, rendant possible le raisonnement sur les entités et la mise en correspondance des ontologies. Ainsi, l'approche suivie pour la construction d'une méta-carte ontologique consiste à stratifier le niveau ontologique en différentes sous-ontologies situées à différents niveaux d'abstraction selon leur couverture, leur granularité sémantique et leur niveau de formalisation. Schématiquement, nous distinguons trois niveaux d'abstraction :

- Au niveau le plus abstrait, des ontologies fondationnelles telles que DOLCE, SUMO ou GFO apportent un ensemble de concepts et de relations abstraits censés permettre de structurer, par spécialisation, la conceptualisation de n'importe quel domaine ;
- à un niveau médian, une méta-carte ontologique a pour objectif de représenter une sémantique partagée par le biais d'un ensemble de primitives conceptuelles reflétant un consensus local (régional) au niveau d'une communauté de pratique ;
- Enfin, au niveau le plus spécifique, les primitives conceptuelles de la méta-carte sont à leur tour spécialisées pour définir des concepts plus concrets.

Le besoin d'un niveau médian est la conséquence du fait que les concepts apportés par une ontologie fondationnelle sont abstraits et ne sont pas directement instanciables. Cependant, les ontologies fondationnelles offrent des catégories de haut niveau permettant de catégoriser les concepts du domaine visé et de représenter formellement leur sémantique inhérente. En s'appuyant sur les travaux de recherche élaborés par [Isa05, Tem08, Cho09], notre choix s'est porté sur l'ontologie DOLCE. Cette décision est motivée par trois facteurs majeurs :

Le premier facteur est lié à l'axiomatisation de DOLCE qui est riche et bien documentée, ainsi que son fondement sur des principes de structuration explicites. De plus, elle est basée sur la méthodologie OntoClean [GW02, GW04], fournissant ainsi un guide précieux pour structurer des ontologies d'application, particulièrement en ce qui concerne les relations taxonomiques.

Le second argument en faveur de DOLCE est la disponibilité de nombreuses extensions sous forme d'ontologies noyaux de plusieurs domaines, qui ont considérablement servi à maintenir une structure et une méthodologie cohérentes dans la construction des ontologies.

Le troisième argument est lié aux principes de base retenus dans DOLCE, que nous considérons particulièrement pertinents dans le contexte de cette thèse. Le choix délibéré du "biais cognitif" (i.e. dépendant fortement des perceptions humaines et des conventions

sociales) peut s'avérer pertinent pour la modélisation d'artefacts humains, tels que les concepts mathématiques ou dans notre cas, les services en général et les services Web en particulier. Particulièrement, la distinction entre les "endurants" physiques et les "endurants" non physiques nous semble très importante dans le domaine biologique.

Tout d'abord, nous présentons notre cadre de référence DOLCE et ensuite la méta-carte ontologique proposée au niveau ontologique du méta-framework BioMEd.

5.3.1 Le cadre de référence ontologique

DOLCE (*Descriptive Ontology for Linguistic and Cognitive Engineering*) fait partie de la bibliothèque Wonder Web des ontologies fondationnelles (WFOL pour *WonderWeb Foundational Ontologies Library*). DOLCE est proposée comme le premier module de ces bibliothèques, servant de module de référence. Elle a été conçue pour permettre de comparer et rendre explicites les relations et les hypothèses fondamentales des futurs modules de ces bibliothèques.

Comme son acronyme le reflète, DOLCE a un biais cognitif clair, dans le sens où elle vise à appréhender les catégories ontologiques fondamentales du langage naturel et du bon sens humain. DOLCE embrasse une approche descriptive multiplicative, et possède une axiomatisation riche et rigoureuse. Tous les concepts de DOLCE sont considérés comme des propriétés rigides, selon la méthodologie OntoClean qui souligne l'importance de se concentrer d'abord sur ces propriétés [GW02, GW04]. DOLCE est une ontologie fondationnelle de "particuliers", dans le sens où son domaine du discours ne concerne que les particuliers. Les universels sont uniquement utilisés pour organiser et classifier ces particuliers, partitionné en quatre sous-domaines :

- Les Endurants sont des entités "*endurantes dans le temps*". Parmi les Endurants sont distingués les *Physical Objects* et les *Non-Physical Objects*, les premiers étant les seuls à posséder des qualités spatiales directes. Le domaine des Non-Physical Objects recouvre le domaine des entités sociales (ex. : la communauté des chercheurs en biologie moléculaire) et des entités cognitives parmi lesquelles figurent les Concepts (ex. : la notion d'ontologie) réifiant des types d'instances.
- Les Perdurants (PD) sont des entités se déroulant dans le temps. Parmi les Perdurants sont distingués, suivant un principe de cumulativité, les *Statives* et les *Events*. Parmi ces derniers, suivant qu'ils sont atomiques ou non, les *Achievements* sont distingués des *Accomplishments*.
- Endurants et Perdurants ont des *Qualities*, que nous percevons et/ou mesurons (ex. : poids, durée, couleur). Ces Qualities prennent une valeur (*Quale*) dans des régions de valeurs qui sont des *Abstracts*.

On notera également la principale relation entre Endurants et Perdurants, la relation ternaire de participation temporelle signifiant que : *un endurant participe à un perdurant durant un intervalle de temps*.

5.3.2 Ontologies et représentation des connaissances

Le problème posé par la représentation des connaissances est de formaliser dans un langage formel de représentation les connaissances permettant de traiter le problème à résoudre. Traditionnellement, un langage formel se définit à partir d'un alphabet comprenant des symboles de variables, des symboles de fonction et des symboles de prédicat et des symboles d'opérateurs logiques (connecteurs), des quantificateurs, etc. Ces symboles sont des primitives à partir desquelles toutes les formules bien formées du langage sont construites selon des règles de construction syntaxique définies par le langage lui-même. Cependant, il ne suffit pas de préciser comment construire les formules du langage, il faut en outre préciser quelle signification ou sémantique elles peuvent recevoir, pour leur associer des connaissances du domaine. C'est pourquoi un langage formel se doit d'être pourvu d'une sémantique formelle précisant quelles significations associer aux formules en fonction des significations des symboles qu'elles contiennent et de la manière dont ils sont assemblés. A ce point, il est nécessaire de distinguer les primitives logiques des primitives non logiques.

Si les primitives logiques ont leur sémantique spécifiée par le langage formel lui-même, c'est qu'ils possèdent une signification indépendante du domaine. C'est en quelque sorte une sémantique intégrée dans le langage formel. Par exemple, parmi les primitives logiques du langage OWL (Ontology Web Language) nous distinguons : *owl :class*, *owl :property*, *owl :allvaluesfrom*, *owl :maxcardinality*, etc. Par contre les primitives non logiques possèdent une signification dépendante du domaine : non spécifiée par le langage formel, elle doit être explicitement déterminée en fonction du domaine. Les primitives non logiques du domaine biologique incluent gène, protéine, métabolisme, structure tertiaire d'une protéine, domaine fonctionnel, etc.

C'est ainsi, la sémantique des connecteurs logiques et de la composition des symboles de fonction avec les symboles de variables est propre au langage formel et ne dépend pas de ce que les symboles représentent et donc du domaine concerné. Par exemple, la sémantique formelle associée au langage spécifie les tables de vérité des connecteurs une fois pour toutes. C'est dans ce sens, que nous adoptons la définition de B. Bachimont selon laquelle définir une ontologie pour la représentation des connaissances, c'est définir, pour un domaine et un problème donnés, la signature fonctionnelle et relationnelle d'un langage formel de représentation et la sémantique associée. Ce n'est qu'une fois l'ontologie définie qu'il est possible d'associer une sémantique dans le domaine aux constructions syntaxiques du langage. Il n'est possible de représenter des connaissances, c'est-à-dire d'exprimer dans un langage formel les connaissances du domaine, que si les formules et les primitives non logiques qu'elles contiennent sont pourvues d'une sémantique qui permette de savoir quelle connaissance est assumée par cette formule.

5.3.3 Le modèle conceptuel de BioMed

La conception de la méta-carte ontologique s'effectue selon les trois étapes suivantes : une première étape d'engagement sémantique permettant de définir une sémantique référentielle en fixant l'interprétation des termes (des unités linguistiques) ; une deuxième étape permet le passage au niveau formel et ce par le biais de l'engagement ontologique qui permet d'as-

socier à un concept sémantique une extension d'objets et enfin la troisième étape permet l'opérationnalisation de la méta-carte obtenue. Ainsi, La méta-carte sera constituée d'un ensemble de primitives sémantiques (conceptuelles) normalisées et formellement définies. Nous explicitons et illustrons par des exemples les trois étapes pré-citées.

5.3.3.1 L'engagement (la normalisation) sémantique

La première étape a pour objectif d'aboutir à un consensus sémantique portant sur la signification (interprétation) des termes (unités linguistiques, labels) utilisés dans le domaine pour dénoter les concepts. Dans cette étape, il s'agit de recenser les *concepts* clés du domaine visé et leurs différentes connotations. En effet, dans la modélisation ontologique, on cherche à construire des *primitives conceptuelles* dont le sens ne dépend pas des autres primitives et ne dépendent pas du contexte. La détermination de signifiés non contextuels correspond à la phase de la normalisation sémantique qui consiste à choisir parmi les signifiés possibles que peut recevoir une unité linguistique dans un contexte donné, celle qui doit être à chaque fois associée. Cela revient en fait à choisir un contexte de référence dans lequel les termes peuvent être interprétés sans ambiguïté. On retrouve ainsi les conditions d'une sémantique pour laquelle les unités signifient toujours la même chose quels que soient les énoncés qui les contiennent. Ce qui n'est donc pas vrai d'un point de vue linguistique doit être imposé par la normalisation sémantique qui porte sur le choix d'un contexte de référence, celui de la tâche ou du problème qui motive l'élaboration d'une représentation formelle des connaissances. Le point de vue de la tâche permet au modélisateur de fixer quelle doit être la signification de l'unité linguistique considérée. Ainsi, il est nécessaire de normaliser les significations des concepts pour ne retenir, pour chacun d'eux, qu'une seule signification, i.e., qu'une seule interprétation possible dans un monde possible. C'est ainsi que l'engagement sémantique vise à résoudre les problèmes de conceptualisation liés à la polysémie lexicale dans le sens où un terme possède plusieurs interprétations selon le contexte d'usage. Nous illustrons la polysémie lexicale dans l'exemple 1.

Exemple 2 *Le concept gène a été perçu de différents points de vue (polysémie lexicale) :*

1. *Un gène est un fragment d'ADN (aspect structurale) comprenant une partie promotrice suivie d'une alternance de régions d'exons (séquences codantes) et d'introns (séquences non codantes) comme le modélise l'ontologie Sequence Ontology ;*
2. *Un gène est composé des acides nucléotidiques : Adénine, Cytosine, Guanine et Thymine (composition biochimique) ;*
3. *Un gène possède une fonction biologique induite par la protéine (ou les protéines qu'ils encodent) comme le modélisent l'ontologie Gene Ontology ;*
4. *Un gène est localisé sur un chromosome.*

À la fin de l'étape de l'engagement sémantique, nous obtenons un ensemble de primitives conceptuelles valable dans la seule région du monde modélisée où les concepts retenus correspondent bien à ceux de l'ontologie qui sont par définition décontextualisées. Ainsi, le sens d'un libellé est principalement défini selon deux types de sémantiques :

- Une sémantique référentielle qui établit le lien entre l'intension et l'extension d'un concept impliquant notamment des relations d'hyperonymie/hyponymie et d'holonymie/méronymie ;
- sémantique différentielle qui associe à chaque unité linguistique les unités voisines (celles qui sont utilisées en même temps qu'elle dans les contextes d'usage) pour la définir par les identités et différences qu'elle entretient avec ses voisines au niveau extensionnel.

Exemple 3 *En l'occurrence, les ADN (DNA) et les ARN (RNA) sont des acides nucléiques composé de nucléotides comprenant trois types de molécules (sémantique référentielle) :*

Nucleotide hasComponent (Heterocyclic Base AND Phosphate AND 5-Carbon Sugar)

Les ADN et les ARN se distinguent par la nature du sucre contenu dans leurs nucléotides. Au niveau de l'ADN, le sucre (5-carbon sugar) est le désoxyribose alors qu'au niveau de l'ARN, il s'agit du ribosome (sémantique différentielle).

Ainsi, la sémantique différentielle permet de décrire les unités entre elles par les identités qui les unissent et les différences qui les distinguent. Toutes les unités se déterminent à partir d'une unité générique ultime, une unité racine, à laquelle elles appartiennent toutes. Par ailleurs, toutes les unités se déterminent d'une part par l'unité générique à laquelle elles appartiennent et d'autre part par les différences qui les distinguent. Cela signifie que le réseau des unités est un réseau d'héritage de propriétés où les unités filles héritent des sèmes d'une unité générique mère. Enfin, l'engagement sémantique dégage une ontologie (ici, une méta-carte ontologique) valable seulement localement (régionalement) dans le cadre d'un domaine et d'une tâche.

5.3.3.2 L'engagement ontologique

Cette étape correspond à la définition d'une sémantique formelle. La sémantique formelle ne considère plus des notions sémantiques (concepts sémantiques) mais des extensions, c'est-à-dire l'ensemble des objets qui vérifient des propriétés définies en intension dans l'étape précédente, propriétés ayant une définition formelle à ce niveau. En effet, les concepts sémantiques s'inter-définissent par identités et différences. L'identité correspond au fait qu'une notion est comprise dans une autre : l'insuline est une protéine ayant une activité hormonale (fonction biologique), de ce fait, elle est une hormone. Les concepts hormone et protéine ne sont pas disjoints (au niveau extensionnel), l'insuline hérite les caractéristiques de protéine (macromolécule) et celle d'hormone (activité hormonale). Cela implique que l'on retrouve dans les relations unissant les concepts formels les relations d'héritage des concepts sémantiques, mais nécessairement pas les exclusions. La structure des concepts formels n'est plus obligatoirement un arbre, mais plus généralement une structure de treillis. Cela se comprend d'ailleurs facilement pour la raison suivante : si la sémantique des concepts est référentielle, les relations entre les concepts sont des relations entre ensemble. La structure des concepts formels doit correspondre à la structure algébrique des ensembles, c'est-à-dire un treillis. La formalisation des relations se définit

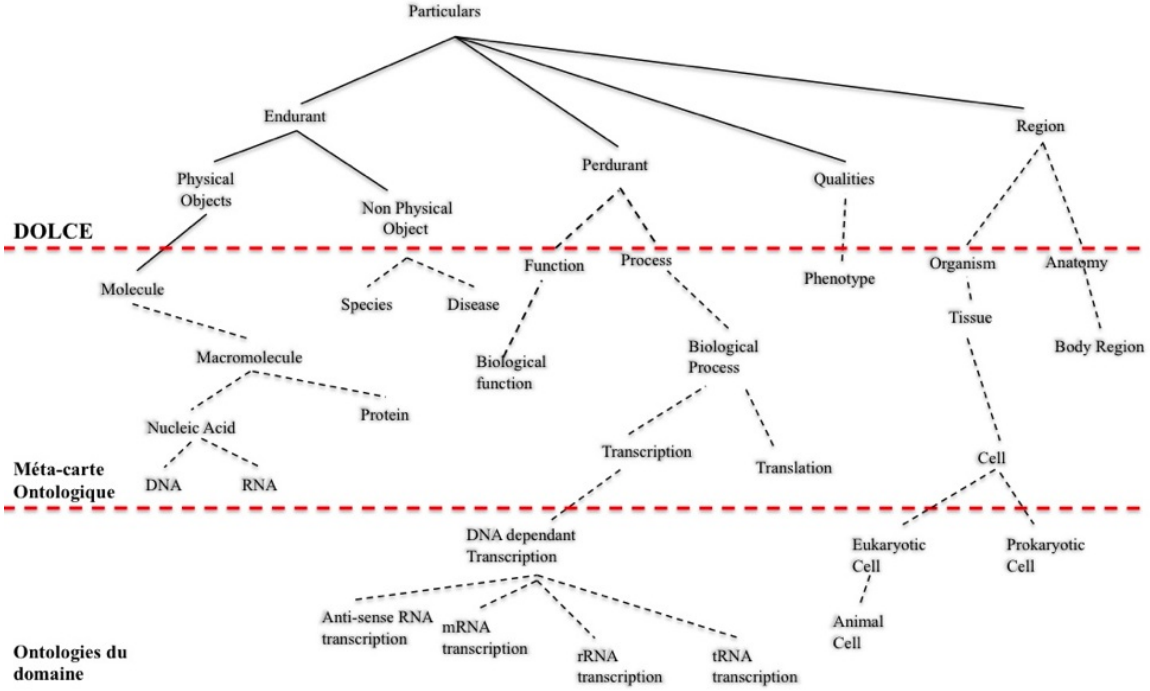


FIGURE 5.6 – La méta-carte ontologique

alors facilement : une relation se définit comme un sous-ensemble du produit cartésien des extensions des concepts composant sa signature.

L'engagement ontologique élaboré pour la conception de la méta-carte émane du soubassement la théorie de l'ontologie fondationnelle DOLCE qui nous permet de qualifier la nature intrinsèque des concepts modélisés en les rattachant à des méta-catégories.

5.4 Annotations sémantiques pour la sémantisation de SW

L'enrichissement sémantique de descriptions de services Web a pour objectif de faciliter la découverte des services pertinents (en réponse à un besoin utilisateur) et l'explicitation de leurs paramètres (fonctionnels) et de leurs profils d'usage. Deux étapes primordiales de cet enrichissement sémantique sont la définition d'un modèle (sémantique) de description de services Web et l'annotation sémantique proprement dite. La définition d'un modèle de description, tel que le modèle d'unification proposé dans la section 2, fournit les primitives conceptuelles (nommées descriptives au niveau du modèle d'unification) permettant de représenter abstraitement un service Web. L'annotation sémantique consiste à instancier les primitives conceptuelles du modèle unifié de descriptions en termes de concepts, relations et instances définis formellement au niveau de la méta-carte ontologique. Nous nommons l'artéfact résultant du processus d'annotation sémantique, dans le contexte des services Web, une ***description sémantisée***. En l'occurrence, le terme courant en littérature est celui de *description sémantique* reflète plutôt le modèle de des-

cription qui a pour objectif de capturer les éléments ou attributs décrivant la sémantique d'un service tels que ceux de sa signature fonctionnelle, sa catégorie, ou son domaine.

Nous explicitons brièvement dans cette section le *quoi* et le *comment* du processus d'annotation sémantique en général et celui de l'annotation sémantique de SW en particulier. Nous présentons quelques définitions de base portant sur l'annotation sémantique. L'annotation sémantique est définie par [KPT+04] comme *la génération de métadonnées spécifiques et d'un schéma d'utilisation, ayant pour but de rendre possible de nouvelles méthodes d'accès à l'information et d'améliorer les méthodes existantes.* Il s'agit d'assigner à des entités dans un texte (dans le cas d'un ressource textuelle) un lien vers leur description sémantique. Cette définition est selon nous incomplète puisqu'elle omet de préciser que ces métadonnées sont générées sur la base d'une ontologie du domaine si on parle précisément d'annotation sémantique. Ainsi, Florence Amardeilh définit l'annotation sémantique comme *une représentation formelle d'un contenu, exprimée à l'aide de concepts, relations et instances décrits dans une ontologie*, et liée à la ressource originelle [ALM05]. L'aspect formel sous entend que les annotations sont formalisées dans un langage formel pour être exploitées par des agents logiciels.

Enfin, Jérôme Euzenat distingue le processus d'annotation de l'artéfact résultant du processus d'annotation. Il introduit une terminologie relative à l'annotation sémantique de documents. Les définitions qui suivent sont extraites de la référence [Euz05] qui définit le processus d'annotation comme étant le processus permettant de rattacher du sens à un ensemble de symboles (texte, image, vidéo) grâce à un ensemble d'assertions. Une assertion peut être *générique* ou *individuelle*, elle est générique si elle s'applique à plusieurs individus et individuelle si elle s'applique à un individu particulier dans un domaine d'interprétation. Un *schéma* est un ensemble d'assertions génériques qui décrit des entités génériques pour exprimer le contenu. Une *description* est un ensemble d'assertions individuelles. La *connaissance de contexte* est un ensemble d'assertions qui peuvent être schématiques ou descriptives. Une ontologie est un ensemble d'assertions qui décrit les concepts impliqués dans le domaine. L'ontologie étant de la connaissance commune utilisée pour comprendre le contenu, elle fait évidemment partie de la connaissance de contexte. Cependant, Euzenat distingue l'ontologie, composée en partie de connaissance générique, et le reste de la connaissance de contexte qui contient exclusivement des descriptions individuelles. Le contenu d'un document est le sens du document envisagé sous l'angle le plus général. Lorsque le contenu est exprimé dans un langage formel et attaché au document, il est appelé **annotation**.

Dans le contexte de notre travail, nous nommons schéma d'annotation un ensemble d'assertions génériques décrivant les propriétés sémantiques d'un service Web. En l'occurrence, le modèle RDF-S de descriptions canoniques de SW présenté en section 2 est un exemple de schéma d'annotation. Ainsi, l'annotation sémantique de services Web permet de décrire le lien entre les primitives d'un schéma d'annotation et un ensemble de connaissances formalisées dans une ontologie. Le résultat du processus d'annotation sémantique est une description sémantisée d'un SW comportant un ensemble d'assertions qui expriment **la sémantique explicite et formelle du service**. En d'autres termes, si $\mathcal{O}$ est une ontologie, $\mathcal{K}$ l'ensemble des connaissances du contexte, D est la description sémantisée du service, $\mathcal{I}_S$ est la sémantique d'un service $\mathcal{S}$ et $\models$ est la conséquence logique, alors

$$\mathcal{O} \bigcup \mathcal{K} \bigcup D \models \mathcal{I}_S.$$

Cette sémantique est explicite si elle est exprimée par le biais d'un ensemble d'assertions dans un langage déclaratif. Elle est formelle si elle est exprimée sur la base d'un schéma d'annotation. Il est important de disposer d'annotations formelles pour pourvoir les réutiliser dans des mécanismes inférentiels. Au niveau du méta-framework BioMed, le schéma d'annotation est formalisé par le modèle RDF-S de la description unifiée de SW. Sur la base de ce schéma d'annotation, les descriptions sémantisées sont réifiées en un ensemble de triplets RDF, constituant un graphe que nous nommons *graphe de services annotés*.

Définition 2 (Graphe de services annotés) *Un graphe de services est un graphe orienté biparti G=(V,E) tel que les noeuds dans V sont soit des services soit des concepts ontologiques. Les arcs dans E désignent les relations entre les services et les concepts ontologiques, tels que, si C est les hyper-concepts de la méta-carte ontologique et S est un service alors :*

- *(S, hasInputType, C) indique que S a pour entrée un paramètre de type sémantique le concept C,*
- *(S, hasOutput, C) indique que S a pour sortie un paramètre de type sémantique le concept C.*

Un exemple de graphe de services annotés est illustré dans la figure 5.7. Enfin, la qualité et la pertinence des annotations dépendent étroitement de la couverture sémantique et de la pertinence de l'ontologie choisie pour annoter le service. Dans notre cas, la couverture sémantique est assez large dés lors que la méta-carte ontologique offre une pertinence élevée et révèle les différentes facettes tant au niveau intensionnel qu'extensionnel.

5.5 Une sémantique axiomatique comme fondement du modèle inférentiel

Le raisonnement déductif est une composante essentielle du modèle conceptuel du méta-framework BioMed. Le moteur de raisonnement proposé permet d'effectuer différents types de raisonnement dans l'objectif de raisonner à la fois sur les connaissances ontologiques de la méta-carte ontologique, les connaissances descriptives et les annotations sémantiques. A chaque typologie de raisonnement, nous définissons un ensemble d'axiomes constituant dans leur ensemble la sémantique axiomatique sur laquelle repose le moteur de raisonnement, nous distinguons les axiomes suivants :

- **Axiomes ontologiques** : les axiomes du niveau ontologique permettent de raisonner au niveau ontologique afin d'inférer de nouvelles connaissances en exploitant par exemple certains propriétés algébriques des relations telles que la transitivité, ou déterminer la clôture transitive d'un concept à partir d'une hiérarchie de concepts.
- **Axiomes des services** : ces axiomes permettent de déduire de nouvelles propriétés des services Web en exploitant à la fois les connaissances du domaine (explicites à partir de la méta-carte ontologique ou implicites sous forme d'axiomes ontologiques) et les annotations sémantiques des services Web. Ces axiomes vont enrichir la couverture sémantique des propriétés des services (par exemple, on associera à un service

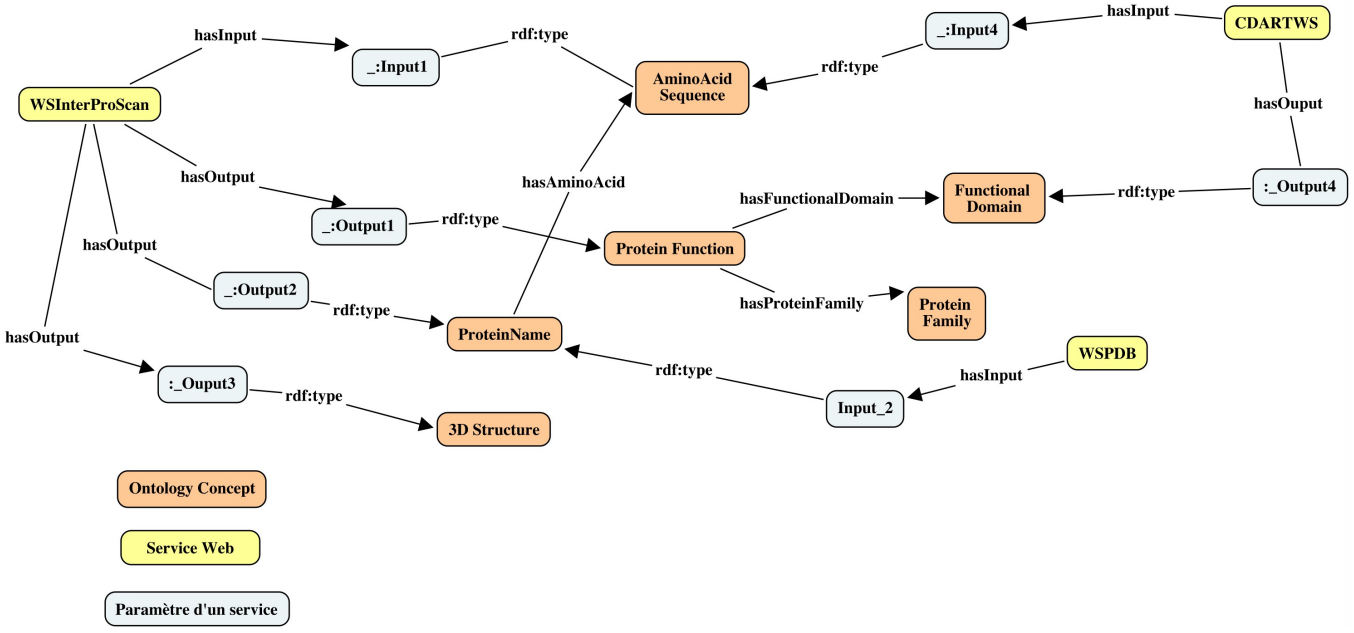


FIGURE 5.7 – Graphe de services annotés

qui accepte en entrée un paramètre de type concept C_1, une annotation comprenant l'arbre ontologique du concept C_1 obtenu par calcul de la clôture transitive du concept).

– **Axiomes des instances** définissent des axiomes pour le concept spécifique/général auquel une instance appartient.

Selon que les annotations sémantiques sont créées en termes d'instances de concepts, de concepts, ou de relations sémantiques nous définissons cinq groupes d'axiomes relatifs à la *relaxation assertionnelle*, la *relaxation par généralisation et la relaxation par spécialisation*, la *relaxation fonctionnelle et la relaxation fonctionnelle inverse*. Cette axiomatique permet d'élargir (enrichir) les descriptions sémantisées de SW en inférant de nouvelles propriétés sémantiques d'où le terme de relaxation. Nous explicitons dans ce qui suit chaque groupes d'axiomes en les illustrant par des exemples.

5.5.1 La relaxation assertionnelle (au niveau des instances)

La relaxation assertionnelle permet de déduire le type sémantique pouvant être accepté en entrée ou renvoyé comme sortie par un service Web. Cette relaxation peut être effectuée aux différents niveaux de stratification du méta-framework BIOMED, en effet, les appariements existants entre le niveau ressource comprenant les instances des services, le niveau description comprenant les descriptions unifiées et le niveau ontologique comprenant les types sémantiques (les hyper-concepts et leurs instances) du domaine. Cette relaxation assertionnelle est réifiée par les deux axiomes suivants :

$$\frac{[(I,\texttt{rdf :type},c_1),(s,\texttt{rdf :type},isService),(s,\texttt{hasInput},I)]}{[(s,\texttt{InputType},c_1)]} \quad \text{(A1)}$$

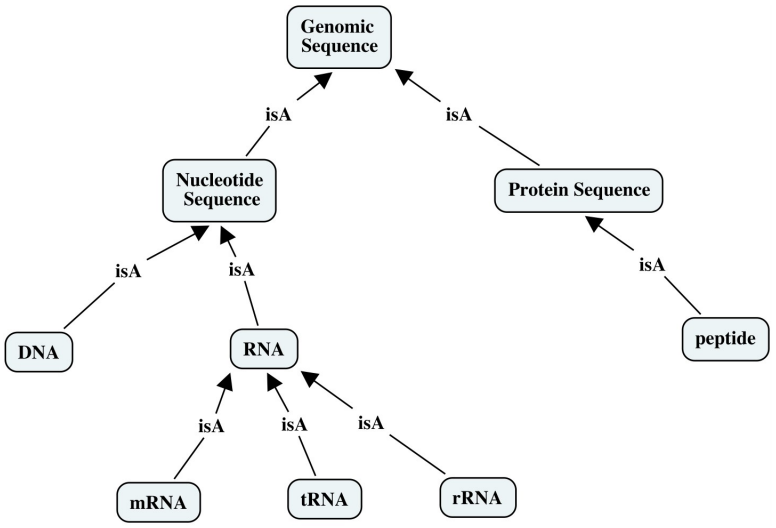


FIGURE 5.8 – Arbre Ontologique du concept *Genomic Sequence*

$$\frac{[(O,\texttt{rdf :type},c_2),(s,\texttt{rdf :type},isService),(s,\texttt{hasOutput},O)]}{[(s,\texttt{OutputType},c_2)]} \quad \text{(A2)}$$

L'axiome (A1) permet de déduire le type sémantique accepté en entrée par un service alors que l'axiome (A2) permet de déduire le type sémantique d'une sortie d'un service.

5.5.2 La relaxation par généralisation et par spécialisation

La relaxation par spécialisation et par généralisation permet de déduire de nouvelles propriétés sémantiques du service en exploitant les connaissances d'un arbre ontologique défini comme suit.

Définition 3 (Arbre Ontologique) *Un arbre ontologique est un ensemble de concepts $\mathcal{A}$ muni d'une relation d'ordre partiel $\leq$ vérifiant les trois propriétés suivantes :*
- $\forall\ c \in \mathcal{A}$, $c \leq c$ *(réflexivité)*
- $\forall\ c_1, c_2, c_3 \in \mathcal{A}$, *si* $c_1 \leq c_2$ *et* $c_2 \leq c_3$ *alors* $c_1 \leq c_3$ *(transitivité)*
- $\forall\ c_1, c_2 \in \mathcal{A}$, *si* $c_1 \leq c_2$ *et* $c_2 \leq c_1$ *alors* $c_1 = c_2$ *(antisymétrie)*

En d'autre termes, un arbre ontologique permet de catégoriser les concepts d'un domaine sous forme d'une hiérarchie de concepts. La figure 5.8 illustre l'arbre ontologique de l'hyper-concept *séquence génomique* de la méta-carte ontologique.

Le principe de la relaxation par spécialisation est défini par la règle d'annotation suivante : si S est un service Web qui accepte en entrée un paramètre I annoté sémantiquement par un concept C, alors I sera annoté par tous les sous-concepts subsumant C dans l'arbre ontologique. La relaxation par spécialisation permet d'étendre la portée sémantique d'une propriété d'un service pour inclure tous les concepts spécifiques d'un super-concept annotant un paramètre. La relaxation par spécialisation est définie formellement par les deux axiomes suivants :

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasInput},I),(I,\texttt{rdf :type},c),(c_1,\texttt{rdfs :subClassOf},c)]}{[(s,\texttt{InputType},c_1)]} \quad \text{(A3)}$$

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasOutput},O),(O,\texttt{rdf :type},c),(c_2,\texttt{rdfs :subClassOf},c)]}{[(s,\texttt{OutputType},c_2)]} \quad \text{(A4)}$$

Le haut d'un axiome définit les conditions devant être vérifiées (prémisses) et le bas de l'axiome définit la conclusion ou l'annotation à réaliser.

D'un autre côté, la relaxation par généralisation se base sur le constat suivant : si S est un service Web qui retourne en sortie un paramètre O annoté sémantiquement par un concept C, alors O sera annoté sémantiquement par tous les super-concepts de C dans l'arbre ontologique. Les axiomes (A5) et (A6) permettent de calculer la fermeture transitive d'un concept et d'inférer de nouvelles annotations sémantiques enrichissant la description sémantisée d'un service Web. La relaxation par généralisation est décrite formellement en utilisant les axiomes suivants :

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasInput},I),(I,\texttt{rdf :type},c_1),(c_1,\texttt{rdfs :subClassOf},c)]}{[(s,\texttt{InputType},c)]} \quad \text{(A5)}$$

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasOutput},O),(O,\texttt{rdf :type},c_2),(c_2,\texttt{rdfs :subClassOf},c)]}{[(s,\texttt{OutputType},c)]} \quad \text{(A6)}$$

Nous illustrons dans l'exemple suivant les relaxations par généralisation et par spécialisation.

Exemple 4 *Blastn est un service Web pour l'alignement multiple de séquences nucléotidiques. L'alignement multiple de séquences permet de retrouver des régions de similarité entre des séquences biologiques. Le graphe de service annoté est illustré par la figure 5.9 montrant les annotations sémantiques du service Blastn.*

Une relaxation par spécialisation permet d'élargir la portée sémantique du paramètre en entrée "Nucleotide sequence" en annotant le paramètre I_1 par les sous-concepts DNA sequence, RNA et mRNA. D'un autre côté, une relaxation par généralisation permet d'annoter le paramètre en sortie O_1 avec les super-concepts du concept "Nucleotide sequence" à savoir le concept "Genomic Sequence".

5.5.3 La relaxation fonctionnelle

La relaxation fonctionnelle permet de déduire de nouvelles annotations sémantiques en se basant les propriétés fonctionnelles d'un concept. Une propriété fonctionnelle se définit comme suit : si P est une propriété fonctionnelle entre deux concepts C_1 et C_2, alors pour chaque instance du concept C_1, il existe une et une seule instance de C_2. Ainsi, l'axiome de la relaxation fonctionnelle se définit comme suit : si S est un service qui accepte en entrée un paramètre I annoté sémantiquement par un concept C_1 alors le paramètre I sera annoté par tous les concepts reliés à C_1 par une propriété fonctionnelle P. Les axiomes (A7) et (A8) formalisent la relaxation fonctionnelle :

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasInput},I),(I,\texttt{rdf :type},c_1),(c_1,p,c_2),(p,\texttt{rdf :type},isFunctionalProperty)]}{[(s,\texttt{hasInputType},c_2)]} \quad \text{(A5)}$$

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasOutput},O),(O,\texttt{rdf :type},c_1),(c_1,p,c_2),(p,\texttt{rdf :type},isFunctionalProperty)]}{[(s,\texttt{hasOutputType},c_2)]} \quad \text{(A6)}$$

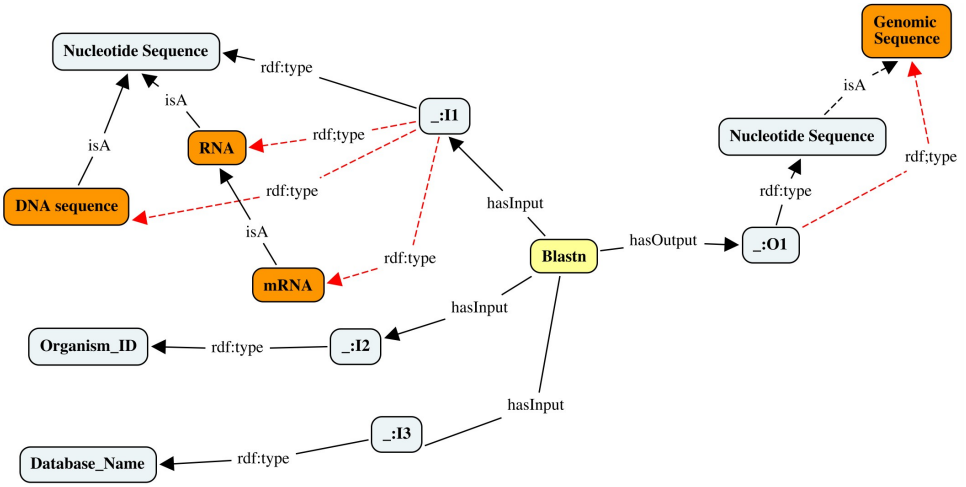


FIGURE 5.9 – Graphe annoté du service Blastn

Exemple 5 *WSInterProScan*[16] *est un service Web prend en entrée une séquence protéique (AminoAcid Sequence) pour retourner l'identifiant de la protéine correspondante à la séquence en entrée, ainsi que la fonction de la protéine et sa structure tertiaire (3D structure). Les annotations sémantiques du service WSInterProScan est illustré par le graphe de la figure 5.10. Au niveau ontologique, La fonction d'une protéine renseigne sur le domaine fonctionnel de la protéine et la famille de la protéine, d'où les propriétés fonctionnelles "hasFunctionalDomain" entre "Protein Function" et "Functional Domain" et "hasProteinFamily" entre "Protein Function" et "Protein Family". La relaxation fonctionnelle permet d'annoter respectivement le paramètre de sortie $Output_1$ par les concepts "Functional Domain" et "Protein Family". par toutes les propriétés fonctionnelles d'un concept annotant.*

5.5.4 La relaxation fonctionnelle inverse

D'un autre côté, une propriété inversement fonctionnelle est définie comme suit : si P est une propriété inversement fonctionnelle entre deux concepts C_1 et C_2 alors pour chaque instance de C_2, il correspondra une et une seule instance de C_2. Par conséquent, l'axiome de la relaxation fonctionnelle inverse est définie comme suit : si S est un service qui accepte en entrée un concept C_2 alors S accepte en entrée tous les concepts reliés par une relation inversement fonctionnelle à C_2 (ce qui correspond au domaine des relations dont le co-domaine est le concept C_2). La relaxation inversement fonctionnelle s'applique aussi bien aux entrées et aux sorties d'un service et se réifie en les axiomes (A7) et (A8) formalisant ainsi la relaxation inversement fonctionnelle.

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasInput},I),(I,\texttt{rdf :type},c_2),(c_1,p,c_2),(p,\texttt{rdf :type},isInverseFunctionalProperty)]}{[(s,\texttt{hasInputType},c_1)]}$$

16. Le service Web WSInterproscan permet d'interroger la base InterPro. Cette base de données fédère un grand nombre de banques de données sur les domaines protéiques (PROSITE, PRINTS, Pfam, ProDom, SMART et d'autres).

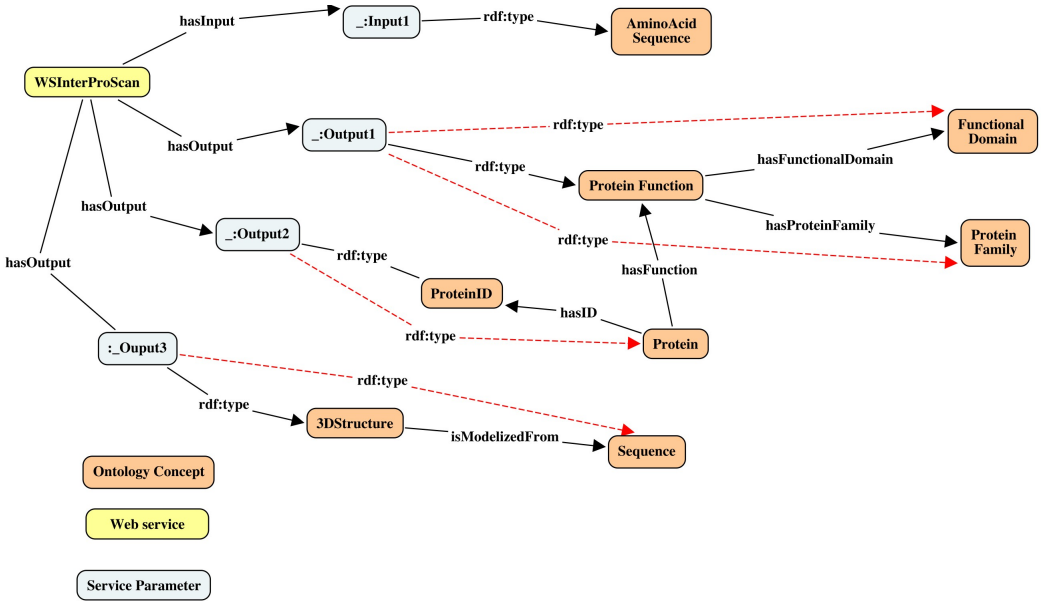


FIGURE 5.10 – Illustration de la relaxation fonctionnelle

(A7)

$$\frac{[(s,\texttt{rdf :type},isService),(s,\texttt{hasOutput},O),(O,\texttt{rdf :type},c_2),(c_1,p,c_2),(p,\texttt{rdf :type},isInverseFunctionalProperty)]}{[(s,\texttt{hasOutputType},c_1)]}$$

(A8)

Exemple 6 *Pour prédire la fonction d'un gène, les protocoles scientifiques en bioinformatique se basent sur la séquence ADN correspondant au gène. "hasDNASequence" est une relation fonctionnelle inverse entre un gène et sa séquence ADN, ceci signifie qu'à une instance de séquence ADN correspond un et un seul gène. Ainsi, un service qui accepte une séquence ADN, acceptera le gène relié (par relaxation fonctionnelle inverse), de plus, toutes les propriétés fonctionnelles du gène telles que son nom (GeneName), son symbole (GeneSymbol) ou son identifiant (GeneID) seront déduits comme de nouvelles propriétés sémantiques du service comme illustrées dans la figure 5.11.*

5.6 Conclusion

Nous avons proposé dans ce chapitre un méta-framework stratifié pour la médiation de services Web sémantiquement hétérogènes dénommé BIOMED. Ce méta-framework intègre un modèle unifié pour la description et l'annotation sémantique de SW. Le modèle d'alignement sémantique de BIOMED permet de normaliser la sémantique des services Web par le biais d'un ensemble de primitives canoniques descriptives. Il s'agit à ce niveau de réconcilier les descriptions émanant des trois principaux frameworks de descriptions sémantiques de SW ; à savoir OWL-S, WSMO, SAWSDL. La construction d'un référentiel de descriptions sémantisées de services Web est réalisée sur la base de ce modèle d'alignement. Les descriptions sont annotées sémantiquement en se basant sur la *méta-carte*

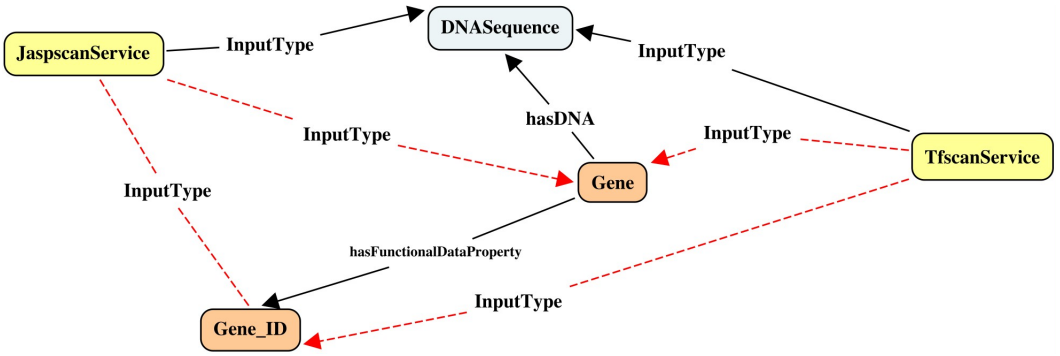


FIGURE 5.11 – Illustration de la relaxation fonctionnelle et fonctionnelle inverse

ontologique du domaine bioinformatique. Le modèle inférentiel de BIOMED se base sur une sémantique axiomatique formalisant une large typologie de raisonnement permettant d'inférer de nouvelles annotations sémantiques élargissant ainsi la couverture sémantique des descriptions sémantisées contenues dans le référentiel. Nous montrons dans le chapitre suivant comment les techniques de raisonnement déductif permettent une médiation sémantique entre les descriptions sémantisées des services et la sémantique des besoins en services des utilisateurs exprimée par le biais de requêtes.

Chapitre 6

La découverte de Services Web dans BioMed

Sommaire

6.1 Introduction

Nous présentons dans ce chapitre une approche originale pour la découverte de services Web combinant à la fois les propriétés fonctionnelles et non fonctionnelles comme critères de sélection. Cette phase de sélection repose sur le moteur inférentiel conçu dans le cadre du méta-framework BioMed et qui permet de déduire l'ensemble des solutions qui satisfont fonctionnellement et non fonctionnellement les contraintes d'une requête de découverte.

Le modèle inférentiel repose sur la sémantique axiomatique présentée dans le chapitre 5 permettant d'élargir la couverture sémantique des annotations des services et ce par biais de techniques de relaxation permettent de déduire un appariement vérifiant ***au mieux*** les contraintes d'une requête de découverte. Une phase de classement (ranking) complète la phase de sélection des solutions possibles et permet de classer l'espace des solutions possibles sur la base de leurs propriétés non fonctionnelles. Les expérimentations menées dans le cadre de l'évaluation de l'approche de découverte montre l'impact des techniques de raisonnement sur la taille de l'espace des solutions possibles d'une requête et l'intérêt des techniques de ranking pour déceler les meilleures solutions de l'espace en termes de propriétés QdS [GVRA09, GVRA10].

L'opérationnalisation du méta-framewok BioMed se base sur l'implémentation d'un **méta-service Web déductif** qui repose sur une base déductive de données [ALVR07, ALV08a]. Nous explicitons cette opérationnalisation dans la section 6.4.

6.2 Formalisation du problème de découverte

Cette section a pour objectif de définir formellement le problème de découverte de services tel qu'abordé dans l'approche BioMed. Nous nous reposons sur des descriptions unifiées de services Web afin de proposer une approche de découverte adaptée aux trois frameworks de descriptions sémantiques ciblés. Nous rappelons qu'une solution de découverte se base sur un algorithme de "*matchmaking*" pour déduire un appariement sémantique entre une description de service Web et une requête de découverte exprimant le besoin de l'utilisateur en termes de propriétés fonctionnelles des services. Les critères de sélection de services retenus dans l'approche de découverte proposée sont sur le plan fonctionnel, les entrées, les sorties et au plan non fonctionnel, les critères QdS. Les entrées et les sorties sont annotées sémantiquement et les propriétés non fonctionnelles décrivent la qualité du service. La définition 4 explicite la description canonique d'un service Web adoptée au niveau de l'approche de découverte proposée.

Définition 4 (Description canonique d'un service Web) *Un service Web S est défini par un tuplet S=<func, QdS> où func désigne un ensemble de propriétés fonctionnelles, en l'occurrence, l'ensemble des entrées typées I nécessaires à l'exécution du service S et l'ensemble des sorties typées O retournées par S après exécution, et QdS est un ensemble de propriétés non fonctionnelles, chaque propriété QdS est représentée par un triplet (p,v,u), où p est un paramètre QdS, v est la valeur du paramètre QdS et u est l'unité dans laquelle la valeur du paramètre est exprimée.*

La signature fonctionnelle du service notée $func$ est annotée sémantiquement en termes de concepts ontologiques. L'exemple 7 illustre la description canonique d'un service Web.

Exemple 7 *Soit S_1 un service Web qui reçoit en entrée un paramètre nommé E_0 désignant l'identifiant d'un gène (GeneID) et qui retourne en sortie trois paramètres E_1, E_2 et E_3 annotés respectivement avec les concepts AAsequence (séquence aminoacide), ProteinID (identifiant d'une proteine) et 3Dstructure (Structure 3D d'une protéine). Ce service permet aux scientifiques d'explorer l'ensemble des protéines (identifiants, séquences et struc-*

tures tertiaires) codées par un gène dont l'identifiant est spécifié en entrée. Les descriptions QdS du service S_1 sont les suivantes : un temps de réponse de 4 secondes, une Disponibilité de 98 pour cent et un Débit de données de 15 invocations par seconde.

Alors, la description canonique du service S_1=(func, QdS) est définie comme suit :
- *func =(I= {E_0 : GeneID}, O={E_1 : AAsequence, E_2 : ProteinID, E_3 : 3Dstructure })*
- *QdS = {(temps de réponse, 4, secondes), (Disponibilité, 98, pourcentage), (Débit de données, 15, invocations par seconde)}*

Les descriptions canoniques sont déclarées en RDF-S au niveau du référentiel de services construit dans le contexte du méta-framework BioMed. chaque service Web est déclaré par un triplet RDF-S comme suit (s, `rdf :type`, isService). Les paramètres en entrée et en sortie sont déclarés respectivement par les triplets (s,`hasInput`, i) et (s, `hasOutput`, o), leurs types sémantiques sont déclarés par (p, `rdf :type`, c) où p est un paramètre en entrée ou en sortie d'un service s. p peut être une collection de concepts, nous disposons de trois types de collections, (p, `rdf :bag`, c), (p, `rdf :alt`, c) et (p, `rdf :seq`, c). Les prédicats (s, `hasQdSParameter`, p), (s, `hasValue`, v) et (p, `hasUnit`, u) permettent de définir respectivement un paramètre QdS d'un service, sa valeur et son unité de mesure. Le graphe RDF de services Web sémantisés est défini comme suit.

Définition 5 (Graphe RDF de services Web sémantisés) *Soit U un ensemble d'URI (de services Web, de paramètres, de concepts ontologiques, de propriétés ontologiques, de types de données et d'ontologies), L un ensemble infini de littéraux, Un graphe RDF de services Web sémantisés est constitué d'un ensemble de triplets RDF (v_1, v_2, v_3) $\in U \times$ (`rdf :type`, `rdf :bag`, `rdf :seq`, `rdf :alt`, `hasInput`, `hasOutput`, `hasQdSParameter`, `hasValue`, `hasUnit`) $\times (U \cup L)$ tel que v_1 est le sujet, v_2 est le prédicat et v_3 est l'objet. Un graphe de services Web sémantisés ne comprend pas de noeud blanc.*

Une requête de découverte permet d'exprimer le besoin de l'utilisateur, elle permet de spécifier un ensemble de contraintes fonctionnelles et/ou non fonctionnelles. La requête de découverte spécifie un ensemble de contraintes fonctionnelles en restreignant les types sémantiques acceptés et retournés par un service ou un ensemble de contraintes non fonctionnelles portant sur les valeurs des paramètres QdS. Nous explicitons dans ce qui suit les notions d'appariement fonctionnel et d'appariement non fonctionnel d'une requête de découverte.

6.2.1 Appariement fonctionnel

La définition 6 définit la notion de requête de découverte de sorte qu'elle exprime uniquement des contraintes fonctionnelles restreignant les types sémantiques en entrée ou en sortie.

Définition 6 (Requête de découverte) *Une requête de découverte est un tuplet Q=<I,-O>, où I est un ensemble de paramètres typés désignant les entrées de la requête Q et O est un ensemble de paramètres typés désignant les sorties de la requête Q.*

Il est primordial de disposer d'un langage de requête destiné à interroger les graphes RDF(S) et qui pourrait être adopté pour exprimer les requêtes de découverte dans le méta-

framework BioMed. Plusieurs langages d'interrogation de ressources RDF(S) existent tels que le langage RQL (RDF Query Language) [KAC+02], le langage TRIPLE [1] [SD02], ou plus récemment le langage SPARQL [2] (SPARQL Protocol and RDF Query Language) [PS08]. Nous adoptons ce dernier pour exprimer les requêtes de découverte dans BioMed, la raison de ce choix est qu'il est le langage de référence pour l'interrogation des annotations RDF selon la recommendation du W3C [3].

Les requêtes SPARQL sont définies à partir de patrons de graphes qui sont fondamentalement des graphes RDF avec des variables (noeuds blancs).

Exemple 8 *Un exemple de requête de découverte exprimée en SPARQL se présente comme suit :*

```
PREFIX rdf : <http ://www.w3.org/1999/02/22-rdf-syntax-ns>
SELECT DISTINCT fiS
WHERE {
   fiS     rdf :type    isService
   fiS    hasInput    p1
   p1 rdf :type GeneId
   fiS hasOutput o1
   o1 rdf :type ProteinId
   fiS hasOutput  o2
   o2 rdf :type Sequence
}
```

La sémantique formelle des graphes RDF est décrite en détails dans [Hay04] et est traitée de la perspective de la logique du premier ordre en se basant sur la notion de modèle, d'interprétation et d'implication logique (Théorie des modèles [4]). Un graphe RDF est un ensemble de triplets (s,p,o), tel que s est un sujet, p est un prédicat (propriété qui met en évidence une relation entre s et o) et o est un objet. Nous définissons l'interprétation

1. http ://triple.semanticweb.org/
2. http ://www.w3.org/TR/rdf-sparql-query/
3. http ://www.w3.org/
4. La théorie des modèles est une branche de la logique mathématique. Elle a été formulée d'une façon complète et cohérente d'abord par Alfred Tarski qui a appelé sa théorie la sémantique du calcul des prédicats et cela pour deux raisons : (1) Elle donne une définition de la vérité et de la conséquence logique, indépendante de ce que donnent les démonstrations en logique, (2) Elle donne une réponse partielle à la question de la signification du langage, parce que les mots ont du sens s'ils permettent de faire des phrases vraies dans un monde possible. Selon Tarski, Un modèle sert d'abord de structure pour valider une théorie logique ou mathématique. Il existe deux notions de ≪ consistance ≫ ou de ≪ cohérence ≫ d'une théorie, a priori différentes, mais équivalentes en logique classique du premier ordre, d'après le théorème de complétude de Gödel : (a) un notion sémantique : une théorie est satisfaisable s'il existe un modèle dans lequel elle est vraie, (b) une notion syntaxique : une théorie est non contradictoire si on ne peut en dériver à la fois une formule et sa négation. Un système de déduction est dit ≪ correct ≫ si l'existence d'un modèle donne la certitude de travailler sur une théorie qui ne débouchera pas sur une contradiction. L'intérêt est que pour montrer la cohérence ou consistance d'une théorie, il est souvent plus facile d'en déterminer un modèle que de montrer qu'on ne peut dériver de contradiction.

   La sémantique formelle du langage RDF basée sur la théorie des modèles rend possible la définition de règles d'inférences valides. Une autre façon de définir la sémantique du langage RDF est de donner une traduction de la syntaxe RDF vers une logique formelle liée à une théorie des modèles. Cette approche dénommée ≪ sémantique axiomatique ≫ a été suggérée dans cette thèse.

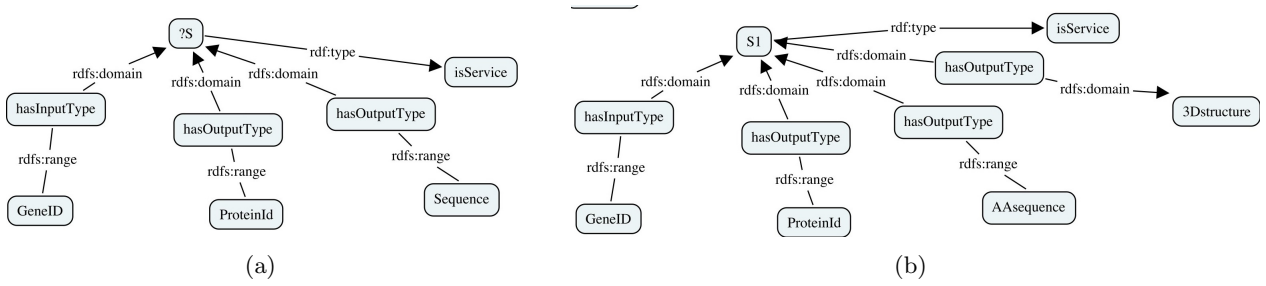


FIGURE 6.1 – Exemples de graphes de requête et de service

$\mathbb{I}$ d'un graphe RDF de services Web sémantisés consiste en : (1) un ensemble non vide Res d'URI de services Web, de variables (paramètres), de concepts ontologiques, de propriétés et d'ontologies, (2) un sous ensemble de littéraux $Lit \subseteq Res$ (3) un ensemble de propriétés $Prop \subseteq Res \times Res$ et une fonction de mapping U $\rightarrow Res \times Prop$ et L $\rightarrow Lit$.

La définition de l'appariement fonctionnel se base sur un *matching structurel exact* entre le graphe d'une requête et celui d'un service en se basant sur la notion d'isomorphisme de graphes.

Définition 7 (Appariement fonctionnel exact) *Etant donnés les graphes RDF d'une requête de découverte Q et d'un service Web S, notés respectivement G_Q et G_S, Il existe un appariement fonctionnel exact entre G et S si et seulement si il existe un isomorphisme entre le graphe de la requête G_Q et le graphe du service G_S. Un isomorphisme est une application bijective π des termes de G_Q dans les termes de G_S de sorte que pour chaque triplet (s,p,o) $\in G_Q$, il existe (π(s), p, π(o)) qui est un triplet de G_S.*

Un isomorphisme implique l'existence d'un appariement exact entre le graphe de la requête et le graphe d'un service au niveau structurel. Ainsi, le service S_1 dont le graphe RDF est illustré dans la figure 6.1.(a) n'est pas une solution exacte de la requête Q dont le graphe RDF est illustré par la figure 6.1.(b).

Cependant, il existe un appariement approximatif entre le service S_1 et la requête Q puisque le service Web accepte les mêmes entrées (compatibilité des types sémantiques) que la requête Q et retourne en sortie l'identifiant d'une protéine (ProteinID) et l'instance d'une séquence d'acides aminés notée *AAsequence* qui est un sous-type de *séquence*.

L'intérêt des techniques de raisonnement présentées dans cette thèse et plus précisément au niveau du modèle inférentiel du méta-framework BIOMED est de reposer sur le principe de la relaxation de paramètres afin de déduire différents degrés d'approximation d'appariement fonctionnel entre une requête de découverte et une description sémantisée d'un SW en reposant sur l'ensemble des connaissances ontologiques fourni par la méta-carte ontologique. En l'occurrence, une relaxation par généralisation permet de mettre en évidence la sémantique référentielle d'un ensemble d'objet en leur associant une intension, ainsi, il est possible de déduire que le service S_1 a comme sortie le concept "*séquence*" comme le montre la figure 6.2.

Nous distinguons cinq niveaux d'appariement fonctionnel entre une requête de découverte et une description sémantisée d'un service Web : le *matching exact*, le *matching fonction-*

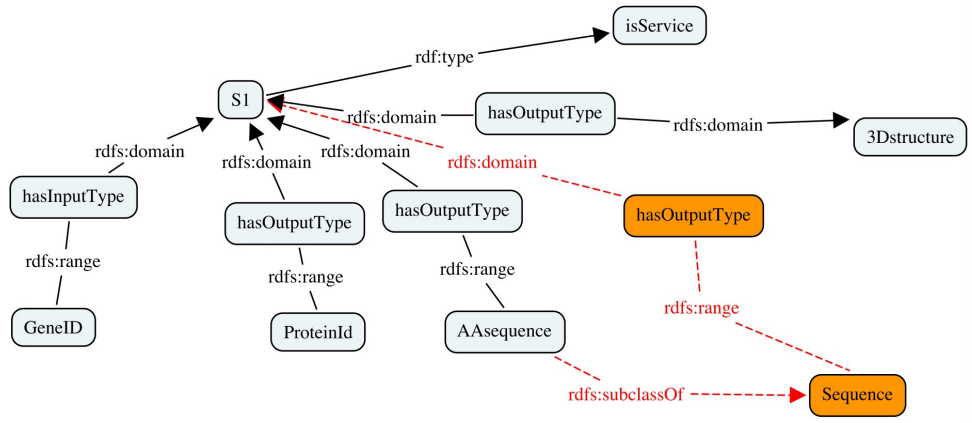


FIGURE 6.2 – Graphe annoté du service S_1

nel, le *matching en plugin* et le *matching en subsomption* que nous définissons comme suit :

1. ***Le matching exact*** entre une requête de découverte Q et un service S, noté Q ≡ S, illustre le cas où pour chaque entrée typée i de l'ensemble I_Q des entrées d'une requête Q (respectivement une sortie typée o de l'ensemble O_Q des sorties), il correspond une entrée $i^{'}$ (respectivement une sortie $o^{'}$) du service S ayant le même type sémantique que i, i.e., le même concept ontologique annotant. Formellement, le matching exact est défini comme suit :
   $\forall\ (i : C) \in I_Q, \exists\ (i^{'} : C^{'}) \in I_S$ tel que $(i : C)$ sémantiquement équivalent à $(i^{'} : C^{'})$
2. ***Le matching fonctionnel*** entre une requête de découverte Q et un service S, noté Q ⇒ S, illustre le cas où pour chaque entrée typée i de l'ensemble I_Q des entrées (respectivement une sortie typée o de l'ensemble O_Q des sorties) d'une requête Q, il correspond une entrée $i^{'}$ (respectivement une sortie $o^{'}$) du service S qui soit :
   - de même type sémantique que le concept annotant i (respectivement o),
   - ou de type sémantique une propriété fonctionnelle du concept annotant i (respectivement o),
   - ou de type sémantique une propriété fonctionnelle inverse du concept annotant i (respectivement o),
3. ***Le matching en plugin*** entre une requête Q et un service S, noté Q ⊑ S, illustre le au cas où :
   - pour chaque entrée typée i de l'ensemble des entrées I_Q d'une requête Q, il correspond une entrée $i^{'}$ du service S qui soit de même type sémantique que le concept annotant i ou un sous-concept du concept annotant i, et
   - pour chaque sortie typée o dans l'ensemble des sorties O_Q d'une requête Q, il correspond une sortie $o^{'}$ du service S qui soit de type sémantique que le concept annotant o ou un super concept du concept annotant o.
4. ***Le matching en subsomption*** entre une requête Q et un service S, noté S ⊑ Q, correspond au cas où :

- pour chaque entrée typée i de l'ensemble des entrées I_Q d'une requête Q, il correspond une entrée o du service S qui soit de type sémantique que le concept annotant de i ou un super concept du concept annotant de i.
- pour chaque sortie typée o dans l'ensemble des sorties O_Q d'une requête Q, il correspond une sortie $o^{'}$ du service S qui soit de type sémantique que le concept annotant de o un sous-concept du concept annotant de o.

Nous illustrons les trois types de matching approximatif (plug-in, subsomption, fonctionnel) avec l'exemple suivant.

Exemple 9 *Nous reprenons la requête de l'exemple 8 exprimant le besoin d'un scientifique de services lui permettant d'étudier les protéines codées par le gène TP53. La spécification de la requête est la suivante :*

- *Q=(I,O) tel que I = { i_1 : geneId} et O={o_1 : ProteinId,o_2 :AAsequence}*

Soit S_1, S_2 et S_3 trois SW ayant respectivement les descriptions fonctionnelles F_{S_1}, F_{S_2}, F_{S_3} déclarées comme suit :

- *$Func_{S_1}$ =(I,O) tel que I= {E_0 :geneId} et O={E_1 :ProteinId, E_2 :Sequence},*
- *$Func_{S_2}$ =(I,O) tel que I= {E_0 :DNAsequence} et O={E_1 :ProteinId, E_2 :AASequence},*
- *$Func_{S_3}$ =(I,O) tel que I= {E_0 :geneId} et O={E_1 :AAsequence}.*

Aucun service ne matche exactement avec la requête Q. Cependant, les matchings approximatifs suivants peuvent être déduits par relaxation des propriétés fonctionnelles des services S_1, S_2 et S_3 :

- *Un matching en plugin entre Q et S_1, $Q \sqsubseteq S_1$,*
- *Un matching fonctionnel entre Q et S_2, $Q \Rightarrow S_2$,*
- *Un matching en subsomption entre Q et S_3, $S_3 \Rightarrow Q$.*

Nous retenons seulement les services S_1 et S_2 comme solutions possibles à la requête Q étant donné que S_3 satisfait partiellement la requête en renvoyant seulement un sous ensemble de l'ensemble des sorties Q.

Un mécanisme de classement de l'ensemble des solutions d'une requête se basera sur le type d'appariement établi et se définit comme suit :

$$exact > fonctionnel > \ plug\text{-}in > subsomption > échec$$

En pratique, le nombre de services pouvant satisfaire fonctionnellement une requête de découverte peut se révéler très élevé. Les techniques de relaxation permettent d'élargir la couverture sémantique des descriptions sémantisées des services induisant l'élargissement de l'espace de solutions d'une requête de découverte. Les contraintes non fonctionnelles permettent d'exprimer les attentes des utilisateurs en termes de qualité de service. L'extension de la requête de découverte par l'ajout de contraintes non fonctionnelles aura pour impact de restraindre l'espace de solutions aux services satisfaisant à la fois les contraintes fonctionnelles et non fonctionnelles.

6.2.2 Appariement fonctionnel et non fonctionnel

La requête de découverte étendue contenant à la fois des contraintes fonctionnelles et non fonctionnelles, est définie comme suit :

Définition 8 (Requête de découverte étendue) *Une requête de découverte étendue Q est définie par un tuplet Q=<F,NF>, où F est l'ensemble de propriétés fonctionnelles et NF est un ensemble de propriétés non fonctionnelles.*

- *Les propriétés fonctionnelles F d'une requête Q sont représentées par un tuplet (I,O) où I est un ensemble de paramètres typéss désignant les entrées de la requête Q et O est un ensemble de paramètres typés désignant les sorties de la requête Q.*
- *Les propriétés non fonctionnelles NF d'une requête Q sont représentées par un ensemble de quadruplets (P,C,V,U) où P est un paramètre non fonctionnel, C est un opérateur logique de comparaison, V est la valeur du paramètre et U est l'unité du paramètre P. L'évaluation de la requête Q doit respecter la conjonction de expressions exprimés par NF.*
  Si (P,** C_i**,** V_i**,** U_i**) et (P,** C_j **,** V_j **,** U_j**)** $\in$ ***NF alors $C_i = C_j$**,** $V_i = V_j$ ***et*** $U_i = U_j$**.**

Exemple 10 *Nous proposons d'étendre la requête de découverte exprimée dans l'exemple 8 par un ensemble de contraintes non fonctionnelles exprimées comme suit : un temps de réponse inférieur ou égal à 5 secondes et une disponibilité plus grande ou égale à 95. La requête Q est exprimée comme suit :*

Q=(F,NF)

- *F=(I,O)*
  - *I = { E_0 : GeneID }*
  - *O = { E_1 : sequence, E_2 : ProteinID }*
- *NF = {(P,C,V,U)}*
  - *NF = {(temps de réponse, $\leqslant$, 5, secondes), (Disponibilité, $\geqslant$,95, pourcentage)}*

Un service S =(*func*, *QdS*) satisfait les contraintes non fonctionnelles d'une requête Q=(F,NF) ssi pour chaque quadruplet (P,C,U,V) $\in NF$, il existe un triplet (p,v,u) $\in QdS$, tels que :

- P et p sont équivalents
- l'expression v C V est valide avec C $\in \{>, <, \geqslant, \leqslant, =, \neq\}$
- les deux paramètres sont exprimés dans la même unité.

Une nouvelle approche de découverte proposée dans le cadre du méta-framework BioMed est présentée dans la section suivante permettant de sélectionner les services qui satisfont fonctionnellement et non fonctionnellement une requête de découverte.

6.3 La solution BioMed pour la découverte de services Web

La solution de découverte proposée permet de déduire un espace de solutions $\mathbb{S}$ pour une requête Q comprenant l'ensemble les meilleurs services qui satisfont fonctionnellement ***et*** non fonctionnellement une requête de découverte et cela en deux phases : une **phase d'appariement fonctionnel** permettant de sélectionner les solutions exactes et approchées d'une requête de découverte et une **phase de sélection et de classement** pour classer l'ensemble des solutions sélectionnées en termes de l'ensemble des attributs QdS spécifié dans la requête étendue. La phase d'appariement fonctionnel repose sur le

moteur de raisonnement qui exploite les descriptions sémantisées des service Web et l'ensemble des connaissances ontologiques du domaine pour inférer les propriétés implicites des services en se basant sur la sémantique axiomatique (inférentielle). Une phase de classement est proposée afin de retourner à l'utilisateur l'ensemble des k-meilleurs services en termes de leurs propriétés non fonctionnelles.

La section 6.3.1 présente les techniques de relaxation proposées dans la phase de raisonnement. Ensuite, la section 6.3.2 explicite l'approche de classement proposée.

6.3.1 Phase de l'appariement fonctionnel

Au niveau de la phase de raisonnement, les différents types de relaxation exécutés sont les suivants :

- *La relaxation par spécialisation des paramètres en entrée d'un service S* est définie comme suit :
  - ♣ si I est un paramètre en entrée d'un service S et I est annoté par un concept C alors I est annoté par tous les sous-concepts de C.
- *La relaxation par généralisation des paramètres en sortie d'un service* est définie comme suit :
  - ♣ si O est un paramètre en sortie d'un service S et O est annoté avec le concept C alors O est annoté par tous les super-concepts de C.
- *La relaxation fonctionnelle et fonctionnelle inverse des paramètre en entrée d'un service* sont définies comme suit :
  - ♣ si I est un paramètre en entrée d'un service S et I annoté par le concept C_1 et P est une propriété fonctionnelle de C_1 vers C_2 alors I est annoté par le concept C_2
  - ♣ si I est un paramètre en entrée d'un service S et I est annoté par le concept C_2 et P est une propriété fonctionnelle inverse de C_1 vers C_2 alors I est annoté par le concept C_1.
- *La relaxation fonctionnelle et fonctionnelle inverse des paramètres en sortie d'un service* sont définies comme suit :
  - ♣ si O est un paramètre en sortie d'un service S et O annoté par un concept C1 et P est une propriété fonctionnelle de C1 vers C2 alors O est annoté par le concept C2
  - ♣ si O est un paramètre en sortie d'un service S et O est annoté avec le concept C2 et P est une propriété fonctionnelle inverse de C1 vers C2 alors O est annoté par le concept C1.

L'algorithme suivant permet de déduire l'ensemble des solutions satisfaisant une requête au niveau fonctionnel. La phase de raisonnement est opérationnalisée par un méta-service Web déductif modélisé par deux composantes : une composante extensionnelle comprenant l'ensemble des connaissances ontologiques et les annotations sémantiques des services Web et une composante intensionnelle comprenant un ensemble de règles logiques implémentant l'algorithme de l'appariement fonctionnel.

Algorithme : **Appariement fonctionnel**

Inputs : G_O : graphe ontologique,
G_S : graphe de services,
$\mathbb{A}$: l'ensemble des axiomes du modèle inférentiel,
G_Q : graphe de requête de découverte
Outputs : S : ensemble de services
Début
– Générer le graphe minimal G'_S par application de chaque axiome $A_i \in \mathbb{A}$ au graphe initial G_S de sorte que :
pour chaque $s \in G_S$ faire
pour chaque $c \in Input(s)$ faire
$Input(s) \leftarrow Input\ (s) \bigcup subclasses(\text{c},G_O) \bigcup functionalProperties(\text{c}, G_O) \bigcup inverseFunctionalProperties(\text{c}, G_O)$
fin Pour
pour chaque $c^{'} \in Output(s)$ faire
$Output(s) \leftarrow Output\ (s) \bigcup superclasses(c^{'},G_O) \bigcup functionalProperties(c^{'}, G_O) \bigcup inverseFunctionalProperties(c^{'}, G_O)$
fin Pour
fin pour
– Normaliser G'_S en fusionnant tous les noeuds se référant à un même concepts.
– Calculer les homorphismes entre Q et G'_S ainsi obtenu.
Fin

6.3.2 Phase de sélection et de classement des services

La seconde phase de l'approche de découverte proposée consiste à classer l'ensemble des services retenus par le moteur inférentiel et vérifiant les contraintes non fonctionnelles de la requête étendue afin de retourner à l'utilisateur les meilleurs services en terme de qualité de service. La solution proposée combine deux approches connues dans le contexte des bases des données à savoir celle se basant sur l'opérateur de *skyline* [BKS01] et celle de l'algorithme *top-k* [IBS08]. Pour introduire l'approche de classement proposée, nous introduisons deux notions fondamentales à la compréhension de l'approche à savoir la dominance et le skyline.

Définition 9 (Dominance) *Etant donné un ensemble points $\mathcal{S}$ dans un espace multidimensionnel $\mathcal{D}$ de dimension n, un tuple t_i domine un tuple t_j dans $\mathcal{D}$, si et seulement si $\forall\ d_x \in \mathcal{D}$, $t_i(d_x) \geqslant t_j(d_x)$ et $\exists\ d_y \in \mathcal{D}$, $t_i(d_y) > t_j(d_y)$.*

Définition 10 (Skyline) *Etant donné un ensemble points $\mathcal{S}$ dans un espace multidimensionnel $\mathcal{D}$ de dimension n, un tuple t_i est un objet skyline dans $\mathcal{D}$ si et seulement si aucun autre tuple $\forall\ t_j\ (\neq t_i) \in \mathcal{S}$ ne domine t_i dans $\mathcal{D}$. SKY($\mathcal{D}$) désigne l'ensemble des objets skyline de $\mathcal{S}$ dans $\mathcal{D}$.*

Ainsi, la technique de skyline consiste en la constitution d'un ensemble de dominance *a la Pareto* (Voir Annexe B). Dans le contexte de la découverte de services, un skyline de

Service	Disponibilité	Temps de réponse
NCBIBlastService	98 %	4 secondes
runBlat	97 %	2 secondes
WSFASTA	96 %	1 seconde

Table 6.1 – Skyline de trois services

services désigne l'ensemble des services qui ne sont dominés par aucun autre service dans un espace multidimensionnel comprenant tous les critères QdS d'une requête de découverte ; ces critères sont supposés être de même poids.

Exemple 11 *Supposons qu'un scientifique recherche un service pour l'alignement multiple de séquences nucléotidiques vérifiant les contraintes non fonctionnelles suivantes : Disponibilité au minimum supérieur à 98 % et un temps de réponse inférieur à 5 secondes. Les trois services Web NCBIBlastService*[5]*, runBlat*[6] *et WSFASTA*[7] *sont des outils d'alignement multiple de séquences et donc satisfont fonctionnellement et non fonctionnellementl le besoin de l'utilisateur comme le montre leurs descriptions non fonctionnelles décrites dans la Table*

De plus, nous supposons que l'utilisateur est intéressé par le meilleur service entre les trois services sélectionnés. Du point de vue disponibilité, runBlat est meilleur WSFASTA et NCBIBlastService sont meilleurs que runBlat. Cependant, en considérant la propriété temps de réponse, WSFATA est meilleur que runBlat et runBlat est meilleur que NCBIBlastService. Etant donné qu'aucun service n'est meilleur que tous les autres pour toutes les propriétés non fonctionnelles, ces services sont incomparables, i.e., ils forment ainsi un skyline de services induits par les propriétés QdS considérés dans la requête de découverte. Il est important de disposer de techniques de classement pour choisir les meilleurs solutions d'un skyline, sachant que ce genre de problème toutes les propriétés QdS sont d'égale importance.

La composante de découverte dans BioMed repose sur l'algorithme WTKSF (Web Top-k Skyline Frequency) proposé dans [GVRA09, GVRA10]. Cet algorithme permet d'identifier les *top-k* services d'un skyline en calculant le nombre de fois qu'un service apparaît dans les différents sous-espaces dans l'espace multi-dimensionnel induits par les n dimensions. Etant donné un espace multi-dimensionnel de dimension n, il existe 2^n-1 sous espaces non vides de propriétés QdS (critères). En l'occurrence, étant donné un critère multidimensionnel comprenant quatre paramètres QdS : la *disponibilité* (noté A pour Availibility), le *débit* (noté T pour Throughtput), la *fiabilité* (noté R pour reliability) et la *documentation* (noté D). L'ensemble des sous-espaces de l'espace multi-dimensionel forment la structure d'un treillis comme l'illustre la figure 6.3 .

La métrique de fréquence skyline permettant de mesurer la qualité d'un service est définie comme suit : SFM (Skyline Frequence Metric) = $\frac{\sum_{\forall q_i \in m} s(q_i)}{2^d - 1}$. La métrique SFM permet de calculer le nombre d'occurrences du service dans les sous-espaces de l'espace multi-

5. http ://www.ebi.ac.uk/Tools/webservices/wsdl/WSNCBIBlast.wsdlfikjkj
6. http ://biomoby.org/services/wsdl/inb.bsc.es/runBlat
7. http ://www.ebi.ac.uk/Tools/webservices/wsdl/WSFasta.wsdl

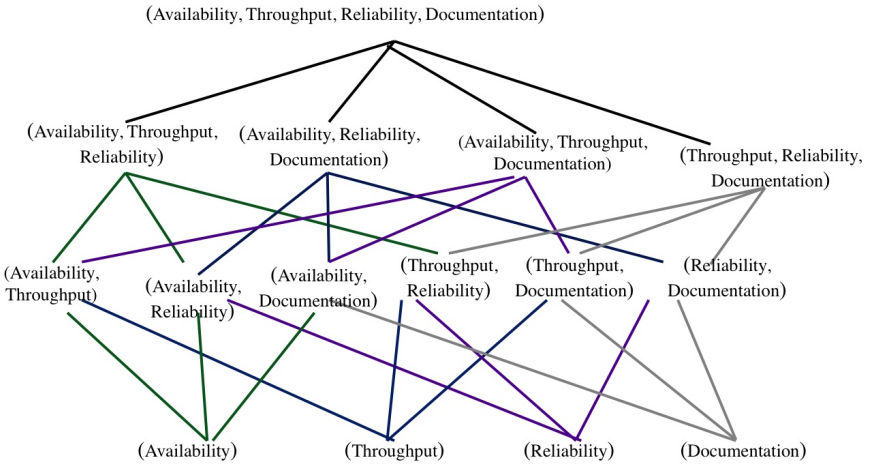


FIGURE 6.3 – Ensemble des sous-espaces multi-dimensionnels

dimensionnel. L'algorithme proposé détermine les *top-k* services satisfaisant une requête de découverte étendue ayant les meilleures valeurs dans la plus grande partie des critères QdS de la requête. L'algorithme WTKSF, est une version améliorée de l'algorithme TKSI (*Top-k SkyIndex*) présenté dans [GV09]. *WTKSF* diffère de l'algorithme *TKSI* du fait qu'il repose sur une heuristique pour guider la recherche des meilleurs services. *WTKSF* présuppose que l'ensemble des services sélectionnés est trié dans d listes ordonnées, d étant le nombre de paramètre QdS d'une requête étendue. L'algorithme se base sur l'heuristique suivante pour décider si un service S_i est meilleur qu'un service S_j :

- S_i est un point extrême ayant les meilleures valeurs dans au minimum $\frac{d}{2}$ paramètres QdS de la requête. Un service est un point extrême dans une dimension m, s'il possède la meilleure valeur de cette dimension.
- Il n'existe pas de service S_k meilleur que le service S_i dans (d-l) dimensions, où l est le nombre de dimensions dans lesquelles S_j est un point extrême.
- Il existe un service S_l meilleur que le service S_j dans (d-l) dimensions, où l est le nombre de dimensions dans lesquelles S_i est un point extrême.

Dans le cas où les deux services ne vérifient pas l'heuristique proposée, l'algorithme WTKSF calcule la valeur de la métrique SFM pour les services S_i et S_j afin de décider lequel des deux services est meilleur. L'algorithme s'arrête quand k services sont déterminés. Ainsi, le skyline des services est incrémentalement construit, et la métrique SFM est calculée seulement pour un nombre réduit de services.

Le problème de découverte des meilleurs services en terme de QdS est défini comme suit dans l'approche BioMed :

Définition 11 (Problème de découverte des Top-k Skyline Services) *Etant donné une requête de découverte étendue Q=(F,NF), un ensemble de services SW = {S_1, S_2,...,S_n} et la métrique de fréquence skyline SFM, le problème de découverte des top-k skyline services est de trouver l'ensemble SW′, tel que*

- $SW' \subseteq SW$,

- *Pour chaque service s ∈ SW', s satisfait les contraintes fonctionnelles et non fonctionnelles de la requête Q,*
- *Il existe au plus k services dans SW' ayant les meilleures valeurs de la fonction SFM, ces k services existent dans le skyline induit par les propriétés non fonctionnelles*

Nous proposons dans ce qui suit une opérationnalisation du méta-framework BioMed comme méta-service Web déductif permettant de représenter à la fois la sémantique descriptive des services Web, la sémantique référentielle et différentielle explicitée par la méta-carte ontologique et la sémantique axiomatique émanant du modèle inférentiel.

6.4 Opérationnalisation du méta-framework BioMed

Dans cette section, nous présentons le cadre opérationnel du méta-framework BioMed comme un méta-service Web déductif comprenant deux méta-composantes :

- Une méta-composante extensionnelle comprenant un ensemble de méta-prédicats extensionnels désignant un ensemble de primitives sémantiques de déclaration de connaissances,
- Une méta-composante intensionnelle comprenant un ensemble de méta-prédicats intensionnels désignant un ensemble d'axiomes.

6.4.1 Fondements de base

Nous considérons l'alphabet d'un langage logique sans symboles de fonctions composé de trois ensembles : VAR, CONST et PRED dénotant respectivement des ensembles de variables, de constantes et de prédicats. VAR, CONST et PRED sont de plus deux à deux disjoints et l'ensemble VAR est infini dénombrable, alors que les ensembles CONST et PRED sont finis. Un terme est défini comme étant une constante ou une variable. Un atome p(t_1 , . . . , t_n) est formé par un symbole de prédicat p d'arité n et une liste d'arguments (t_1 , . . . , t_n) où chaque t_i ($1 \leqslant i \leqslant n$) est un terme. Nous utilisons aussi la notation p($\bar{t}$) pour représenter l'atome p(t_1 , . . . , t_n). Un littéral est un atome p(t_1 , . . . , t_n) (littéral positif) ou sa négation *not* p(t_1 , . . . , t_n) (littéral négatif). Un littéral clos contient seulement des constantes parmi ses arguments. Un fait est un littéral positif clos.

Sur la base d'un tel langage logique, une théorie ontologique peut être exprimée pour décrire les connaissances spécifiques d'un domaine ainsi que l'ensemble des axiomes permettant d'inférer de nouvelles connaissances. Nous introduisons la définition d'une base de connaissances ontologiques.

Définition 12 (Base de Connaissances Ontologiques) *Une base de connaissances ontologiques $\mathcal{KB}$ d'une ontologie $\mathcal{O}$ est un tuple $\mathcal{KB}$ = ($\mathcal{F}$,$\mathcal{I}$), tel que $\mathcal{F}$ est un ensemble de faits représentant les connaissances explicites de l'ontologie à savoir les concepts, les propriétés des concepts, les relations sémantiques (taxonomiques ou non) entre concepts ainsi que les instances, et $\mathcal{I}$ est un ensemble d'axiomes permettant d'inférer de nouveaux faits.*

Nous nous sommes inspirés des travaux de [RRV08] formalisant les ontologies comme une base de données déductive [RU93] nommée par les auteurs *Deductive Ontology Data-*

PREDICATS EXTENSIONNELS	**DESCRIPTION**
`isOntology(O)`	An ontology has an Uri `O`
`isImpOntology(O1,O2)`	Ontology `O1` imports ontology `O2`
`isClass(C,O)`	`C` is a class in ontology `O`
`isOProperty(P,D,R)`	`P` is an object property with domain `D` and range `R`
`isDProperty(P,D)`	`P` is a data property with domain `D`
`isTransitive(P)`	`P` is a transitive property
`isSymmetric(P)`	`P` is a symmetric property
`subClassOf(C1,C2)`	`C1` is subclass of `C2`
`subPropertyOf(P1,P2)`	`P1` is subproperty of `P2`
`AllValuesFrom(C,P,D)`	`C` has property `P` with all values in `D`
`maxCardinality(C,P,N)`	`C` has property `P` with maximum cardinality `N`
`isIndividual(I,C)`	`I` is an individual belonging to class `C`
`isStatement(I,P,J)`	`I` is an individual that has property `P` with value `J`
PREDICATS INTENSIONNELS	**DESCRIPTION**
`areSubClasses(C1,C2)`	`C1` are the direct and indirect subclasses of C2
`areImpOntologies(O1,O2)`	`O1` import the ontologies `O2` directly and indirectly
`areSubProperties(P1,P2)`	`P1` are the direct and indirect subproperties of `P2`
`areClasses(C,O)`	`C` are all the classes of an ontology and its imported ontologies `O`
`areStatements(S,P,O)`	`(S,P,O)` are the statements for inverse, transitive or symmetric `P`
`areIndividuals(I,C)`	`I` are the individuals of a class and all of its direct and indirect superclasses `C I` are the individuals that participate in a property and belong to its domain or range `C`, or are values of a property with all values in `C`

Table 6.2 – Méta-prédicats Extensionnels et Intensionnels du formalisme DOB

base (DOB). Comme toute base déductive, une DOB comprend une base extensionnelle (EDB : extensional DataBase) et une base intensionnelle (IDB : Intensional DataBase). La base extensionnelle d'une DOB est conçue de sorte qu'elle comprend toutes les connaissances explicites d'une ontologie par le biais d'un ensemble de méta-prédicats extensionnels correspondant aux primitives de déclaration de connaissances dans un langage ontologique. La base intentionnelle correspond aux différents axiomes permettant de raisonner sur les connaissances de la base extensionnelle. Comme langage de représentation d'ontologies, les auteurs ont considéré le langage OWL (Ontology Web Language). La table 6.2 illustre la traduction d'un sous-ensemble des constructeurs du langage de définition d'ontologies OWL vers le formalisme DOB.

L'exemple 12 illustre la traduction d'un fragment de l'ontologie Galen [8] en une DOB.

8. http ://www.co-ode.org/galen/

isOntology('galen.owl')
isClass('factkb :Abdomen','galen.owl')
isClass('factkb :AbdominalAorta','galen.owl')
isFunctional('factkb :hasAbnormalityStatus')
isProperty('factkb :actsOn','Ontologies :galen.owl')
isTransitive('factkb :hasCause')
someValuesFrom('factkb :AdhesivePericarditis','factkb :hasOutcome','factkb :Adhesion')
subClassOf('factkb :AdductorMagnus','factkb :NAMEDMuscle')
subClassOf('factkb :AdductorTubercle','factkb :Eminence')
subPropertyOf('factkb :hasLayer','factkb :StructuralPartitiveAttribute')
subPropertyOf('factkb :hasLeftRightSelector','factkb :hasPositionalSelector')

Table 6.3 – Fragment de l'ontologie GALEN traduite en DOB

Exemple 12 *Un fragment de l'ontologie Galen traduit en DOB est illustré dans la Table 6.3. Les méta-prédicats extensionnels sont les suivants : isClass, isFunctional, isProperty, isTransitive, someValuesFrom, subclassOf, subpropertyOf. Ces méta-prédicats extensionnels permettent de déclarer respectivement des concepts, des propriétés, des propriétés fonctionnelles, des propriétés transitives, des restrictions de valeurs, des sous-classes et des sous- propriétés.*

Les prédicats intensionnels du formalisme DOB représentent l'ensemble des axiomes ontologiques du modèle inférentiel.

6.4.2 Le méta-service Web déductif dans BioMed

Nous nous proposons d'étendre le formalisme DOB pour représenter la dimension inférentielle du méta-framework BioMed. La formalisation proposée s'inspire de la représentation classique des bases de connaissances où on regroupe un ensemble de faits du monde réel et un ensemble de règles ce qui nous permet de raisonner sur ces faits.

Définition 13 (Méta-service Web Déductif) *Un méta-service Web déductif est un couple (EB,IB) comprenant :*

- *une base extensionnelle notée ESB (Extensional Service Base) composée d'un ensemble de méta-prédicats extensionnels EP permettent de représenter des descriptions sémantisées de SW et des connaissances ontologiques du domaine,*
- *une base intensionnelle notée ISB (Intensional Service Base) composée d'un ensemble de méta-prédicats intensionnels IP représentant l'ensemble des axiomes du modèle de raisonnement.*

Définition 14 (Méta-prédicat extensionnel) *Un méta-prédicat extensionnel de la forme* $ep(t_1,...,t_n)$ *est un littéral positif clos tel que ep est le symbole du méta-prédicat extensionnel faisant référence à une primitive sémantique.*

Les méta-prédicats extensionnels permettent d'exprimer la sémantique extensionnelle d'un service dans une forme déclarative. Nous distinguons deux types de méta-prédicats exten-

Méta-Prédicat Extensionnel	*Description*
`isService(S)`	*S est le nom du service Web*
`hasDescription(S,Desc)`	*Desc est la description du service S*
`hasInput(S,I)`	*le service S a comme entrée la variable I*
`hasOutput(S,O)`	*le service S a comme sortie la variable O*
`hasParameter(S,P)`	*le service S a comme paramètre la variable P*
`isTypeOf(X,C,O)`	*la variable X est de type le concept C de l'ontologie O*
`hasPrecondition(S,Pre)`	*Pre est une précondition du service S*
`hasEffect(S,Eff)`	*Eff est un effet du service S*
`hasQdSParameter(S,QdSName)`	*QdSName est un paramètre QdS décrivant le service S*
`hasQdSValue(S,QdSName,V)`	*le paramètre QdSName du service S possède la valeur V*
`isQdSparameter(QdSName)`	*QdSName est le nom d'un paramètre QdS*
`isQdSUnit(QdSName,U)`	*Le paramètre QdSName est exprimé dans l'unité U*

Table 6.4 – Méta-prédicats extensionnels du méta-service Web déductif BioMed

sionnels : ceux qui représentent des primitives de représentation de connaissances d'un langage de définition d'ontologie tels que OWL ou pour notre cas RDF-S (Table 6.2) et ceux qui représentent des primitives descriptives psour la déclaration des propriétés sémantiques de services tels que ceux du notre modèle unifié de descriptions canonique présenté précédemment. La Table 6.4 illustre un ensemble de méta-prédicat pour expliciter la sémantique extensionnelle d'un service.

Définition 15 (Méta-prédicat intensionnel) *Un méta-prédicat intensionnel est défini par le biais d'une règle R de la forme : $H(\bar{X}) \leftarrow \exists\ \bar{Y}\ B(\bar{X},\bar{Y})$, où H est la tête de R, B son corps, $\bar{X}$ et $\bar{Y}$ sont respectivement des variables distinctes et non distinctes. Le symbole H désigne le symbole du méta-prédicat intensionnel et B est la conjonction d'un nombre fini de littéraux positifs.*

Nous distinguons les méta-prédicats intensionnels qui correspondent aux axiomes du niveau ontologique (tels que l'axiome de la clôture transitive) des méta-prédicats intensionnels qui correspondent aux axiomes du niveau service. En l'occurrence, au niveau service, deux méta-prédicats intensionnels **InputType** et **OutputType** sont définis comme formalisation logique des axiomes du modèle inférentiel présenté dans le chapitre 5. La table 6.5 comprend la définition de ces deux méta-prédicats intensionnels.

Définition 16 (Requête de découverte) *Une requête de découverte (étendue ou non) s'exprime comme une requête conjonctive de la forme : $Q(\bar{X}) \leftarrow \exists\ \bar{Y}\ G(\bar{X},\bar{Y})$, où $\bar{X}$ et $\bar{Y}$ sont respectivement des variables distinctes et non distinctes. $G(\bar{X},\bar{Y})$ est la conjonction d'un ensemble de méta-prédicats extensionnels et/ou intensionnels.*

Selon que la requête exprime des contraintes fonctionnelles et/ou fonctionnelles, le corps de la requête conjonctive correspond à la conjonction des prédicats `InputType`, `OutputType`, `hasInput`, `hasOutput` pour exprimer des contraintes fonctionnelles et les prédicats `hasQoS` et `hasQoSValue` pour exprimer des contraintes non fonctionnelles.

Relaxation Assertionnelle	
`InputType(S,C)`	`:-hasInput(S,X),isTypeOf(X,C,O).`
`OutputType(S,C)`	`:-hasOutput(S,X),isTypeOf(X,C,O).`
Relaxation par spécialisation	
`InputType(S,C)`	`:-hasInput(S,X),isTypeOf(X,C,O), areSubClasses(`C_1`,C)`
Relaxation par généralisation	
`OutputType(S,`C_1`)`	`:-hasOutput(S,X),isTypeOf(X,C,O),areSubClasses(`C_1`,C).`
Relaxation fonctionnelle	
`InputType(S,`C_1`)`	`:-hasInput(S,X),isTypeOf(X,C,O),isOProperty(P,C,`C_1`), isFunctionalProperty(P).`
`OutputType(S,`C_1`)`	`:-hasOutput(S,X),isTypeOf(X,C,O), isDProperty(C,`C_1`), isFunctionalProperty(`C_1`).`
Relaxation fonctionnelle Inverse	
`InputType(S,`C_1`)`	`:-hasInput(S,X),isTypeOf(X,C,O),isOProperty(P,C,`C_1`), isInverseFunctionalProperty(P).`
`OutputType(S,`C_1`)`	`:-hasOutput(S,X),isTypeOf(X,C,O), isDProperty(C,`C_1`), isFunctionalProperty(`C_1`).`

Table 6.5 – Méta-Prédicats Intensionnels du Méta-Service Déductif BioMed

6.4.3 Sémantique formelle

La sémantique du méta-service Web déductif se base sur celle des bases de données déductives qui est définie comme suit.

Définition 17 (Valuation) *Etant donné un ensemble de variables V et un ensemble de constantes C, un mapping ou une valuation γ est une fonction $\gamma : V \rightarrow \mathcal{C}$.*

Définition 18 (Instanciation valide) *Etant donné un méta-service Web déductif S, un ensemble de constantes C dans S, un ensemble de variables V, un prédicat R, et une interprétation I de S qui correspond au modèle minimal [AHV95], une valuation γ est une instanciation valide de R si et seulement si $\gamma(R)$ est évaluée à vrai dans I .*

Définition 19 (Interpretation) *Une **Interpretation** $I = (\Delta^I, \mathcal{P}^I, .^I)$ consiste en :*
- *Un domaine d'interprétation non vide Δ^I correspond à l'union des ensemble de URIs d'ontologies, des classes, des services, des paramètres, des associations sémantiques, des types de données et des instances de classes. Ces ensembles sont disjoints deux à deux.*
- *L'ensemble d'interprétations $\mathcal{P}^I$ des prédicats extensionnels EP et des prédicats intensionnels IP.*
- *Une fonction d'interprétation $.^I$ qui fait correspondre à chaque prédicat n-aire $p^I \in \mathcal{P}^I$ une relation n-aire $\prod_{i=1}^{n} \Delta^I$.*

Pour un prédicat extensionnel EP $p(t_1, ..., t_n)$ et une interprétation I, $I \models p(t_1, ..., t_n)$, i.e., I satisfait p, si et seulement $(t_1, ..., t_n) \in p^I$.

Pour un prédicat intensionnel IP de la forme $R : H(\overline{X}) \leftarrow \exists \overline{Y} B(\overline{X}, \overline{Y})$ et une interprétation I, $I \models R$, i.e., I satisfait R, si et seulement si, chaque fois que I satisfait un prédicat appartenant au corps B, elle satisfait également la tête H.

Définition 20 (Modèle) *Etant donné un méta-service Web déductif S et une interprétation I, $I \models \mathcal{S}$ si et seulement I modélise chaque prédicat EP et IP dans S.*

Définition 21 (Satisfiabilité) *Etant donné $q : Q(X) : -\exists Y G(X, Y)$ une requête de découverte, S un méta-service déductif, une instanciation de X, $\phi(X)$, satisfait q, si et seulement si il existe une instanciation de Y, $\phi(Y)$, telle que l'évaluation $G(X, Y)[\phi(X), \phi(Y)]$ appartient au modèle minimal de S.*

Fort de ces fondements théoriques, nous allons appliquer cet ensemble de concepts et l'axiomatique pour illustrer l'originalité du processus de découverte du méta-framework BioMed.

6.5 Découverte de services Web en bioinformatique

Nous présentons dans cette section une étude de cas menée dans le cadre de cette thèse pour illustrer l'intérêt des contributions proposées. Nous nous intéressons au problème de découverte de services dans le domaine bioinformatique, au sein duquel, les services Web constituent des interfaces à des sources de données scientifiques hétérogènes et diverses. En effet, dans ce domaine, les expérimentations "*in silico*" adoptent de plus en plus la technologie des services Web pour implémenter des flots de données (*workflows*) répondant à une question scientifique complexe. Cependant, la découverte et la composition de services constitue un vrai challenge du fait que le nombre de ressources biologiques disponibles sur le Web a nettement augmenté ces dix dernières années. Comme cela a été mentionné auparavant dans ce mémoire, la *Databases Issue* de la revue Nucleic Acids Research (NAR) recense plus de 1230 ressources en 2010.

Grimoires et **BioMoby** central repository constituent tous les deux les plus importants annuaires de services Web sémantiques dans le domaine bioinformatique. Grimoires [MPP+04] est un annuaire de services Web basé sur la technologie UDDI et proposant des extensions permettant d'annoter sémantiquement les entités UDDI afin de faciliter la découverte de services. Selon [LA10], BioMoby est un annuaire de services Web dédié à la communauté bioinformatique comptant aujourd'hui 1673 services publiés par plus de 100 fournisseurs différents. Nous illustrons la problématique de découverte de service dans l'annuaire BioMoby par un ensemble de questions biologiques (requêtes de découverte). Nous étudions la pertinence des résultats du processus de découverte en comparant l'espace et la qualité des solutions obtenues avec et sans usage du raisonneur BioMed.

6.5.1 Le projet BioMoby

Initié en 2001, BioMoby [WL02] est un projet de recherche qui vise à proposer une plate-forme d'interopérabilité pour la publication et la découverte de services Web dans le domaine bioinformatique. Auparavant, la découverte de services Web était associée à

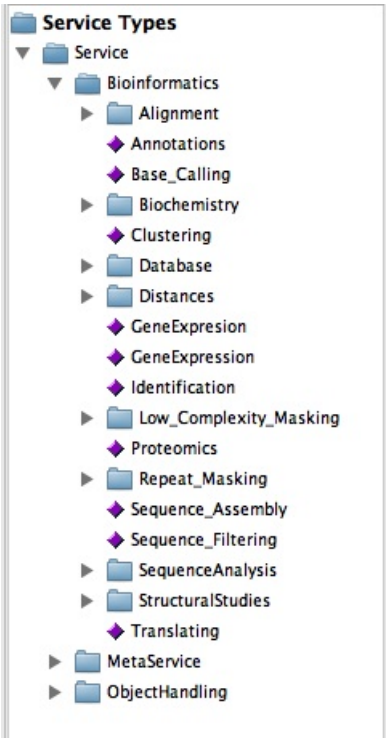


FIGURE 6.4 – Fragment de la hiérarchie *Service Type* dans BioMoby

des systèmes de workflows tels que *Taverna* [OGA+06] ou *Kepler* [LAB+06]. Dans ce type de systèmes, les services Web sont découverts sur la base du format syntaxique de leurs entrées et sorties plutôt qu'en se basant sur la classe du service ou sa qualité.

Pour décrire la sémantique d'un service Web, BioMoby repose sur deux types d'hiérarchies : une hiérarchie définissant les catégories de service (*Service Type Ontology*) et une hiérarchie de types de données (*datatype ontology*). La première permet de spécifier la tâche du service Web alors que la seconde permet de définir le type des entrées et des sorties d'un service Web.

Ainsi, la publication de ressources dans BioMoby est réalisée en deux étapes :

– La première étape consiste à classer la ressource dans la hiérarchie de catégories de services Web définie par l'ontologie de type de service selon la tâche réalisée par le service (figure 6.4), Le fournisseur de service peut classer son service par rapport à une catégorie existante ou créer une nouvelle catégorie. Cinq catégories primitives sont définies pour classifier les ressources bioinformatiques : *Analyse* (Analysis), *Parsing*, *Enregistrement* (Registration), *Recherche* (Retrieval) et *Résolution* (resolution). L'analyse présenté dans [LA10] permet de révéler de nombreuses inconsistances au niveau de l'ontologie de types de services à cause de la liberté donnée au fournisseur qui ont la possibilité de rajouter de nouvelles catégories. La figure 6.5 illustre un exemple d'inconsistance au niveau de la hiérarchie de types de services. Le service *NCBI_Blast* est créé comme une catégorie de services (sous-catégorie de la

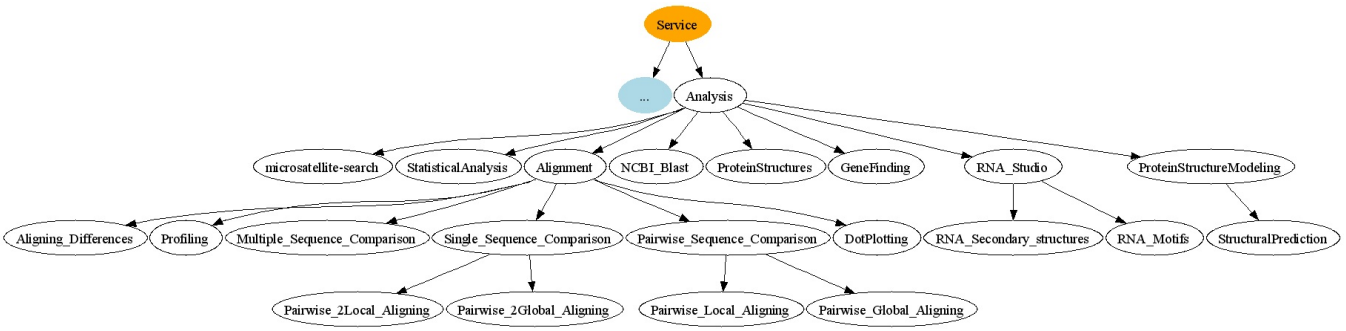


FIGURE 6.5 – Zoom sur la catégorie *Analyse* de l'ontologie *service type* dans BioMoby [LA10]

catégorie Analyse) alors qu'il désigne un service d'alignement et non une catégorie.
- Une fois le service Web enregistré dans une catégorie de services, ses entrées et sorties doivent être identifiées en termes de concepts de l'ontologie des types de données. Comme l'ontologie de type de services, l'ontologie de types de données présente plusieurs inconsistances plus d'ordre conceptuel que structurel comme c'est le cas de la première ontologie. En effet, l'ontologie des types de données ne capture pas les concepts sémantiques du domaine concerné, elle est plutôt orientée syntaxe (format de représentation) en ce sens qu'elle différencie les différents types de formats syntactiques représentant les différents concepts clés du domaine et non la sémantique des concepts en terme de propriétés et de relations. Par exemple, la figure 6.6 montre une sous-hiérachie de types syntactiques. En conclusion, l'ontologie de types de données manque de conceptualisation et se concentre uniquement sur le niveau opérationnel.

La découverte de services dans BioMoby se fait uniquement par le biais d'une API accessible via le Web (figure 6.7) ; la découverte repose seulement sur la navigation (et non sur une recherche par mots-clés) et les services sont classés par ordre alphabétique, par datatype ou par catégorie. Seulement, le nombre de services étant important, la recherche d'un service pour répondre à un besoin spécifique peut prendre beaucoup de temps. Il est indispensable de disposer de techniques pour l'automatisation du processus de découverte dans BioMoby. Afin de remédier aux inconsistances structurelles et conceptuelles des ontologies conçues dans le cadre du projet BioMoby,[LA10] proposent une approche semi-automatique pour la classification des services BioMoby dans une ontologie du domaine tout en nettoyant les ontologies existantes des inconsistances repérées. Au niveau de notre travail, nous faisons usage de la méta-carte ontologique proposée comme support d'annotation et de raisonnement pour pallier les inconsistances et les incohérences ontologiques. De plus, nous proposons des techniques de raisonnement pour guider la découverte de services dans BioMoby qui reste toujours limitée à la navigation dans [LA10].

6.5.2 Un scénario de découverte dans BioMoby

Un scientifique souhaite étudier les mutations connues des gènes APP et Préséniline 1 (PSEN1) chez l'homme dont des recherches ont montré l'implication des mutations de

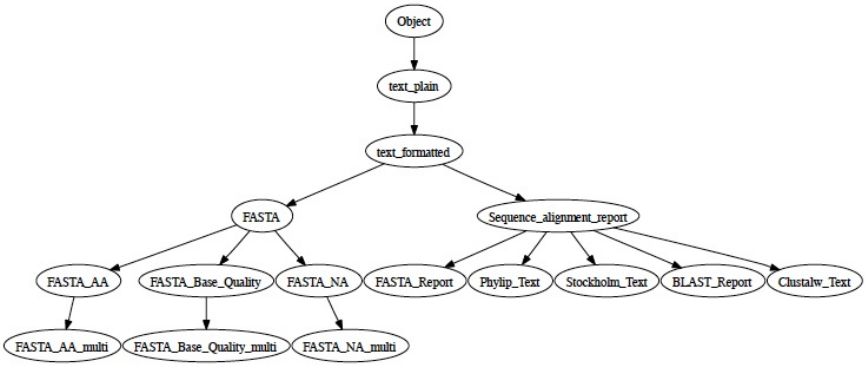


FIGURE 6.6 – Zoom sur le type `text_plain` de l'ontologie *Datatype Ontology* dans BioMoby [LA10]

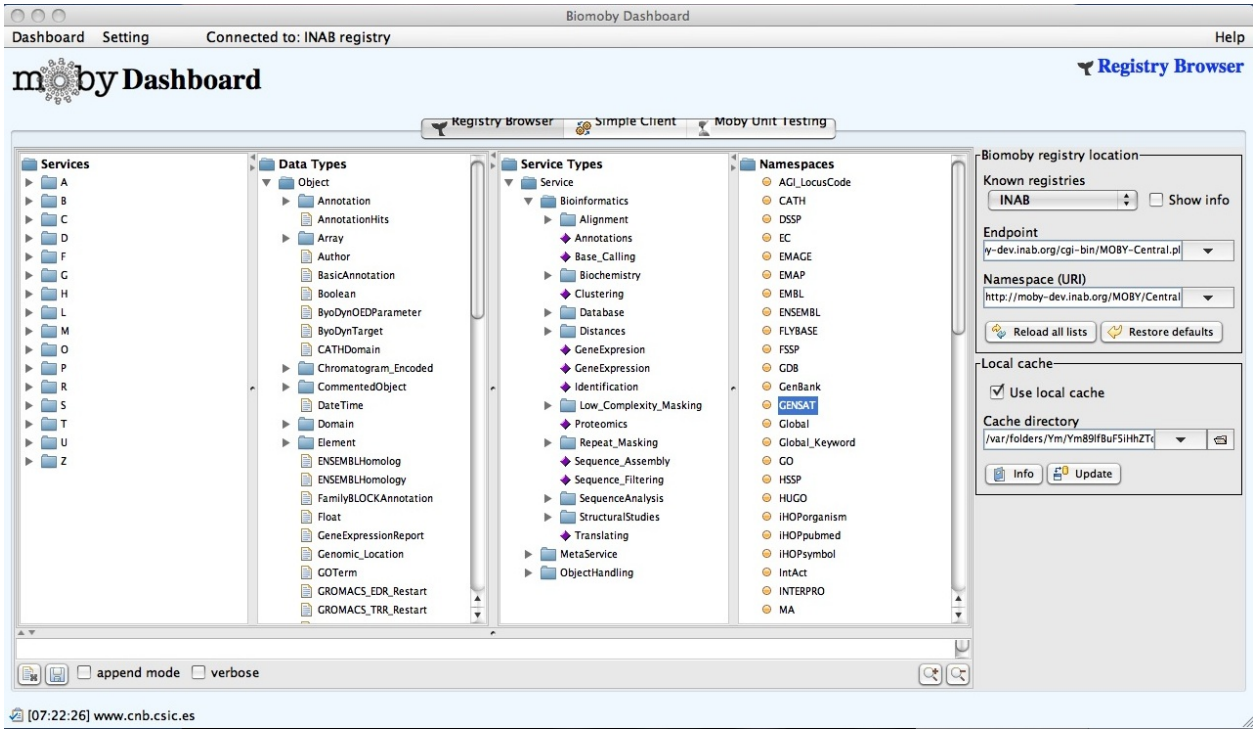


FIGURE 6.7 – Interface de l'API BioMoby

ce gène dans la maladie d'Alzheimer. Le scientifique aurait besoin d'un certain nombre de services afin de répondre à ses besoins. Les questions scientifiques pour lesquelles le scientifique désire sélectionner des services sont les suivantes :

- (Q_1) Quels sont les services Web permettant de retourner l'ensemble des séquences nucléotidiques d'un gène dont on connaît l'identifiant par exemple les identifiants `APP` ou `PSEN1` ?
- (Q_2) Quels sont les services Web qui permettent de rechercher les séquences d'acides aminées ayant une similitude avec une séquence d'acide aminés résultante de la traduction de la séquence nucléotidique du gène APP ?
- (Q_3) Le scientifique souhaiterait également disposer d'un service pour la prédiction de la structure 3D d'une séquence protéique ?
- (Q_4) Quels sont les services permattant de générer un arbre phylogénétique étant donné une séquence d'acides aminés ?

Les requêtes de découverte exprimées par le scientifique seront formulées sous forme d'une requête conjonctive comme suit :

- (Q_1) : q(S) :-InputType(S, `GeneId`), hasInput(S, '`homosapiens`'), rdfType('`homosapiens`', `ScientificOrganismName`), OutputType(S, `NucleotideSequence`).
- (Q_2) : q(S) :-InputType(S, `NucleotideSequence`), OutputType(S, A), rdfBag(A, `AminoAcidSequence`).
- (Q_3) : q(S) :-InputType(S, `AAsequence`), OutputType(S, `3DStructure`).
- (Q_4) : q(S) :-InputType(S, `AminoAcid`), InputType(S, '`homosapiens`'), OutputType(S,`Phylogenetic_Tree`).

Nous désignons par $\mathcal{KB}_O$, la base déductive ontologique du méta-service déductif de BioMed. $\mathcal{KB}_O$ comprend dans sa partie extensionnelle, l'ensemble de faits correspondant aux assertions de déclaration de connaissances de la méta-carte ontologique et dans sa partie intensionnelle, l'axiome de fermeture transitive définie par les deux règles suivantes [9] :

areSubclasses(C_1, C_2) :-subClassOf(C_1, C_2).

areSubclasses(C_1, C_2) :-subClassOf(C_1, C_3), areSubclasses(C_3, C_2).

D'un autre côté, nous désignons par $\mathcal{KB}_S$, la base déductive du méta-service Web déductif comprenant dans sa partie extensionnelle, l'ensemble des annotations sémantiques des services contenues dans le catalogue BioMoby et dans sa partie intensionnelle les huits axiomes explicitant la sémantique inférentielle du méta-framework BioMed.

Il existe au moins une solution à la requête q si et seulement si $\mathcal{KB}_O \bigcup \mathcal{KB}_S \models q$. Pour évaluer la requête q, nous nous reposons sur le moteur de raisonnement BioMed développé avec le langage de programmation logique Prolog, nous décrirons en détail le fonctionnement du moteur de raisonnement BioMed dans la section 6.6.3.

La Table 6.6 résume les solutions des requêtes Q_1, Q_2, Q_3 et Q_4, nous distinguons un seul matching exact entre les contraintes de la requête Q_4 et le service runPhylipProtpars permettant de générer un arbre phylogénétique étant donné une séquence d'acides aminés. La Table 6.6 montre que le nombre de services découverts augmente considérablement avec l'utilisation des techniques de relaxation proposées dans BioMed.

9. Pour illustrer, nous prenons à titre d'exemple seulement l'axiome de la fermeture transitive

Requête	Sans raisonneur	Avec raisonneur BioMed
Q_1	Aucun service	DDBJGetEntry eFetchSequence DBFetch retreive_ensembl_sequence
Q_2	Aucun service	runNCBIblastp
Q_3	Aucun Service	ShowPDBfromFASTA PDBWebService
Q_4	runPhylipProtPars	runClustalWService PhylipService

Table 6.6 – Solutions des requêtes avec et sans l'utilisation de techniques de raisonnement

Des appariements fonctionnels approximatifs ont été obtenus par le raisonneur BioMed pour toutes les requêtes. Pour la requête Q_1, comme appariement fonctionnel en subsomption, nous obtenons les services DDBJGetEntry [10], eFetchSequence [11], DBFecth [12] et retreive ensembl sequence [13]. Les quatre services offrent un accès à des bases génomiques permettant de retrouver les séquences connues d'un gène. La description des services est plus générale que celle de la requête du fait que ces services retournent en sortie une séquence, couvrant ainsi tous les types de séquences, d'où l'appariement en subsomption. Pour la requête Q_2, nous obtenons comme appariement fonctionnel en plugin le service runNCBIBlastp pour l'alignement des séquences nucléotidiques des gènes étudiés après les avoir traduit en six positions de lecture et aligné contre une base de séquences protéiques. Pour la requête Q_3, nous identifions comme appariement fonctionnel les services PDB [14] et ShowPDBFromFASTA. Enfin, comme appariement en subsomption pour la requête Q_4, PhylipService [15] et ClustalW [16] sont sélectionnés.

6.6 Expérimentations et Résultats

Comme l'illustre la figure 6.9, la composante de découverte du méta-service BioMed repose sur un catalogue de ressources (référentiel de services Web sémantisés), un moteur d'exécution permettant l'appariement fonctionnel des requêtes de découverte étendues en se basant sur les axiomes du modèle inférentiel et un moteur de classement pour le scoring et le rangement des services satisfaisant les contraintes non fonctionnelles de la dite requête. Afin d'évaluer le passage à l'échelle de l'approche proposée, nous avons choisi de générer automatiquement des *datasets* de services de différentes tailles (nombre de services) allant jusqu'à 16000 services dans le plus grand dataset. Le catalogue de ressources (i.e., la

10. http ://xml.nig.ac.jp/wsdl/GetEntry.wsdl
11. http ://www.ncbi.nlm.nih.gov/entrez/eutils/soap/v2.0/efetch seq.wsdl
12. http ://www.ebi.ac.uk/Tools/webservices/wsdl/WSDbfetch.wsdl
13. http ://rsat.bigre.ulb.ac.be/rsat/web services/RSATWS.wsdl
14. http ://www.pdb.org/pdb/services/pdbwsfiwsdl
15. http ://www.vbi.vt.edu/ pathport/services/wsdls/phylip.wsdl
16. http ://www.ebi.ac.uk/Tools/webservices/wsdl/WSClustalW.wsdl

Requête	Services Sélectionnés	Type d'appariement approximatif
Q_1	`DDBJGetEntry`	*Appariement en subsomption*
	`eFetchSequence`	*Appariement en subsomption*
	`retreive_ensembl_sequence`	*Appariement en subsomption*
	`DBFetch`	*Appariement en subsomption*
Q_2	`runNCBIblastp`	*Appariement en plugin*
Q_3	`ShowPDBfromFASTA`	*Appariement fonctionnel*
	`PDBWebService`	*Appariement fonctionnel*
Q_4	`runClustalWService`	*Appariement en subsomption*
	`PhylipService`	*Appariement en subsomption*

TABLE 6.7 – Les services sélectionnés par requête et par degré d'appariement

base extensionnelle du méta-service Web déductif) comprend les descriptions unifiées, les annotations sémantiques et les connaissances ontologiques. Nous présentons les ontologies MESH et PROTEIN ONTOLOGY utilisées pour générer un dataset de services Web dont les entrées et sorties sont annotés avec des concepts des ontologies.

Au niveau opérationnel, le catalogue de ressources correspond à la base extensionnelle d'un programme DATALOG. Quant à la base intensionnelle du programme DATALOG, elle comprend un ensemble de règles logiques implémentant tous les axiomes de la sémantique axiomatique présentée dans le chapitre 5. Le moteur d'exécution est implémenté en tant qu'un méta-interpréteur PROLOG. En premier lieu, le moteur évaluera la base extensionnelle en calculant le modèle minimal de la base et la requête de découverte est évaluée considérant le modèle minimal calculé pour générer l'ensemble des services satisfaisants les contraintes d'une requête de découverte étendue.

Ensuite, un moteur de scoring et classement prendra le relais pour déceler les meilleures solutions parmi celles seléctionnées par le moteur d'exécution. Nous supposons que tous les critères QdS sont d'égale d'importance et nous cherchons les services qui ont les meilleures valeurs du plus grand nombre de critères QdS. Enfin, le fonctionnement de la composante de découverte dans BIOMED est illustré par la figure 6.8. Nous présentons dans ce qui suit et en détails, le catalogue des ressources, le moteur de raisonnement et les résultats des expérimentations.

6.6.1 Catalogue de ressources

L'étude menée dans cette expérimentation a considéré un ensemble de services générés automatiquement dont les entrées et les sorties sont annotés avec des termes du vocabulaire contrôlé *Mesh* et de l'ontologie *Protein Ontology*. Nous présentons dans ce qui suit une description de ces ontologies. De plus, nous nous sommes basés sur des techniques d'échantillonnage pour associer aux services générés 10 paramètres QdS (dont les valeurs sont normalisées entre 0 et 1) et ce d'une manière aléatoire.

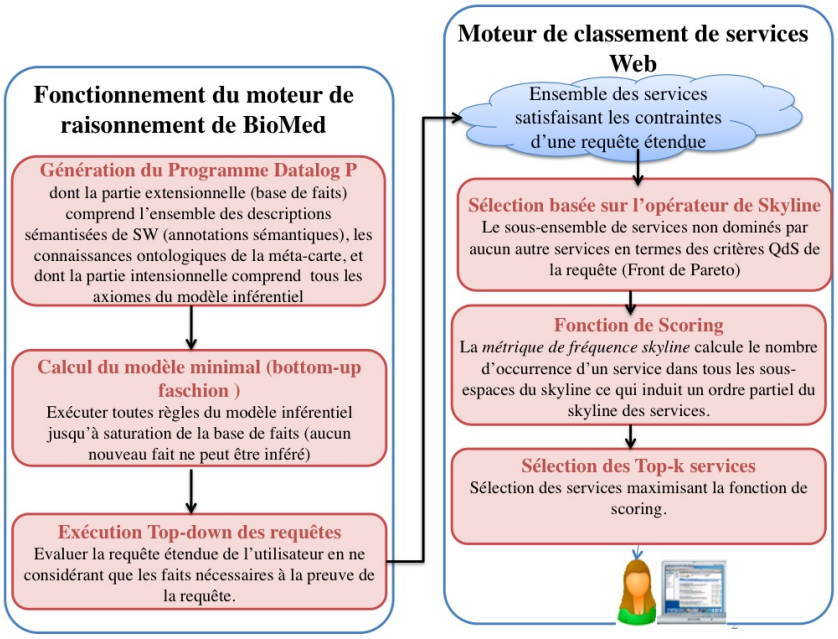


FIGURE 6.8 – Fonctionnement Général de la composante de découverte dans BIOMED

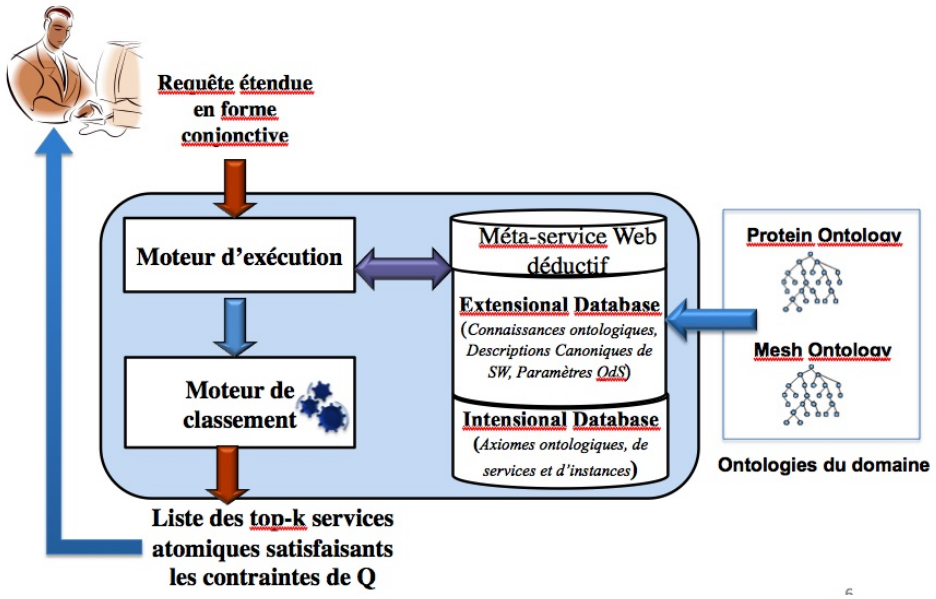


FIGURE 6.9 – Architecture de la composante de découverte dans BIOMED

code hiérarchique	terme
C01	maladies bactériennes et mycoses
C01.539	infections
C01.539.830	suppuration
C01.539.830.025	abcès
C01.539.830.025.160	abcès cerveau
C01.539.830.025.325	abcès péridural
C01.539.830.025.490	*abcès poumon*
[...]	
C08	maladies de l'appareil respiratoire
C08.381	poumon, maladies
C08.381.125	dysplasie bronchopulmonaire
[...]	
C08.381.348	hémoptysie
C08.381.449	*abcès poumon*
C08.730	appareil respiratoire, infection
C08.730.099	bronchite
C08.730.407	*abcès poumon*

FIGURE 6.10 – Extrait de l'arbre ontologique Mesh

6.6.1.1 Le thésaurus Mesh

MeSH (acronyme de **M**edical **S**ubject **H**eading) est le thésaurus de référence dans le domaine biomédical produit par le NLM (U.S. National Library of Medicine) [17]. La structure du MeSH est à trois niveaux : un ensemble de termes dont un préférentiel, désigne un *concept* qui fait partie d'une classe de concepts appelée *descripteur*. La navigation dans le thésaurus se fait par recherche ou en entrant par les Main Headings. Le MeSH contenait près de 25 000 descripteurs et plus de 455 000 termes en janvier 2009. Trois types de relations, ne reposant pas sur une logique formelle, sont utilisées : hiérarchique, synonymique et proxémique (proximité sémantique).

Produit depuis 1960, le MeSH est utilisé pour l'indexation par de nombreuses bibliothèques et institutions à travers le monde. Le thésaurus a été traduit dans de nombreuses langues mais il est à déplorer qu'il n'existe aucune possibilité de navigation à travers le multilinguisme. Le MeSH n'est conforme à aucune norme, il permet seulement des échanges par le biais du format pivot XML.

Nous avons fait usage de l'arbre ontologique du thésaurus MeSh pour illustrer la relaxation par généralisation et par spécialisation que nous avons proposée. Une des spécificités de l'arbre ontologique MeSh est qu'il présente des cas d'*héritage multiple* comme l'illustre la figure 6.10. Le terme *abcès poumon* subsume les trois termes "*abcès*", "*poumon, maladie*" et "*appareil respiratoire, poumon*".

6.6.1.2 L'ontologie PO (Protein Ontology)

L'ontologie Protein Ontology (PO) [SDC06] offre un vocabulaire unifié capturant les connaissances du domaine des protéines. D'une part, PO fournit un méta-modèle en définissant les propriétés sémantiques du concept Protéine. Etant donné que ce concept est décrit d'une manière hétérogène entre les sources, PO fournit une sémantique unifiée

17. http ://www.nlm.nih.gov/

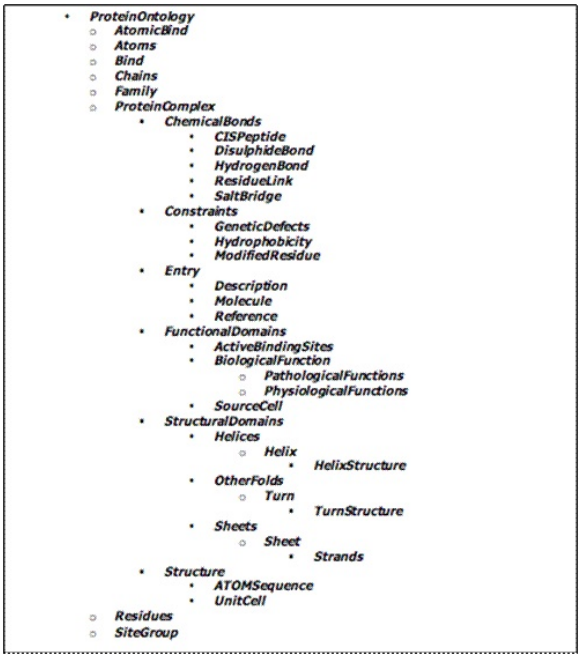


FIGURE 6.11 – Modélisation du concept Protéine au niveau de l'ontologie PO

telle qu'illustrée par la figure 6.11. D'autres part, PO offre un entrepôt d'instances de protéines selon le format PO, il compte aujourd'hui 7398 protéines classées par famille de protéines (e.g., *growth factor*, *kinase*, *glycoprotein*, etc.). Nous distinguons comme relations sémantiques dans PO, en plus de la relation de subsomption (`subclassOf`), les relations suivantes `PropertyOf` (propriétés sémantiques), `PartOf` (relations de meronymie), `InstanceOf` (instances de concepts) et `ValueOf` (valeurs de propriétés).

6.6.2 L'implémentation du méta-service Web déductif dans BioMed

Le méta-service Web BIOMED se présente en l'occurrence comme une base déductive où un langage déclaratif est utilisé pour spécifier des règles DATALOG qui est un sous-ensemble de PROLOG. Nous nous sommes particulièrement intéressés aux techniques développées autour du langage DATALOG ce qui nous incite à les présenter ci-après.

6.6.2.1 Datalog

Une limite importante du calcul (algèbre) relationnel(le) est qu'il n'est pas possible d'exprimer des requêtes contenant des chemins entre instances, comme par exemple trouver la fermeture transitive d'un graphe dont les arcs sont stockés dans une relation. DATALOG étend les requêtes conjonctives avec la récursivité pour pouvoir traiter de telles requêtes. La récursivité consiste à utiliser les mêmes noms de relation au niveau de la tête et le corps d'une requête. La sémantique de DATALOG peut être définie à travers trois approches équivalentes : la théorie des modèles, la théorie du point fixe et la théorie de la

preuve. DATALOG hérite cette propriété de la programmation logique et son langage phare PROLOG.

Syntaxe Datalog

La syntaxe de DATALOG est la même que celle du langage de la programmation logique PROLOG en éliminant les symboles de fonctions et en considérant les constantes, variables et prédicats.

Définition 22 *Une règle* DATALOG *est définie comme suit :*

$$T(\mathrm{x}) : -q(\mathrm{x}, \mathrm{y})$$

tel que $\mathrm{x} = x_1, \ldots, x_n$ *est un tuple de variables distinctes,* $\mathrm{y} = y_1, \ldots, y_m$ *est un tuple de variables "existentiellement quantifiées", T est une relation et q est la conjonction d'atomes. La partie gauche est la tête de la règle alors que la partie droite est le corps de la règle. Notons que toutes les variables apparaissant dans la tête d'une règle doivent impérativement apparaître dans au moins un atome du corps de la règle*[18]*. Un programme Datalog P est un ensemble fini de règles. L'instanciation d'une règle est la substitution de chaque variable par une constante. L'instanciation d'un programme P ; notée Inst(P) ; est obtenue en instanciant les règles de P de toutes les façons possibles, en utilisant uniquement les constantes présentes dans les règles et dans les faits. Les symboles des relations (prédicats) qui apparaissent seulement dans le corps des règles d'un programme sont nommés prédicats extensionnels, ceux qui apparaissent dans la tête de règles sont nommés prédicats intensionnels. Un programme Datalog définit une requête datalog lorsqu'un prédicat intensionnel est spécifié comme la sortie du programme.*

Par exemple, considérons le programme DATALOG suivant pour calculer la fermeture transitive d'une hiérarchie de concepts comprenant les deux règles IDB_1 et IDB_2 et un ensemble de trois prédicats extensionnels formant l'EDB du programme DATALOG :

$$\begin{aligned} IDB_1 &: \textit{areSubclasses(X,Y)} && \textit{:-subClassOf(X,Y)} \\ IDB_2 &: \textit{areSubClasses(X,Y)} && \textit{:-subClassOf(X,Z), areSubClasses(Z,Y)} \\ EDB &= && \{\textit{subClassOf}(\texttt{Peritoneum}, \texttt{Abdominal Cavity}), \\ & && \textit{subClassOf}(\texttt{Abdominal Cavity}, \texttt{Abdomen}), \\ & && \textit{subClassOf}(\texttt{Abdomen}, \texttt{Body Region})\ \} \end{aligned}$$

Sémantique DATALOG

Nous distinguons trois approches différentes mais équivalentes pour définir la sémantique d'un programme DATALOG P : *théorie des modèles*, *Point fixe*, *théorie de la preuve*. Pour présenter ces approches, nous présentons une synthèse des chapitres 12 et 13 de l'ouvrage de **Abiteboul et al.** intitulé *Foundations of Databases* [AHV95].

18. Nous nommons ce type de règles, règles saines (*safe rules*)

Théorie des modèles

Dans cette approche, le programme Datalog P est vu comme un ensemble de formules du premier ordre. Rappelons qu'à une requête conjonctive de la forme : T(x) :- q(x, y) correspond une formule (en logique du premier ordre) $\forall$ x ($\exists$ y q(x, y) $\rightarrow$ T(x)). Les formules de la logique du premier ordre associées à un programme DATALOG sont d'un type spécial de clause, appelées *clauses de Horn*. Nous rappelons à ce sujet qu'une clause contient au plus un littéral positif alors qu'une clause de Horn contient exactement un littéral positif.

Une interprétation donne à chaque symbole de formule une signification. Nous disons qu'une interprétation I satisfait une formule close φ ($I \models \varphi$) si φ est vrai dans I. Dans ce cas, I est un modèle de φ. En général, si une interprétation I satisfait toutes les formules d'un ensemble de formules closes E ($I \models E$), alors I est un modèle de E. Si tous les modèles d'un ensemble de formules E satisfont une formule close φ ; alors φ est la conséquence logique de E (E $\models \varphi$).

Dans le contexte DATALOG, il est question de tester si un fait est une conséquence d'un programme DATALOG. Néanmoins, il est inconcevable qu'il soit nécessaire de vérifier un nombre infini d'interprétations. Comme un programme DATALOG est constitué d'un ensemble de clauses de Horn, nous parlons d'*interprétations de Herbrand* dont la caractéristique est d'interpréter chaque constante en elle-même. Par conséquent, deux interprétations de Herbrand distinctes interprètent les constantes de la même façon et diffèrent seulement par leur interprétation des symboles de prédicats.

Définition 23 (Interprétation de Herbrand) *Soit L un langage du premier ordre. Une interprétation de Herbrand pour L est définie de la façon suivante :*

1. *Le domaine de l'interprétation est l'Univers de Herbrand (HU), c'est- à-dire, l'ensemble de tous les termes clos qui peuvent être construits avec les constantes et les symboles de fonctions de L.*
2. *Les constantes dans L sont associées à elles-mêmes dans HU :*
3. *Si f est un symbole de fonction d'arité n dans L ; alors nous attribuons à f la fonction de $(HU)^n$ dans HU définie par (t_1,..., t_n) $\mapsto$ f(t_1,..., t_n).*
4. *Une fonction qui associe à un élément de $(HU)^n$ un élément de {vrai ; faux} ; c'est-à-dire, une relation sur $(HU)^n$ est associée à chaque symbole de prédicat d'arité n dans L.*

Dans le contexte de DATALOG, le point 3 ci-dessus peut être omis puisque le langage L considéré ne contient pas de symbole de fonction. Par conséquent, une *interprétation de Herbrand* est caractérisée par l'ensemble de tous les faits qui sont vrais par rapport à cette interprétation. L'ensemble de tous les faits que l'on peut écrire avec les éléments de l'alphabet (c'est-à-dire, avec le langage courant) est appelé *Base de Herbrand (HB)*, une interprétation de Herbrand est un sous-ensemble de HB. Une interprétation de Herbrand qui satisfait une formule close φ (ou un ensemble de formules closes E) est appelée Modèle de Herbrand de φ (ou de E).

Toujours dans le contexte de DATALOG, nous considérons que les seuls constantes et prédicats à prendre en compte sont ceux qui apparaissent dans le programme DATALOG.

Nous pouvons alors reformuler la notion de conséquence logique de la manière suivante. Un fait f est conséquence d'un ensemble de clauses de Horn définies E (dénoté $E \models f$) si et seulement si chaque interprétation de Herbrand qui satisfait toutes les clauses de E satisfait f.

Définition 24 (Modèle Minimal) *Etant donné un programme* DATALOG *P, Σ_P l'ensemble des formules du premier ordre correspondant à P, $\mathcal{X}$ l'ensemble des modèles de Herbrand satisfaisant Σ_P, alors :*

1. *Le modèle (Herbrand) minimal est l'intersection de tous les modèles $\bigcap \mathcal{X}$ satisfaisant Σ_P,*
2. *Le modèle minimal constitue la sémantique de P.*

Point Fixe

Dans cette approche, la sémantique d'un programme DATALOG est définie comme étant la solution d'une équation de point fixe. Il s'agit d'une approche procédurale où le programme est vu comme un ensemble de faits et de règles sur lequel un mécanisme d'inférence est appliqué pour engendrer de nouveaux faits. Un opérateur, appélé opérateur de conséquence immédiate, est défini pour formaliser l'idée de production de nouveaux faits.

Définition 25 (Opérateur de conséquence immédiate) *Soit P un programme* DATALOG *et soit $\mathcal{P}_{HB}$ l'ensemble des interprétations de Herbrand de P. L'opérateur de conséquence immédiate associé à P ; noté T_P est une fonction de $\mathcal{P}_{HB}$ dans $\mathcal{P}_{HB}$ définie par :*

$T_P(I) = \{$ *tête(r)* $\mid r \in Inst_P \wedge \forall\ L \in corps(r),\ I \models L\}$ *; pour tout I de* $\mathcal{P}_{HB}$.

Pour un programme Datalog, l'opérateur T_P est monotone et continu et il est défini sur le treillis complet $(\mathcal{P}_{HB}, \subseteq)$. Par conséquent, la suite définie par $T_P^0(\emptyset) = \emptyset$, $T_P^k(\mathrm{I}) = \mathrm{T}(T_P^{k-1}(\mathrm{I}))$, pour $k > 0$, possède une limite. Cette limite est le plus petit point fixe de T_P par rapport à I, notée, $lfp(T_P)$. Le théorème suivant établit le rapport entre le modèle minimal et l'opérateur de conséquence immédiate.

Théorème 1 *Soit P un programme Datalog et soit M_P le modèle minimal associé à P : Le plus petit point fixe de l'opérateur de conséquence immédiate T_P associé à P coïncide avec M_P ; c'est-à-dire, $M_P = lfp(T_P)$.*

En conclusion, cette approche donne une définition constructive de la sémantique d'un programme DATALOG, et peut être considérée comme une implantation de la théorie de modèles.

Théorie de la preuve

Dans cette approche, la sémantique d'un programme DATALOG est définie en fonction de preuves, c'est-à-dire, la sémantique d'un programme P par rapport à une interprétation I est l'ensemble de faits à prouver en utilisant P et I : Pour définir la notion de preuve, nous devons définir d'abord l'**arbre de preuve d'un fait**.

Définition 26 (Arbre de preuve d'un fait) *Soit P un programme Datalog et I une interprétation. Un arbre de preuve d'un fait f à partir de I et P est défini de la façon suivante :*

1. *Chaque noeud de l'arbre est étiqueté par un fait.*
2. *Chaque feuille est étiquetée par un fait de I :*
3. *La racine est étiquetée par f :*
4. *Pour chaque noeud interne, il existe une instanciation $A_0 \leftarrow A_1,...,A_n$ d'une règle de P telle que le noeud est étiqueté par A_0 et ses fils sont étiquetés par $A_1,...,A_n$.*

Un arbre de preuve représente la preuve d'un fait. De plus, dans [AHV95] le résultat suivant est présenté : Un fait f est dans M_P si et seulement si il y a un arbre de preuve pour f à partir de I et de P.

*É*tant donné un fait f à prouver, il y a deux méthodes pour chercher la preuve de f : bottom up et *top down*.

La méthode *bottom up* est une autre façon de traiter l'approche par point fixe. Commençant par les faits dans *EDB* ; les règles sont appliquées pour engendrer de nouveaux faits, jusqu'au moment où aucun nouveau fait n'est dérivé. La méthode *top-down* permet l'orientation de la recherche. Si nous voulons prouver f ; il est nécessaire de prouver seulement des faits qui ont un rapport avec f. Ainsi, la méthode *top-down* inhibe les inférences inutiles à la preuve du fait considéré. La méthode *top down* est fondée sur l'idée suivante : étant donné une formule close φ du premier ordre et un ensemble E de formules closes du premier ordre, φ est conséquence logique de E si et seulement si $E \bigcup \{\neg\, \varphi\}$ n'est pas satisfaisable.

Le moteur inférentiel de BIOMED se base sur l'approche par point fixe en intégrant les deux techniques d'évaluation *top-down* et *bottom-up*.

6.6.2.2 Le moteur inférentiel de BioMed

Comme la syntaxe Datalog est un sous-ensemble de la syntaxe PROLOG, nous implémentons le moteur BIOMED en PROLOG tout en adoptant les techniques d'évaluation de programmes DATALOG. Ainsi, le moteur de raisonnement a été implémenté comme un méta-interpréteur Prolog implémentant l'approche d'évaluation semi-naive de la technique *bottom-up* du DATALOG.

En effet, une stratégie d'évaluation simple en bottom-up, appelée *évaluation naïve*, est basée sur la sémantique du point fixe de DATALOG. L'idée est d'appliquer l'opérateur de conséquence immédiate sur les résultats de toutes les étapes précédentes (en commençant par la base extensionnelle initialement définie). Une évaluation naïve aboutit à une redondance des faits inférés. Afin de pallier cette insuffisance, une évaluation semi-naïve permet

de produire à chaque étape seulement les faits qui peuvent être déduits par au moins un des faits inférés pendant l'étape précédente.

L'implémentation du programme *Prolog* de l'approche semi-naive se présente comme suit :

La règle PROLOG implémentant la relaxation assertionnelle est codée comme suit :

```
inputtypes(S,C) :- hasinput(S,X), (isoftype(X,C,'mesh',S),addInput(S,C).
```

Les entrées et les sorties des services sont représentées sous forme de listes. A chaque axiome de relaxation, le prédicat addInput permet d'ajouter un nouveau concept à la liste des concepts, tout en vérifiant qu'il n'appartient pas déjà à la liste en retirant de la liste tous ses éléments (prédicat retractInput) et vérifiant si le concept en question est dans la liste (prédicat checkInput). Le prédicat addInput est défini d'une manière disjonctive comme suit :

```
addInput(S,C) :-retractInput(L,S),checkInput(L,S),not(member(C,L)),append(L,[C],L1),
notExistInput(S,L1),assert(hasInput(S,L1)).
addInput(S,C) :-not(hasInput(S,_)),assert(hasInput(S,[C])).
addInput(_,_).
```

L'axiome AddOutput est défini de la manière similaire que AddInput comme suit :

```
addOutput(S,C) :-retractOutput(L,S),checkOutput(L,S),not(member(C,L)),append(L,[C],L1),
notExistOutput(L1,S),assert(output(L1,S)).
addOutput(S,C) :-not(output(_,S)),assert(output([C],S)).
addOutput(_,_)
```

L'axiome de relaxation par spécialisation est implémenté avec la règle PROLOG suivante :

```
inputtypes(S,Y) :-hasinput(S,X), isoftype(X,C,O,S)),(aresubclasses(C,Y),addInput(S,C)).
```

Le prédicat isoftype est un prédicat avec quatre arguments spécifiant le type sémantique d'un paramètre X d'un service S en terme d'un concept C d'une ontologie O. Bien que

les services Web considérés dans notre catalogue aient été annotés en termes des concepts, propriétés et relations de la méta-carte ontologique, nous prévoyons le cas où les paramètres des services sont annotés en nous basant sur plusieurs ontologies dans des travaux futurs.

L'axiome de relaxation par généralisation est implémenté comme suit :

outputTypes(S,Y) :-hasOutput(S,X), isoftype(X,C,O,S)),(aresubclasses(C,Y),addOutput(S,C)).

Les axiomes de relaxation fonctionnelle sont implémentés comme suit :

inputtypes(S,Y) :-inputtypes(S,C),(propertyDomain(P,C),(functionalProperty(P),
(propertyRange(P,Y),addInput(S,Y)).

outputtypes(S,Y) :-outputtypes(S,C),(propertyDomain(P,C),(functionalProperty(P),
(propertyRange(P,Y),addOutput(S,Y)).

Les axiomes de relaxation fonctionnelle inverse sont implémentés comme suit :

inputtypes(S,C) :-inputtypes(S,Y),(propertyRange(P,Y),(inverseFunctionalProperty(P),
(propertyDomain(P,C),addInput(S,C)).

outputtypes(S,C) :-outputtypes(S,Y),(propertyRange(P,Y),(inverseFunctionalProperty(P),
(propertyDomain(P,Y),addOutput(S,Y)).

Sachant que la définition des règles de relaxation fonctionnelle et non fonctionnelle correspondent à des règles récursives, les prédicats propertyRange et propertyDomain permettent de définir le domaine et le co-domaine d'une propriété. Les prédicats unaires functionalProperty et inverseFunctionalProperty permettent de déclarer les propriétés fonctionnelles et fonctionnelles inverse.

Nous avons utilisé l'environnement SWI-Prolog [19] pour développer le moteur de raisonnement de BioMed. SWI-Prolog est un compilateur Prolog rapide et puissant, qui possède un large panel de prédicats. Il offre un environnement rapide et robuste. Les services Web générés étaientt stockés dans une base de données Oracle 8i. Les tests du moteur de raisonnement BioMed sont décrits dans la section suivante.

6.6.3 Résultats des expérimentations

Une première batterie d'expérimentations a été menée avec des *datasets* comprenant entre 1000, 1500, 2000 et 2500 services annotés sémantiquement avec des termes Mesh et de Protein Ontology. Trente requêtes ont été générées comprenant chacune entre 5 et 10 contraintes fonctionnelles et entre 6 et 9 contraintes non fonctionnelles. Les résultats de cette expérimentation sont illustrés dans la figure 6.12. Certes le nombre de services sélectionnés en utilisant des techniques de raisonnement augmente d'une magnitude (d'un

19. http ://www.swi-prolog.org/

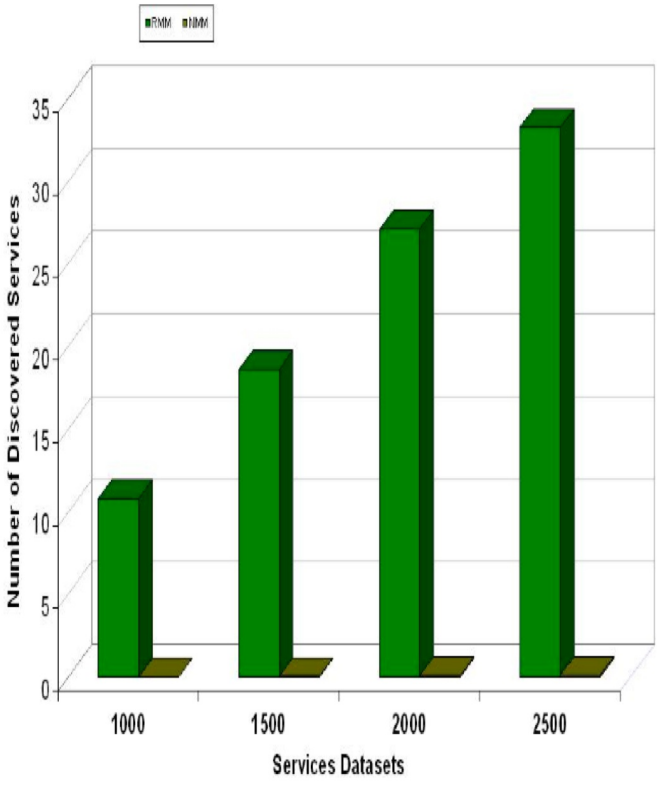


FIGURE 6.12 – Résultats de la première batterie d'expérimentation

ordre de grandeur) par rapport au nombre de services découverts sans l'usage de ces techniques et cela pour chaque taille du dataset. Cependant, il est à noter que le calcul du modèle minimal est exponentiel par rapport à la taille du dataset ce qui a pris plusieurs jours (3 jours) pour le plus grand dataset.

Une seconde batterie d'expérimentations a été menée avec des datasets comprenant un nombre plus important de services (comparé à la première expérimentation), le processus s'effectuant sur huit machines. Nous avons étudié les performances des tâches de raisonnement en reportant le nombre de services sélectionnés lorsque les tâches de raisonnement sont exécutées. Dans la figure 6.13, nous comparons le nombre moyen de services sélectionnés (à l'échelle logarithmique) dans le cas où le processus de découverte repose sur des tâches de raisonnement (RT : Reasoning Task) ou non (WRT : Without Reasoning Task). Nous avons exécuté les trente requêtes en considérant différents datasets comprenant respectivement 1000, 2000, 4000, 6000 et 16000 de services Web. Nous observons que le nombre de services découverts augmente de deux magnitudes (ordres de grandeur sur l'échelle logarithmique) lorsqu'on utilise des tâches de raisonnement. A titre d'exemple, nous extrayons les trois requêtes de découverte étendues suivantes parmi les trente définies dans les expérimentations.

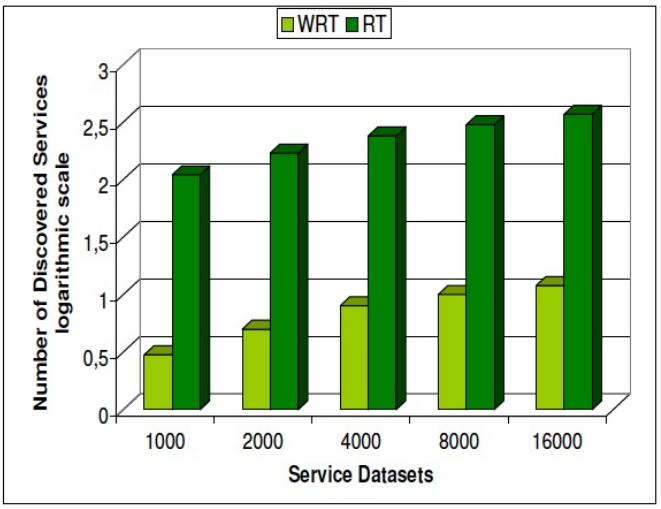


FIGURE 6.13 – Résultats de la seconde expérimentation

query1(S,T,A) :-solveQuery((ifostype(inputatt1,'C04.557.386.480.750.199',_,S), nonFunctionalCondition(S,X,100)),T,A).

query2(S,T,A) :-solveQuery((inputtypes(S,'C04.557.386.480.750.199'), nonFunctionalCondition(S,qosatt4,100)),T,A).

query3(S,T,A) :-solveQuery((((((((((((((inputtypes(S,'D12'), inputtypes(S,'C10.177')), inputtypes(S,'C04')), inputtypes(S,'D01.146')), inputtypes(S,'D12.776.828')),inputtypes(S,'D02.522')), outputtypes(S,'C05')), outputtypes(S,'C15.378')), outputtypes(S,'D08.811')),outputtypes(S,'A06')), outputtypes(S,'C17.800')),outputtypes(S,'D03')), nonFunctionalCondition(S,qosatt2,100)),nonFunctionalCondition(S,qosatt3,100)), nonFunctionalCondition(S,qosatt5,100)),nonFunctionalCondition(S,qosatt8,100)), nonFunctionalCondition(S,qosatt9,100)),T,A).

L'algorithme WTKSF a été implémenté en Java 1.6. L'expérimentation a considéré un dataset de 10.000 services en exécutant 30 requêtes, définissant chacune 10 paramètres QdS. Dans la figure 6.14, nous montrons le ratio de la taille du skyline par rapport au nombre total de services sélectionnés par le moteur de découverte (satisfaisant les contraintes fonctionnelles de la requête de découverte). D'une manière générale, on observe que la taille du skyline est plus grande pour les requêtes ayant un grand nombre de critères QdS. Dans les requêtes étudiées, la taille skyline varie entre 9% et 37% du nombre total des services qui satisfont fonctionnellement la requête.

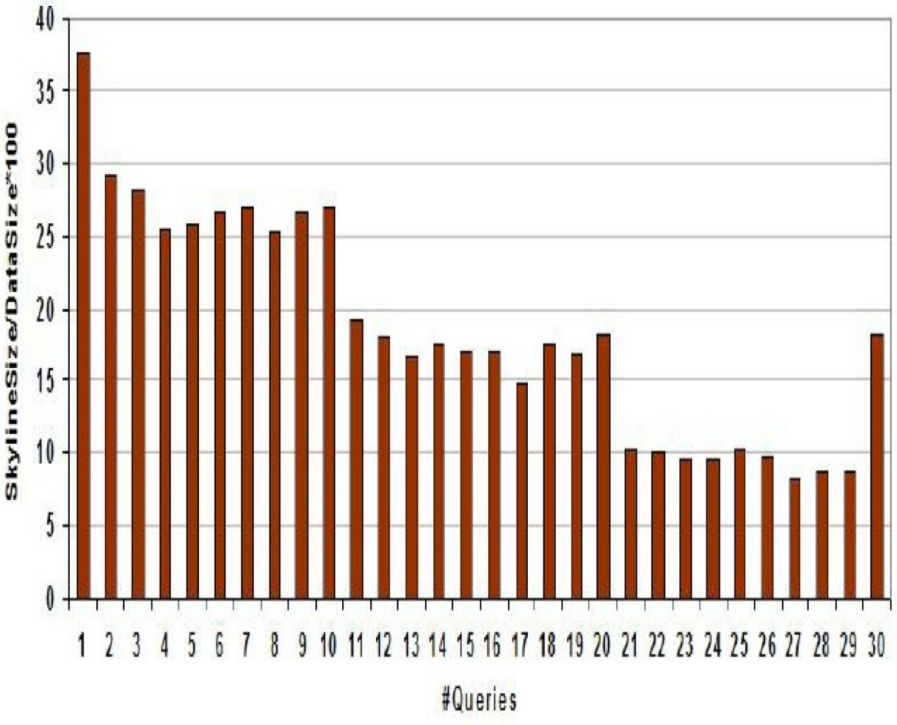


FIGURE 6.14 – Taille du skyline pour les trente requêtes définies pour les expérimentations

Les expérimentations menées montrent l'impact des techniques de relaxation sur la taille de l'espace de recherche des services découverts (augmentant de deux magnitudes dans la seconde expérimentation). Par ailleurs, la taille du skyline pour les requêtes évaluées variant entre 8 et 37 services justifie l'intérêt des techniques de classement proposées.

6.7 Conclusion

Dans ce chapitre, nous avons présenté notre approche de découverte de services Web conçue dans le cadre du méta-framework BioMed. Dans un premier temps, l'approche proposée repose sur les techniques de raisonnement fondées sur l'axiomatique du méta-framework pour déduire l'ensemble des solutions (services) qui satisfont (dans le cas idéal) de manière exacte ou (dans la plupart des cas) de manière approximative les contraintes de la requête. Les requêtes de découverte ici considérées sont étendues c'est à dire couvrent tant les propriétés fonctionnelles que non fonctionnelles d'un service Web. Nous avons montré l'intérêt des techniques de relaxation développées dans cette thèse en considérant un cas de pratique de découverte de services dans le domaine bioinformatique. En l'occurrence, nous avons étudié le problème de découverte dans le catalogue BioMoby, l'un des plus importants catalogues de services dans ce domaine. La découverte dans ce catalogue est basée uniquement sur la navigation. Bien que certains travaux récents en l'occurrence [LA10] aient étudié le problème de découverte dans l'annuaire BioMoby, ils n'ont pas proposé des techniques de raisonnement pour améliorer la qualité du processus comme nous l'avons fait ici. En considérant un certain nombre de questions scientifiques pertinentes, nous avons montré l'impact des approches et techniques développées dans cette thèse sur la taille et la pertinence de l'espace des solutions.

Nous avons proposé une opérationnalisation du méta-framework BioMed dans sa partie inférentielle qui a été conçue comme un méta-service Web déductif. Ce méta-service est représenté par une base déductive Datalog, le niveau extensionnel du méta-service comprend la déclaration de toutes les connaissances descriptives (descriptions sémantisées des services) et les connaissances ontologiques de la méta-carte, le niveau intensionnel comprenant un ensemble de règles implémentant les axiomes du modèle inférentiel. Les techniques de Datalog ont été utilisées afin de représenter le méta-service mais également pour évaluer une requête de découverte représentée sous forme conjonctive.

Les expérimentations menées ont considéré des datasets de services Web synthétiques de différentes tailles. Les services synthétiques ont été générés automatiquement à partir de termes de Mesh et de concepts de Protein Ontology. Les premières expérimentations ont révélé que le calcul du modèle minimal pour un programme Datalog est exponentiel au nombre de services.

En second lieu, l'approche de découverte a intégré une phase de classement permettant de déceler les meilleurs services parmi ceux sélectionnés par le moteur de raisonnement. L'approche a permis de déterminer le **skyline de services** désignant l'ensemble des services non dominés par aucun autre service au niveau des critères de Qds. La sélection des top-k meilleurs services a été effectuée grâce à l'heuristique maximisant une fonction de score calculant la fréquence du service dans tous les sous-espaces multi-dimensionnels. Cette heuristique repose sur l'algorithme WTKSF [GVRA09, GVRA10].

Enfin, les expérimentations effectuées montrent aussi bien la robustesse que la scalabilité de l'approche proposée.

Dans le chapitre suivant, nous présentons notre approche de composition de services Web qui vise à étendre les techniques de raisonnement afin de résoudre une requête étendue nécessitant l'interaction de plusieurs services.

Chapitre 7

La composition de Services Web dans BioMed

Sommaire

7.1 Introduction

Le dernier chapitre de cette thèse présente l'approche de composition de services Web proposée dans le cadre du méta-framework BioMed. A l'instar de l'approche de découverte, l'approche de composition proposée considère à la fois des contraintes fonctionnelles et non fonctionnelles afin de sélectionner l'ensemble des services composables fonctionnellement, vérifiant les contraintes NF et ayant les valeurs QdS les plus satisfaisantes. Pour ce faire, nous ajoutons deux nouveaux axiomes, au niveau de la sémantique axiomatique, définissant la notion de **composabilité fonctionnelle** entre services. Ainsi, un graphe de composition déduit par le moteur de raisonnement de BioMed comprend l'ensemble des services vérifiant les critères de composabilité. Ce graphe peut consister en un (ou plusieurs) plans de composition possibles, chacun sera évalué en termes de critères QdS afin de distinguer les **solutions admissibles** (celles qui satisfont les contraintes non fonctionnelles) et ensuite sélectionner les meilleures solutions parmi celles admissibles en se basant sur un algorithme de *ranking*.

Dans la première section de ce chapitre, nous formalisons le problème de composition et définissons la notion de composabilité fonctionnelle. Ensuite, nous discutons la composabilité non fonctionnelle.

7.2 Composabilité fonctionnelle

Nous définissions la notion de requête étendue de composition comprenant un ensemble de contraintes fonctionnelles et non fonctionnelles.

Définition 27 (Requête de composition) *Une requête de composition est définie par un tuplet Q=<F, NF>, où F et NF désignent respectivement un ensemble de propriétés fonctionnelles et un ensemble de propriétés non fonctionnelles définies telles que :*

- *F=(I,O) où I est un ensemble d'entrées typées de la requête Q et O est l'ensemble de sorties typées de la requête Q.*
- *NF est un ensemble de quadruplets (P,C,V,U) où P est un paramètre non fonctionnel, C est un opérateur logique de comparaison, V est la valeur du paramètre et U est l'unité du paramètre P. L'évaluation de la requête Q doit respecter la conjonction de expressions exprimés par NF. Si (P, C_i, V_i, U_i) et (P, C_j , V_j , U_j) ∈ NF alors $C_i = C_j$, $V_i = V_j$ et $U_i = U_j$.*

Au niveau du méta-framework BioMed, une requête utilisateur est prise en compte par le moteur de découverte, en premier lieu, si aucun service atomique ne satisfait la requête alors le moteur de composition qui prend le relais.

La composabilité fonctionnelle permet de définir un (voire plusieurs) plan(s) de composition en réponse à une requête Q en sélectionnant **un ensemble de services fonctionnellement composables** et ce en considérant uniquement leurs signatures fonctionnelles (en termes d'entrées et de sorties). Un graphe de composition est l'union de tous les plans de composition identifiés.

Au niveau de ce graphe, les services sont inter-reliés par deux types de dépendance : **logique** ou **empirique** et cela selon l'approche utilisée pour déduire cette dépendance. Sachant qu'une dépendance dans un graphe de composition entre deux services indique que ces deux derniers sont composables. Ainsi, deux services S_i et S_j peuvent être interconnectés par dépendance logique, si (idéalement) l'ensemble des sorties de S_i est sémantiquement équivalent à l'ensemble des entrées de S_j, i.e., $\forall\ o \in O_i : \exists\ i \in I_j,\ o \equiv i$. Les techniques de relaxation présentées au niveau du modèle inférentiel de BioMed permettent de relaxer cette propriété afin de déduire une dépendance logique approximative entre services.

Le second type de dépendance dite **empirique** est déduite par calcul de similarité sémantique entre les concepts en sortie d'un service et les entrées de son successeur. Nous faisons usage de certaines métriques exploitant les relations taxinomiques au niveau de la méta-carte ontologique pour déduire une dépendance empirique. Nous annotons les dépendances entre services au niveau du graphe de composition par la valeur de similarité γ_D obtenue lors de l'évaluation de la métrique. Sachant que les dépendances logiques sont annotées par l'unique valeur γ_D=1.

Nous explicitions dans la prochaine section deux nouveaux axiomes proposés formalisant la notion de composabilité fonctionnelle. Nous montrons que ces axiomes garantissent la définition de quatre patrons de composition (sequential, And Join, And Split et Loop illustrés par la figure 7.2.). La section 7.2.2. présente un aperçu des mesures de similarité pouvant être adoptées pour évaluer la similarité entre concepts en se basant sur une

hiérarchie de concepts.

7.2.1 Axiomatique inférant une dépendance logique

Nous proposons d'étendre l'axiomatique du modèle inférentiel BioMed par deux nouveaux axiomes permettant de déduire des dépendances entre services. L'axiome 9 définit formellement trois conditions de composabilité séquentielle entre deux services S_1=(I_1, O_1) et S_2=(I_2, O_2). Le service S_3=(I_1, O_2) est défini comme le service résultat de la composition en séquence des services. L'axiome 9 est énoncé comme suit :

Axiome 9

$$\frac{S_1 \odot S_2}{S_3} \begin{cases} \textit{Composabilité exacte :} \\ \forall\ c \in O_2, \exists\ c^{'} \in I_1 \text{ tel que, } c\ \textit{isEquivalentTo}\ c^{'}\ ; \\ \textit{Composabilité partielle :} \\ \exists\ c \in O_2, \exists\ c^{'} \in I_1 \text{ tel que, } c\ \textit{isEquivalentTo}\ c^{'}\ ; \\ \textit{Composabilité approximative :} \\ \forall\ c \in O_2, \exists\ c^{'} \in I_1 \text{ tel que, } c^{'} \in \textit{instancesOf}(c) \cup \\ \textit{subClasses}(c) \cup \textit{functionalProperties}(c) \cup \\ \textit{InversefunctionalProperties}(\text{c}). \end{cases}$$

L'axiome 9 définit les règles de composabilité permettant de déduire une dépendance logique entre services Web en distinguant la composabilité exacte, la composabilité partielle et la composabilité approximative.

1. **La composabilité exacte** implique l'équivalence sémantique entre l'ensemble des sorties d'un service et des entrées de son successeur.
2. **La composabilité partielle** permet de déduire une dépendance logique entre S_1 et S_2 en considérant seulement un sous-ensemble des sorties de S_1 plutôt que l'ensemble de tous les paramètres ; en effet, seulement un sous-ensemble des paramètres en sorties de S_1 sera matché à un sous-ensemble de paramètres en entrée de S_2.
3. **La composabilité approximative** repose sur les différentes techniques de relaxation :
   - *La relaxation par subsomption.* Plutôt que d'avoir une compatibilité exacte, une relaxation par subsomption permet de relaxer les paramètres de sorte d'accepter comme entrées S_2 toutes les sorties de S_1 qui sont subsumés par une entrée de S_2.
   - *La relaxation assertionnelle.* Si la sortie o d'un service S_1 appartient à l'ensemble des instances d'une entrée i en entrée à un service S_2, une dépendance logique sera établie entre S_1 et S_2.
   - *La relaxation fonctionnelle et fonctionnelle inverse.* Enfin, il est possible de relaxer les paramètres d'un service avec toutes les propriétés fonctionnelles et non fonctionnelles de ses entrées.

L'axiome 10 permet de déduire un nouveau service S_3 comme l'union de deux services S_1 et S_2 comme suit :

Phase du Workflow	Services sélectionnés
Rechercher les séquences d'un gène	`GetEntryDDBJ` [1], `eFetch` [2], `eFetchSequence` [3]
Recherche de séquences similaires	`WSNCBIBlast` [4], `WSFASTA` [5], `runBLATService` [6]
Construction d'un arbre phylogénétique	`runPhylipFromDna` [7], `runCreateTreeFromClustalW` [8]

TABLE 7.1 – L'ensemble des services sélectionnés

Type de patron	Définition
Sequential	$S_1 \odot S_2$
And Split (Fork)	$S_1 \odot (S_2 \oplus S_3 .... \oplus S_{n+1})$
And Join (Merge)	$(S_1 \oplus S_2 .... \oplus S_n) \odot S_{n+1}$
Loop	$S_1 \oplus S_1$

TABLE 7.2 – Définitions de quatre patrons de composition

Axiome 10

$$\frac{S_1 \oplus S_2}{S_3} \begin{cases} si \quad \mathrm{S_1}= (I_1, O_1) \text{ et } S_2=(I_2, O_2) \text{ alors} \\ \mathrm{S_3}=(I_3, O_3) \text{ tel que } I_3= I_1 \cup I_2 \text{ et } O_3=O_1 \cup O_2. \end{cases}$$

Les deux axiomes proposés permettent de définir quatre parmi les six patrons de composition comme illustré par la figure 7.2 (exposés dans le chapitre 3 de l'état de l'art). La Table 7.2 résume la définition des quatre patrons de composition en utilisant les deux axiomes.

Exemple 13 *La conception et l'implémentation de workflows scientifiques constituent un des plus puissants outils de la discipline de la biologie "in silico" qui est la bioinformatique. Les workflows scientifiques visent à définir une chaîne de traitement afin de répondre à une question scientifique complexe. Pour illustrer ceci, nous prenons l'exemple d'un scientifique qui désirer étudier l'homologie d'un gène humain avec d'autres gènes provenant d'autres espèces (on parle précisément dans ce cas de gènes orthologues). Ceci peut être réalisé par la construction d'un arbre phylogénétique permettant de mesurer le degré de parenté entre les taxons (les organismes vivants) mettant en évidence les gènes homologues à un gène humain spécifique. Ce workflow nécessite trois étapes : en premier lieu, il s'agit de requêter à partir d'une base de séquence la séquence ADN du gène, puis exécuter un alignement de séquence pour rechercher les séquences similaires à la séquence du gène pour enfin exécuter un service d'analyse phylogénétique afin de générer un arbre phylogénétique.*

Comme l'illustre la Table 7.1, un ou plusieurs services peuvent être sélectionnés et puis orchestrés selon les règles de composabilité définis par l'axiome 9.

7.2.2 Mesures de similarité inférant une composabilité empirique

Le raisonneur déductif de BioMed permet d'inférer des appariements approximatifs entre les concepts annotant les entrées et les sorties d'un service Web et une requête de découverte comme présenté dans le chapitre 6 de ce mémoire. De plus, par le biais des

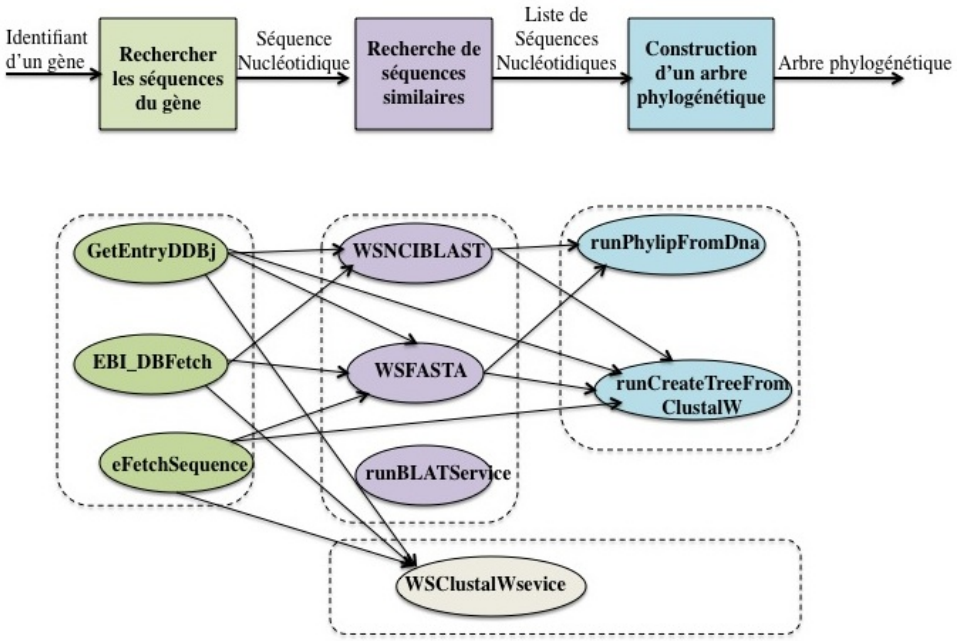


FIGURE 7.1 – Le Workflow scientifique de l'étude phylogénétique et les différentes alternatives de solutions de composition

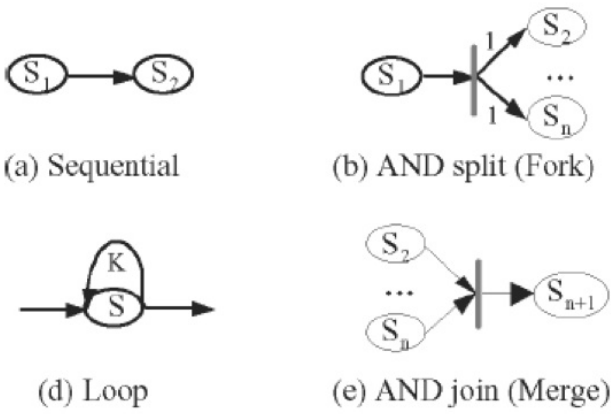


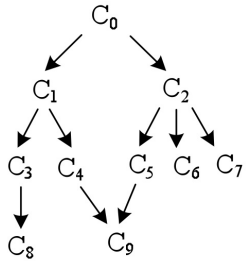
FIGURE 7.2 – Les quatre patrons choisis : Sequantial, AndSplit, AndJoin et Loop

Plans	Services
$Plan_1$	GetEntryDDBJ, WSNCBIBlast, runPhylipFromDna
$Plan_2$	GetEntryDDBJ, WSNCBIBLAST, runCreateTreeFromClustalW
$Plan_3$	GetEntryDDBJ, WSFasta, runPhylipFromDna
$Plan_4$	GetEntryDDBJ, WSFasta, runCreateTreeFromClustalW
$Plan_5$	GetEntryDDBJ, runClustalW
$Plan_6$	DBFetch, WSNCBIBlast, runPhylipFromDna
$Plan_7$	DBFetch, WSNCBIBLAST, runCreateTreeFromClustalW
$Plan_8$	DBFetch, WSFasta, runPhylipFromDna
$Plan_9$	DBFetch, WSFasta, runCreateTreeFromClustalW
$Plan_{10}$	DBFetch, runClustalW
$Plan_{11}$	eFetchSequence, WSNCBIBlast, runPhylipFromDna
$Plan_{12}$	eFetchSequence, WSNCBIBLAST, runCreateTreeFromClustalW
$Plan_{13}$	eFetchSequence, WSFasta, runPhylipFromDna
$Plan_{14}$	eFetchSequence, WSFasta, runCreateTreeFromClustalW
$Plan_{15}$	eFetchSequence, runClustalW

TABLE 7.3 – Ensemble des différents plans de composition

axiomes 9 et 10 définis dans la section précédente il est possible de déduire une composabilité fonctionnelle empirique entre les entrées d'un service et les sorties de son prédécesseur. Les techniques de relaxation présupposent que les concepts se situent sur le même chemin taxinomique dans l'arbre ontologique. Cependant, si deux concepts ne se trouvent pas sur le même chemin dans l'arbre ontologique, l'appariement approximatif échoue. Nous proposons d'étendre les techniques d'appariement basées initialement sur seulement le raisonnement déductif avec des techniques de mesures de similarité sémantique. L'adoption des mesures de similarité peut être proposée dans le contexte de la découverte et/ou la composition de services Web afin de pallier les insuffisances du raisonnement déductif. Nous illustrons notre propos avec l'exemple suivant.

Exemple 14 *Nous considérons l'exemple de trois services S_1, S_2 et S_3. Le service S_1 accepte une entrée de type sémantique un concept C_2 et retourne deux sorties de type C_4 et C_2. Le service S_2 accepte en entrée un concept de type C_3 et retourne deux sorties de type C_9 et C_7. Enfin, S_3 prend en entrée un concept C_5 et renvoie en sortie un concept C_3 et C_6. La hiérarchie de concept est illustrée par la figure (a). Enfin, la requête Q exprime un besoin d'un service qui accepte en entrée un concept C_8 et retourne en sorties deux paramètres de type sémantique respectivement C_4 et C_7. Il est clair qu'aucun service ne matche exactement la requête Q. Cependant, le service S_1 matche en plugin la requête ; en opposition au service S_2 qui matche en subsomption la requête Q. Par contre, il est impossible de déduire un matching entre S_3 et Q puisque d'une part les concepts C_5 et C_8 ne sont pas sur le même chemin taxinomique et d'autre part les concepts en sortie ne se trouvent pas sur le même chemin. L'intérêt de techniques de similarité est de mesurer le degré de similarité entre deux concepts ne se trouvant pas sur le même chemin. Evidemment, face à ce problème, la majorité des mesures utilise les liens taxinomiques pour calculer le degré de similarité.*



(a) Hiérarchie de Concepts

	INPUT	OUTPUT
Q	C_8	C_4, C_7
S_1	C_1	C_4, C_2
S_2	C_3	C_9, C_7
S_3	C_5	C_3, C_6

(b) Descriptions de Services et d'une requête

7.2.2.1 Notion de similarité, de distance et de degré de relation sémantique

Il existe trois types de mesures sémantiques, chacune ayant des propriétés différentes :
- **La similarité sémantique** : On parle de mesure de similarité sémantique (i.e., *semantic similarity*) lorsque la mesure calcule si deux concepts sont similaires sémantiquement, au sens où ils partagent des propriétés et attributs communs.
- **Le degré de relation sémantique** : On parle de degré de relation sémantique (i.e., *semantic relatedness*) lorsque la mesure calcule si deux concepts sont reliés sémantiquement, au sens où ils sont liés dans leur fonction.
- **La distance sémantique** : On parle de distance sémantique (i.e. semantic distance) lorsque la mesure calcule si deux concepts sont distant sémantiquement.

La similarité sémantique et le degré de relation sémantique sont intéressants à présenter ensemble car ils ont le même type de monotonie : plus deux concepts sont reliés ou similaire, plus le score sémantique est grand. Autrement dit, soit $scr(x, y) \in [0, 1]$ une mesure donnant un score sémantique entre 0 et 1, alors que scr représente une similarité sémantique ou un degré de relation sémantique, nous avons $scr(x, x) = 1$. Néanmoins, le degré de relation sémantique tient compte de plus d'informations que la similarité sémantique, puisqu'il utilise aussi les relations fonctionnelles entre les concepts. Ainsi, le degré de relation sémantique peut être considéré comme une généralisation de la similarité sémantique [Res95].

D'autre part, la courbe d'une fonction de distance sémantique est l'exact opposé de la courbe de similarité ou de degré de relation (les conversions linéaires d'une forme à l'autre sont ainsi couramment utilisées [Res95, JC97]). En ce sens, en reprenant la notation $scr(x,y)$ précédente, dans une distance sémantique nous avons $scr(x, x) = 0$. L'inconvénient principal de la notation de distance est qu'elle est ambiguë. En effet, elle désigne aussi bien l'inverse de la similarité sémantique que l'inverse du degré de relation sémantique.

L'évaluation du lien sémantique entre deux concepts dans une ontologie est un problème qui se pose de longue date dans le domaine de l'intelligence artificielle. La similarité sémantique est une évaluation du lien sémantique entre deux concepts dont le but est d'estimer le degré par lequel les concepts sont proches dans leur sens [Res99]. La définition donnée par Lin de la similarité sémantique repose sur trois suppositions [Lin98]. La similarité entre deux concepts est liée aux caractéristiques qu'ils ont en commun (plus ils ont de caractéristiques communes, plus les concepts sont similaires) et à leurs différences

(plus deux concepts sont différents, moins ils sont similaires). La similarité maximale est obtenue lorsque deux concepts sont identiques.

7.2.2.2 Mesures sémantiques exploitant des taxinomies de concepts

Au départ, nous fixons un certain nombre de notations que nous utiliserons pour décrire les formules sémantiques des différentes mesures présentées dans cette section :

- c_1 et c_2 : Les concepts dont nous chercherons la distance.
- $dist_{XXX}$ et sim_{XXX} : Notation pour la distance et la similarité sémantique. Nous remplacerons XXX par l'acronyme du nom des auteurs.
- $ccp(c_1, c_2)$: Le noeud de la hiérarchie qui est le plus proche parent commun de c_1 et c_2 (i.e. *closest common parent*, notation de Zhong [ZZL+02]). Cette formulation est évidemment plus courante dans les systèmes dont les connaissances sont représentées uniquement sous forme de hiérarchie, mais elle est aussi souvent utilisée dans les systèmes possédant d'autres types de relations, la relation de hiérarchie ayant en général un traitement spécial prioritaire.
- *root* : Le noeud racine de la hiérarchie. Il représente le concept le plus général, englobant tous les concepts. Il est souvent nommé *thing* dans la hiérarchie de concepts (root étant le nom plus calculatoire lié à la structure arborescente d'une hiérarchie).
- *depth*(c) : La profondeur du noeud c dans la hiérarchie avec *depth(root)* = 0.
- $sp(c_1, c_2)$: Le plus court chemin entre c_1 et c_2 en suivant la hiérarchie (i.e., *shortest path*). Ce plus court chemin est un ensemble de couples de concepts, un couple représentant une arête et l'ensemble de ces arêtes le chemin.
- $len(c_1,c_2)$ =sp(c_1, c_2) : La longueur du plus court chemin entre c_1 et c_2.
- $MAX = maxdepth$(c) : La hauteur maximale de la hiérarchie.

Il est intéressant de noter que par construction, le plus court chemin entre deux concepts dans une hiérarchie passe toujours par leur plus proche parent commun [9]. Cette propriété, bien que rarement directement nommée, permet de rapprocher certaines formules de distance à première vue radicalement différentes.

Nous présentons dans ce qui suit les mesures sémantiques se basant sur les longueurs de chemins. La source de connaissance est vue comme un arbre ontologique et les formules utilisent la longueur des chemins (pondérés ou non) pour déterminer la distance entre deux concepts. Cette idée part d'une intuition naturelle : dans une hiérarchie sémantique, plus deux concepts sont éloignés physiquement, plus ils sont éloignés sémantiquement.

7.2.2.3 Rada

Rada et al. fut le premier à envisager la distance entre concepts comme une distance métrique[RMB+89]. Il utilise comme source de connaissances la hiérarchie MeSH. En ce sens, il affirme que toute distance sémantique se doit de respecter les hypothèses de bases d'une distance métrique, à savoir si $f(x,y)$ est une distance métrique :

9. Cette hypothèse n'est valable que dans le cas où la hiérarchie n'autorise pas de multi-héritage. Dans le cas du multi-héritage, elle ne reste valable que si il n'existe qu'un seul et unique chemin hiérarchique entre les concepts cibles

1. Existence du zéro : $\forall\ x,\ f(x,\ x) = 0$
2. Symétrie : $\forall$(x, y), f(x,y) = f(y, x)
3. Positivité : $\forall$(x, y) f(x, y) ≥ 0
4. L'inégalité triangulaire : $\forall$(x, y, z), f(x,y) + f(y, z) $\geq$ f(x ; z)

Les trois premiers points sont admis par toutes les mesures actuelles. En revanche, l'inégalité triangulaire est souvent discutée et finalement rarement respectée. En effet, si les arêtes sont pondérées, alors elle n'est souvent plus valide [ZZL+02]. De même, la présence d'héritage multiple invalide cette propriété.

Rada et al. définit sa mesure de distance simplement :

$$dist_{RAD}(\text{c1,c2}) = \frac{1}{len(c_1,c_2)}$$

où *len* est la fonction donnant la longueur du plus court chemin entre c1 et c2.

7.2.2.4 Zhong

Zhong dans [ZZL+02] utilise le principe de profondeur de sorte que la profondeur d'un noeud est représentative de la spécificité d'un concept. Pour ce faire, Zhong définit un coefficient (le *milestone*) pour chaque noeud n de sa hiérarchie. Cette valeur décroît en fonction de la profondeur et rentre en compte dans la formule finale :

$$milestone(\text{n}) = \frac{1}{k^{depth(n)+1}}$$

où k est un paramètre permettant d'intensifier ou de diminuer la rapidité d'évolution du *milestone* en fonction de la profondeur. En pratique, k est toujours instancié à 2 (tel qu'utilisé entre autres dans le moteur de recherche sémantique Corese [CDF04]). La distance entre deux concepts c_1 et c_2 est alors définie par le *milestone* de c_1 et c_2 et par celui de leur plus proche parent commun $ccp(c_1,\ c_2)$. On obtient alors : $dist_{ZHO}(\text{c}_1,\ c_2) = 2 * milestone(ccp(c_1,\ c_2))$ - $(milestone(c_1) + milestone(c_2))$

On remarque qu'en fait cette formule calcule une distance entre c_1 et $ccp(c_1,\ c_2)$ (avec le calcul $milestone(\text{ccp}(c_1,\ c_2)$ - $milestone(c_1))$ puis une distance entre c_2 et $\text{ccp}(c_1,\ c_2)$ (avec le calcul $milestone(\text{ccp}(c_1,\ c_2)$ - $milestone(c_2))$ et les additionne, permettant de faire l'analogie entre cette distance et une distance métrique pondérée.

7.2.2.5 Wu et Palmer

Wu et Palmer [WP94] travaillent sur la traduction automatique de l'anglais vers le Mandarin. Leur source de connaissance est WordNet. Leur formule utilise le plus proche parent des deux concepts x et y et de sa profondeur.

La distance sémantique s'énonce :

$$dist_{WP}(\text{c}_1,\ c_2) = \frac{len(c_1,ccp(c_1,c_2))+len(c_2;ccp(c_1,c_2))}{len(c_1,ccp(c_1,c_2))+len(c_2,ccp(c_1,c_2))+2*depth(ccp(c_1,c_2))}$$

Cette distance utilise encore une fois l'hypothèse de profondeur. En effet, si l'on fixe l'ensemble des longueurs entre deux concepts (i.e. len(c_1, $ccp(c_1, c_2)$) et $len(c_2, ccp(c_1, c_2))$ sont alors constants), la distance diminue au fur et à mesure que la profondeur augmente (i.e. le fait que *depth*(ccp(c1, c2)) soit présent au dénominateur diminue la distance quand la profondeur augmente). **Cette mesure est ainsi sûrement l'une des plus précises dans la catégorie des mesures à longueur de chemin**. Cependant, il est à noter que les mesures utilisant la théorie de l'information (i.e., un corpus informationnel) obtiennent encore de meilleurs résultats. Un état de l'art et une comparaison de toutes les mesures dans le mémoire de thèse de Ferihane Kboubi [Kbo10] .

Enfin, nous définissons un graphe de composition comme un plan (chemin) de services fonctionnellement composables comme suit.

Définition 28 (Graphe de composition) *Un graphe de composition est un graphe étiqueté de services* $P= \langle \mathbb{S}, D, c, \gamma_D \rangle$ *tel que :*

- $\mathbb{S}=\{S_1,...,S_n\}$ *est un ensemble fini de services Web de cardinalité n,*
- $D : \mathbb{S} \times \mathbb{S}$ *est un ensemble fini de dépendances entre services,*
- $c : D \rightarrow$ {*logique, empirique*} *permettant d'associer à chaque dépendance un type soit logique ou soit empirique (étiquette de la dépendance),*
- $\gamma_D : D \rightarrow [\,0,1.$

La première phase de l'approche de composition BioMed est à construire des chemins (plans) de composition constitués d'un ensemble de services atomiques composables de manière exacte ou approchée selon un ou plusieurs patrons de composition, le tout constituant un graphe de composition.

Un graphe de composition est généré à la suite de cette phase pouvant contenir un ou plusieurs plans de composition entre les entrées et les sorties souhaitées. Chaque plan de composition contient un nombre variable noté p de services atomiques composables. La valeur p est variable d'un plan de composition possible à un autre ; la valeur minimale de p étant 2 (un plan comprenant seulement deux services atomiques). Dans une seconde phase, nous identifions parmi les plans de composition ceux qui satisfassent les contraintes non fonctionnelles. Un mécanisme de classement prend place pour identifier les plans les plus optimaux en termes de QdS.

7.3 Composabilité non fonctionnelle et Sélection des meilleurs plans

Comme signalé dans le chapitre 4 de ce mémoire la recherche du plan le plus satisfaisant de composition étant un problème NP-difficile nous proposons deux heuristiques pour résoudre approximativement le problème de l'optimisation de la composabilité non fonctionnelle [10]. Ces deux heuristiques permettent d'identifier les plans de composition satisfaisant au mieux les contraintes non fonctionnelles d'une requête Q. Le première est

10. Nous avons consacré l'annexe B à l'optimisation multi-critère et à l'heuristique Skyline. Outre cette dernière heuristique, d'autres heuristiques non Pareto-like existent comme les algorithmes génétiques, l'Analyse Formelle des Concepts, le clustering, etc.

Propriété QoS	*Séquentiel*	*And*	Loop
le temps de réponse	$\sum_{i=1}^{n} rt_i$	$max(rt_i)$	$rt \times k$
Le débit	$\min(th_i)$	$\min(th_i)$	th
La disponibilité	$\prod_{i=1}^{n} av_i$	$\prod_{i=1}^{n} av_i$	av^k
La fiabilité	$\prod_{i=1}^{n} re_i$	$\prod_{i=1}^{n} re_i$	re^k

TABLE 7.4 – Agrégation des paramètres QoS dans un patron de composition

Service	Temps de réponse	Débit	Disponibilité
GetEntryDDBJ (S_1)	15	10	95
DBFetch (S_2)	10	12	98
eFetchSequence(S_3)	10	18	96
WSNCBIBlast (S_4)	8	20	98
WSFASTA (S_5)	10	15	98
runBlatService (S_6)	10	10	99
runPhylipFromDna (S_7)	10	15	97
runCreateTreeFromClustalW(S_8)	8	15	98
runClustalW (S_9)	13	15	97

TABLE 7.5 – Les services sélectionnés pour le workflow de l'analyse phylogénétique

dite approche de composabilité non fonctionnelle *globale* dans le sens où elle évalue les plans en termes de QdS agrégés et la seconde est dite *locale* dans le sens où les contraintes non fonctionnelles sont évaluées au niveau de chaque service atomique S_{ij} d'un plan de composition PC_i

7.3.1 Approche globale

Etant donné un graphe de composition, la seconde phase de l'approche de composition proposée dans cette thèse consiste à étiqueter chaque plan de composition PC_i par les valeurs globales (agrégées) de ses attributs QdS. Le graphe de composition est représenté une matrice de sorte que chaque plan $Plan_i$ sera associé à une valeur de QdS par l'agrégation des valeurs des critères de ses services atomiques selon le Tableau 7.4.

La matrice des plans de composition et des valeurs agrégées de leur QdS se présente comme suit :

$$\begin{pmatrix} & \text{QdS}_1 & \text{QdS}_2 & .... & \text{QdS}_r \\ \text{Plan}_1 & ... & ... & .. & \\ \text{Plan}_2 & .. & .. & .. & \\ ... & & & & \\ \text{Plan}_n & & & & \end{pmatrix}$$

A une requête étendue comprenant m contraintes QdS parmi les r contraintes possibles, la matrice (n,r) sera réduitre à une matrice (n,m) telles que les m colonnes parmi les r correspondent aux attributs QdS choisis par le client. Nous procédons ensuite en deux étapes :

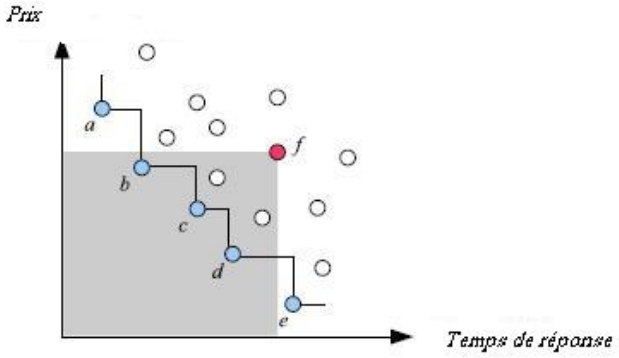


FIGURE 7.3 – Exemple de Skyline de Plan de composition

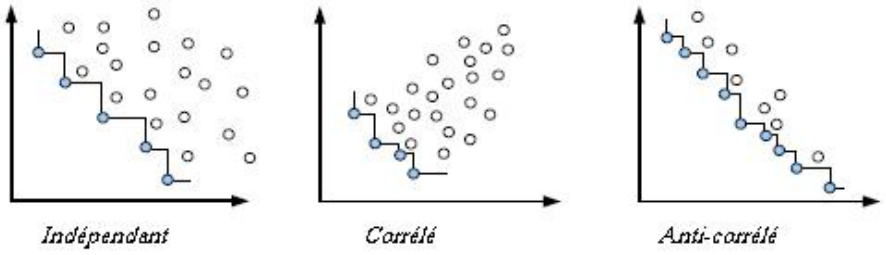


FIGURE 7.4 – Skyline de différents types de datasets

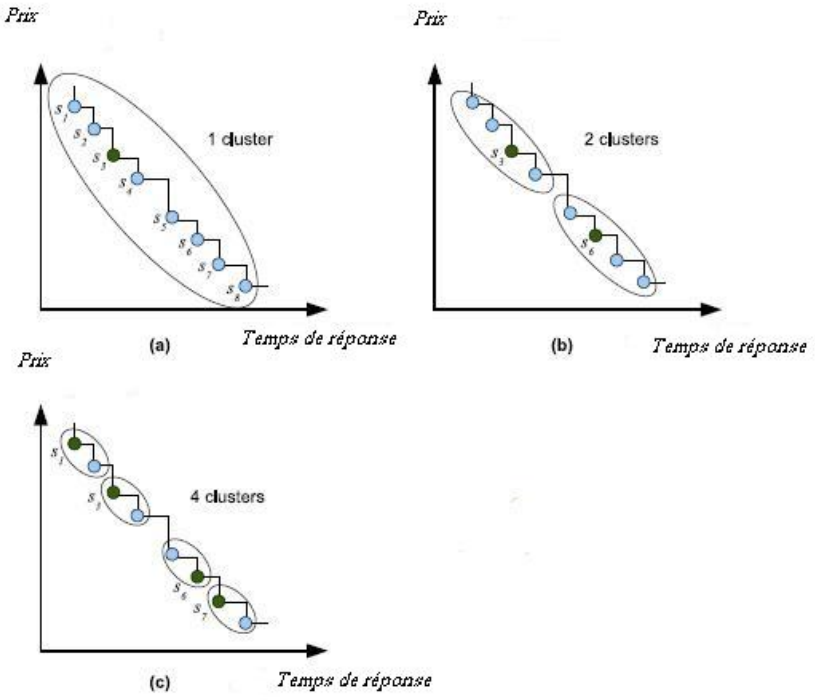


FIGURE 7.5 – Clusterisation hiérarchique d'un skyline de services

1. La première étape correspond à un filtrage permettant de ne retenir parmi les plans de composition que ceux dont toutes les valeurs QdS agrégées vérifient les contraintes globales de la requête. **Ces plans seront dits admissibles**.

2. La seconde étape consiste à chercher parmi les plans admissibles le meilleur plan, i.e., celui qui maximise au plus les valeurs agrégées des attributs '*positifs*' et minimise au plus les valeurs agrégées des attributs '*négatifs*' et ce, en utilisant une méthode de décision multicritère basée l'opérateur *Skyline*. La technique de skyline nous permettra de déterminer l'ensemble de services qui ne sont dominés par aucun autre service. La figure 7.3 illustre un skyline de plans de composition dans un espace bidimensionnel dont le premier axe désigne la disponibilité et le second le temps de réponse. Nous remarquons que le plan a appartient au skyline puisqu'il n'existe un plan ayant un meilleur temps de réponse **et** en même temps un prix plus petit. C'est le cas des plans b, c, d et e. Contrairement au service f qui ne fait pas partie du skyline et il est dominé par les plans b, c et d. Notons qu'un skyline de plans admissibles de composition comprend un ensemble de plans incomparables entre eux, en l'absence de préférences prédéfinies. En l'occurrence, un client pourrait être intéressé par le plan a à cause de son temps de réponse minime malgré son prix élevé, un autre client serait plutôt intéressé par le plan e. Cependant la taille d'un skyline peut être importante et cela dépend étroitement de la distribution des données QdS et la corrélation entre les différents paramètres QdS. La figure 7.4 illustre l'exemple de trois configurations de taille de skyline dans un espace bi-dimensionnel : (a) des dataset indépendants où les valeurs des deux paramètres QdS sont indépendantes, (b) des datasets corrélés, un service qui est bon par rapport à critère, il l'est aussi par rapport à l'autre critère, (c) des datasets anti-corrélés lorsqu'il y a une véritable distribution au niveau des datasets. Le nombre de services dans un skyline est relativement petit pour les datasets corrélés, moyen pour les datasets indépendants et large pour les datasets anti-corrélés.

   Lorsque la taille d'un skyline des plans admissibles de composition est large, il est possible de classer ces plans grâce à une méthode de ranking comme celles de FAGIN et al. (TA : Threshold Algorithm) ou NRA (No Random Access) exposées dans [FLN01, FKS03], LARA (Lattice-based Rank Aggregation) proposée dans [MYCC07], ou WTKSF [GVRA09] ou encore TKSI proposée dans [GV09]. Bien d'autres algorithmes efficients de ranking sont disponibles et sont cités dans la bibliographie de ce document comme par exemple [YST+09] ou [KPSV09]. Il est aussi tout à fait possible de se baser sur une méthode de clusterisation hiérarchique pour identifier des clusters contenant les plans les plus représentatifs du skyline comme l'illustre la figure 7.5.

Exemple 15 *Nous considérons l'ensemble des services sélectionnés comme candidats pour l'implémentation de l'exemple 13. Nous spécifions pour chacun des services la valeur de leurs propriétés QdS relatives au temps de réponse, débit et disponibilité. Le tableau 7.5 (page 187) illustre toutes les valeurs QdS pour chaque service. Le graphe de composition déduit comprend quinze plans de composition possibles dont les valeurs QdS agrégées sont illustrés par la matrice suivante :*

	QdS_1	QdS_2	QdS_3
$Plan_1$	33	10	96
$Plan_2$	31	10	97
$Plan_3$	35	10	97
$Plan_4$	33	11	97
$Plan_5$	28	11	97
$Plan_6$	28	11	97.6
$Plan_7$	28	11	97.6
$Plan_8$	30	11	97.6
$Plan_9$	28	11	98
$Plan_{10}$	26	11	98
$Plan_{11}$	26	15	97
$Plan_{12}$	26	15	97.33
$Plan_{13}$	30	15	97
$Plan_{14}$	28	15	97.33
$Plan_{15}$	23	15	97

Supposons que le scientifique soit intéressé par un plan de composition dont le temps de réponse ne dépasse pas 30 ms, avec un débit au moins égal à 10 et une disponibilité au moins égale à 97%. Ainsi, les plans admissibles sont les suivants :

	QdS_1	QdS_2	QdS_3
$Plan_6$	28	11	97.6
$Plan_7$	28	11	97.6
$Plan_8$	30	11	97.6
$Plan_9$	28	11	98
$Plan_{10}$	26	11	98
$Plan_{11}$	26	15	97
$Plan_{12}$	26	15	97.33
$Plan_{13}$	30	15	97
$Plan_{14}$	28	15	97.33
$Plan_{15}$	23	15	97

Le skyline des services correspondant à l'espace de solutions admissibles comprend les trois services suivants :

	QdS_1	QdS_2	QdS_3
$Plan_{10}$	26	11	98
$Plan_{12}$	26	15	97.33
$Plan_{15}$	23	15	97

7.3.2 Approche locale

La première étape consiste à sélectionner parmi les plans possibles de composition ceux qui vérifient toutes les contraintes QdS localement, i. e. que tous les services atomiques composant le plan vérifient les contraintes QdS de la requête. La seconde étape est une

étape de classement des services composites sélectionnés lors de l'étape précédente sur la base de leur **fonction d'utilité globale**. Nous nommons la première étape **optimisation locale**. En partant du graphe de composition, nous allons éliminer les plans qui ne répondent pas aux exigences de qualité de la requête au niveau des QdS. Ensuite, pour chaque plan de composition (service composite), nous allons considérer l'ensemble de ses services atomiques lesquels seront affectés de leur vecteur d'attributs QdS

L'heuristique ici proposée vise à ne sélectionner parmi les services composites que ceux dont tous les services atomiques vérifient toutes les m contraintes exigées par la requête. L'optimisation locale est, en fait, computationnellement peu coûteuse contrairement aux algorithmes dont la complexité est exponentielle.

Nous allons considérer la matrice suivante listant pour chaque plan de composition $Plan_i$, i allant de 1 à q où q est le nombre de plans de composition comprenant un ensemble de services atomiques fonctionnellement composables, ces services atomiques lesquels seront affectés leurs vecteurs normalisés d'attributs QdS

$$\begin{pmatrix} Plan & Serviceatomique & \text{QdS}_1 & \text{QdS}_2 & ... & \text{QdS}_m \\ \text{PC}_1 & \text{SA}_{11} & 10 & 96 & 0 & 0 \\ & \text{SA}_{21} & 10 & 97 & 0 & 0 \\ & \text{SA}_{31} & 10 & 97 & 0 & 0 \\ \text{PC}_2 & \text{SA}_{12} & 11 & 97 & 0 & 0 \\ & \text{SA}_{22} & 11 & 97 & 0 & 0 \\ \text{PC}_3 & \text{SA}_{13} & 11 & 97.6 & 0 & 0 \\ .... & & 11 & 97.6 & 0 & 0 \\ ... & & 11 & 97.6 & 0 & 0 \\ ... & & 11 & 98 & 0 & 0 \\ \text{PC}_n & \text{SA}_{1n} & 11 & 98 & 0 & 0 \\ & \text{SA}_{2n} & 15 & 97 & 0 & 0 \end{pmatrix}$$

Chaque valeur q d'un attribut QdS positif sera normalisé comme suit :

$$q_{norm} = (\text{q} - q_{min}) \ / \ (q_{max} - q_{min})$$

où q_{min} (resp. q_{max}) est la plus petite (resp. grande) valeur de l'attribut QdS concerné dans tous les services atomiques des divers plans possibles de composition.

Pour chaque attribut QdS_j les valeurs correspondant aux services atomiques sont comprises initialement dans l'intervalle $[q_{jmin}, q_{jmax}]$ avec j ∈ [1, m], une fois les valeurs nomalisées elles seront comprises dans l'intervalle [0, 1].

Exemple 16 *Pour l'ensemble des services du tableau 7.5, les différents intervalles de qualité identifiés sont :*

- *L'intervalle $[q_{1min}, q_{1max}]$ est [8, 15].*
- *L'intervalle $[q_{2min}, q_{2max}]$ est [10, 20].*
- *L'intervalle $[q_{3min}, q_{3max}]$ est [97, 99].*

Nous allons choisir l niveaux de qualité (le nombre l est fixé a priori et plus souvent en accord avec les experts du domaine). Ceci va nous donner la structure suivante : en colonnes

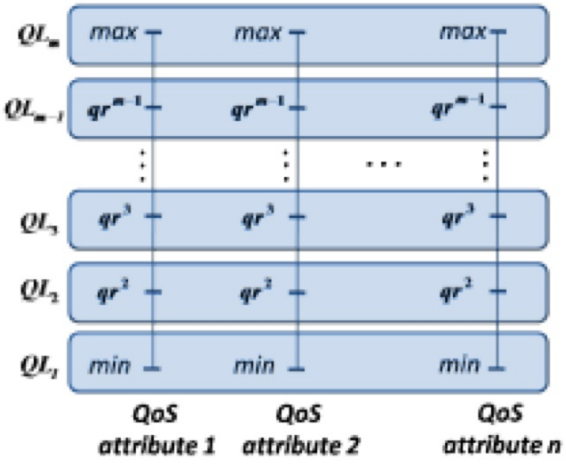


FIGURE 7.6 – Différents niveaux de qualité de service

les attributs QdS et en lignes les intervalles de qualité notés IQ. Ainsi, nous disposons de l niveaux de qualité IQ_k avec k variant de 1 à l tel que illustré par la figure 7.6.

Considérons l-2 intervalles de qualité en excluant les niveaux *min* et *max*. Pour l'exemple précédent, en choisissant $l = 5$, nous obtiendrons pour les intervalles normalisés les niveaux 0, 0,25, 0,5, 0,75 et 1. Les niveau de qualité qui seront les centroïdes des clusters seront les trois suivants 0,25, 0,5 et 0,75.

Niveaux de qualité	Temps de réponse	Débit	Disponibilité
IQ_5 = max (1)	15	20	99%
IQ_4= 0,75	13	17,5	98%
IQ_3= 0, 5	11	15	97%
IQ_2=0, 25	9	12,5	96%
IQ_1=min (0)	7	10	95%

Une fois les (l-2) valeurs calculées comme valeurs des centroïdes des clusters, nous procédons à la clustérisation à l'aide d'un classifieur sur la matrice donnant en ligne les services candidats $Scandidats$ à la composition et comme colonnes le vecteur d'attributs QdS de chaque $S_{candidat}$, ce qui va donner $(l \times 2) \times m$ clusters. Dans notre exemple, il s'agira de générer avec l'algorithme de K-Means pour chaque attribut QdS parmi les trois choisis, 3 clusters correspondant aux intervalles de qualité 0,25, 0,5 et 0,75. Cependant , il est possible d'utiliser une des bibliothèques GnuGPL WEKA (Waikato Environment for Knowledge Analysis) de l'université Waikato [11] en Australie ou TANAGRA de l'Université de Lyon 2 en France [12].

En effet, la clustérisation va servir à réduire l'espace de sélection des plans de compo-

11. Le site de WEKA est à l'adresse suivante : http ://www.cs.waikato.ac.nz/ml/weka/
12. Le site de TANAGRA est à l'adresse suivante : http ://eric.univ-lyon2.fr/ricco/tanagra/fr/tanagra.html

sition comme suit :

La clustérisation fait regrouper au sein de chaque cluster et pour chaque attribut QdS les services dont les valeurs sont proches de la valeur du centroïde. Ainsi, pour une requête ayant spécifié une contrainte par exemple temps de réponse inférieur à 8 ms, le cluster autour de la valeur du centroïde 0,25 (normalisée) ou 9 ms comme valeur non normalisée va contenir obligatoirement les services dont la valeur de l'attribut n°1 est comprise entre 7 et 9 ms. Dans ce cas c'est le cluster *min* (pour un attribut '*négatif*') qui sera choisi comme espace pouvant contenir les services vérifiant l'attribut n°1 du client.

Les services atomiques d'un cluster seront triés par valeurs décroissantes entre 0,25 et 0 (comme valeurs normalisées) ; la valeur normalisée de la contrainte usager étant égale à 0,125, seuls les services dont les valeurs normalisées de l'attribut 1 comprises entre 0,125 et 0 seront retenus comme candidats à l'admissibilité. Pour un attribut '*positif*', ce sera le cluster *max* qui sera choisi avec une sélection appropriée à l'intérieur de ce cluster des services atomiques respectant la contrainte du client. Un tri croissant à l'intériuer du sous-cluster choisi permettra de sélectionner les services atomiques candidats à l'admissibilité pour ledit attribut entre la valeur normalisée de la contrainte QdS de l'usager et la valeur maximale 1.

Lorsque le nombre de niveaux de qualité est assez élevé, il est possible que dans le cas d'attributs négatifs (resp. positifs), ce soit *hyper-cluster* qui contient les services atomiques susceptibles de vérifier la contrainte QdS de l'usager hyper-cluster constitué par le cluster *min* (resp. *max*) et un ou plusieurs clusters correspondants à des niveaux supérieurs (resp. inférieurs) de qualité. Le tri par valeurs décroissantes (resp. croissantes) à l'intérieur de l'hyper-cluster permet de sélectionner les services atomiques admissibles. Une fois, pour chaque service atomique et pour les attributs QdS, la sélection de ceux qui sont admissibles est effectuée, nous revenons à la matrice des plans possibles de composition en substituant aux valeurs des attributs QdS dans la cas de vérification de la contrainte de l'usager, la valeur 1, sinon 0.

Les éléments de la matrice seront mis à zéro initialement, comme suit :

Plan	*ServicesAtomique*	QdS_1	QdS_2	...	QdS_m
PC_1	$\text{SA}_1$1	0	0	0	0
	SA21	0	0	0	0
	SA21	0	0	0	0
PC_2	SA12	0	0	0	0
	SA22	0	0	0	0
PC_3	SA13	0	0	0	0
....		0	0	0	0
...		0	0	0	0
...		0	0	0	0
PC_n	SA1n	0	0	0	0
	SA2n	0	0	0	0

Ensuite, à partir des m sous-clusters contenant les services atomiques dont l'attribut de qualité k (variant de 1 à m) vérifie la contrainte de l'usager, l'élément de la ligne du service existant dans le sous-cluster et pour la colonne correspondant à l'attribut k sera

mis à 1.

Parmi les N plans possibles seuls les plans dont tous les éléments de la sous-matrice sont à 1 (i. e. tous les services atomiques vérifient toutes les contraintes de l'usager) seront retenus. Le cas illustré dans la matrice suivante montre que seuls les plans PC_1 et PC_n sont à retenir :

Plan	*ServicesAtomique*	QdS_1	QdS_2	...	QdS_m
PC_1	$\mathrm{SA}_1$1	1	1	1	1
	SA21	1	1	1	1
	SA21	1	1	1	1
PC_2	SA12	1	0	1	1
	SA22	1	1	1	1
PC_3	SA13	1	1	0	0
....		0	0	0	0
...		0	0	0	0
...		0	0	0	0
PC_n	SA1n	1	1	1	1
	SA2n	1	1	1	0

Pour l'obtention des ((l-2) $\times$ m) clusters en utilisant l'algorithme K-Means, le coût total est en $\mathbf{O}(m\ (l\text{- }2)\ n\ k)$ où m est le nombre d'attributs QdS de la requête qualité de l'usager, l est le nombre d'intervalles de qualité, n le nombre total de services présents dans tous les plans possibles de composition et k le nombre maximal d'itérations de l'algorithmes K-Means.

En conséquence, le coût total de cette première étape est proprement **polynomial** comprenant le coût du prétraitement, le coût de la clustérisation, le coût de l'identification du cluster (ou de l'hyper-cluster) de sélection et le coût du tri. Parmi les services composites admissibles ainsi sélectionnés, il faudra dans l'étape suivante sélectionner le ou les plus satisfaisant (s).

La dernière étape de classement se base sur la fonction globale d'utilité [13] de sorte que chaque service *admissible* sélectionné au cours de la première étape sera affecté de sa fonction d'utilité globale. Pour un service composite SC_k le vecteur QdS s'écrit comme suit : $Q_{SC_k} = [Q^1_{SC_k},...,Q^m_{SC_k}]$ tel que $\mathrm{Q}^l_{SC_k}$ est la valeur agrégée de l'attribut QdS_l parmi les m attributs de la requête de l'usager calculée avec les fonctions d'agrégation correspondant à la nature de l'attribut et aux patrons de composition intervenant dans le plan de composition de SC_k pouvant contenir n services atomiques.

Pour un plan de composition SC_k, la fonction d'utilité globale normalisée s'écrit :

$\mathcal{F}_{SC_k} = (\sum_j Q^m_{SC_k}\ )$ / m, j $\in$ [1,...,m].

Pour les attributs qualitatifs négatifs, on prendra le complément à 1 comme valeurs de ces attributs (qu'elles soient normalisées ou non), de sorte que le meilleur service composite est celui qui a la plus forte valeur comme fonction d'utilité globale. Alors, un simple tri des services composites admissibles permet d'identifier le meilleur plan. La méthode reste peu coûteuse même si le nombre de services candidats est très élevé.

13. La formule de cette fonction a été présenté dans le chapitre 4 page 89

7.4 Conclusion

L'approche de composition proposée dans ce chapitre permet de sélectionner un ensemble de services atomiques vérifiant à la fois les conditions de composabilité exacte ou approchée au niveau fonctionnel et les contraintes non fonctionnelles d'une requête étendue. Pour ce faire, nous avons proposé : (1) deux nouveaux axiomes permettant d'inférer une composabilité fonctionnelle logique (exacte ou approchée) entre services et (2) d'utiliser des techniques de similarité sémantique pour calculer la similitude entre concepts au cas où les techniques de raisonnement échouent pour déduire une composabilité fonctionnelle empirique (exacte ou approchée). Un graphe de composition est généré comprenant l'ensemble des chemins (plans) de composition constitués par les services Web atomiques composables de manière exacte ou approchée selon un ou plusieurs patrons de composition ; chaque plan de composition est étiquetée par la *valeur agrégée de ses critères QdS*. La seconde étape consiste à sélectionner parmi les plans de composition possibles(i.e., vérifiant la composabilité fonctionnelle exacte ou approchée) ceux qui sont dits admissibles (i.e. vérifiant les contraintes de qualité de la requête de l'usager) pour ensuite procéder au classement de ces derniers.

Deux heuristiques de sélection et de classement des plans de composition sont proposées dans cette thèse :

- La première consiste à appliquer la même technique utilisée dans la découverte à savoir une combinaison de l'heuristique de skyline et l'algorithme de classmeent top-k.
- La seconde consiste à sélectionner les plans de composition, parmi les plans possibles de composition, et dont les services atomiques les composant, vérifient toutes les contraintes QdS de la requête de l'usager. Cette sélection s'effectue une fois les valeurs des attributs de qualité normalisées de ces services atomiques et les services atomiques de tous les plans possibles de composition clustérisés en fonction d'un certain nombre de niveaux de qualité choisi entre [0, 1] à l'aide d'un classifieur non supervisé. Une fois les plans de composition admissibles sélectionnés, une étape de classement de ces derniers est effectuée sur la base de la valeur normalisée de la fonction d'utilité globale de chacun des plans admissibles.

Conclusion Générale et Perspectives

La thématique de cette thèse traite du partage de données, d'applications et/ou de services dans le domaine bioinformatique ; partage favorisant les recherches multi-centriques tant fondamentales que cliniques dans un domaine qui connaît un grand essor exigeant la mise en œuvre d'infrastructures favorisant le partage, la réutilisation, l'interopérabilité et le retour d'expériences. Rappelons que l'objectif visé par la notion de service Web sémantique est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines et ce, en utilisant les couches techniques de l'infrastructure existante des services Web et sans être conceptuellement indépendants. Des frameworks de services Web sémantiques ont été proposés depuis l'apparition du concept du Web sémantique en 2001 ayant pour objectifs d'expliciter les propriétés sémantiques pertinentes des services Web garantissant l'automatisation des processus de découverte, sélection, invocation et composition (i.e., tout au long du cycle de vie d'un service Web). Cependant, cet effort de sémantisation apporté par chacun des frameworks a fait émerger des descriptions hétérogènes entre elles rajoutant ainsi une nouvelle forme d'hétérogénéité.

Aujourd'hui que des portails de services Web à l'instar de BioMoby ou Seekda voient leur volume (en termes de nombre de services et de fournisseurs) augmenter à grande allure, il est devenu urgent selon nous de pallier la nouvelle hétérogénéité sémantique induite par les frameworks proposés tant au niveau des descriptions qu'au niveau des annotations qui sont elles-mêmes définies à partir d'ontologies hétérogènes. En effet, au niveau des frameworks de descriptions de services Web sémantiques, le recours aux ontologies est devenu un passage obligé tant que les ontologies favorisent le partage, la réutilisation et l'interopérabilité, mais là aussi une nouvelle hétérogénéité est apparue dès lors que le recours aux ontologies peut se traduire par un choix parmi une large éventail d'ontologies allant de l'informel (thésaurus ou méta-thésaurus comme ceux du domaine du domaine biomédical) au formel pour aboutir aux ontologies lourdes.

Les contributions de la thèse

Nous avons proposé dans cette thèse un méta-framework stratifié couvrant des ressources hétérogènes, des descriptions hétérogènes et des ontologies hétérogènes. Dans le cadre du méta-framework proposé, le travail mené a consisté à proposer une méthodologie de sémantisation de services Web créés sous différents frameworks réalisant une méta-normalisation sémantique de leurs descriptions donnant lieu à un ensemble de descriptions canoniques, une réconciliation tant sémantique qu'ontologique sur lesquelles s'adossent les services Web sémantiques à une méta-carte ontologique médiant les ontologies de référence et une ontologie fondationnelle. Cette méta-carte est constituée d'hyper-concepts reliés par le biais d'une sémantique référentielle aux concepts (intensions) et leurs instances (extensions) des ontologies originelles. C'est à travers cette sémantique référentielle que se réalise la conceptualisation partagée (méta-conceptualisation) conciliant les points de vue conceptuels des ontologies originelles. C'est aussi sur la base de cette sémantique référentielle et méta-conceptuelle que se sont effectuées les annotations sémantiques des services Web. Par ailleurs, l'engagement sémantique de la méta-carte ontologique vis-à-vis de l'ontologie fondationnelle DOLCE (choisie parmi d'autres) nous a permis de faire émerger une

sémantique inférentielle offrant la possibilité de raisonner sur les connaissances émergentes des descriptions canoniques, des annotations sémantiques et des concepts unifiés grâce à une sémantique formelle valide dans l'univers du discours garantissant ainsi la validité des inférences effectuées.

En résumé, les contributions majeures de cette thèse est un modèle d'alignement sémantique réalisant une méta-normalisation des descriptions émanant des trois frameworks OWL-S, WSMO et SAWSDL, une méta-carte ontologique reflétant les sémantiques référentielle et différentielle des concepts du domaine ciblé et une sémantique axiomatique comprenant des axiomes de relaxation permettant d'élargir la portée et la couverture des annotations sémantiques tout en exploitant les connaissances ontologiques de la méta-carte. Deux grandes classes de processus sont considérées : la découverte et la compositon (vue dans notre approche comme une chorégraphie) de services Web décrits canoniquement et annotés sémantiquement sur la base de la méta-carte ontologique. Ces deux processus sont effectués à travers un moteur inférentiel DATALOG-*like* conçu comme un méta-service Web déductif et sur la base de l'axiomatique inférentielle proposant des opérateurs de relaxation élargissant, dans le cas d'une découverte, la couverture sémantique des descriptions et l'espace de recherche d'une requête de découverte et, dans le cas d'une composition, en précisant les règles de composabilité des services atomiques. Les expérimentations menées montrent l'impact des techniques de relaxation sur la taille de l'espace de recherche des services découverts au niveau de l'approche de découverte.

Dans le cas de la découverte comme dans celui de la composition, nous avons considéré des requêtes étendues couvrant à la fois des propriétés fonctionnelles et d'autres non fonctionnelle notamment relatives à la QdS. Pour la partie non fonctionnelle, nous avons proposé une heuristique basée sur l'opérateur skyline (qui est une méthode PARETO-*like* pour la problématique de sélection des services Web réponsant aux critères non fonctionnels) suivie d'une procédure de classement des top-k meilleurs services.

L'approche de composition proposée dans ce mémoire se base aussi sur le moteur inférentiel afin de déduire des chemins possibles de services composables constitués de l'ensemble des services atomiques vérifiant les critères de composabilité exacte et approchée. Nous avons redéfini la notion de composabilité afin disposer d'une approche flexible permettant de déduire des dépendances logiques (en se basant sur les techniques de raisonnement) ou empirique (en se basant sur des mesures de similarité).

De ce fait, notre approche est originale à plus d'un titre en offrant une triple sémantique à la fois *interprétative* avec un ensemble de descriptions canoniques, *formelle* grâce aux annotations sémantiques définies sur la base de la méta-carte ontologique et *inférentielle* basée sur une sémantique axiomatique. Enfin, notre approche de découverte et de composition est originale du fait de l'inexistence de techniques de raisonnement et de classement dans la plupart des projets existants, dans la littérature.

Perspectives

Les contributions proposées dans cette thèse nous ont amené à envisager des perspectives intéressantes tant à caractère théorique que pratique. Nous présentons dans ce qui suit celles qui nous semblent les plus prometteuses.

L'ensemble des techniques de raisonnement proposées dans cette thèse exploitant la triple sémantique interprétative (descriptive), formelle (ontologique) et inférentielle induite par le modèle d'alignement et la méta-carte ontologique est totalement généralisable à n'importe quel domaine ce qui pourrait rendre le méta-framework BioMed générique. La première perspective à court terme de ce travail de recherche consiste à enrichir les techniques de raisonnement proposées avec de nouveaux opérateurs de relaxation permettant d'exploiter les primitives non logiques d'un domaine ciblé. Notre objectif serait d'enrichir la méta-carte ontologique par un ensemble de règles permettant d'exprimer des contraintes du domaine telles que les propriétés spatio-temporelles des entités décrites par la méta-carte. Cet ensemble de règles serait généré par le langage de règles SWRL [14] (Semantic Web Rule Language) qui a été proposé par le W3C pour le Web sémantique combinant les langages OWL et RuleML.

D'un autre côté, il serait utile de rechercher des techniques d'optimisation de la complexité temporelle du temps de calcul du modèle minimal de la base déductive, l'objectif final étant de considérer des référentiels contenant les descriptions de plusieurs dizaines de milliers de services Web, voire plus. En outre, des techniques telles que celle des *Magic Sets* proposée dans le contexte des bases de données déductives pourrait être suggérées. De plus, un benchmark et une comparaison de le moteur inférentiel de BioMed avec des raisonneurs tels que KAON2 [15], Pellet [16] ou Racer [17] sont envisageables, sachant que la typologie de techniques de raisonnement de notre approche est, a priori, plus riche que celles de raisonneurs existants. Au niveau empirique, il serait utile d'envisager d'autres expérimentations afin d'évaluer l'approche de composition combinant une large palette d'opérateurs de relaxation pour une composabilité fonctionnelle logique et une composabilité approximative à base de mesures de similarité.

Aussi, nous proposons d'investiguer une perspective considérée actuellement dans le domaine bioinformatique d'un très grand intérêt qui est celle de l'étude de la traçabilité des usagers de workflows scientifiques (*Scientific Workflows*). Dans ce contexte, il est possible de capitaliser l'usage des Workflows en proposant un ou plusieurs workflows scientifiques les plus indiqués (les plus satisfaisants) au contexte de l'usager, à ses besoins et aux QdS exigés.

Enfin, nous envisageons de considérer les profils et les préférences des utilisateurs au sein du processus de découverte et de composition des services à l'instar des propriétés non fonctionnelles (QdS). Les critères de QdS proposés dans un cadre assez général sont indépendants de tout domaine, il serait intéressant de proposer de nouvelles métriques dépendant du domaine afin de guider l'utilisateur dans le choix d'un service ou d'un plan de composition par rapport à d'autres, des métriques liées à la qualité de la découverte (Quality of Discovery, QoD) ou de la composition (Quality of composition, QoC) peuvent être proposées sur la base des préférences des utilisateurs et/ou de leurs *feedbacks*.

14. http ://www.w3.org/Submission/SWRL/
15. http ://kaon2.semanticweb.org/reasoning
16. http ://www.mindswap.org/2003/pellet/index.shtml
17. http ://www.racer-systems.com/

Bibliographie

[ABP06a] Nadia Yaacoubi Ayadi, Mohamed BenAhmed, and Yann Pollet. Formal framework for semantic interoperability. In *Proceedings of the First International Conference on Software and Data Technologies (ICSOFT'06)*, pages 139–144, 2006.

[ABP06b] Nadia Yaacoubi Ayadi, Mohamed BenAhmed, and Yann Pollet. Ontology-Based Meta-Model for Semantically Interoperable Systems. In *Proceedings of the 8th International Conference on Information Integration and Web-based Applications Services IIWAS'06*, pages 413–422, 2006.

[ABPR04] Daniel Austin, Abbie Barbir, Ed Peters, and Steve RossTalbot. Web Services Choreography Requirements. Technical report, W3C, 2004.

[ACD+03] Tony Andrews, Francisco Curbera, Hitesh Dholakia, Yaron Goland, Johannes Klein, Frank Leymann, Kevin Liu, Dieter Roller, Doug Smith, Satish Thatte, Ivana Trickovic, and Sanjiva Weerawarana. Business Process Execution Language for Web Services (BPEL4WS) version 1.1, May 2003.

[ACKM04] Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services : Concepts, Architecture and Applications*. Springer Verlag, 2004.

[ADH+04] Xin Dong Alon, Xin Dong, Alon Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Similarity search for web services. In *Proceedings of Very Large Data Bases (VLDB'04)*, pages 372–383, 2004.

[AFJ+05] R. Akkiraju, J. Farrell, J.Miller, M. Nagarajan, M. Schmidt, A. Sheth, and K. Verma. Web service semantics -WSDL-S, version 1.0. Technical report, April 2005.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[AL07] Nadia Yacoubi Ayadi and Zoé Lacroix. Resolving scientific service interoperability with schema mapping. In *Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering (BIBE'07)*, pages 448–455, 2007.

[ALM05] Florence Amardeilh, Philippe Laublet, and Jean-Luc Minel. Document annotation and ontology population from linguistic extractions. In *Proceedings of the 3rd international conference on Knowledge capture (K-CAP '05)*, pages 161–168, 2005.

[ALV08a] Nadia Yacoubi Ayadi, Zoé Lacroix, and Maria-Esther Vidal. BiOnMap : a deductive approach for resource discovery. In *Workshop on Resource Discovery in conjunction with International Conference on Information Integration and Web-based Applications (RED'08)*, pages 477–482, 2008.

[ALV08b] Nadia Yacoubi Ayadi, Zoé Lacroix, and Maria-Esther Vidal. A deductive approach for resource interoperability and well-defined workflows. In *OTM Workshops*, pages 998–1009, 2008.

[ALVR07] Nadia Yacoubi Ayadi, Zoé Lacroix, Maria-Esther Vidal, and Edna Ruckhaus. Deductive Web Services : An Ontology-Driven Approach for Service Interoperability in Life Science. In *OTM Workshops (2)*, pages 1338–1347, 2007.

[AMM07a] Eyhab Al-Masri and Qusay H. Mahmoud. Crawling multiple UDDI business registries. In *Proceedings of the 16th international conference on World Wide Web (WWW '07)*, pages 1255–1256, 2007.

[AMM07b] Eyhab Al-Masri and Qusay H. Mahmoud. QoS-based Discovery and Ranking of Web Services. In *Proceedings of the 16th International Conference on Computer Communications and Networks (ICCCN 2007)*, pages 529–534, 2007.

[AMSK01] M. Mostofa Akbar, Eric G. Manning, Gholamali C. Shoja, and Shahadat Khan. Heuristic Solutions for the Multiple-Choice Multi-dimension Knapsack Problem. In *Proceedings of the International Conference on Computational Science (ICCS '01)*, pages 659–668, 2001.

[AP05] Danilo Ardagna and Barbara Pernici. Global and Local QoS Constraints Guarantee in Web Service Selection. In *Proceedings of the IEEE International Conference on Web Services (ICWS '05)*, pages 805–806, 2005.

[AP07] Danilo Ardagna and Barbara Pernici. Adaptive service composition in flexible processes. *IEEE Transactions on Software Engineering*, 33(6) :369–384, 2007.

[APB06] Nadia Yaacoubi Ayadi, Yann Pollet, and Mohamed BenAhmed. A Semantic-Based Approach to Interoperability of UML Schemas. In *IAT Workshops*, pages 316–319, 2006.

[AR09] Mohammad Alrifai and Thomas Risse. Combining global optimization with local selection for efficient QoS-aware service composition. In *Proceedings of the 18th international conference on World wide web (WWW'09)*, pages 881–890, 2009.

[ARDN08] Mohammad Alrifai, Thomas Risse, Peter Dolog, and Wolfgang Nejdl. A scalable approach for QoS-based web service selection. In *Service-Oriented Computing – ICSOC 2008 Workshops*, pages 190–199, 2008.

[Bac00] Bruno Bachimont. *Engagement Sémantique et Engagement Ontologique : Conception et Réalisation D'ontologies dans Ingénierie Des Connaissances*, chapter 19, pages 305–324. 2000.

[Bal08] Fabien Baligand. Une approche déclarative pour la gestion de la qualité de service dans les compositions de services. Thèse de Doctorat. Ecole Nationale Supérieure des Mines de Paris, 2008.

[BAM08] Devis Bianchini, Valeria De Antonellis, and Michele Melchiori. Flexible semantic-based service matchmaking and discovery. *World Wide Web*, 11(2) :227–251, 2008.

[BDDM05] B. Benatallah, R.M. Dijkman, M. Dumas, and Z. Maamar. *Service composition : concepts, techniques, tools and trends*, pages 48–66. Idea Group Publishing, 2005.

[BHL01] Tim BernersLee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5) :35–43, May 2001.

[BHL+04] Mark Burstein, Jerry Hobbs, Ora Lassila, Drew Mcdermott, Sheila Mcilraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, and Katia Sycara. OWL-S : Semantic Markup for Web Services. Website, November 2004.

[BK06] Abraham Bernstein and Christoph Kiefer. Imprecise RDQL : towards generic retrieval in ontologies using similarity joins. In *Proceedings of the ACM symposium on Applied computing (SAC '06)*, pages 1684–1689, 2006.

[BKS01] Stephan Börzsönyi, Donald Kossmann, and Konrad Stocker. The Skyline Operator. In *Proceedings of the 17th International Conference on Data Engineering (ICDE'01)*, pages 421–430, 2001.

[BLM+05] Franz Baader, Carsten Lutz, Maja Milicic, Ulrike Sattler, and Frank Wolter. A description logic based approach to reasoning about web services. In *Proceedings of the Workshop on Web Service Semantics (WSS2005)*, 2005.

[BLPF06] Jos De Bruijn, Holger Lausen, Axel Polleres, and Dieter Fensel. The Web Service Modeling Language WSML : An overview. In *Proceedings of the European Semantic Web Conference (ESWC'06)*, pages 590–604, 2006.

[Bou05] Sarah Cohen Boulakia. Intégration de données biologiques : Sélection de sources centrée sur l'utilisteur. Thèse de Doctorat. Université Paris Sud, 2005.

[CAkH05] Daniela Barreiro Claro, Patrick Albers, and Jin kao Hao. J.k. : Selecting web services for optimal composition. In *Proceedings of the 2nd International Workshop on Semantic and Dynamic Web Processes (SDWP 2005)*, pages 32–45, 2005.

[CCDS04] Fabio Casati, Malu Castellanos, Umesh Dayal, and Ming-Chien Shan. Probabilistic, context-sensitive, and goal-oriented service selection. In *Proceedings of the 2nd international conference on Service oriented computing (ICSOC '04)*, pages 316–321, 2004.

[CCGM06] Valeria Cardellini, Emiliano Casalicchio, Vincenzo Grassi, and Raffaela Mirandola. A framework for optimal service selection in broker-based architectures with multiple qos classes. In *Proceedings of the IEEE Services Computing Workshops (SCW '06)*, pages 105–112, 2006.

[CDF04] Olivier Corby, Rose Diengkuntz, and Catherine Faronzucker. Querying the semantic web with the corese search engine. In *Proceedings of European Conference on Artificial Intelligence (ECAI'04)*, pages 705–709, 2004.

[CDG+06] Liliana Cabral, John Domingue, Stefania Galizia, , Alessio Gugliotta, Vlad Tanasescu, Carlos Pedrinaci, and Barry Norton. IRS-III : A broker for semantic web services based applications. In *International Semantic Web Conference (ISWC'06)*, volume 4273, pages 201–214, 2006.

[Cha07] Yasmine Charif. Chorégraphie dynamique de services basée sur la coordination d'agents introspectifs. Thèse de Doctorat. Université Pierre et Marie Curie, 2007.

[Cho09] Olfa Chourabi. Un cadre générique de modélisation, de capitalisation et de partage de connaissances métiers situées en ingénierie systèmes. Thèse de Doctorat. Conservatoire National des Arts et Métiers (France) et l'Ecole Nationale des Sciences de l'Informatique (Tunisie), Décembre 2009.

[CIJ+00] Fabio Casati, Ski Ilnicki, Li-jie Jin, Vasudev Krishnamoorthy, and Ming-Chien Shan. Adaptive and dynamic service composition in eflow. In *Proceedings of the 12th International Conference on Advanced Information Systems Engineering (CAiSE '00)*, pages 13–31, 2000.

[CLC07] Lei Cao, Minglu Li, and Jian Cao. Using genetic algorithm to implement cost-driven web service selection. *Multiagent Grid Systems*, 3(1) :9–17, 2007.

[CMML03] Monica Crubézy, Mark A. Musen, Enrico Motta, and Wenjin Lu. Configuring online problem-solving resources with the Internet Reasoning Services. *IEEE Intelligent Systems*, 18(2) :34–42, 2003.

[CMSA02] Jorge Cardoso, John Miller, Amit Sheth, and Jonathan Arnold. Modeling quality of service for workflows and web service processes. Technical report, University of Georgia, 2002.

[CMSA04] Jorge Cardoso, John Miller, Amit Sheth, and Jonathan Arnold. Quality of service for workflows and web service processes. *Journal of Web Semantics*, 1(3) :281–308, 2004.

[CPE+06] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, Francesco Perfetto, and Maria Luisa Villani. Service composition (re)binding driven by application-specific QoS. In *Proceedings of 4th International Conference Service-Oriented Computing (ICSOC 2006)*, pages 141–152, 2006.

[CS01] Fabio Casati and Ming-Chien Shan. Dynamic and adaptive composition of e-services. *Inf. Syst.*, 26(3) :143–163, 2001.

[CSG+03] Liming Chen, Nigel Shadbolt, Carole A. Goble, Feng Tao, Simon J. Cox, Colin Puleston, and Paul R. Smart. Towards a knowledge-based approach to semantic service composition. In *Proceedings of International Semantic Web Conference (ISWC'03)*, pages 319–334, 2003.

[DCG+08] John Domingue, Liliana Cabral, Stefania Galizia, , Vlad Tanasescu, Alessio Gugliotta, Barry Norton, and Carlos Pedrinaci. IRS-III : A broker-based approach to semantic web services. *Journal of Web Semantics*, 6(2) :109–132, 2008.

[DPEV+06] Massimiliano Di Penta, Raffaele Esposito, Maria Luisa Villani, Roberto Codato, Massimiliano Colombo, and Elisabetta Di Nitto. WS binder : a framework to enable dynamic binding of composite web services. In *Proceedings*

of the 2006 international workshop on Service-oriented software engineering (SOSE '06), pages 74–80, 2006.

[DY06] Wen-Li Dong and Hang YU. Optimizing web service composition based on qos negotiation. In *EDOCW '06 : Proceedings of the 10th IEEE on International Enterprise Distributed Object Computing Conference Workshops*, page 46, Washington, DC, USA, 2006. IEEE Computer Society.

[Euz05] Jérôme Euzenat. L'annotation formelle de documents en (8) questions. In *Ingénierie des connaissances*. L'Harmattan, 2005.

[FB02] Dieter Fensel and Christoph Bussler. The web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, 1(2) :113–137, 2002.

[FKS03] Ronald Fagin, Ravi Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data (SIGMOD '03)*, pages 301–312, 2003.

[FKT09] Federico Michele Facca, Srdjan Komazec, and Ioan Toma. WSMX 1.0 : A Further Step toward a Complete Semantic Execution Environment. In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web (ESWC'09)*, pages 826–830, 2009.

[FLN01] Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS'01)*, pages 102–113. ACM, 2001.

[FMH+99] Dieter Fensel, Enrico Motta, Frank Van Harmelen, V. Richard Benjamins, Monica Crubezy, Mauro Gaspari, Rix Groenboom, William Grosso, Mark Musen, Enric Plaza, Guus Schreiber, Rudi Studer, and Bob Wielinga. The Unified Problem-solving Method Development Language UPML. *Knowledge and Information Systems*, 5 :2003, 1999.

[Fro07] Annick Fron. *Architecture réparties en Java*. Dunod, 2007.

[GCC07] Chunming Gao, Meiling Cai, and Huowang Chen. QoS-aware Service Composition Based on Tree-Coded Genetic Algorithm. In *Proceedings of the 31st Annual International Computer Software and Applications Conference (COMPSAC '07)*, pages 361–367, 2007.

[GD04] S. Galizia and J. Domingue. Towards a choreography for IRS-III. In *Proceedings of the Workshop on WSMO Implementations (WIW 2004)*, pages 29–30, 2004.

[GGM99] Jean-Marc Geib, Christophe Gransart, and Philippe Merle. *CORBA : des concepts à la pratique*. Dunod, 1999.

[Gri07] Stephan Grimm. *Discovery Identifying relevant services*, pages 211–244. Springer, 2007.

[Gru93] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2) :199–220, 1993.

[Gua95] Nicola Guarino. Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human-Computer Studies*, 43(5-6) :625–640, 1995.

[Gua99] Nicola Guarino. The role of identity conditions in ontology design. In *Proceedings of the International Conference of Spatial Information Theory : Cognitive and Computational Foundations of Geographic Information Science (COSIT '99)*, volume 1661, pages 221–234, 1999.

[GV09] Marlene Goncalves and Maria-Esther Vidal. Reaching the Top of the Skyline : An Efficient Indexed Algorithm for Top-k Skyline Queries. In *Proceedings of the International Conference on Database and Expert Systems Applications (DEXA'09)*, pages 471–485, 2009.

[GVRA09] Marlene Goncalves, Maria-Esther Vidal, Alfredo Regalado, and Nadia Yacoubi Ayadi. Selecting the Top-k Skyline Services. In *Proceedings of Resource Discovery workshop in conjunction with VLDB'09*, pages 10 – 25, 2009.

[GVRA10] Marlene Goncalves, Maria-Esther Vidal, Alfredo Regalado, and Nadia Yacoubi Ayadi. Efficiently Selecting the Best Web Services. *To appear*, 2010.

[GW02] Nicola Guarino and Christopher Welty. Evaluating ontological decisions with ontoclean. *Commununication of The ACM*, 45(2) :61–65, 2002.

[GW04] Nicola Guarino and Christopher A. Welty. An Overview of OntoClean. In *Handbook on Ontologies*, pages 151–172. 2004.

[Hay04] Patrick Hayes. RDF Semantics. W3C Recommendation, February 2004.

[HCM+05] Armin Haller, Emilia Cimpian, Adrian Mocan, Eyal Oren, and Christoph Bussler. WSMX - A Semantic Service-Oriented Architecture. In *Proceedings of the International Conference on Web Service (ICWS'05)*, pages 321–328, 2005.

[HK97] Markus Horstman and Mary Kirtland. DCOM Architecture. Technical report, Microsoft Corp., 1997.

[IBS08] Ihab F. Ilyas, George Beskales, and Mohamed A. Soliman. A survey of top-k query processing techniques in relational database systems. *ACM Computing Surveys*, 40(4) :1–58, 2008.

[Isa05] Antoine Isaac. Conception et utilisation d'ontologies pour l'indexation de documents audiovisuels. Thèse de Doctorat. Université Paris IV-Sorbonne, Décembre 2005.

[JC97] Jay J. Jiang and David W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, 1997.

[JC02] Tyler Jewell and David Chappell. *Java Web Services*, chapter UDDI : Universal Description, Discovery, and Integration. O'Reilly, 2002.

[JRL+05] Michael Jaeger, Gregor Rojecgoldmann, Christoph Liebetruth, Gero Mühl, and Kurt Geihs. Ranked Matching for Service Descriptions using OWL-S. In *In Proceedings of Kommunikation in Verteilten Systemen (KiVS'2005), Informatik Aktuell*, pages 91–102, 2005.

[KAC+02] Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, and Michel Scholl. RQL : A Declarative Query Language for RDF. In *Proceedings of the 11th international conference on World Wide Web (WWW'02)*, pages 592–603, 2002.

[KB08] Christoph Kiefer and Abraham Bernstein. The creation and evaluation of iSPARQL strategies for matchmaking. In *Proceeding of European Semantic Web Conference (ESWC'08)*, pages 463–477, 2008.

[Kbo10] Férihane Kboubi. Médiation et navigation sémantiques dans un corpus textuel annoté conceptuellement et thématiquement. Thèse de Doctorat. Ecole Nationale des Sciences de L'informatique, 2010.

[KFS06] Matthias Klusch, Benedikt Fries, and Katia Sycara. Automated semantic web service discovery with OWLS-MX. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems (AAMAS '06)*, pages 915–922, 2006.

[KFS09] Matthias Klusch, Benedikt Fries, and Katia Sycara. OWLS-MX : A hybrid semantic web service matchmaker for OWL-S services. *Web Semantics*, 7(2) :121–133, 2009.

[KG05] Matthias Klusch and Andreas Gerber. Semantic web service composition planning with OWLS-XPlan. In *Proceedings of the 1st Int. AAAI Fall Symposium on Agents and the Semantic Web*, pages 55–62, 2005.

[KK08] Matthias Klusch and Patrick Kapahnke. Semantic web service selection with SAWSDL-MX. In *Proceedings of the Second International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2'08)*, 2008.

[KK09] Matthias Klusch and Frank Kaufer. WSMO-MX : A hybrid semantic web service matchmaker. *Web Intelligence and Agent Systems*, 7(1) :23–42, 2009.

[KKD+98] Shahadatullah Khan, C Flmd Shahadatullah Khan, Supervisors Dr, Kin F. Li, Dr. Eric, and G. Manning. Quality adaptation in a multisession multimedia system : Model, algorithms and architecture. Technical report, 1998.

[KKL03] Sravanthi Kalepu, Shonali Krishnaswamy, and Seng Wai Loke. Verity : A qos metric for selecting web services and providers. In *Proceedings of the International Web Information Systems Engineering Workshops (WISE Workshops)*, pages 131–139, 2003.

[KKR07] Ulrich Kuster and Birgitta Konig-Ries. Semantic service discovery with DIANE service descriptions. In *Proceedings of the International Workshop on Service Composition & SWS Challenge at the 2007 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2007)*, pages 152–156, 2007.

[KLL+05] Uwe Keller, Rubén Lara, Holger Lausen, Axel Polleres, and Dieter Fensel. Automatic Location of Web services. In *Proceedings of 2nd European Semantic Web Conference (ESWC'05)*, volume 3532, pages 1–16, 2005.

[KLW95] Michael Kifer, Georg Lausen, and James Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM (JACM)*, 42(4) :741–843, 1995.

[KMW03] Rania Khalaf, Nirmal Mukhi, and Sanjiva Weerawarana. Service-Oriented Composition in BPEL4WS. In *WWW (Alternate Paper Tracks)*, 2003.

[KPSV09] Ravi Kumar, Kunal Punera, Torsten Suel, and Sergei Vassilvitskii. op-k aggregation using intersections of ranked inputs. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM '09 :)*, pages 222–231, 2009.

[KPT+04] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. Semantic annotation, indexing, and retrieval. *Web Semantics : Science, Services and Agents on the World Wide Web*, 2(1) :49–79, December 2004.

[KVBF07] Jacek Kopecký, Tomas Vitvar, Carine Bournez, and Joel Farrell. SAWSDL : Semantic annotations for WSDL and XML schemas. *IEEE Internet Computing*, 11(6) :60–67, 2007.

[KZ07] Ziad Kobti and Wang Zhiyang. An Adaptive Approach for QoS-Aware Web Service Composition Using Cultural Algorithms. In *Australian Conference on Artificial Intelligence (ASWC'07)*, pages 140–149, 2007.

[LA10] Zoé Lacroix and Maliha Aziz. Resource Description, Ontology, and Resource Discovery. *International Journal of Metadata, Semantics and Ontologies (IJMSO) Special Issue On resource Discovery*, (To appear), 2010.

[LAB+06] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A. Lee, Jing Tao, and Yang Zhao. Scientific workflow management and the Kepler system. *Concurrency and Computation : Practice and Experience*, 18(10) :1039–1065, 2006.

[Lam02] S. Lammermann. Runtime Service Composition via Logic-Based Program Synthesis. Thèse de Doctorat. Department of Microelectronics and Information Technology, Royal Institute of Technology, Stockholm, 2002.

[Lin98] Dekang Lin. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98)*, pages 296–304, 1998.

[LRL+97] Hector J. Levesque, Raymond Reiter, Yves Lespérance, Fangzhen Lin, and Richard B. Scherl. Golog : A logic programming language for dynamic domains. *Journal of Logic Programming*, 31, 1997.

[MB10] Georgios Meditskos and Nick Bassiliades. Structural and Role-Oriented Web Service Discovery with Taxonomies in OWL-S. *IEEE Trans. on Knowl. and Data Eng.*, 22(2) :278–290, 2010.

[MBE03] Brahim Medjahed, Athman Bouguettaya, and Ahmed K. Elmagarmid. Composing Web services on the Semantic Web. *The VLDB Journal*, 12(4) :333–351, 2003.

[MBM+07] David Martin, Mark Burstein, Drew Mcdermott, Sheila Mcilraith, Massimo Paolucci, Katia Sycara, Deborah L. Mcguinness, Evren Sirin, and Naveen Srinivasan. Bringing Semantics to Web Services with OWL-S. *World Wide Web*, 10(3) :243–277, 2007.

[Mcd02] Drew Mcdermott. Estimated-Regression Planning for Interactions with Web Services. In *Proceedings of the 6th International Conference on AI Planning and Scheduling*, pages 204–211, 2002.

[MDCG03] Enrico Motta, John Domingue, Liliana Cabral, and Mauro Gaspari. IRS-II : A Framework and Infrastructure for Semantic Web Services. In *Proceeding of the International Conference on Semantic Web (ISWC'03)*, volume 2870, pages 306–318, 2003.

[MGK+98] Drew McDermott, Malik Ghallab, Craig Knoblock, Adele Howe, Aschin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL - The Planning Domain Definition Language. Technical report, Yale Center for Computational Vision and Control, 1998.

[MM10] Moreno Marzolla and Raffaela Mirandola. QoS Analysis for Web Service Applications : a Survey of Performance-oriented Approaches from an Architectural Viewpoint. Technical report, Department of Computer Science, University of Bologna, 2010.

[MPP+04] Simon Miles, Juri Papay, Terry Payne, Michael Luck, and Luc Moreau. Towards a protocol for the attachment of metadata to grid service descriptions and its use in semantic discovery. *Scientific Programming*, 12(4) :201–211, 2004.

[MS04a] E. Michael Maximilien and Munindar P. Singh. A framework and ontology for dynamic web services selection. *IEEE Internet Computing*, 8(5) :84–93, 2004.

[MS04b] E. Michael Maximilien and Munindar P. Singh. Toward autonomic web services trust and selection. In *ICSOC '04 : Proceedings of the 2nd international conference on Service oriented computing*, pages 212–221, New York, NY, USA, 2004. ACM.

[MSZ01] S. McIlraith, T.C. Son, and H. Zeng. Semantic Web Services. *IEEE Intelligent Systems. Special Issue on the Semantic Web*, 16(2) :46–53, March/April 2001.

[MvH04] Deborah L. McGuinness and Frank van Harmelen. OWL Web Ontology Language Overview. Technical report, W3C Consortium, 2004.

[MW92] Zohar Manna and Richard J. Waldinger. Fundamentals of deductive program synthesis. *IEEE Trans. Software Eng.*, 18(8) :674–704, 1992.

[MYCC07] Nikos Mamoulis, Man Lung Yiu, Kit Hung Cheng, and David W. Cheung. Efficient top-k aggregation of ranked inputs. *ACM Transactions on Database Systems*, 32(3) :19, 2007.

[NIK+03] Dana Nau, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman. SHOP2 : An HTN planning system. *Journal of Artificial Intelligence Research*, 20 :379–404, 2003.

[NM02] Srini Narayanan and Sheila A. McIlraith. Simulation, verification and automated composition of web services. In *Proceedings of the 11th international conference on World Wide Web (WWW '02)*, pages 77–88, 2002.

[OCF+03] Borys Omelayenko, Monica Crubézy, Dieter Fensel, V. Richard Benjamins, Bob J. Wielinga, Enrico Motta, Mark A. Musen, and Ying Ding. UPML :

The Language and Tool Support for Making the Semantic Web Alive. In *Spinning the Semantic Web*. 2003.

[OGA+06] Tom Oinn, Mark Greenwood, Matthew Addis, M. Nedim Alpdemir, Justin Ferris, Kevin Glover, Carole Goble, Antoon Goderis, Duncan Hull, Darren Marvin, Peter Li, Phillip Lord, Matthew R. Pocock, Martin Senger, Robert Stevens, Anil Wipat, and Chris Wroe. Taverna : lessons in creating a workflow environment for the life sciences : Research Articles. *Concurr. Comput. : Pract. Exper.*, 18(10) :1067–1100, 2006.

[OS99] Aris M. Ouksel and Amit P. Sheth. Semantic interoperability in global information systems : A brief introduction to the research area and the special section. *SIGMOD Record*, 28(1) :5–12, 1999.

[Pap03] Mike P. Papazoglou. Service -oriented computing : Concepts, characteristics and directions. In *Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE '03)*, pages 3–12, 2003.

[Pee04] Joachim Peer. A PDDL based tool for automatic web service composition. In *Proceedings of the Second International Workshop on Principles and Practice of Semantic Web Reasoning (PPSWR)*, pages 149–163, 2004.

[Pee05] Joachim Peer. Web Service Composition as AI Planning - A Survey. Technical report, University of St. Gallen, Switzerland, 2005.

[PF02] Shankar R. Ponnekanti and Armando Fox. SWORD : A developer toolkit for web service composition. In *Proceedings of the 11th International WWW Conference (WWW2002)*, pages 83–107, 2002.

[PKCH05] Jyotishman Pathak, Neeraj Koul, Doina Caragea, and Vasant G. Honavar. A framework for semantic web services discovery. In *Proceedings of the 7th annual ACM international workshop on Web information and data management (WIDM '05)*, pages 45–50, 2005.

[PKP+02a] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, , and Katia Sycara. Importing the semantic web in UDDI. In *Proceedings of International Workshop on Web Services, E-Business, and the Semantic Web*, pages 225–236, 2002.

[PKP+02b] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, , and Katia Sycara. Semantic matching of web services capabilities. In *Proceedings of International Semantic Web Conference (ISWC'02)*, pages 333–347, 2002.

[POSV04] Abhijit A. Patil, Swapna A. Oundhakar, Amit P. Sheth, and Kunal Verma. Meteor-s web service annotation framework. In *Proceedings of the 13th international conference on World Wide Web (WWW '04)*, pages 553–562, New York, NY, USA, 2004. ACM.

[Pre04] Chris Preist. A conceptual Architetcure for Semantic Web Services. In *Proceedings of the International Semantic Web Conference (ISWC'04)*, pages 395–409, 2004.

[PS08] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. W3C Recommendation, January 2008.

[PSL03] Chintan Patel, Kaustubh Supekar, and Yugyung Lee. A QoS Oriented Framework for Adaptive Management of Web Service based Workflows. In *Proceedings of the Conference on Database and Expert Systems (DEXA'03)*, pages 826–835. Springer, 2003.

[Ran03] Shuping Ran. A model for web services discovery with QoS. *SIGecom Exch.*, 4(1) :1–10, 2003.

[Rao04] Jinghai Rao. Semantic web service composition via logic-based program synthesis. Thèse de Doctorat. Department of Computer and Information Science, Norwegian University of Science and Technology, December 2004.

[Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI'95)*, pages 448–453, 1995.

[Res99] Philip Resnik. Semantic similarity in a taxonomy : An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research (JAIR)*, 11 :95–130, 1999.

[RKL+05a] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Polleres, Cristina Feier, Cristoph Bussler, and Dieter Fensel. Web Service Modeling Ontology. *Applied Ontology*, 1(1) :77–106, 2005.

[RKL+05b] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Cristoph Bussler, and Dieter Fensel. Web Service Modeling Ontology. *Applied Ontology*, 1(1), 2005.

[RMB+89] Roy Rada, Hafedh Mili, Ellen Bicknell, , and Maria Blettner. Development and application of a metric on semantic nets. *IEEE Transactions on Systems, Man and Cybernetics*, 19(1) :17–30, January/February 1989.

[RPD06] Florian Rosenberg, Christian Platzer, and Schahram Dustdar. Bootstrapping performance and dependability attributes ofweb services. In *Proceedings of the IEEE International Conference on Web Services (ICWS '06)*, pages 205–212, Washington, DC, USA, 2006. IEEE Computer Society.

[RRV08] Edna Ruckhaus, Eduardo Ruiz, and Maria-esther Vidal. Query evaluation and optimization in the semantic web. *Theory Pract. Log. Program.*, 8(3) :393–409, 2008.

[RS03] Jinghai Rao and Xiaomeng Su. Toward the composition of semantic web services. In *Proceedings of the Second International Workshop on Grid and Cooperative Computing (GCC'03)*, pages 760–767, 2003.

[RS04] Jinghai Rao and Xiaomeng Su. A survey of automated web service composition methods. In *Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC'04)*, pages 43–54, 2004.

[RU93] Raghu Ramakrishnan and Jeffrey D. Ullman. A survey of research on deductive database systems. *Journal of Logic Programming*, 23 :125–149, 1993.

[Sal88] G. Salton. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5) :513–523, 1988.

[SBF98] Rudi Studer, V. Richard Benjamins, and Dieter Fensel. Knowledge engineering : Principles and methods. *Data and Knowledge Engineering*, 25(1-2) :161–197, March 1998.

[Sch06] Hans Albrecht Schmid. Service congestion : The problem, and an optimized service composition architecture as a solution. In *Proceedings of the IEEE International Conference on Web Services (ICWS '06)*, pages 505–514, Washington, DC, USA, 2006. IEEE Computer Society.

[SD02] Michael Sintek and Stefan Decker. TRIPLE - A Query, Inference, and Transformation Language for the Semantic Web. In *Proceedings of the First International Semantic Web Conference on The Semantic Web (ISWC'02)*, pages 364–378, 2002.

[SDC06] Amandeep S. Sidhu, Tharam S. Dillon, and Elizabeth Chang. Integration of protein data sources through PO. In *Proceedings of Database and Expert Systems Applications (DEXA 2006)*, pages 519–527, 2006.

[SdF03] Mithun Sheshagiri, Marie desJardins, and Timothy Finin. A Planner for Composing Services Described in DAML-S. In *Proceedings of the AAMAS Workshop on Web Services and Agent-based Engineering*, 2003.

[SDHS05] M. Adel Serhani, Rachida Dssouli, Abdelhakim Hafid, and Houari Sahraoui. A qos broker based architecture for efficient web services selection. In *Proceedings of the IEEE International Conference on Web Services (ICWS'05)*, pages 113–120, 2005.

[SGR08] Amit P. Sheth, Karthik Gomadam, and Ajith Ranabahu. Semantics enhanced services : METEOR-S, SAWSDL and SA-REST. *IEEE Data Engineering Bulletins*, 31(3) :8–12, 2008.

[SHS08] Michael Schumacher, Heikki Helin, and Heiko Schuldt. *CASCOM- Intelligent Service Corrdination in the Semantic Web*, chapter Service Discovery (Chapter 10), pages 209–238. 2008.

[SK05] Larisa N Soldatova and Ross D King. Are current ontologies in biology good ontologies. *Nature Biotechnology*, 23(9) :1095–1098, 2005.

[SP04] Evren Sirin and Bijan Parsia. Planning for Semantic Web Services. In *Proceedings of Semantic Web Services Workshop at 3rd International Semantic Web Conference*, 2004.

[SPS04] Naveen Srinivasan, Massimo Paolucci, and Katia Sycara. An efficient algorithm for OWL-S based semantic search in UDDI. In *First International Workshop Semantic Web Services and Web Process Composition*, pages 96–110, 2004.

[SSGF06] Gareth Shercliff, Jianhua Shao, W. Alex Gray, and Nick J. Fiddian. Qos assessment of providers with complex behaviours : An expectation-based approach with confidence. In *Proceddings of Service-Oriented Computing (ICSOC 2006)*, pages 378–389, 2006.

[SVSM03] Kaarthik Sivashanmujgam, Kunal Verma, Amit Sheth, and John Miller. Adding Semantics to Web Services Standards. In *Proceedings of the International Conference on Web Services (ICWS'03)*, pages 395–401, 2003.

[SWK+02] Katia Sycara, Seth Widoff, Matthias Klusch, , and Jianguo Lu. Larks : Dynamic matchmaking among heterogeneous software agents in cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5(2) :173–203, 2002.

[Tem08] Lynda Temal. Ontologie de partage de données et d'outils de traitement dans le domaine de la neuroimagerie. Thèse de Doctorat. Université de Rennes I, Juin 2008.

[TP04] P. Traverso and M. Pistore. Automated composition of semantic web services into executable processes. In *Proceedings of the 3rd International Semantic Web Conference (ISWC'04)*, pages 380–394, 2004.

[udd01] Universal description, discovery, and integration of business for the web. Technical report, UDDI Specification Technical Commitee, October 2001.

[VC05] Emanuele Della Valle and Dario Cerizza. Cocoon glue : a prototype of wsmo discovery engine for the healthcare field. In *In Proceedings of the WIW 2005 Workshop on WSMO Implementations*, pages 1–12, 2005.

[VHA05] Le-Hung Vu, Manfred Hauswirth, and Karl Aberer. Qos-based service selection and ranking with trust and reputation management. In *OTM Conferences (1)*, pages 466–483, 2005.

[VRB08] Yves Vanrompay, Peter Rigole, and Yolande Berbers. Genetic algorithm-based optimization of service composition and deployment. In *SIPE '08 : Proceedings of the 3rd international workshop on Services integration in pervasive environments*, pages 13–18, 2008.

[VSS+05] Kunal Verma, Kaarthik Sivashanmugam, Amit Sheth, Abhijit Patil, Swapna Oundhakar, and John Miller. METEOR-S WSDI : A scalable P2P infrastructure of registries for semantic publication and discovery of web services. *Journal of Information Technology and Management, Special Issue on Universal Global Integration*, 6(1) :17–39, 2005.

[W3C03] W3C. QoS for web services : Requirements and possible approaches. Technical report, W3C Working Group Note, 2003.

[Wal01] Richard J. Waldinger. Web agents cooperating deductively. In *Proceedings of the First International Workshop on Formal Approaches to Agent-Based Systems (FAABS '00)*, pages 250–262, 2001.

[WfM05] WfMC. Workflow Reference Model, Workflow Management Coalition. [Online] : http ://www.wfmc.org, January 2005.

[Wie92] Gio Wiederhold. Mediators in the Architecture of Future Information Systems. *IEEE Computer*, 25(3) :38–49, 1992.

[WL02] Mark D. Wilkinson and Matthew Links. BioMOBY : An Open Source Biological Web Services Proposal. *Briefings in Bioinformatics*, 3(4) :331–341, 2002.

[WP94] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, 1994.

[WPS+03] Dan Wu, Bijan Parsia, Evren Sirin, James A. Hendler, and Dana S. Nau. Automating DAML-S Web Services Composition Using SHOP2. In *Proceedings of International Semantic Web Conference (ISWC'03)*, pages 195–210, 2003.

[WSH+03] Dan Wu, Evren Sirin, James Hendler, Dana Nau, and Bijan Parsia. Automatic Web Services Composition Using SHOP2. In *Proceedings of the Workshop on Planning For Web services*, 2003.

[XB05] Jin Xiao and Raouf Boutaba. QoS-aware service composition and adaptation in autonomic communication. *IEEE Journal on Selected Areas in Communications*, 23(12) :2344–2360, 2005.

[YL04] Tao Yu and Kwei-Jay Lin. The Design of QoS Broker Algorithms for QoS-Capable Web Services. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, pages 17–24, Washington, DC, USA, 2004. IEEE Computer Society.

[YL05a] Tao Yu and Kwei-Jay Lin. A Broker-Based Framework for QoS-Aware Web Service Composition. In *Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'05)*, pages 22–29, 2005.

[YL05b] Tao Yu and KweiJay Lin. Service selection algorithms for composing complex services with multiple qos constraints. In *Proceedings of the 3rd Int. Conf. on Service Oriented Computing (ICSOC'05)*, pages 130–143, 2005.

[YST+09] Jing Yuan, Guang-Zhong Sun, Ye Tian, Guoliang Chen, and Zhi Liu. Selective-NRA Algorithms for Top-k Queries. In *Proceedings of the Joint International Conferences on Advances in Data and Web Management*, pages 15–26, 2009.

[Yu06] Tao Yu. Quality of service (Qos) in web services : model, architecture and algorithms. Thèse de Doctorat. California State University, 2006.

[YZL07] Tao Yu, Yue Zhang, and Kwei-Jay Lin. Efficient algorithms for Web services selection with end-to-end QoS constraints. *ACM Transactions on the Web*, 1(1) :6, 2007.

[ZBD+03] Liangzhao Zeng, Boualem Benatallah, Marlon Dumas, Jayant Kalagnanam, and Quan Z. Sheng. Quality driven web services composition. In *Proceedings of the 12th international conference on World Wide Web (WWW '03)*, pages 411–421, 2003.

[ZBN+04] Liangzhao Zeng, Boualem Benatallah, Anne H.H. Ngu, Marlon Dumas, Jayant Kalagnanam, and Henry Chang. QoS-Aware Middleware for Web Services Composition. *IEEE Transactions on Software Engineering*, 30 :311–327, 2004.

[ZSC06] Chengwen Zhang, Sen Su, and Junliang Chen. A Novel Genetic Algorithm for QoS-Aware Web Services Selection. In *Proceedings of Data Engineering Issues in E-Commerce and Services (DEECS'02)*, volume 4055, pages 224–235, 2006.

[ZZL+02] Jiwei Zhong, Haiping Zhu, Jianming Li, , and Yong Yu. Conceptual Graph Matching for Semantic Search. In *Proceedings of the 10th International Conference on Conceptual Structures (ICCS '02)*, pages 92–196, 2002.