République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de
la **R**echerche **S**cientifique
Université Ibn Khaldoun – Tiaret –

Faculté des Sciences et de la Technologies et Sciences de la Matière

Département d'Informatique

MEMOIRE EN VUE DE L'OBTENTION DU DIPLÔME DE MAGISTER

SPECIALITE : Sciences et Technologies de l'Information et de la Communication

OPTION : Systèmes d'Information et de Connaissances (SIC)

Présenté par :

KHATEMI Fouad

SUJET DU MEMOIRE

Approche P2P hybride pour la composition automatique des services Web sémantiques

SOUTENU LE 03/11/2012 Devant Le Jury Composé de :

Mr. Amar BALLA	Professeur	E.S.I. Alger	Président
Mr. Youcef AKLOUF	M.C.A	U.S.T.H.B. Alger	Rapporteur
Mr. Youcef DAHMANI	M.C.A	U.I.K. Tiaret	Examinateur
Mr. Djilali BEHLOUL	M.C.A	U.S.T.H.B. Alger	Examinateur
Mr. Samir KECHID	M.C.A	U.S.T.H.B. Alger	Examinateur

Année Universitaire 2011/2012

Remerciements

« Après avoir remercié ALLAH le tout puissant »

En premier lieu, je tiens à exprimer ma profonde reconnaissance à Monsieur ***Youcef AKLOUF****, mon Directeur de Mémoire pour tous ses conseils, ses critiques, ses encouragements dont j'ai bénéficié tout au long de ce travail, et pour m'avoir témoigné sa confiance totale.*

Je tiens à remercier très particulièrement le Professeur ***Amar BALLA*** *de m'avoir fait l'honneur de présider le jury. Je remercie aussi les membres du jury qui ont accepté de juger mon travail.*

J'adresse mes sincères remerciements à mon ami Mr. Hadj Madani MEGHAZI pour son aide précieuse, ses conseils et surtout sa modestie.

Mes remerciements et ma gratitude s'adressent aussi à l'ensemble de mes enseignants de l'Université Ibn Khaldoun de Tiaret et de l'ESI, pour l'effort qu'ils ont déployé afin d'assurer ma formation et pour leurs compétences. Que Fatiha et Djamel du Département d'Informatique soient aussi remerciés pour leurs soutiens.

Merci à tous mes amis, Boubakeur, Khaled, Dilmi, Smail, Djilali, Ali, Adel, Mourad, Farida, Nassima, Lynda, Imane, Nour, Hayet, Sabrina, Fatiha, Mokhtaria, Bisra sans oublier tous mes collègues de l'INSFEP de Tiaret.

Mes derniers remerciements et non les moindres, iront à mon promoteur et à mes proches, et en particulier aux membres de ma famille, qui m'ont toujours apporté leur soutien sans faille. Je les remercie de toute l'affection et tout l'amour qu'ils m'ont témoignés.

Un Grand merci à ceux qui ont une empreinte dans ce travail.

Dédicaces

« Louange à ALLAH, le tout puissant ».

A mes chers parents,
Que Dieu les protège.

A mes sœurs et mon frère
Que Dieu les bénisse.

A mes grands parents
A ma tante et mes oncles.

A mon promoteur Dr. Youcef AKLOUF
Que Dieu exauce ses vœux les plus chers.

Et à tous ceux qui me sont chers.

A TOUS, je dédie ce travail en leur adressant tous mes sentiments d'affection et de considération.

ملخص: بالرغم من التطور في خدمات الواب داخل المؤسسات واستعمالهم بقدر كبير، إلا انه لا يمكنهم دوماً حل كل المشاكل، تركيب خدمات الواب يسمح بالإجابة على المتطلبات المتزايدة التعقيد للمستعملين، بالتوفيق بين عدة خدمات الواب. لكن الاكتشاف شكل مشكلا أساسيا للتركيب الآلي لخدمات الواب. هذه الأخيرة يجب أن تتفاعل تلقائيا بدون تدخل بشري قدر الإمكان. لتحسين الاكتشاف الآلي لخدمات الواب تُستعمل تكنولوجيات مختلفة لإعطاء حلول جديدة. في هذا الخضم، الواب المعنوي و أنظمة الند للند تعتبر التكنولوجيات الأكثر ملائمة للحلول المعتمدة على خدمات الواب. في عملنا هذا سنقدم حلا بالند للند يعتمد على هندسة هجينة من أجل الاكتشاف والتركيب الآلي لخدمات الواب المعنوية. الفكرة الأساسية لحلنا تكمن في تقسيم الأنداد إلى قسمين: *أنداد محلية* و *أنداد جماعات*، تنظيم الأنداد بهذه الكيفية يضمن استقرارا أحسنا و استمرارا في الأداء الجيد لكل الشبكة. في هذا الحل نستعمل أيضا جدولا موزعا من أجل حفظ طرق تركيبات الخدمات السابقة التي تم إتمامها.

كلمات مفتاح: خدمات الواب المعنوية، اكتشاف و تركيب آلي لخدمات الواب، WSMX، الند للند، JXTA.

Résumé : Malgré l'évolution des services Web au sein des organismes et leur utilisation à une grande échelle, ils ne sont pas toujours capables de résoudre tous les problèmes, la composition de services Web permet de répondre aux besoins de plus en plus complexes des utilisateurs, par la combinaison de plusieurs services Web. Cependant, la découverte a présenté un problème important pour la composition automatique des services Web. Ces derniers doivent interagir dynamiquement avec le minimum d'intervention humaine. Pour améliorer la découverte automatique des services Web, différentes technologies sont utilisées pour fournir de nouvelles solutions. Dans ce contexte, le Web sémantique et les systèmes P2P sont les technologies les plus adaptées aux approches orientées services. Dans notre travail, nous présentons une approche P2P scalable basée sur une architecture hybride pour la découverte et la composition automatique des services Web sémantiques. L'idée principale de notre solution est de diviser les pairs en deux segments : *pairs locaux* et *pairs groupes*, une telle organisation des pairs garantie une meilleure stabilité et scalabilité de tout le réseau. Dans cette approche nous utilisons aussi une table distribuée pour préserver les traces des compositions déjà réalisées.

Mots clés : Services Web sémantiques, découverte et composition automatique de services Web, WSMX, P2P, JXTA.

Abstract : Despite the evolution of Web services within organizations and their use in a large scale, they are not always able to solve all problems, the composition of Web services can meet the increasingly complex needs of users, by the combination of several Web services. However, the discovery presented a major problem for the automatic composition of Web services. They must interact dynamically with minimal human intervention. To improve the automated discovery of Web services, different technologies are used to provide new solutions. In this context, the Semantic Web and P2P are the most adapted technologies to the service oriented approaches. In our work, we present a scalable P2P approach based on hybrid architecture for the discovery and automatic composition of semantic Web services. The main idea of our solution is to divide the peers into two segments: *local peers* and *group peer*, such an organization of peers guarantee a better stability and scalability of the whole network. In this approach we also use a distributed table to preserve the traces of compositions already completed.

Keywords : Semantic Web services, Web services discovery and automatic composition, WSMX, P2P, JXTA.

Table des matières

Introduction générale 8
1. Contexte 10
2. Problématique 11
3. Contribution 12
4. Organisation du mémoire 13
Chapitre 1 DEFINITION DES SERVICES WEB 15
1. Introduction 16
2. Définition des services Web 16
3. Caractéristiques des services Web 17
4. Architecture des services Web 17
5. Conclusion 19
Chapitre 2 LES SERVICES WEB SEMANTIQUES 20
1. Introduction 21
2. Les ontologies 21
2.1. Définition 21
2.2. Les composantes d'une ontologie 22
2.3. Langage d'ontologie 22
3. Le web sémantique 23
3.1. Définition 23
3.2. Les composantes du web sémantique 24
4. Les services web sémantiques 25
5. Quelques approches proposées pour la réalisation des services web sémantiques 26
5.1. Web Ontology Language for Web Services (OWL-S) 27
5.2. Web Service Modeling Ontology (WSMO) 28
5.3. METEOR for Semantic Web Services (METEOR-S) 29
6. Conclusion 32
Chapitre 3 LA COMPOSITION DE SERVICES WEB 33
1. Introduction 34
2. Définition 34

3. Types de composition de services Web 35
4. Problématique de la découverte des services Web 37
4.1. Méthodes centralisées de découverte 37
4.2. Découverte distribuée des Web services 37
4.2.1. Utilisation des SMA (Systèmes Multi Agents) 38
4.2.2. Utilisation des réseaux P2P 38
5. Conclusion 39
Chapitre 4 LES SYSTEMES PAIR-A-PAIR 41
1. Présentation : 42
2. Définition 43
3. Architecture des systèmes P2P 44
3.1. Architecture centralisée 44
3.2. Architecture décentralisée 45
3.3. Architecture hybride 46
4. Plateformes de développement d'application P2P 47
5. Conclusion 48
Chapitre 5 APPROCHE P2P HYBRIDE POUR UNE COMPOSITION AUTOMATIQUE DES SWS 51
1. Introduction 52
2. Approche P2P hybride pour la découverte et la composition automatique des SWS 53
2.1. Motivations de l'utilisation des réseaux P2P 53
2.2. Description générale de l'approche 53
2.2.1. Pairs Locaux (PL) : 54
2.2.2. Pairs Groupes (PG) : 54
2.2.3. Table de composition 55
3. Architecture de la solution proposée 56
3.1. Module de composition locale 56
3.2. Module de composition P2P 56
4. Scénario de l'exécution de l'approche proposée pour une composition automatique des SWS dans une plateforme P2P hybride 58
5. Discussion 63
5.1. Avantages de l'approche proposée 63
5.2. Inconvénients 64

6. Conclusion 64
Chapitre 6 IMPLEMENTATION ET MISE EN ŒUVRE 66
1. Introduction 67
2. Présentation des outils technologiques utilisés 67
3. Implémentation de l'architecture P2P proposée 68
3.1. Création des Pairs Groupes 68
3.2. Création des Pairs Locaux 68
4. WSMXEntryPoints 68
5. Préparation d'un nœud WSMX 70
6. Transfert de données 71
7. Conclusion 71
Conclusion et perspectives 72
1. Conclusion 73
2. Perspectives 74
Bibliographie 75
Annexes 85

Liste des figures

Figure 1. Architecture des web services……..……18
Figure 2. L'architecture en couches du web sémantique…….……25
Figure 3. La convergence du Web sémantique et les services Web……...…….26
Figure 4. Ontologie OWS-S…….……..……..27
Figure 5. Les éléments de WSMO……28
Figure 6. Architecture de METEOR-S……...……31
Figure 7. Orchestration et chorégraphie de services Web……...……..35
Figure 8. Architecture P2P centralisée…….…….45
Figure 9. Architecture P2P décentralisée (réseau GNUtella)……46
Figure 10. Architecture P2P Hybride……47
Figure 11. Architecture de l'approche proposée……57
Figure 12. Exemple d'un service Web composite calculant le prix d'un livre……..…61
Figure 13. Exemple d'une composition de services Web dans un réseau P2P hybride…...…62
Figure 14. Description graphique du ficher WSDL de WSMXEntryPoints……..69
Figure 15. Pipe bidirectionnel JXTA……..71
Figure 16. Etapes d'invocation d'objets distants avec SOAP…….86
Figure 17. Structure d'un message SOAP…….…..87
Figure 18. Mécanismes d'accès aux services de UDDI……..89
Figure 19. Modèle de données de l'annuaire UDDI……..90
Figure 20. Relation entre UDDI et WDSL……..……94
Figure 21. Exemple d'un document XML……..……96
Figure 22. Exemple de graphe RDF……..…….97
Figure 23. Exemple de graphe RDFS…….……98
Figure 24. La vision générique (overlay) de la plate-forme JXTA…….99
Figure 25. Architecture logique du framework JXTA……...101
Figure 26. Architecture de WSMX……104

Introduction générale

1. Contexte

Avec une évolution exponentielle dans le temps, les SI (Systèmes d'Informations) sont devenus plus complexes et plus hétérogènes en raison de la diversité des besoins, des exigences des clients, et de la grande masse d'information stockées et manipulées par les internautes.

Assurer l'interopérabilité est l'une des clés de succès de développement et d'intégration des grands systèmes d'information. Cet objectif présente le noyau des travaux de recherches dédiés à l'amélioration des architectures, des plateformes et des technologies de développement des systèmes d'information depuis les méthodes cartésiennes jusqu'aux architectures orientées services, plus connue avec l'abréviation SOA (pour Service Oriented Architecture). Ces dernières présentent actuellement la vision architecturelle des SI modernes.

Dans ce contexte, les services Web proposent une architecture par composants qui permet à une application de faire l'usage d'une fonctionnalité située dans une autre application. Cependant, cette solution repose sur l'ubiquité de l'infrastructure d'Internet alors que les autres architectures reposent chacune sur sa propre infrastructure [MEL, 04]. L'interopérabilité est donc une caractéristique intrinsèque aux services Web parce qu'ils sont basés sur des technologies Web dérivées du fameux standard XML.

Cependant, les services Web possèdent aussi des inconvénients. En effet, il se peut qu'un client ait des besoins spécifiques qui ne sont rendus par aucun service Web. A cause de l'élargissement d'internet, il est rare pour un consommateur de découvrir un Web service qui réponde à ses besoins. Pour cette raison, plusieurs services peuvent interagir et échanger dynamiquement des informations. L'objectif de cette interaction est l'accomplissement d'un but complexe non concevable par un seul service Web. Cette agrégation des compétences pour réaliser un but commun est connue par **« Composition des services Web »**.

Pour composer des services Web, ces derniers devraient être capables de découvrir d'autres services qui ont des capacités bien définies et qui réalisent des tâches précises. Pour garantir cela, il faut que l'infrastructure du système de recherche des Web services fournisse une description détaillée des fonctionnalités offertes par les services disponibles. Cette description doit être compréhensible par la machine pour faciliter la recherche dynamique et pour trouver les services adéquats. Ces défis, souvent rencontrés dans les travaux de recherche des Web services, sont fortement liés à l'opération de **« découverte des services Web »**.

Le processus de découverte classique est réalisé en utilisant des systèmes d'annuaires. Dans la majorité des cas ce sont des serveurs UDDI (Universal Discovery, Discription and Integration), où il est (le processus de découverte) basé essentiellement sur une recherche syntaxique des descriptions WSDL (Web Service Discription Language) des services web. Cette méthode centralisée est malheureusement mal adaptée aux interactions dynamiques dans un environnement évolutif (scalable) et flexible [MAN, 07], [HU, 05].

Cependant, comme nous l'avons déjà précisé, les services Web sont conçus pour être composés afin d'accomplir des buts complexes non réalisables par un seul service Web. Pour aboutir à ces objectifs, les services Web doivent interagir dynamiquement avec le minimum d'intervention humaine. Ceci nécessite la prestation d'une description plus intelligible et plus compréhensible par la machine. C'est dans ce contexte que **« le Web sémantique »** peut intervenir pour régler le problème de découverte basé sur la description technique des services Web (réalisée en WSDL). En effet, une description sémantique est plus riche et plus compréhensible par la machine, ce qui facilite la découverte dynamique des services Web. La combinaison entre les services Web et les technologies du Web sémantique ont donné lieu à la naissance des **«services Web sémantiques »**.

Avec les avantages apportés par les technologies du Web sémantique à l'opération de découverte des services Web, UDDI ne saisit pas les relations entre les entités dans son répertoire et n'est donc pas capable de faire usage de l'information sémantique pour déduire les relations en cours de la recherche [BAT, 10]. De plus, UDDI supporte uniquement la recherche basée sur des informations de haut niveau spécifiques aux entreprises et aux services.

Afin de régler ce problème, plusieurs solutions ont été proposées pour procéder à la découverte distribuée des Web services. La plupart des travaux de recherche récents présentent des méthodes de découverte basées sur les systèmes **P2P (Peer-to-Peer)** [MAN, 07]. Le paradigme P2P est considéré comme une évolution pour les systèmes distribués qui se concentrent sur la gestion des réseaux et le partage des ressources avec une meilleure fiabilité et scalabilité.

Les systèmes P2P fournissent une alternative évolutive aux systèmes centralisés en distribuant les Web services sur tous les pairs du réseau. Les approches fondées sur les systèmes P2P offrent un contexte décentralisé et auto-organisé, où l'interaction entre les Web services se fait dynamiquement.

2. Problématique

L'utilisation et la combinaison des technologies pour résoudre des problèmes dans un contexte bien défini n'est pas un travail facile ou toujours réalisable. Récemment, plusieurs solutions ont été proposées pour procéder à la découverte distribuée des services Web. Les solutions proposées sont cependant fortement liées au type de réseau P2P utilisé : centralisé, distribué et hybride.

De ce fait, pour employer les systèmes P2P dans le domaine de la découverte et composition des services Web sémantiques, il faut répondre à plusieurs questions:

Selon les avantages et les inconvénients de chaque type des systèmes P2P, quel est la structure de réseau P2P la plus adaptée à la découverte dynamique des services Web ?

Une bonne solution doit exploiter les avantages des différentes structures tout en préservant l'efficacité et la scalabilité des réseaux P2P lorsque leur taille devient plus importante.

Plusieurs travaux de recherches proposent la découverte et la composition des services Web dans les réseaux P2P structurés comme Chord [STO, 01], Pastry [ROW, 01] et CAN [RAT, 01] qui ont apporté plusieurs solutions aux problèmes rencontrés dans les premières générations des réseaux P2P (centralisés et non structurés).

Comment découvrir des services Web décris sémantiquement pour répondre à une requête précise tout en assurant une composition dynamique ?

Autrement dit, comment implémenter une méthode de découverte distribuée ? et comment peut-on exploiter les technologies du Web sémantique pour minimiser le plus possible l'intervention humaine ? Dans ce contexte, il faut que les requêtes échangées, entre les différents nœuds du réseau, soient compréhensibles par tous les participants.

Comment minimiser le temps de réponse et le coût de composition ?

Suivant le type de réseau, sa taille et les protocoles utilisés, le temps de réponse peut différer d'une solution à une autre. Cependant, l'objectif principal de toute méthode de découverte des services Web est de minimiser le temps de recherche et de réponse. D'autre part, dans le contexte des systèmes P2P, le passage à l'échelle est un facteur important qu'il faut prendre en considération parce qu'il influe directement sur le coût de l'opération de recherche et par voie de conséquence sur celui de la composition.

3. Contribution

Dans notre travail, nous nous intéressons aux systèmes décentralisés, spécialement aux réseaux P2P. Ces derniers présentent plusieurs avantages et caractéristiques qui les positionnent comme étant le type de réseau le plus approprié à la découverte des Web services sémantiques.

Généralement, dans ce contexte, le demandeur (le service consommateur) recherche une capacité, un but ou une propriété d'une ressource (service Web). Ainsi, quand il envoie une requête il n'est pas nécessaire qu'un seul service Web puisse répondre aux besoins du demandeur, il est donc indispensable que plusieurs services Web collaborent ensemble afin de fournir une réponse appropriée au demandeur.

L'objectif de notre solution est de fournir une architecture qui supporte la composition automatique des services Web sémantiques distribués sur un réseau P2P dans un environnement d'exécution spécifique nommé WSMX (*Web Service Modeling eXecution environment*) qui couvre tout le cycle de vie d'un service Web.

Afin d'atteindre cet objectif, notre contribution s'articulent autour des points suivants :

La première étape de notre solution consiste à organiser les pairs du réseau en *pairs locaux* et *pairs groupes* suivant une structure hybride, L'objectif est de créer un espace collaboratif entre les différents pairs participants à la réalisation d'un but commun, tout en préservant la stabilité et la scalabilité du réseau.

Dans la deuxième étape, nous avons utilisé une table dite table de composition afin de préserver la trace du chemin de composition pour une éventuelle future réutilisation. En effet, la réutilisation des compositions déjà réalisées présente un facteur important qui peut minimiser le temps et le coût de réponse.

4. Organisation du mémoire

Ce mémoire est scindé en deux parties majeures :

La première partie est consacrée à un état de l'art relatif aux domaines abordés dans ce mémoire, elle est subdivisée en quatre chapitres :

Le chapitre 1 : aborde les technologies orientées service et plus particulièrement la technologie des services Web.

Le chapitre 2 : dans ce chapitre, nous introduisons le concept du Web sémantique avant de présenter le paradigme des services Web sémantiques, les langages de descriptions des services Web sémantiques et quelques approches dans le domaine.

Le chapitre 3 : présente la composition des services Web. Nous présentons aussi la problématique de la découverte des services Web sémantiques et les différentes issues de la composition de services Web dans les réseaux P2P proposées par différents acteurs.

Le chapitre 4 : est consacré à la présentation des systèmes P2P et leurs applications. Nous montrons les différents types des réseaux P2P, et leurs plateformes de développement.

La deuxième partie est le cœur de notre travail. Elle est consacrée à notre contribution et est composée de deux chapitres :

Le chapitre 5 : présente notre contribution, nous commençons par les motivations essentielles derrière l'utilisation des réseaux P2P, ensuite nous présentons notre approche basée sur un modèle P2P hybride.

Le chapitre 6 : présente les détails techniques concernant l'implémentation de notre prototype basé sur l'approche proposée.

Première partie

L’Etat de l’Art

Chapitre 1

DEFINITION DES SERVICES WEB

1. Introduction

Le web contient des vastes bases de données avec plusieurs buts et de multiples sources non homogènes. Ceci prouve qu'il est nécessaire d'améliorer l'accessibilité à cette importante masse d'informations, et de disposer d'outils plus sophistiqués pour une meilleure recherche et organisation au sein du web.

Les services Web fournissent une nouvelle manière de développer des applications conformes aux besoins de l'Internet, et ils semblent être la solution la plus adaptée pour assurer l'interopérabilité, qui permet de transmettre les données entre les différentes applications d'une organisation comme l'entreprise, la société ou l'individu ; ainsi la technologie des web services permet de réaliser le traitement de ces données, et gérer les liaisons entre les différentes applications.

Il existe aujourd'hui une grande variété de services Web capables de répondre aux différents types de requêtes des utilisateurs (prévisions météorologiques, différentes réservations comme pour l'hôtel ou les voyages, recherche, …).

2. Définition des services Web

Les services Web permettent de faire communiquer des sous-systèmes ensemble et cela indépendamment des architectures utilisées. Ces systèmes vont pouvoir s'échanger des services ou des traitements applicatifs. Les services Web constituent un moyen de distribuer un service de manière standard grâce à XML tout en respectant un modèle de développement ayant déjà fait ses preuves comme CORBA ou DCOM.

Le W3C a défini les services Web comme étant « un système logiciel conçu pour supporter l'interaction interopérable de machine à machine sur un réseau. Il possède une interface décrite en WSDL (Web Services Description Language) qui est un format exploitable par la machine. D'autres systèmes interagissent avec les services Web d'une façon prescrite par sa description en utilisant des messages SOAP (Simple Object Access Protocol), typiquement en utilisant HTTP (HyperText Transfer Protocol) avec une sérialisation XML en même temps que d'autres normes du Web». [W3C, 04]

Exemples des Services Web:

Plusieurs compagnies publient des interfaces publiques des services Web, à savoir:

- Google: http://www.google.com/apis, une application qui emploie une ontologie pour mettre en valeur une recherche Google.
- Prévisions météo via le service du National Digital Forecast Database XMLWeb: http://www.weather.gov/xml.

- Amazon: http://simplest-shop.com/camera (e-commerce)
- MapQuest (pour localiser un lieu et donner des itinéraires), FedEx, etc.

3. Caractéristiques des services Web

Selon [CER, 02], plusieurs acteurs définissent les web services par des caractéristiques technologiques distinctives, les plus importantes sont :

- **Un web service est une application logicielle qui est reconnue par un URI :** URI est la façon d'identifier un point de contenu sur le web comme un document tel qu'un texte, audio ou vidéo. L'URI le plus connue est l'adresse d'une page web, le web service est donc accessible en spécifiant son URI, c'est-à-dire que le web service est caractérisé par un seul objet et une seule fonctionnalité, c'est un prolongement de la programmation orientée objet et à partir de cela, on peut faire la construction d'une application logicielle très large comportant plusieurs fonctionnalités, afin de sélectionner les fonctionnalités qui sont recherchées par les URI spécifiques.
- **Capacité des interfaces et liaisons (bindings) d'être publiées, localisées et invoquées via le langage XML :** les principales tâches d'un web service sont : la publication dans un registre, la localisation en interrogeant ce registre qui l'héberge et l'invocation par un ou plusieurs web services après sa localisation. Ces tâches sont réalisées en utilisant le langage XML.
- **Capacité d'interagir avec les composantes des logiciels via des éléments XML avec l'utilisation des protocoles Internet standards :** un web service est créé pour être interrogé par d'autres logiciels contrairement à une page web, ou à une autre application qui n'utilise pas les web services. L'interopérabilité est basée sur l'utilisation du XML et des protocoles Internet standards, tels que, le HTTP qui est le protocole du web, le SMTP qui est le protocole du courrier électronique,…etc.
- **Composante logicielle légèrement couplée à interaction dynamique :** un web service avec un programme qui l'invoque est appelé le consommateur de web service, et qui sont indépendants l'un de l'autre. Si une modification est à faire sur le consommateur, on n'a pas besoin de connaître la machine, le langage de programmation, le système d'exploitation ou autres paramètres, afin d'établir à nouveau une communication entre le web service et son consommateur. Le consommateur possède une fonctionnalité qui consiste à faire une localisation et une invocation du web service, au moment de l'exécution du programme de web service, de manière automatique.

4. Architecture des services Web

L'architecture de référence des web services vise trois objectifs importants :

L'identification des composants fonctionnels, la définition des relations entre ces composants et l'établissement d'un ensemble de contraintes sur chaque composant de manière à garantir les propriétés globales de l'architecture. [PAT, 04], [SOF, 07] L'architecture de référence comporte les trois éléments suivants :

- **Le fournisseur de service :** c'est le propriétaire du service. D'un point de vue technique, il est constitué par la plateforme d'hébergement du service.
- **Le client :** c'est le demandeur de service. Techniquement, il est constitué par l'application qui va rechercher et invoquer un service. Une application cliente peut être elle-même un web service
- **L'annuaire des services :** c'est un registre de descriptions de services offrant des facilités de publication de services pour les fournisseurs de services ainsi que des facilités de recherche de services pour les clients.

Ces trois éléments de l'architecture interagissent entre eux selon trois types d'opérations : les opérations de publication, de recherche et de liens.

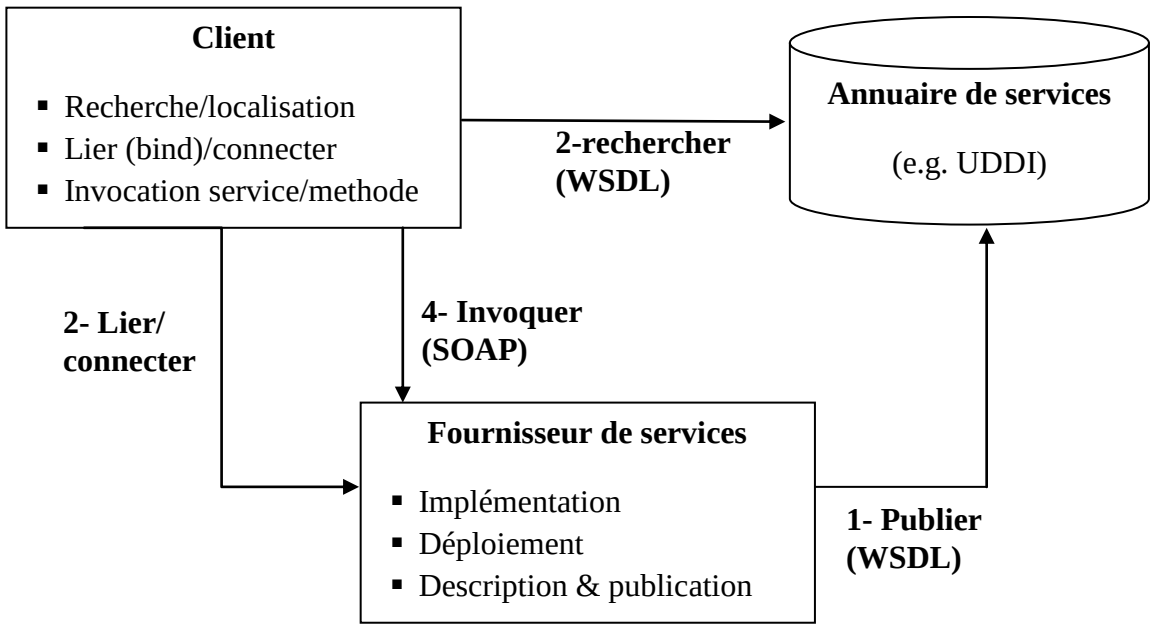


Figure 1. Architecture des web services. [PAT, 04]

Le fournisseur de services définit la description de son service et la publie dans un annuaire de service UDDI qui peut être public ou privé. Le client utilise les facilités de recherche disponibles au niveau de l'annuaire pour retrouver et sélectionner un service donné. Il récupère ensuite les informations nécessaires sous format WSDL, à partir de la description du service sélectionné, lui

permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré. La communication entre le demandeur de service et le fournisseur est assurée par le SOAP et les autres protocoles de communication. Le demandeur de service envoie une requête SOAP vers le fournisseur de service, cette requête est véhiculée par le HTTP jusqu'au fournisseur. Ensuite le web service du fournisseur de service renvoie sa réponse au demandeur sous la forme d'un document XML via SOAP et HTTP.

Plusieurs standards ont été proposés pour assurer l'interaction entre les trois opérations précédentes (publication, recherche et lien), entre autre nous citons les standards suivants :

- SOAP (Simple Object Access Protocol) est un protocole d'échange inter application indépendant de toute plateforme, basé sur le langage XML. Un appel de service SOAP est un flux ASCII encadré dans des balises XML et transporté dans le protocole HTTP.
- WSDL (Web Services Description Language) introduit une grammaire commune pour la description des services. Il donne la description au format XML des web services en précisant les méthodes pouvant être invoquées, et le point d'accès (URL, port, etc..).
- UDDI (Universal Description, Discovery and Integration) normalise une solution d'annuaire distribué de web services, il fournit l'infrastructure de base pour la publication et la découverte des web services. UDDI se comporte lui-même comme un web service dont les méthodes sont appelées via le protocole SOAP.

(Pour plus de détails concernant ces langages, voir l'Annexe A).

5. Conclusion

Les services Web ont pour objectif l'interopérabilité entre applications via le web en vue de rendre le web plus dynamique. Ils facilitent l'accès aux applications et les échanges de données entre entreprises. Ils poursuivent un vieux rêve de l'informatique distribuée où les applications pourraient s'interopérer à travers le réseau, indépendamment de leur plateforme et de leur langage d'implémentation.

Les diverses applications qui utilisent les web services utilisent les annuaires UDDI, ces dernières n'offre pas des informations riches pour la recherche des web services. D'où la nécessité de rajouter une couche sémantique à ces services. Le web sémantique et les ontologies sont les outils utilisés pour cette tâche, cela fait l'objet du chapitre suivant.

Chapitre 2

LES SERVICES WEB SEMANTIQUES

1. Introduction

Les services Web sont des applications modulaires qui fournissent un modèle simple de programmation et de déploiement d'applications, basées sur des normes et s'exécutant au travers de l'infrastructure Web. Les Web services réalisent des fonctionnalités allant des simples requêtes aux processus métiers sophistiqués. Ils sont significatifs seulement si les utilisateurs peuvent trouver des informations suffisantes pour leurs exécutions.

Les services Web sont basés sur plusieurs standards à savoir le standard UDDI (Universal Description, Discovery and Integration) qui est la définition d'un ensemble de services soutenant la description et la découverte des entreprises, organismes, d'autres fournisseurs de web services, les web services qu'ils rendent disponible et les interfaces techniques qui peuvent être employées pour accéder à ces services. Donc les web services deviennent significatifs à l'aide d'UDDI.

Les services Web assurent, à travers UDDI, la publication et la découverte des applications (des web services). La réalisation de ces tâches, de manière manuelle, est clairement fastidieuse, notamment pour la tâche de découverte dont l'utilisateur doit chercher manuellement le service pertinent qui répond à ses besoins dans une grande gamme de services. Les solutions d'automatisation de cette tâche devenaient de plus en plus indispensables. Et c'est dans ce contexte que sont nés les premiers concepts d'automatisations de la découverte, ces concepts consistent, pour la plupart de temps, à ajouter de la sémantique aux web services.

2. Les ontologies

2.1. Définition

Afin de décrire ce qu'est une ontologie, de nombreuses définitions ont été proposées. Parmi les plus répondues, on citera :

Selon [GRU, 93] : "*Une ontologie est une spécification explicite d'une conceptualisation*". Cette définition s'appuie sur deux dimensions :

- **« conceptualisation »** correspond au «modèle abstrait » d'une partie du monde réel, sur lequel doit travailler le système considéré. Qui se présente comme un ensemble de définitions de concepts munis de propriétés et de relations entre ces concepts.
- **« spécification explicite »** signifie que le modèle en question doit être décrit de façon non ambiguë dans un langage. Ce langage peut être une langue naturelle (ex : français, anglais) ou un langage formel (ex : logique du 1er ordre, réseau sémantique).

Selon [BOR, 97] : *"une ontologie est définie comme étant une spécification formelle d'une conceptualisation partagée*". On entend par :

- **« explicite » :** signifie que l'ensemble des concepts utilisés et leurs contraintes d'utilisation sont définis d'une façon explicite ;
- **« formel » :** précise que l'ontologie construite doit être lisible par un ordinateur ;
- le terme **« partagée »** montre qu'une ontologie fournit un vocabulaire conceptuel commun et une compréhension partagée par la communauté visée.

En informatique, une ontologie est un ensemble structuré de concepts. Les concepts sont organisés dans un graphe dont les relations peuvent être :

- des relations sémantiques ;
- des relations de composition et d'héritage (au sens objet).

Il a été bien reconnu dans la communauté du Web Sémantique que les ontologies jouent un rôle clé dans la livraison du Web Sémantique en facilitant le partage d'information entre les communautés d'humains et des agents logiciels. [BOU, 04]

2.2. Les composantes d'une ontologie

Les connaissances exprimées par les ontologies sont véhiculées à l'aide de cinq éléments: Concepts, Relations, Fonctions, Axiomes et Instances [GOM, 99].

- **Concepts:** Ils sont également appelés termes ou classes de l'ontologie. Un concept est un constituant de la pensée (un principe, une idée, une notion abstraite) sémantiquement évaluable et communicable [BEN, 05]. Les concepts peuvent être classifiés selon plusieurs dimensions: niveau d'abstraction (concret ou abstrait), atomicité (élémentaire ou composé) et niveau de réalité (réel ou fictif).
- **Relations**: Elles servent à exprimer les associations existant entre les concepts présents dans le segment analysé de la réalité. Elles regroupent les associations suivantes : sous – classe – de (spécialisation, généralisation), partie – de (agrégation ou composition); associé – à, instance – de, est – un, etc. Les relations permettent d'apercevoir la structuration et l'interrelation des concepts, les uns par rapport aux autres.
- **Fonctions:** Elles représentent des cas particuliers de relation où un élément (le nième) est défini en fonction des n-1 éléments précédents.
- **Axiomes:** Constituent des assertions, admises comme vraies, à propos des abstractions du domaine traduit par l'ontologie.
- **Instances:** Elles constituent la définition extensionnelle de l'ontologie; ces objets véhiculent les connaissances (statiques, factuelles) à propos du domaine du problème.

2.3. Langage d'ontologie

Le W3C a proposé un standard, connu actuellement sous le nom d'OWL (Ontology Web Language). Le langage OWL est construit sur RDFS, et apporte ainsi aux langages du Web

sémantique, l'équivalent d'une logique de description tout en disposant aussi d'une syntaxe XML. Sans être exhaustif, il ajoute à RDF la possibilité de définir des classes de manière plus complexe correspondant aux connecteurs de la logique de description équivalente (intersection, union, restrictions diverses, etc.), les classes disjointes, les propriétés inverses ou transitives ou bien encore les restrictions de cardinalité sur les propriétés. L'accès à ces ontologies et leur rôle pour le Web sémantique reste encore largement à explorer, même si on peut partager les perspectives de Patel-Schneider : « les utilisateurs d'un serveur Web d'ontologies auront accès à de nombreux services puissants pour stocker et analyser la connaissance disponible sur le Web et faire des inférences à partir de cette connaissance, fournissant aux programmeurs beaucoup plus que les données contenues explicitement dans les documents XML » [LAU, 02].

3. Le web sémantique

Le web sémantique, proposé par le W3C, est une infrastructure visant à rendre le contenu sémantique des ressources web interprétables non seulement par l'homme mais aussi par les programmes dont le but est d'avoir une meilleure coopération entre humains et machines. Cette infrastructure permet l'exploitation de connaissances formalisées en plus du contenu informel actuel du web, elle permet de localiser, d'identifier et de transformer des ressources de manière robuste tout en renforçant l'esprit d'ouverture du web avec sa diversité d'utilisateurs. Le web sémantique n'est pas un web distinct mais bien un prolongement du web que l'on connaît, dans lequel une signification clairement définie est attribuée à l'information, ce qui permet une collaboration plus étroite entre machines et humains.

Le web sémantique vise à satisfaire les fonctionnalités avancées pour l'interaction utilisateur-utilisateur, utilisateur – machine et machine – machine. Il permet le partage des ressources et le raisonnement sur le contenu de ces dernières. Afin de rendre la recherche et la sélection d'information facile aux programmes chargés de ces fonctionnalités, les documents sont structurés à l'aide de méta données ou d'annotations ou d'ontologies. [BEN, 05], [ABE, 05], [BOU, 04].

3.1. Définition

L'expression Web sémantique, attribuée à Tim Berners-Lee au sein du W3C, fait d'abord référence à la vision du Web de demain comme un vaste espace d'échange de ressources entre êtres humains et machines permettant une exploitation, qualitativement supérieure, de grands volumes d'informations et de services variés. Espace virtuel, il devrait voir, à la différence de celui que nous connaissons aujourd'hui, les utilisateurs déchargés d'une bonne partie de leurs tâches de recherche, de construction et de combinaison des résultats, grâce aux capacités accrues des machines à accéder aux contenus des ressources et à effectuer des raisonnements sur ceux-ci. [LAU, 02].

Comme l'écrit, en substance, Tim Berners-Lee, ***[BER 01]*** « le Web sémantique est ce que nous obtiendrons si nous réalisons le même processus de globalisation sur la représentation des connaissances que celui que le Web fit initialement sur l'hypertexte ». [LAU, 02], [BOU, 04]

Le Web Actuel	**Le Web Sémantique**
Ensemble de documents	Ensemble de connaissances
Basé essentiellement sur HTML	Basé sur XML et RDF(S)
Recherche par mots clés	Recherche par concepts
Utilisable par l'humain	Utilisable par la machine

Tableau1. Comparaison entre le web actuel et le web sémantique

3.2. Les composantes du web sémantique

Les travaux visant la réalisation du Web sémantique se situent à des niveaux de complexité très différents. Les plus simples utilisent des jeux plus ou moins réduits de méta-données dans un contexte de recherche d'information ou pour adapter la présentation des informations aux utilisateurs. Dans ce cas, des langages de représentation simples sont suffisants. Disposer de chacun de ces langages est indispensable au développement des fonctionnalités correspondantes du Web sémantique. Ces langages permettront diverses applications nouvelles telles que :

- La recherche d'information fondée sur des descriptions formelles ;
- La composition de services en fonction de leur description ;
- L'interconnexion de catalogues sur la base de leur description.

L'infrastructure du Web sémantique s'articule autour de trois composantes essentielles: Métadonnées ou annotations sémantiques, Ontologies, Systèmes de raisonnement. Ajoutant à cela, l'infrastructure du Web sémantique fait appel à une autre notion essentielle qui est : URI (Uniform Ressource Identifier).

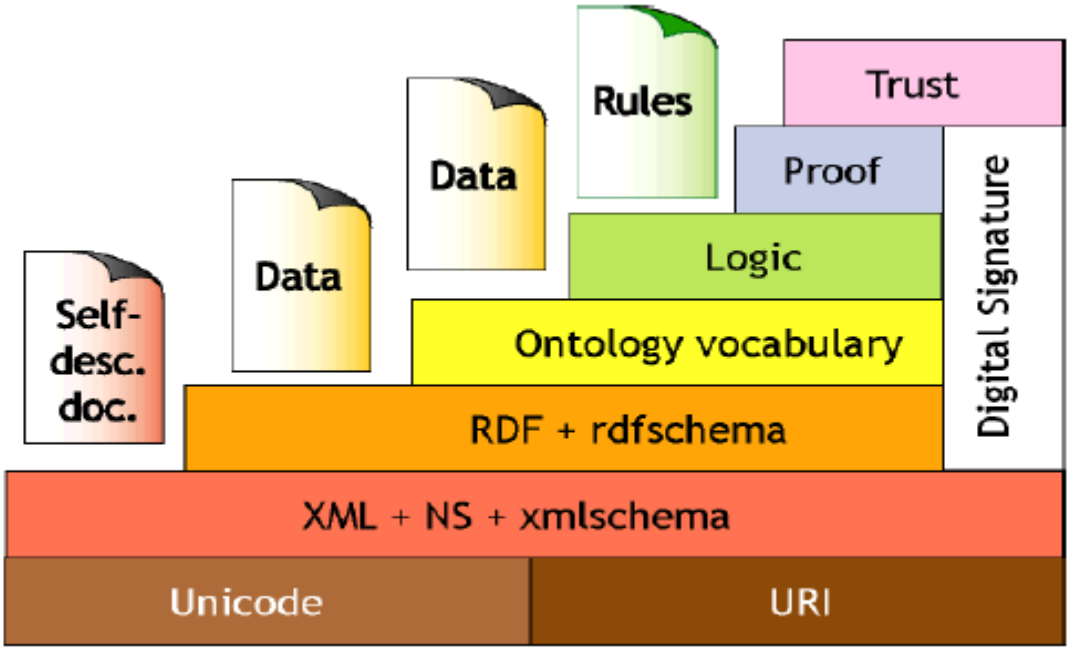


Figure 2. L'architecture en couches du web sémantique

(Pour plus de détails concernant les langages du Web sémantique, voir Annexe B).

4. Les services web sémantiques

L'objectif ultime de l'approche services Web est de transformer le Web en un dispositif distribué de calcul où les programmes (services) peuvent interagir de manière intelligente en étant capables de se découvrir automatiquement, de négocier entre eux et de se composer en des services plus complexes. En d'autres termes, l'idée poursuivie avec les services Web, est de mieux exploiter les technologies de l'Internet en substituant, autant que possible, les humains qui réalisent actuellement un certain nombre de tâches, par des machines en vue de permettre une découverte et/ou une composition automatique de services sur l'Internet.

L'automatisation est donc un concept clé qui doit être présent à chaque étape du processus de conception et de mise en œuvre des services Web.

Pour pallier ces difficultés, les services Web sémantiques (Semantic Web Services) ne se contentent pas de définitions syntaxiques, mais apportent plus d'informations pour permettre de composer, de coordonner, et de surveiller plusieurs services Web.

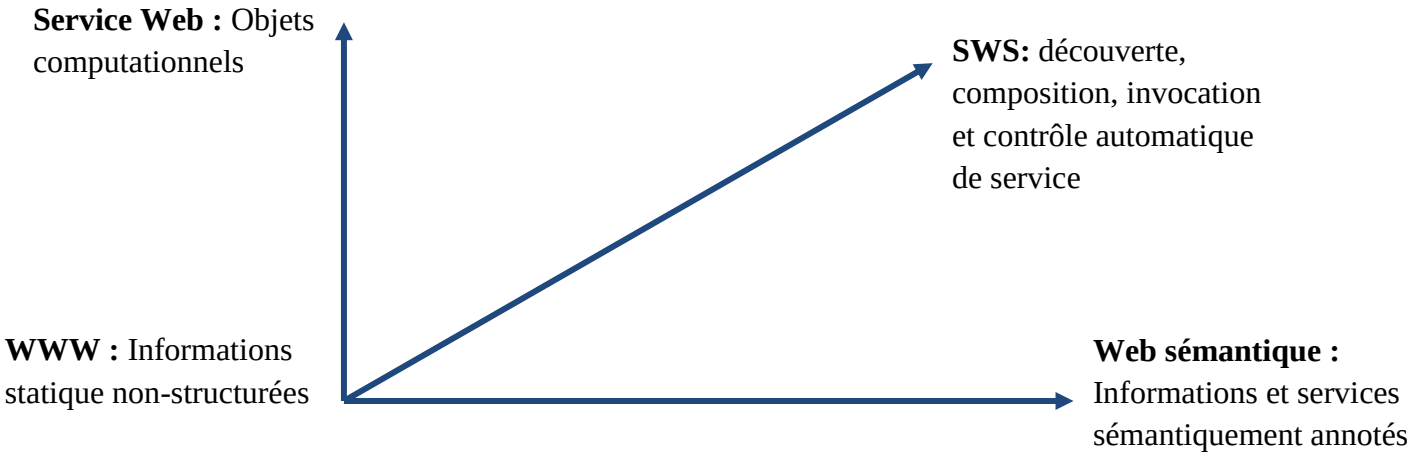


Figure 3. La convergence du Web sémantique et les services Web [DAV, 04]

De manière générale, l'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les interfaces, les capacités 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. La sémantique ainsi exprimée permettra l'automatisation des fonctionnalités suivantes qui sont nécessaires pour une collaboration et une interaction inter-entreprises efficace [KEL, 03]:

- Processus de description et de publication des services ;
- Découverte de services ;
- Sélection des services ;
- Composition de services ;
- Fourniture et administration des services ;
- Négociation des contrats.

5. Quelques approches proposées pour la réalisation des services Web sémantiques

L'approche Service Web Sémantique (SWS) 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. En conséquence, les technologies et les outils développés dans le contexte du Web sémantique peuvent compléter la technologie des services Web en vue d'apporter des réponses crédibles au problème de l'automatisation. La notion d'ontologie peut jouer un rôle prépondérant pour permettre d'expliciter la sémantique des services facilitant les communications hommes-machines, d'une part, et les communications machines-machines, d'autre part. Ainsi, le domaine des services Web sémantiques se situe au croisement du Web

sémantique et des services Web (Figure. 11). De nombreux langages et architectures sont proposés afin de décrire les services Web sémantiques, citons IRS-II (Internet Reasoning Service) [MOT, 03], WSMO (Web Service Modeling Ontology) [ARR, 04], OWL-S (Web Ontology Language for Web Services) [OWL, 04], et METEOR-S (METEOR for Semantic Web Services) [MET 05].

5.1. Web Ontology Language for Web Services (OWL-S)

Le 'Web Ontology Language for Web Services' (OWL-S)[1] [OWL, 04], anciennement DAML-S est une ontologie pour la description sémantique des services Web qui a été développée dans le cadre du projet DAML. OWL-S est un sous-ensemble du langage 'Web Ontology Langage' (OWL) [GOM, 04].

OWL-S décrit ce que le service est capable de faire et pas seulement comment il le fait, ce qui rend les ressources Web accessibles aussi bien par le contenu que par des mots clés. Comme illustré dans la *figure 4*, une description d'un service Web utilisant OWL-S s'articule autour de trois classes d'informations : ServiceProfile (permet une description du service offert), ServiceModel (présente la définition et le fonctionnement des différents processus du service Web) et le ServiceGrounding (définit les modalités d´accès au service).

Dans l'approche OWL-S, la partie profil contient l'information la plus utile pour la découverte et la composition de services du fait qu'elle est utilisée à la fois par les fournisseurs de services pour la publication et par les clients pour l'interrogation.

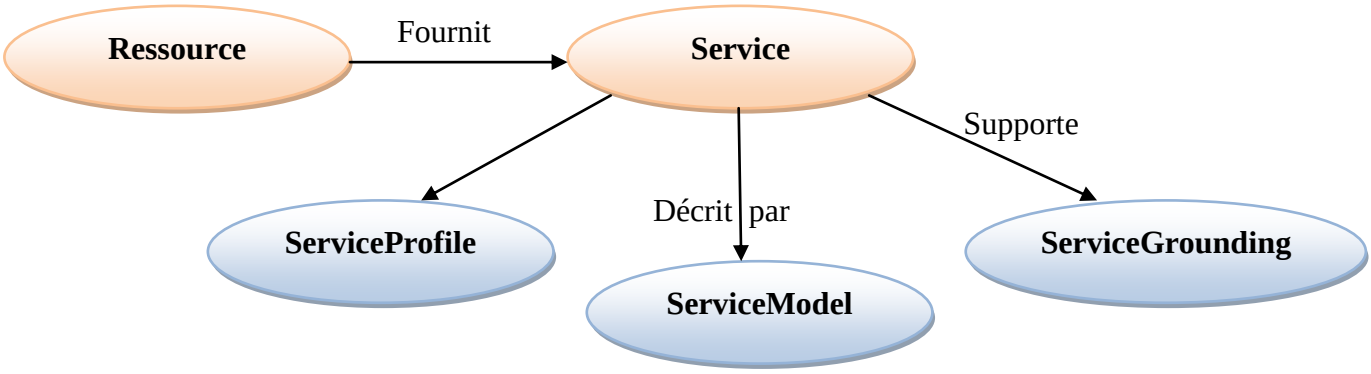


Figure 4. Ontologie OWS-S

[1] http://www.daml.org/services/owl-s/

5.2. Web Service Modeling Ontology (WSMO)

L'architecture Web Service Modeling Ontology (WSMO)[2] [ARR, 04] est une extension WSMF (Web Service Modeling Framework) [FEN, 02], proposée par le laboratoire DERI.

C'est une architecture conceptuelle, ou métamodèle, visant à expliciter la sémantique des services Web. WSMO est un langage formel et comprend des ontologies qui permettent de décrire des aspects variés des services Web sémantiques [BUS, 04]. Elle est organisée autour de quatre éléments essentiels *(Figure. 5)* :

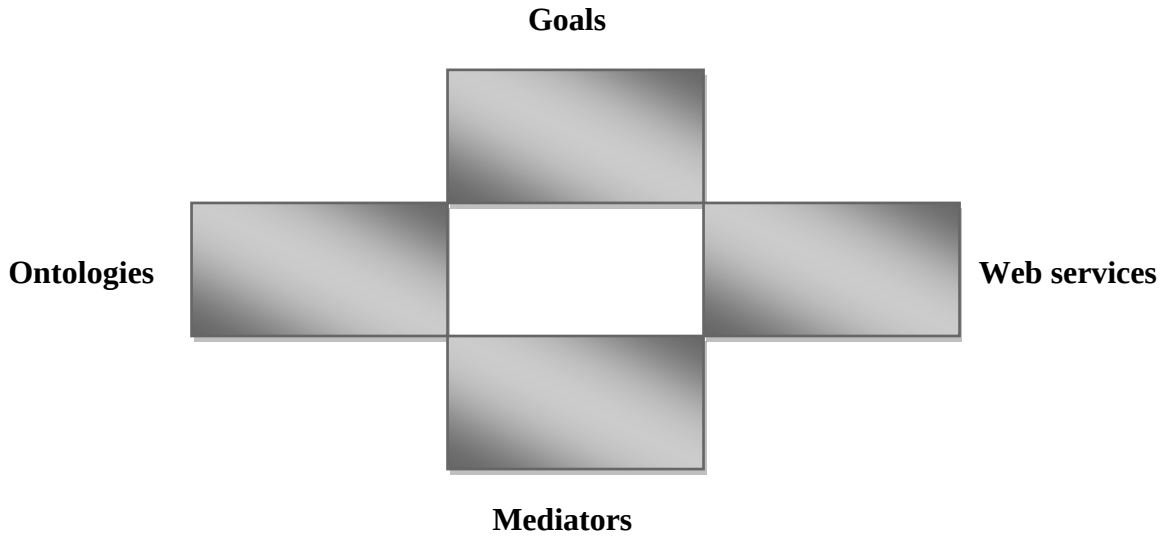


Figure 5. Les éléments de WSMO

- **Les services Web** sont définis comme des « entités de calcul » qui fournissent une fonctionnalité. Une description est associée à chaque service, dont le but est de décrire sa fonctionnalité, son interface et ses détails internes.
- **Les Objectifs** servent à décrire les souhaits des utilisateurs en termes de fonctionnalités requises. Les objectifs constituent une vue, orientée utilisateur, du processus d'utilisation des services Web. Un objectif décrit la fonctionnalité, les entrées/sorties, les préconditions et les postconditions d'un service Web.
- **Les Médiateurs** sont utilisés pour résoudre de nombreux problèmes d'incompatibilité, telles que les incompatibilités de données dans le cas où les services Web utilisent différentes terminologies, les incompatibilités de processus dans le cas de la combinaison de services Web, et les incompatibilités de protocoles lors de l'établissement des communications. WSMO inclut les médiateurs comme des composants centraux de son architecture. Les

[2] http://www.wsmo.org/

hétérogénéités rencontrées lors de l'utilisation de services Web sont gérées par les types de médiation suivants :

- La **médiation de données** 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.
- Les **Ontologies** fournissent la terminologie de référence aux autres éléments de WSMO, afin de spécifier le vocabulaire du domaine de connaissance d'une manière interprétable par les machines. [MRI, 07]

Dans WSMO, on distingue deux composants principaux qui sont WSML (Web Service Modeling Language) [BRU, 05] et WSMX (Web Service Execution Environment) [BUS, 04].

(Pour plus de détails concernant l'environnement WSMX, voir Annexe D).

WSML fournit un langage formel pour la description des éléments définis dans l'architecture WSMO. Il est basé sur différents langages logiques qui sont la logique de description, la logique de premier ordre et la logique de programmation. WSMX est un environnement d'exécution qui offre un certain nombre de modules utilisables en temps d'exécution permettant de prendre en charge la découverte, la sélection, la médiation et l'invocation des services sémantiques.

5.3. METEOR for Semantic Web Services (METEOR-S)

METEOR for Semantic Web Services (METEOR-S)[3] [MET, 05] est une initiative du laboratoire LSDIS de Géorgie et qui a pour objectif de fournir des services Web sémantiques dans le cadre du projet METEOR (Managing End-To-End OpeRations). Il a précisément pour but d'intégrer les standards des services Web (BPEL4WS, WSDL et UDDI) avec les technologies du Web sémantique pour l'annotation, la découverte, la composition, la qualité de service et l'exécution de services Web.

Le projet METEOR-S s'est développé selon trois principales phases qui ont permis d'introduire les trois concepts importants de cette initiative :

- ***Mise en place de l'infrastructure*** : elle consiste à installer une infrastructure de découverte sémantique définie au dessus d'un registre UDDI et qui est appelée MWSDI (*METEOR-S Web Service Discovery Infrastructure*);
- ***Annotation sémantique*** : elle définit l'outil MWSAF (METEOR-S Service Web Annotation Framework) et qui permet d'enrichir sémantiquement les services Web en utilisant une extension enrichie de WSDL appelée WSDL-S (WSDL-Semantics). Le rôle

[3] http://lsdis.cs.uga.edu/projects/meteor-s/

de WSDL-S [WSD, 05] est d'ajouter un niveau sémantique aux services Web à travers l'enrichissement des fichiers WSDL par des annotations, qui sont stockées dans la structure existante de l'UDDI;

- ***La composition et l'exécution de services*** : elle a pour rôle d'assembler des services pour composer des services complexes en se basant sur BPEL4WS. L'outil se chargeant de cette tâche s'appelle MWSCF (METEOR-S *Web Service Composition Framework*).

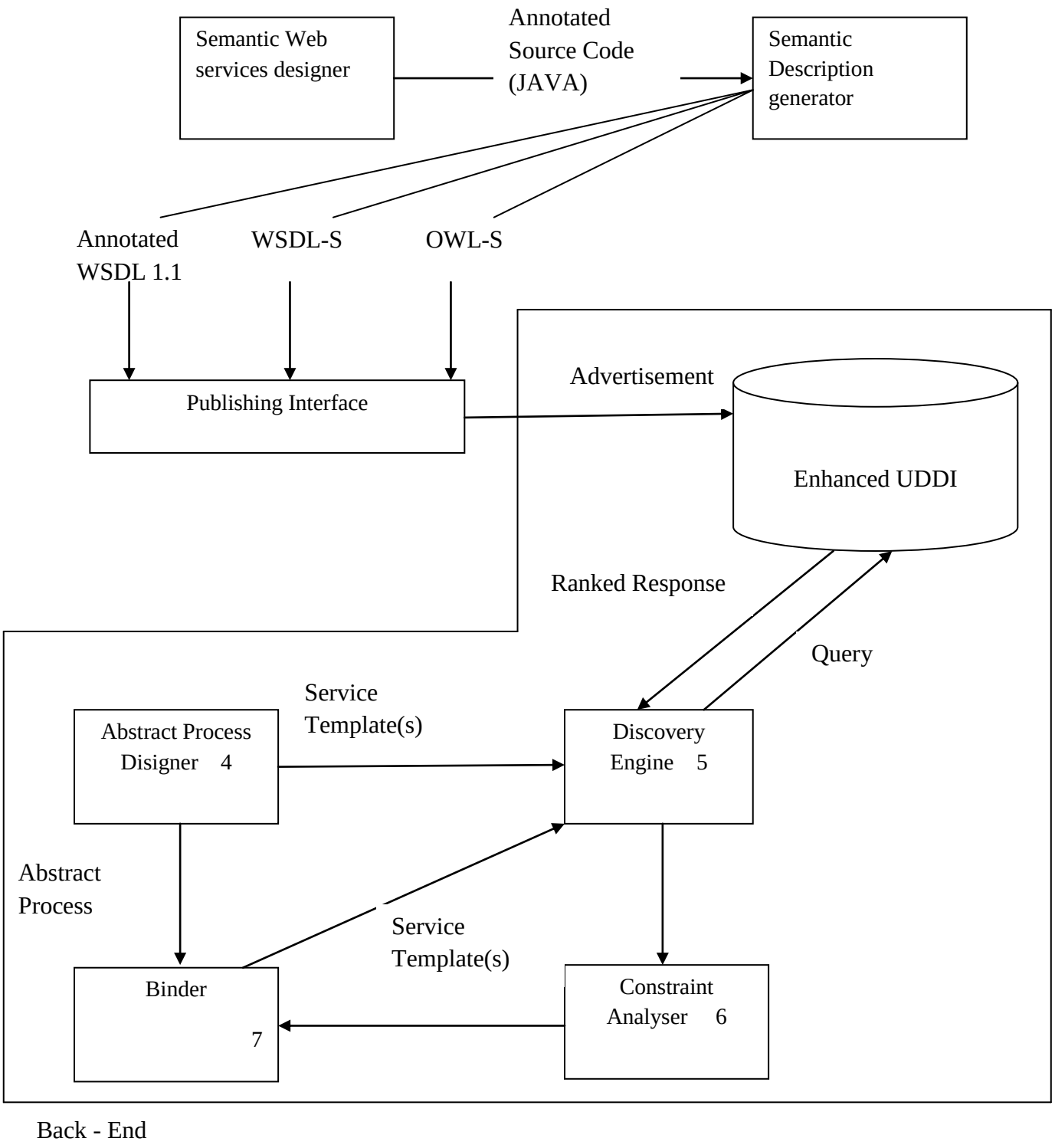


Figure 6. Architecture de METEOR-S [MET, 05]

6. Conclusion

Les web services sémantiques visent à faire une combinaison entre le web sémantique et la technologie des web services afin de permettre une interaction automatique et dynamique entre les systèmes. L'annuaire UDDI reste toujours l'entité qui serve d'appui à la découverte de web service pour les applications clientes.

Avec l'avènement des web services sémantiques, de nouvelles approches sémantiques ont été proposées, ces dernières donnent de bons résultats comparés aux anciennes approches (les approches syntaxiques).

A travers l'étude de ces approches, nous avons remarqué que les web services renforcés par les descriptions sémantiques commencent à trouver leurs applications dans de nombreux domaines.

Chapitre 3

La composition de services Web

1. Introduction

Les composants sont destinés à être composés [BRO, 96]. La composition, et donc la réutilisation, est l'une des caractéristiques les plus importantes des systèmes à composants. Réellement, il n'est pas toujours facile de trouver des services Web qui s'apparient avec les requêtes des utilisateurs. Par conséquent, la composition des services satisfaisant la requête est un besoin grandissant de nos jours. La composition de services est une tâche complexe. Cette complexité est due principalement aux raisons suivantes [BUC, 06] :

- Le nombre de servicesWeb disponibles sur leWeb est potentiellement très important et en constante augmentation ;
- Les services Web peuvent être créés et modifiés à la volée, le système de composition doit donc détecter et gérer ces changements ;
- Les services Web sont créés par des organisations différentes qui n'ont pas forcément les mêmes modèles de concepts pour décrire ces services.

Ces dernières années, la recherche dans le domaine des services Web a été très développée. Une grande partie de cette recherche a été consacrée à la découvert et la composition de services Web [CHA, 10]. L'idée originale de la composition de service Web n'est pas vraiment nouvelle dans la conception d'applications. La composition de services qui autorisent la composition des services simples, afin d'obtenir un service complexe qui répond aux exigences d'un utilisateur.

2. Définition

La composition des services Web est le processus de construction de nouveaux services Web à valeur ajoutée, à partir de deux ou plusieurs services Web déjà présents et publiés sur le Web. Un service Web est dit composé ou composite lorsque son exécution implique des interactions avec d'autres services Web, et des changements des messages entre eux afin de faire appel à leurs fonctionnalités. La composition de services Web spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction [ARE, 06].

En d'autres termes, la composition des services Web est la combinaison des services Web existants pour former de nouveaux services. L'objectif principal de la composition de service Web est la possibilité de créer de nouvelles fonctionnalités d'un nouveau service Web, en combinant des fonctionnalités offertes par d'autres services Web existants. Elle implique la capacité de sélectionner, de coordonnée, d'interagir, et de faire interopérer des services Web existants.

Dans le domaine de la composition de services Web, il existe deux manières différentes pour spécifier ce domaine, qui sont : l'orchestration et la chorégraphie (Voir la Figure 7).

L'orchestration décrit, du point de vue d'un service Web, les interactions de celui-ci ainsi que les étapes internes (par exemple, transformations de données, invocations à des modules internes) entre ses interactions [BRY, 04].

L'orchestration explique la manière avec laquelle les services Web peuvent interagir entre eux selon un scénario prédéfini au niveau des messages dans une vision opérationnelle, avec des structures de contrôle, incluant la logique métier et l'ordre d'exécution des interactions. Un service Web prend le contrôle du déroulement de la composition et coordonne, donc, les différentes opérations des différents services Web. À aucun moment, les services Web servant à la composition n'ont connaissance de cette composition, ils remplissent leur rôle de service sans se soucier si une application interagit avec eux.

La chorégraphie décrit la collaboration entre une collection de services dont le but est d'atteindre un objectif donné. L'accomplissement de ce but commun se fait alors par des échanges ordonnés de messages [BRY, 04].

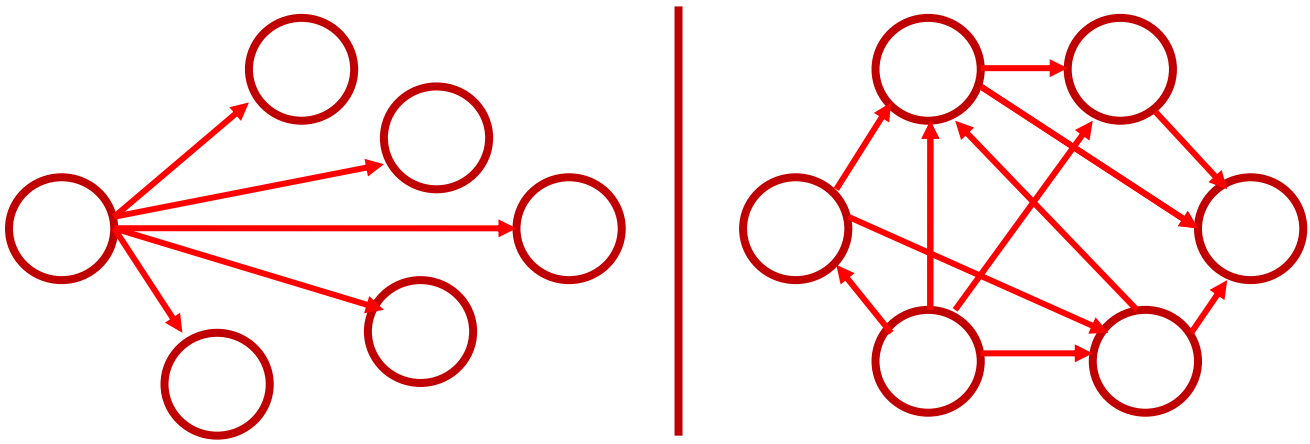


Figure 7. Orchestration et chorégraphie de services Web

Une différence importante entre l'orchestration et la chorégraphie est que l'orchestration offre une vision centralisée, c'est-à-dire, que le procédé est toujours contrôle de la perspective d'un des partenaires. En revanche, la chorégraphie offre une vision globale et plus collaborative de la composition des services Web. Elle décrit le rôle qui joue chaque service Web impliqué dans l'application.

Il existe plusieurs langages d'orchestration et de chorégraphie, qui sont proposés par la composition des services Web.

3. Types de composition de services Web

L'idée de la composition de services Web consiste à définir comment les services Web vont être rassemblés selon certains règles, pour atteindre le but demander par un utilisateur. Une fois que, la description de la composition est réalisée, il est possible de savoir facilement

quels services Web appartiendront à cette composition. Les solutions proposées peuvent être classifiées selon deux axes :

En fonction du degré de participation de l'utilisateur dans la définition du schéma de composition, ces propositions sont manuelles, semi-automatiques ou automatiques [CHA, 07]:

- **La composition manuelle :** La composition manuelle des services Web suppose que l'utilisateur gère la composition à la main via un éditeur de texte et sans l'aide d'outils dédiés.
- **La composition semi-automatique :** Les techniques de composition semi-automatiques sont un pas en avant en comparaison avec la composition manuelle, dans le sens qu'elles font des suggestions sémantiques pour aider à la sélection des services Web dans le processus de composition.
- **La composition automatique :** La composition totalement automatisée prend en charge tout le processus de composition et le réalise automatiquement, sans qu'aucune intervention de l'utilisateur ne soit requise.

Selon que, la sélection des services Web et la gestion du flot soient faites ou non a priori, une autre approche sera dite statique ou dynamique [POU 07] :

- **La composition statique :** 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.
- **La composition dynamique :** Si les services Web sont sélectionnés et composés à la volée en fonction des besoins formulés par l'utilisateur.

Une composition est statique quand tous les services Web qui font partie de la composition sont connus, ainsi que leur ordre d'exécution. Dans cette composition, les services sont connus avant leur exécution, c'est-à-dire, au moment ou la composition est faite. De plus, nous connaissons également l'ordre d'exécution ainsi que la localisation et la description des services Web (le fichier WSDL). Dans la composition statique, tout est complètement spécifié et en particulier tous les services Web sont connus.

De plus, la création de ces processus métier se fait durant le développement du système et reste statique pendant son utilisation, or l'environnement des services Web est dynamique. Pour ces raisons, les chercheurs fournissent beaucoup d'efforts pour la création des processus métier et le rendre plus dynamique en restreignant, autant que possible, l'intervention du développeur dans le choix et la composition des services Web. Ce type de composition est la composition dynamique. Une composition est dynamique quand les services Web qui font partie de la composition sont connus progressivement, ainsi que l'ordre d'exécution nécessaire pour la composition.

Une approche dynamique pour la composition de services offre le potentiel de réaliser des 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. Par conséquent, la composition dynamique de services Web est propice dans un environnement, tel que le Web et l'informatique pervasive où les composants disponibles sont dynamiques et les attentes des utilisateurs variables et personnalisées.

Cependant, une bonne composition des services dépend fortement de la découverte des meilleurs services qu'offrent les fournisseurs.

4. Problématique de la découverte des services Web

L'objectif principal de la technologie des Web services est de fournir l'intégration et l'interaction dynamique des systèmes hétérogènes, et de ce fait, faciliter la coopération rapide et efficace entre les différentes entités dans un environnement collaboratif [EME, 04], [SAH, 05], [ESS, 05].

Pour atteindre ces objectifs, les Web services doivent agir automatiquement avec le minimum d'intervention humaine; ils doivent être capables de découvrir d'autres services qui ont des capacités particulières et réalisent des tâches précises [ESS, 05].

4.1. Méthodes centralisées de découverte

La première génération des travaux de recherche effectuée sur les architectures des services Web, proposent des solutions basées sur les méthodes de découverte centralisées (comme UDDI) où les Web services sont décrits par les interfaces (*signatures*) de leurs fonctions. Dans une telle architecture, les fournisseurs des services Web publient les capacités et les fonctionnalités des ces services dans un enregistrement central.

Cependant, ces méthodes souffrent d'un nombre de problèmes classiques connus dans les systèmes centralisés comme le seul point centralisé (*point d'échec*) et le coût élevé de la maintenance. De plus, ces méthodes ne garantissent pas l'évolutivité (scalabilité) requise pour soutenir un environnement flexible et dynamique [MAN, 07], [HU, 05].

4.2. Découverte distribuée des Web services

D'autres méthodes de découverte et de composition des Web services ont été proposées pour résoudre les limites des méthodes centralisées. Ce type de méthodes est caractérisé par la distribution du mécanisme de publication et de découverte. Elles ont pour objectif principal la composition dynamique des Web services et il existe deux techniques de bases qui sont utilisées pour implémenter les méthodes décentralisées : les Systèmes Multi Agents et les réseaux P2P.

4.2.1. Utilisation des SMA (Systèmes Multi Agents)

Le couplage des Web services et des systèmes multi agents est bidirectionnel. D'une part les agents peuvent utiliser les Web services pour exposer leurs capacités au monde extérieur. Dans ce cas, les SMA hétérogènes utilisent les Web services pour raison d'interopérabilité [SEG, 04]. D'autre part, les SMA sont utilisés généralement pour implémenter la composition chorographique des Web services.

Dans ce type d'architecture, chaque agent est responsable d'un ou de plusieurs Web services. Sont but est la négociation et la communication d'une manière intelligente avec les autres agents, afin d'accomplir un objectif commun à travers la composition de plusieurs services [VAD, 11], [BOU, 07], ces derniers pouvant appartenir à différents fournisseurs.

Cependant, les SMA sont généralement utilisés pour composer les Web services dans un environnement coopératif fermé (pour échanges B2B par exemple) [BRA, 09].

Habituellement, ce type de solutions ne propose pas un mécanisme de découverte des Web services. L'objectif principal des ces solutions est la proposition d'une architecture SMA ainsi que des protocoles et des algorithmes pour composer dynamiquement les Web services.

4.2.2. Utilisation des réseaux P2P

Le deuxième type des méthodes décentralisées dédiées à la découverte et la composition des services Web permet d'implémenter les systèmes P2P. Ces derniers ont été impliqués dans ce contexte parce qu'ils fournissent une alternative évolutive aux systèmes centralisés et ce en distribuant les Web services sur tous les pairs du réseau. De plus, la convergence entre les technologies du Web sémantique, des Web services et du P2P, présente des nombreux avantages, notamment pour le partage des connaissances dans les réseaux à grande échelle. Dans ce contexte, nous allons présenter dans ce qui suit quelques récents travaux de recherche proposés dans ce contexte.

[MAN, 07] ont présenté l'architecture de FLO2WER qui est une plateforme supportant une large collaboration des services Web sémantique dans un contexte P2P hybride et hétérogène. L'approche qu'ils ont adopté exploite les avantages de l'annuaire centralisé, tout en préservant le dynamisme et la scalabilité des réseaux P2P non structuré, l'idée principale derrière la plateforme FLO2WER est que tout en décentralisant les descriptions fonctionnelles des services disponibles à travers le réseau, les descriptions des objectifs de ces services nommés Goals sont gardées centralisé, chaque Goal représente un sous réseau de services spécifiques et est enregistré dans un annuaire approprié appelé Goals Repository. Toutefois, ils n'ont pas décrit comment les Goals sont découverts, en plus leur solution reste vulnérable au phénomène de l'échec du nœud central.

[ZHE, 09] ont proposé une architecture P2P scalable pour la composition de services Web sémantiques appelée P2PSWS, cette dernière combine les avantages des structures centralisées et décentralisées du système P2P, ainsi, les nœuds (pairs) sont divisés en nœuds

locaux (LWP), nœuds de groupes (GWP) et nœuds publics (CWP) et un système de réseaux associatifs orientés contenu (CAN) assure la gestion des services Web dans les nœud publics, les résultats obtenus ont prouvé que la structure P2PSWS aide à surmonter le problème de la vulnérabilité du nœud central et garantie une meilleure composition des SWS en terme d'efficacité et de scalabilité.

Dans leurs travaux, [GHA, 11] ont présenté une approche de découverte et de composition de SWS dans un réseau P2P non structuré, l'objectif principal de leur solution et de composer un service Web répandant à une requête particulière, un service Web composite est constitué d'un ensemble de services Web distribué à travers plusieurs pairs sur le réseau. Afin de permettre la découverte des services Web appropriés des algorithmes épidémiques basés sur un matching Input/Output ont été élaborés, de plus, tous les pairs participants à une composition donnée enregistrent dans une table distribuée toutes les données concernant cette composition, cette table sera utilisé comme un historique pour permettre de fournir des réponses rapides à d'autres requêtes de composition similaires.

D'autres travaux de recherches, s'inscrivant dans ce contexte, sont bien décrits dans [BIA, 06] avec une étude comparative détaillée.

5. Conclusion

La composition de services Web est un point crucial, qui a suscité l'intérêt de nombreux organismes de recherche académiques et industriels. Beaucoup d'efforts ont été fourni afin de permettre une composition utilisable et acceptable de services Web. Ces efforts ont été concrétisés par plusieurs standards et approches de compositions, qui varient de celles qui aspirent à devenir des normes de l'industrie à celles qui sont beaucoup plus abstraites.

La composition de services Web a pour but d'utiliser les compétences de plusieurs services, afin de résoudre un problème qu'aucun ne saurait résoudre individuellement. Le résultat de cette composition est un enchaînement des services Web, qui permet de définir la façon dont les données calculées par les uns sont consommées par les autres. L'objectif de la composition automatique de services Web est de gagner du temps et de faciliter la tâche d'un programmeur, afin de découvrir ou de créer un nouveau service Web complexe.

Après que le Web sémantique ait résolu le problème des standard de descriptions classiques comme WSDL et UDDI, qui ne décrivent pas suffisamment les connaissances d'un Web service, il s'est avéré que la composition dynamique des Web services nécessite un moyen plus efficace qu'UDDI qui ne permet pas d'agréger des Web services d'une manière adaptative et correcte.

Afin de régler ce problème, les systèmes P2P ont été impliqués dans ce domaine. Le paradigme P2P est considéré comme une évolution pour les systèmes distribués qui se concentre sur la gestion des réseaux et le partage des ressources avec une meilleure fiabilité et

scalabilité. Afin de mieux comprendre le concept des systèmes P2P, nous avons consacré le chapitre suivant à cette fin.

Chapitre 4

Aperçu sur les systèmes pair-à-pair.

1. Présentation :

Les systèmes pair à pair[4] se présentent actuellement comme une solution viable pour permettre le partage de ressources à l'échelle de l'Internet. En effet, aussi bien d'un point de vue commercial que scientifique, les architectures pair à pair suscitent un véritable engouement. Le paradigme du P2P garantit un fonctionnement à large échelle. Un très grand nombre de pairs peut interagir dans le réseau, de manière à permettre le partage d'une grande quantité de ressources. Aussi appelé d'égal à égal, chaque participant à un système P2P peut être à la fois client et serveur. Le fonctionnement du système ne repose sur aucune coordination centralisée. Ainsi, le comportement global du réseau résulte uniquement des interactions locales entre les pairs qui se connectent et se déconnectent.

Le succès des systèmes P2P est justifiable grâce à des points qui sont considérés comme des inconvénients des systèmes Client/Serveur et peut se résumer en :

- Le nombre de clients, le téléchargement et la demande en bande passante grandissante peuvent amener les serveurs à ne plus servir d'avantage de clients.
- Une caractéristique de cette architecture (Client/Serveur), est qu'elle nécessite très peu de puissance de calcul et de ressources du côté client, ce qui a été justifié à l'ère où cette architecture a vu le jour, alors qu'à notre ère, la plupart des machines clientes sont devenus très puissantes.
- Le client dans une telle architecture agit de manière passive : il demande des services auprès des serveurs mais il est incapable d'offrir des services aux autres.

Contrairement à l'architecture client/serveur, un réseau P2P ne dépend pas d'un serveur central pour accéder à un service quelconque. Le réseau P2P évite l'organisation centralisée de l'architecture client/serveur, en adoptant une organisation horizontale, fortement connectée et permettant aux machines à connexion intermittente de se retrouver sur le réseau en agissant tantôt en client et tantôt en serveur. L'avantage principal de réseaux P2P est qu'ils distribuent les responsabilités de fournir des services sur l'ensemble des pairs du réseau.

En plus, ces réseaux exploitent mieux la bande passante en utilisant différents canaux de communications. Le problème des goulots d'étranglement au niveau des serveurs fortement sollicités ne se pose pas dans les réseaux P2P, car plusieurs pairs peuvent fournir des services et router d'autres pairs aux fournisseurs de services similaires afin de réduire la congestion du réseau.

[4] Plusieurs appellations peuvent être utilisées pour identifier le concept peer-to-peer : P2P, Pair à Pair, Egal à Egal et Poste à Poste

2. Définition

Plusieurs définitions ont étés attribués au concept P2P dont voici une des plus significatives et complètes :

« Le P2P est une classe de systèmes ***auto-organisants*** qui permettent de réaliser une fonction globale16 de manière décentralisée en tirant parti du partage des ressources et de la puissance de calcul disponibles sur un grand ***nombre d'extrémités d'Internet*** » [RIC, 00].

Les termes *« auto-organisants, grand nombre, extrémités de l'internet »* expriment clairement les trois principaux enjeux du paradigme P2P qui sont (respectivement) : *l'autoorganisation, scalabilité et volatilité des pairs.*

- **L'auto-organisation :** Qui se traduit par le fait que le réseau n'est sous l'emprise d'aucune administration ou contrôle centralisé. Les pairs communiquent et s'organisent entre eux de manière autonome et coopérative. Vu l'étendue des réseaux et la diversité des plates-formes (système d'exploitation et équipements réseaux) une gestion du réseau est nécessaire à cause de :
  - L'hétérogénéité des plates-formes ;
  - L'accès aux ressources à distance (contourner les Firewalls et les NAT[5]).

- **La scalabilité :** Qui veut dire qu'un véritable réseau P2P devrait être capable de garder les mêmes performances en terme de localisation de données, de routage de messages et d'insertion de nouveaux pairs quelque soit le nombre de pairs composant le réseau P2P.

- **La volatilité des pairs :** Qui est synonyme d'un problème de disponibilité des ressources. En effet, le fait que les pairs deviennent détenteurs des ressources, il faudrait s'assurer de la disponibilité de ces ressources en absence des pairs qui les détiennent (chose qui est prise en charge par le serveur dans les réseaux client/serveur) ce qui constitue l'un des défis majeurs des développeurs des applications P2P (surtout pour le calcul distribué), dont il faut prendre en compte:
  - Les déconnexions violentes des pairs ;
  - La découverte automatique des ressources nouvellement disponibles sur le réseau.

[5] **NAT :** Network Address Translation

3. Architecture des systèmes P2P

Il existe trois grandes familles d'architecture des systèmes P2P, qui sont : l'architecture *centralisée*, l'architecture *décentralisée* et l'architecture *hybride.*

3.1. Architecture centralisée

Ce type d'architecture repose sur un serveur central sur lequel se connectent les utilisateurs et qui permet la gestion des partages, la recherche et l'insertion d'informations. Chaque utilisateur recherche un service désiré sur le serveur. Ce dernier lui indique alors la liste des pairs sur lesquels il est susceptible de le trouver. L'utilisateur se connecte alors directement à l'un de ces pairs pour l'exécution du service. Ce qui est très similaire au processus de découverte-sélection et d'invocation des services web en utilisant un simple UDDI.

Cette façon de procéder est très fragile puisque le serveur central est indispensable au réseau. Ainsi, si le serveur est débranché, à la suite d'une panne (par exemple) c'est tout le réseau qui s'effondre. Le réseau Napster est un très bon exemple de cette architecture.

Les avantages de cette architecture sont :

- L'efficacité de la recherche par une indexation centralisée ;
- La facilité de la mise en œuvre ;
- La mise à jour en temps réel de l'index central.

En contre partie

- le réseau est complètement dépendant du serveur central qui est une source de vulnérabilité;
- et l'anonymat n'est pas garanti ;

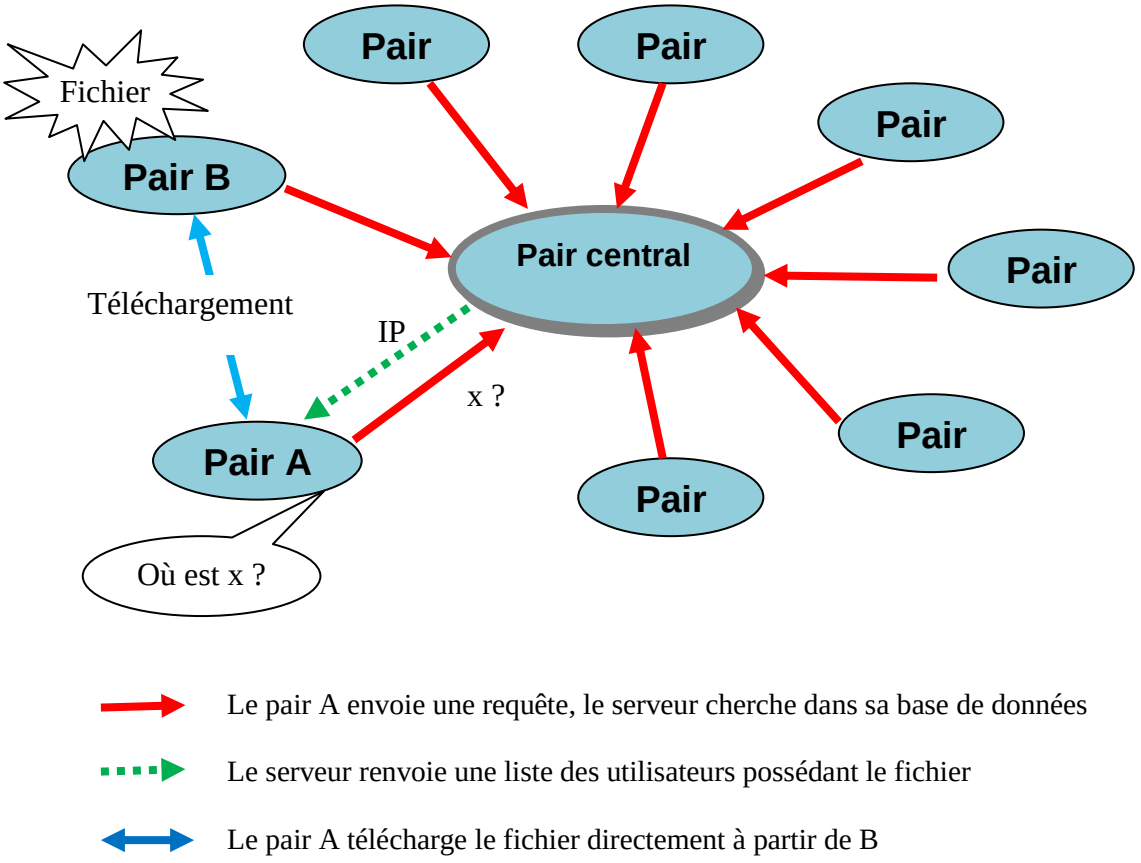


Figure 8. Architecture P2P centralisée

3.2. Architecture décentralisée

C'est l'architecture P2P pure où il n'y a pas de serveur central et tous les pairs ont un rôle équivalent. Les membres du réseau se découvrent dynamiquement en relayant les requêtes de proche en proche (Figure 9.). Un pair transmet une requête à ses voisins qui font de même et ainsi de suite (notion d'inondation) jusqu'à arriver à destination. Pour limiter la génération d'un nombre exponentiel des messages par ce procédé, on limite la propagation des messages jusqu'à un horizon défini par un paramètre de durée de vie de la requête, dit : « TTL[6] » qui est décrémenté sur chaque pair. Une fois un pair possédant la ressource est trouvée, une connexion directe s'établit entre les pairs impliqués.

Ce modèle a pour avantages la non vulnérabilité des pairs, du moment qu'il est un système pair à pair pur, ainsi que l'anonymat des utilisateurs. En contrepartie, il génère beaucoup de trafic. Cependant, des protocoles optimisés ont pu être mis en place, basés sur les tables de hachage distribuées permettant de réaliser des recherches en un nombre de messages croissant de façon logarithmique en fonction du nombre des pairs du réseau comme CAN, Chord, Freenet, Tapestry et Pastry [SOY 05], [DOY 05]. Les réseaux Gnutella sont un très bon exemple de cette architecture.

[6] **TTL :** Time To Live

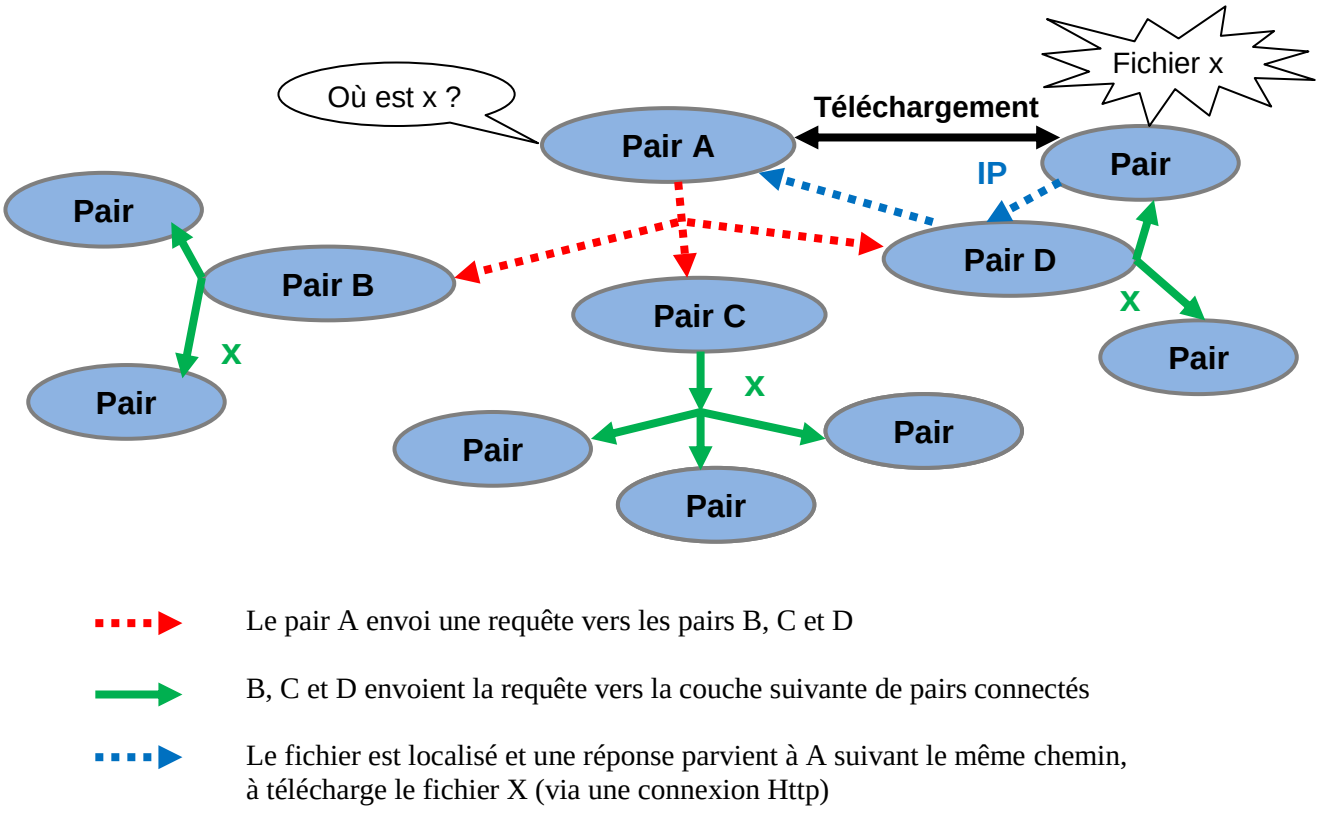


Figure 9. Architecture P2P décentralisée (réseau GNUtella)

3.3. Architecture hybride

Une solution hybride consiste à utiliser des Superpeers (super pairs), éléments du réseau choisis en fonction de leur puissance de calcul et de leur bande passante. Ces derniers gèrent un groupe de postes, jouent le rôle d'intermédiaire dans la transmission des requêtes et réalisent des fonctions utiles au réseau, comme l'indexation des informations. Lorsqu'un poste se connecte, il est affecté à l'un de ces groupes.

Les serveurs sont reliés entre eux suivant un mode décentralisé. Chaque serveur référence le contenu de son groupe, si un serveur ne possède pas le fichier (la ressource) demandé par un des pairs de son groupe, il transmet à ce dernier l'adresse d'un autre super pair où sera renouvelée la requête. Cette solution, rend le réseau un peu moins robuste que dans un réseau de type Gnutella. Cependant, si un super-pair tombe en pane, son groupe de pairs sera déconnecté au reste du réseau et non plus le réseau tout entier comme le cas d'un réseau Napster. Cette architecture est illustrée dans la figure 10.

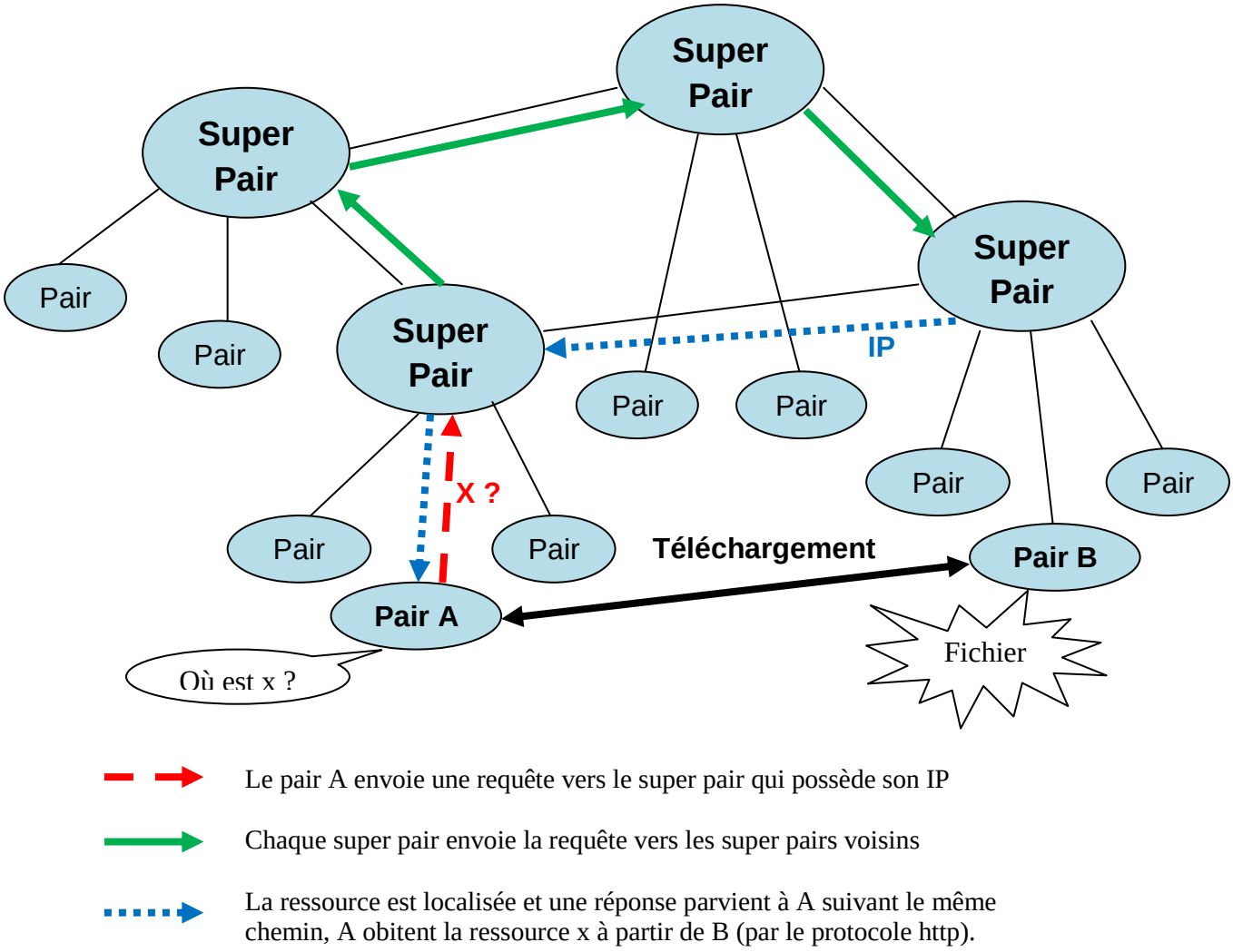


Figure 10. Architecture P2P Hybride

4. Plateformes de développement d'application P2P

La plupart des logiciels P2P existants ont été développés d'une manière spécifique sans se référencer à des standards propres au P2P. Des initiatives ont été entreprises pour offrir un tel standard. Elles proposent un environnement permettant de développer des applications P2P en offrant les fonctionnalités de base telles que la gestion des pairs, des groupes de pairs, l'indexation des ressources, la découverte des ressources, la communication entre pairs et autres aspects comme la sécurité.

Ainsi, *Sun* propose sa plateforme P2P *JXTA*[7] basée sur les technologies Java et XML, Microsoft de sa part, intègre dans sa plate-forme *.Net* des outils pour développer des

[7] JXTA pour JuXTApose -- https://jxta.dev.java.net.

applications P2P, tels que les *services web*, les *services processes*[8], *Windows Forms*[9] et *Web Forms*[10], Une autre importante plateforme est celle de *Jini*[11] qui délivre un ensemble de mécanismes (un ensemble d'APIs sous forme d'interfaces Java) qui permettent aux pairs de s'interconnecter et de fournir des services pour les membres du réseau.

- **JXTA vs Microsoft.Net :** Le développement des applications P2P avec l'architecture .NET a comme principal désavantage par rapport à JXTA de ne pas être créé spécifiquement pour ce genre d'applications et demande de la part des programmeurs une plus grande charge de travail. En effet, l'utilisateur doit, par exemple, utiliser les services web pour construire une application P2P. JXTA propose par contre une architecture étendue et complète spécifique aux systèmes P2P. De plus, cette architecture est en pleine expansion et intègre de nombreux concepts propres aux systèmes P2P.
- **JXTA vs Jini [KUR, 06] :** Même si les deux plateformes partagent les mêmes objectifs de permettre une organisation et une découverte dynamique des ressources d'un réseau en respectant l'ubiquité et l'interopérabilité entre les dispositifs membres du réseau, ils diffèrent par le fait que Jini est intimement lié au langage de programmation Java. En effet, contrairement à JXTA qui défini un ensemble de protocole pour publier et rechercher des ressources dans le réseau, dont l'implémentation peut être faite avec n'importe quelle langage de programmation, Jini défini un ensemble d'API sous forme d'interfaces Java qui implémentent les concepts utilisés pour manipuler les services.

  Autre inconvénient de Jini par rapport à JXTA est l'utilisation du modèle de données de Java, donc pour faire passer ces données dans le réseau, Jini est obligé d'encapsuler ces données dans des objets Java. Par contre, JXTA s'appuie sur les documents XML pour représenter les données, et dans ce cas un simple parsing[12] du document est suffisant.

(Pour plus de détails concernant la plateforme JXTA, voir Annexe C).

5. Conclusion

Dans ce chapitre, nous avons présenté l'un des types d'applications distribuées appartenant à la deuxième génération des architectures réparties. Pratiquement, après le grand succès de la première génération présentée par l'architecture client/serveur, cette dernière a rencontré plusieurs problèmes liés beaucoup plus à la centralisation des ressources de stockage et de traitement dans la partie serveur.

[8] Permettent de gérer la découverte des pairs lorsque le protocole HTTP n'est pas utilisé.
[9] Permettent d'écrire des interfaces utilisateurs graphiques permettant à un utilisateur de s'authentifier, d'effectuer des recherches et de partager du contenu.
[10] Permettent de retourner facilement le contenu des pairs sous forme HTML.
[11] http://www.jini.org
[12] Le **parsing** est le traitement syntaxique d'un document.

Pour régler les problèmes de l'architecture client/serveur et pour assurer des nouvelles fonctionnalités au niveau des applications distribuées, le paradigme P2P a vu le jour. Il est basé sur l'égalité des nœuds d'un réseau, où tous les pairs jouent le même rôle dans une architecture plate, chacun d'entre eux peut être client et/ ou serveur en même temps. Cette caractéristique de base a permis aux systèmes P2P d'être impliqués dans plusieurs types d'applications distribuées : calcul distribué, diffusion de contenu, collaborations, moteurs de recherches et plateformes.

Le projet JXTA a été conçu en respectant la norme SOA (Service-Oriented Architecture) et comprend plusieurs services de haut niveau qui facilite la création et l'utilisation des applications P2P. En plus, JXTA intègre un nombre de protocoles générique qui permet à n'importe quel dispositif connecté au réseau de communiquer et de collaborer en tant que pair.

Dans le chapitre suivant, nous allons décrire notre approche qui utilise des technologies du Web sémantique et celles des systèmes P2P pour offrir une solution permettant la découverte et la composition dynamiques des Web services sémantiques.

Deuxième partie

La Contribution

Chapitre 5

Approche P2P hybride pour une composition automatique des SWS

1. Introduction

Les services Web sont considérés comme des implémentations de SOA, ils sont devenus incontournables pour la réalisation d'applications réparties sur le Web. La technologie des services Web est, aujourd'hui, largement utilisée comme support de l'interopérabilité entre les applications distribuées, qui fonctionnent indépendamment des caractéristiques conceptuelles (architecture, modèle d'interaction, …), et des caractéristiques techniques (plateforme, langage de programmation, …), afin de réaliser une fonctionnalité établie auparavant.

Nous avons aussi expliqué le bénéfice d'utilisation du Web sémantique pour décrire les différents services Web fournis. Les langages de description sémantique issus (OWL-S, WSMO,…) permettent de décrire les connaissances liées au domaine d'application d'un Web service, ce qui rend sa manipulation plus intelligible et exploitable par la machine. Cette caractéristique offre un atout pour la découverte automatique des Web services et implicitement pour sa composition dynamique.

Après que le Web sémantique ait résolu le problème des standards de descriptions classiques comme WSDL et UDDI, qui ne décrivent pas suffisamment les connaissances d'un Web service, il s'est avéré que la composition dynamique des Web services nécessite un moyen plus efficace qu'UDDI qui ne permet pas d'agréger des Web services d'une manière adaptative et correcte.

Afin de régler ce problème, les systèmes P2P ont été impliqués dans ce domaine. Le paradigme P2P est considéré comme une évolution pour les systèmes distribués qui se concentre sur la gestion des réseaux et le partage des ressources avec une meilleure fiabilité et scalabilité. Les systèmes P2P ont été inclus dans plusieurs domaines de recherche comme le travail collaboratif et la découverte des ressources pertinentes y compris les Web services. Les systèmes P2P fournissent une alternative évolutive aux systèmes centralisés en distribuant les Web services sur tous les pairs du réseau. Ainsi chaque pair peut être à la fois demandeur et fournisseur de services, c'est pourquoi les systèmes P2P fournissent un environnement idéal pour l'implémentation des services Web [JIN, 10].

Dans cette partie du mémoire, nous allons présenter une architecture P2P hybride pour la découverte et la composition automatique des services Web sémantiques, cette dernière combine les avantages des structures centralisées et décentralisées d'un système peer-to-peer. La découverte des services Web participants, leur composition ainsi que leur interopération doivent être effectuées de façon automatique et transparente à l'utilisateur. Notre approche implémente une table distribuée, dite « table de composition », pour préserver la trace des Web services déjà composés dans le réseau. Cette table vise à accélérer le temps de recherche des services Web à travers la réutilisation des compositions déjà réalisées.

2. Approche P2P hybride pour la découverte et la composition automatique des SWS

Nous avons vu dans le chapitre précédent, que les plus importants travaux de recherche menés dans le domaine de la composition des services Web sémantique dans les plateformes P2P ont été concentrés principalement sur deux aspects :

- La résolution du problème du serveur central (UDDI) qui représente le maillon faible des services Web dans les plateformes centralisées ;
- La structure à adopter pour les plateformes P2P afin que ces dernières garantissent la stabilité et la scalabilité du réseau lorsque le trafic des messages (requêtes/réponses) est important entre les paires.

Dans cette section, nous allons d'abord motiver l'utilisation des réseaux P2P dans notre solution. Ensuite, nous présentons les deux idées de base de cette solution qui sont l'architecture P2P hybride dédiée à la découverte et la composition des Web services sémantiques, et la table de composition.

2.1. Motivations de l'utilisation des réseaux P2P

La décision d'utiliser les réseaux P2P dans notre travail est justifiée par plusieurs raisons :

- Pour plus de deux décennies, les systèmes P2P ont prouvé leur efficacité en terme de robustesse et de scalabilité comme des systèmes distribués pour l'échange d'informations ;
- Les avantages de la combinaison des structures centralisées et décentralisées qu'offre le modèle P2P hybride en termes de partage, de réduction des coûts et de tolérance aux pannes.

2.2. Description générale de l'approche

Rappelons que l'objectif principal de cette solution est de fournir un scénario décentralisé qui supporte la composition automatique des services Web sémantiques distribués sur un réseau P2P. A partir d'un réseau hybride, nous voulons composer un Web service qui répond à une requête particulière. Ce service peut être composé de plusieurs services appartenant à différents pairs. L'objectif est de créer un espace collaboratif entre les différents pairs participants à la réalisation d'un but commun.

Dans cette première étape de notre travail, nous présentons notre architecture de découverte et de composition des Web services dans un réseau P2P hybride. L'idée principale est d'organiser les pairs en deux types de groupe appelés : *Pairs Locaux (PL)* et *Pairs Groupes (PG)* (*que nous détaillons dans la section suivante)*, une telle organisation permet de limiter le flot de messages

circulant dans le réseau et ainsi garantir la stabilité et la scalabilité de tout le système. Aussi, dans cette solution, nous avons utilisé une table dite « *table de composition* » afin de préserver la trace du chemin de composition pour une éventuelle future réutilisation. Cette table est considérée comme une mémoire cache utilisée pour accélérer la recherche des Web services déjà composés.

Dans notre approche, *JXTA* a été choisie comme plateforme P2P et *WSMX* comme environnement d'exécution des services Web sémantiques.

(Pour plus de détails concernant la plateforme JXTA et l'environnement WSMX, voir Annexe C et D).

2.2.1. Pairs Locaux (PL) :

Ces pairs ont les particularités d'être privés, sont désignés statiquement et forment un groupe avec les pairs adjacents. Les pairs dans le même groupe ont un intérêt commun et sont issus généralement d'une même organisation. Ces paires peuvent être demandeur ou fournisseurs de services et ne communiquent qu'avec le Pair Groupe de leur propre groupe.

2.2.2. Pairs Groupes (PG) :

De chaque groupe de pairs on choisit un super pair qui sera appelé *Pair Groupe (PG)*, ce dernier en plus de son rôle de demandeur et fournisseur de service, gère la communication entre les pairs de son groupe et communique avec les autres *PG* sur le réseau. Chaque Pair Groupe contient une copie de la table de composition (*voir figure 11*). L'ensemble des PG constitue un groupe public.

2.2.3. Table de composition

Dans l'approche proposée, nous avons utilisé une table -dite table de composition distribuée sur tous les pairs groupes du réseau. Elle est constituée d'un ensemble d'attributs qui décrivent es différentes compositions réalisées dans le réseau *(voir tableau 2).*

Pair-Initiateur	ID-Compo	Init-Input	Init-Output	But	Services Web Exécutés	Pairs Précédents	Pairs Suivants
PL1	0001	…	…. ….	… ….	……		
PG2	0002	…..	…..	… ……	…..	…	…
…	….	……. …	…..	…. …	.		

Tableau 2. Table de composition

- **Pair-Initiateur :** Le premier pair dans l'enchainement (qui exécute le (les) premier Web service) est appelé le pair initiateur (le pair initiateur est un pair participant).
- **ID-Compo :** l'identificateur de composition. Chaque composition a un seul identifiant composé de deux valeurs (Pair-Initiateur, N°-composition). Lorsque deux compositions ont le même pair initiateur, elles ont automatiquement deux numéros différents, parce que le numéro de composition est généré par le pair initiateur lors de la sauvegarde de la composition.
- **Init-Input :** l'entrée initiale du service Web composite.
- **Init-Output :** la sortie initiale du service Web composite.
- **But (Goal) :** le but du service Web. Un but est une conceptualisation du domaine des services dont les objectifs finaux sont identiques ou similaires. Le but d'un Web service est spécifié dans sa description sémantique.
- **Services Web Exécutés :** sont les services Web exécutés localement par un pair donné pour composer un Web service dans le réseau.
- **Pairs Précédents :** les pairs qui exécutent les services Web précédents dans une composition.
- **Pairs Suivants :** les pairs qui exécutent les services Web suivants dans une composition

3. Architecture de la solution proposée

L'idée principale de l'approche est de fournir une découverte et une composition des services Web sémantique dans une structure P2P hybride, dans ce contexte, la table de composition distribuée sur chacun des pairs groupes présente une mémoire cache qui sauvegarde les traces des différentes compositions réalisées avec succès.

3.1. Module de composition locale

L'objectif de cette couche est de découvrir ou de composer un service Web local qui répond à une requête du demandeur, elle est exécutée grâce au moteur de composition inclus dans WSMX. Si cette opération est possible, on n'a pas besoin de retransmettre la requête sur tout le réseau, sinon on passe à la composition P2P.

3.2. Module de composition P2P

Ce module présente une interface entre le pair et le réseau P2P. C'est le composant principal qui gère la communication et permet d'établir des compositions collaboratives entre les différents pairs participants.

L'architecture générale de notre approche est représentée dans *la figure 11*.

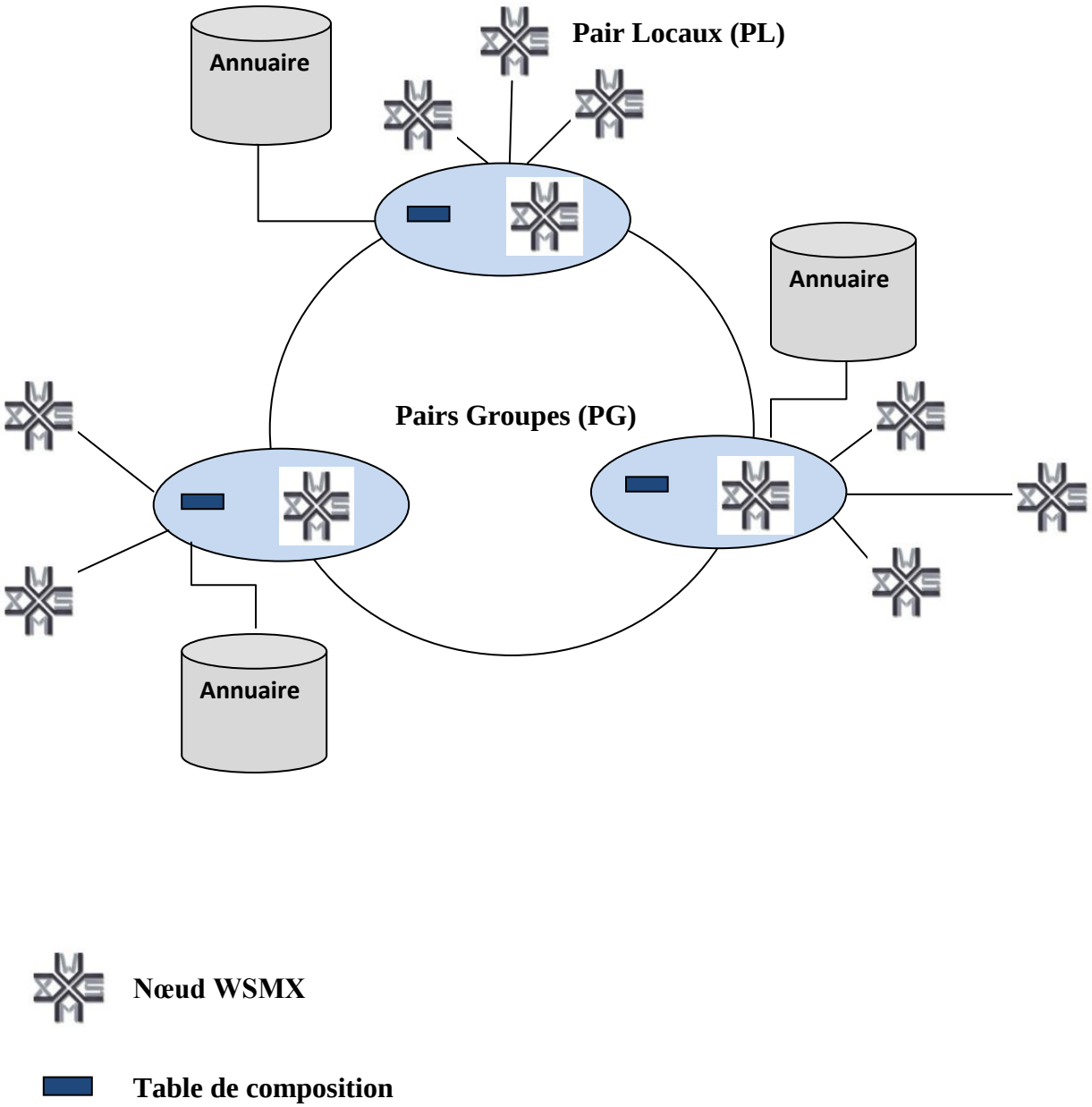


Figure 11. Architecture de l'approche proposée

4. Scénario de l'exécution de l'approche proposée pour une composition automatique des SWS dans une plateforme P2P hybride

Pour bien comprendre le principe de la solution proposé, nous avons jugé utile de présenter la séquence des événements d'un simple scénario d'exécution d'une composition d'un SWS :

Etape 1 : un *Pair Local* (nœud WSMX) reçoit une requête de composition dans un message WSML *(Init-Input, Init-Output, Goal).*

Etape 2 : ce pair retransmet la requête à son *Pair Groupe*

Etape 3 : une fois que le PG ait reçu la requête, il effectue une recherche dans son propre annuaire *(voir Algorithme 1).*

Etape 4 : si le résultat est positif *(c.-à-d. il existe des services locaux qui satisfont la requêtes)*, alors le PG envoie un message de réponse à l'expéditeur avec la liste des pairs concernés, et il enregistre les informations de la composition effectuée dans la table de composition et la diffuse à tous les pairs groupes. Sinon il passe à l'étape suivante

Etape 5 : le *PG* effectue une recherche dans la table de composition pour tenter de trouver des compositions précédentes qui lui sont similaires *(voir Algorithme 2).*

Etape 6 : si une correspondance est trouvée dans la table de composition alors le PG renvoie la réponse au pair expéditeur, sinon il passe à l'étape suivante.

Etape 7 : le PG diffuse la requête de composition à tous les autre PG du réseau en utilisant le mécanisme de diffusion de JXTA (*Propagation Pipes).*

Début

1. Le pair groupe reçoit une requête de type « search (init-Input, Init- Output, But) ».
2. Chercher un service Web local qui répond à la requête ;
3. **Si** (il existe un WS local) **alors**

   Envoyer la réponse favorable; Aller à **Fin ;**
4. **Si** (il n'existe pas un WS local) **alors**

   Essayer de composer un WS localement pour répondre à la requête.
5. **Si** (c'est possible de composer un WS localement) **alors**

   Envoyer la réponse favorable; aller à **Fin ;**
6. **Si** (il n'existe pas une composition locale qui répond à la requête) **alors**

   Search-in-Composition-Table (Init-Input, Init-Output, But);
7. **Si** (il n'existe pas une composition dans la table) **alors**

   Diffuser la requête aux autres pairs groupes ;

Fin.

Algorithme 1. Recherche des services Web locaux

L'opération de recherche dans la table de composition présentée dans l'étape 6 de l'algorithme précédent *(Search-in-Composition-table (Init-Input, Init-Out-put, But)* est implémentée par l'algorithme suivant :

Search-in-Composition-Table ()

Début

1. **Select** [Pair-Initiateur, ID-Compo] **from** la table de composition
   where [(l'entrée initiale de la composition=Init-input) **&** (la sortie initiale de la composition=Init-Output) **&** (le but de la composition=But)] /*le résultat de cette sélection est une liste nommée selection-list*/
2. **SI** (selection-list est non vide) **alors**
3. **Tant que** (selection-list est non vide) **Faire**
4. Sélectionner un Pair-Initiateur de la liste selection-list;
5. Envoyer un message: search (ID-Compo, Init-Input, init-Output, But) au Pair-Initiateur;

Fin tant que ;

Fin.

Algorithme 2. Recherche dans la table de composition

Afin de mieux illustrer le concept de la table de composition et son rôle dans l'approche proposée, nous considérons un exemple d'un service Web composite qui calcule le prix d'un livre en Dinar Algérien (DA) en entrant son titre et le nom de son auteur, Ce service Web est composé de quatre services (SW1, SW2, SW3 et SW4) basiques distribués sur le réseau (voir figure 12.), chaque service est déployé par un pair (PL1, PL2, PG1 et PL5 respectivement).

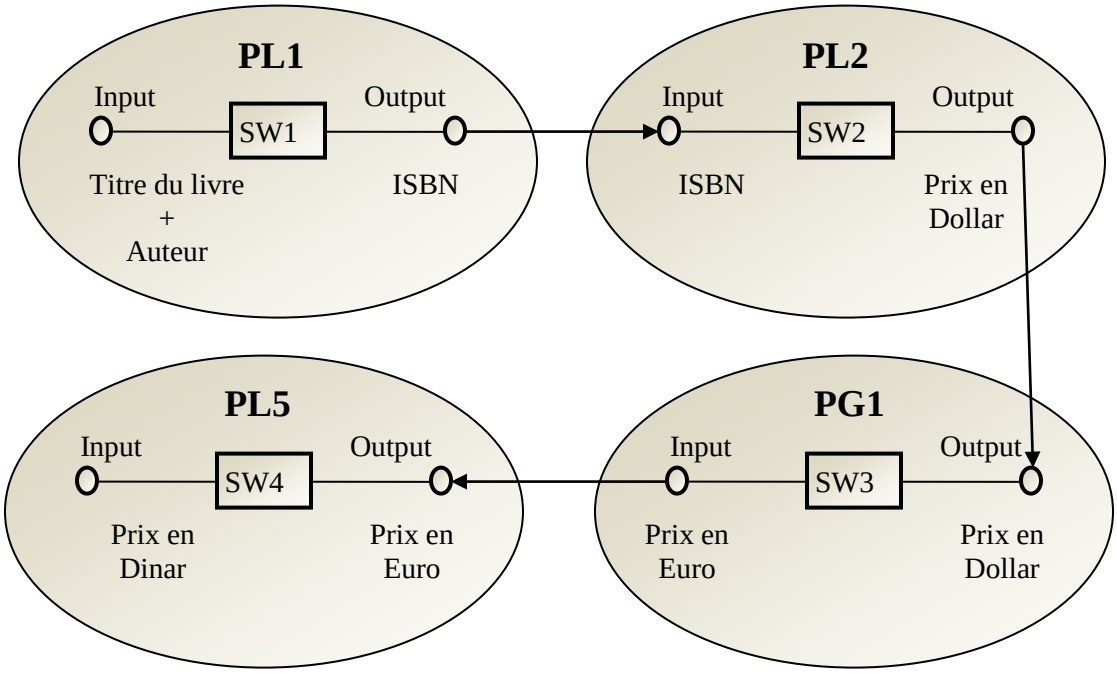


Figure 12. Exemple d'un service Web composite calculant le prix d'un livre

L'opération de découverte des différents services Web est représentée dans la figure 13.

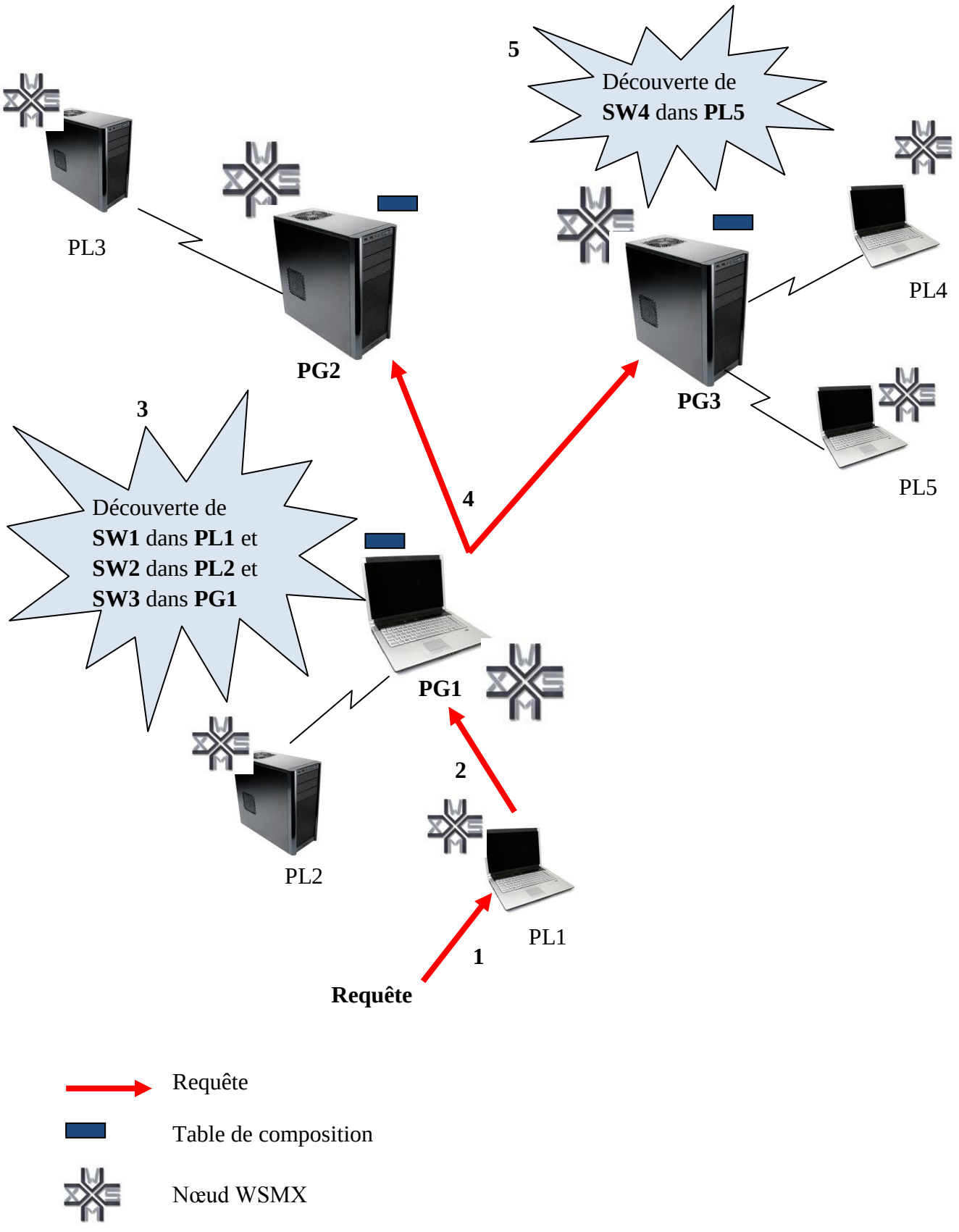


Figure 13. Exemple d'une composition de services Web dans un réseau P2P hybride.

Suivant cet exemple, la ligne encadrée en rouge dans le *tableau 3* présente la composition sauvegardée par le pair **PL2** *(figures 12 et 13).*

Pair-Initiateur	ID-Compo	Init-Input	Init-Output	But	Services Web Exécutés	Pairs Précédents	Pairs Suivants
PL1	0001	…	…. ….	… ….	……		
				…		…	…
PL2	0002	Titre Auteur	Prix	Calculer le prix du livre en DA	SW2	PL1	PG1
…	….	……. …	…..	…. … ….	…..	…	….
PL7	0010	…	…..	…. ……	……	….	…

Tableau 3. Contenu de la table de composition (exemple du pair PL2)

5. Discussion

Comme toute solution, notre proposition présente des avantages et inconvénients que nous pouvons les résumer comme suit :

5.1. Avantages de l'approche proposée

Dans notre solution proposée de composition automatique de SWS dans un environnement P2P, nous avons surmonté deux inconvénients majeurs :

- **La vulnérabilité du système :** qui se traduit par le problème du serveur central qui pourrait affecter tout le réseau s'il tombe en panne. Pour contourner ce problème nous avons pensé à une architecture hybride qui exploite les avantages des structures centralisée et décentralisée
- **La surcharge du réseau par les messages :** pour remédier à ce problème, notre solution était d'organiser les pairs en Pairs Locaux et Pairs Groupes, ainsi, s'il est

possible d'effectuer une composition localement (au sein d'un groupe), le réseau ne sera pas submergé, ce qui garantit une meilleure stabilité et scalabilité du système entier.

On peut dire que l'organisation en groupes permet de mieux structurer les pairs WSMX en communautés, ce qui permet une meilleure exploitation des ressources du réseau, et éviter l'inondation de tout le réseaux avec des messages (requêtes/réponses).

La table de composition permet de créer des espaces collaboratifs, ainsi chaque pair participant sauvegarde son historique des compositions et peut exploiter les expériences des autres pairs ce qui représente un gain de temps et de ressources.

5.2. Inconvénients

Cependant, notre solution proposée pose deux problèmes principaux :

- *Problème de la Qualité de Services :* la nécessité de la définition d'un ensemble de métriques permettant la sélection des services découverts.
- *Le non dynamisme des pairs :* tous les pairs sont créés statiquement et joints à leurs groupes, et notre approche ne prend pas en considération l'augmentation du réseau par d'autres pairs *(c.-à-d. l'inscription de nouveaux pairs)*, ainsi que la réduction du réseau *(c.-à-d. lorsque des pairs quittent le réseau ou tombent en panne).*

6. Conclusion

Les avantages apportés par les technologies du Web sémantique à l'opération de découverte des Web services sont bien réels mais un frein leur est imposé par l'UDDI qui ne supporte que la recherche basée sur des informations de haut niveau spécifiques aux entreprises et aux services. En effet, UDDI ne reçoit pas les spécificités des capacités des services pendant le mapping sémantique (Machmaking) et ne saisit pas les relations entre les entités dans son répertoire et n'est donc pas capable de faire usage de l'information sémantique pour déduire les relations en cours de la recherche.

Pour régler ces problèmes, les réseaux P2P sont devenus des solutions incontournables comme des plateformes pour les services Web sémantiques grâce à leur caractère fortement distribué. Dans ce chapitre, nous avons présenté une solution hybride pour la découverte et la composition des services Web sémantique qui combine les avantages des structures centralisées et distribuées. L'idée principale derrière notre approche est de diviser les pairs en deux segments : *pairs locaux* et *pairs groupes*, une telle organisation des pairs garantie une meilleur stabilité et scalabilité de tout le réseau.

Afin d'accélérer le temps de recherche et d'optimiser le passage à l'échelle dans le réseau, nous avons implémenté une table de composition permettant de préserver la trace des compositions déjà réalisées dans le réseau. Cette table nous permet de réutiliser les

expériences des différents pairs participants, et de créer un espace de collaboration entre les pairs fournissant des services Web.

Chapitre 6

Implémentation et mise en œuvre

1. Introduction

L'objectif essentiel de ce chapitre est la réalisation d'un prototype de l'approche proposée pour la découverte et la composition automatique des services Web sémantiques pour la plateforme WSMX.

Pour implémenter ce prototype, nous avons essentiellement fait appel à deux plateformes : la plateforme JXTA pour implémenter une infrastructure de communication P2P et la plateforme WSMX pour la définition et la description des services Web sémantiques (selon l'approche WSMO).

En plus de ces plateformes, nous avons fait appel à plusieurs outils techniques qui sont tous « libres » et d'autres « open sources ». Dans la section suivante, nous présentons brièvement quelques outils que nous avons utilisés dans notre travail

2. Présentation des outils technologiques utilisés

a) Langage

Le langage de développement utilisé est le langage Java dans sa version d'entreprise « Java EE 5 SDK ». Le Kit Java EE 5 SDK peut être téléchargé à partir du lien : http://java.sun.com/javaee/

b) Serveurs

- Nous avons choisi le serveur http « Apache 2.2 », car il est léger et très simple d'installation et d'utilisation, le serveur Apache 2.2 est disponible depuis le lien : http://apache.org
- L'autre serveur important pour implémenter notre application est le serveur « Axis 2 » d'Apache Foundation, *Axis* est utilisé comme serveur SOAP pour les services Web, il est téléchargeable gratuitement à partir du site : http://axis.apache.org

c) Système de gestion de base de données

Nous avons choisi le SGBD Java DB version 10.4 inclus dans la Java SE Development Kit, il est disponible sur le site : http://developers.sun.com/javadb/

d) Environnement de développement

Nous avons opté pour Eclipse comme environnement de développement dans sa version *Helios Service Release 1,* téléchargeable à partir de : http://www.eclipse.org.

3. Implémentation de l'architecture P2P proposée

Dans cette section, nous allons présenter les détails techniques de l'implémentation de notre architecture P2P hybride de composition automatique des services Web sémantiques pour la plateforme WSMX.

3.1. Création des Pairs Groupes

Le package *net.jxta.peergroup.** de la plateforme JXTA fournit des classes pour créer des pairs groupes, le code suivant illustre la création de ce type de pairs.

```
PeerGroupID myNewPeerGroupID = (PeerGroupID)
net.jxta.id.IDFactory.newPeerGroupID();
```

3.2. Création des Pairs Locaux

De la même façon, mais avec le package *net.jxta.peer.**, nous pouvons créer des pairs locaux, comme illustré dans les lignes suivantes :

```
PeerID myNewPeerID = (PeerID)
net.jxta.id.IDFactory.newPeerID();
```

4. WSMXEntryPoints

La plateforme WSMX n'offre pas d'API pour invoquer ses modules à partir d'un code, mais elle offre la possibilité de l'utiliser en tant qu'un serveur à part (*Stand-alone server*) en utilisant ses *WSMXEntryPoints*.

Comme son nom l'indique, WSMXEntryPoints est une classe Java de WSMX qui offre des points d'entrées (interfaces) pour accéder aux fonctionnalités importantes de la plateforme WSMX.

Cependant pour une éventuelle utilisation sans avoir besoin de Java pour recompiler toute la plateforme, les développeurs de WSMX ont pensé à offrir aux autres développeurs qui souhaitent utiliser cette plateforme la possibilité d'interagir avec un serveur WSMX comme un simple service web.

Le fichier de description WSDL de ce service web est disponible depuis l'URL « *http://Nom_de_la_machine:8050/axis/services/WSMXEntryPoints?wsdl* », avec une instance en exécution de la plateforme WSMX sur la machine. Ce fichier permet de lister les méthodes publiques disponibles ainsi qu'une description de leurs paramètres, comme il fournit des détails techniques sur la manière d'accéder au service.

Eclipse permet d'avoir une description graphique de ce fichier WSDL, ceci est représenté dans la figure 14.

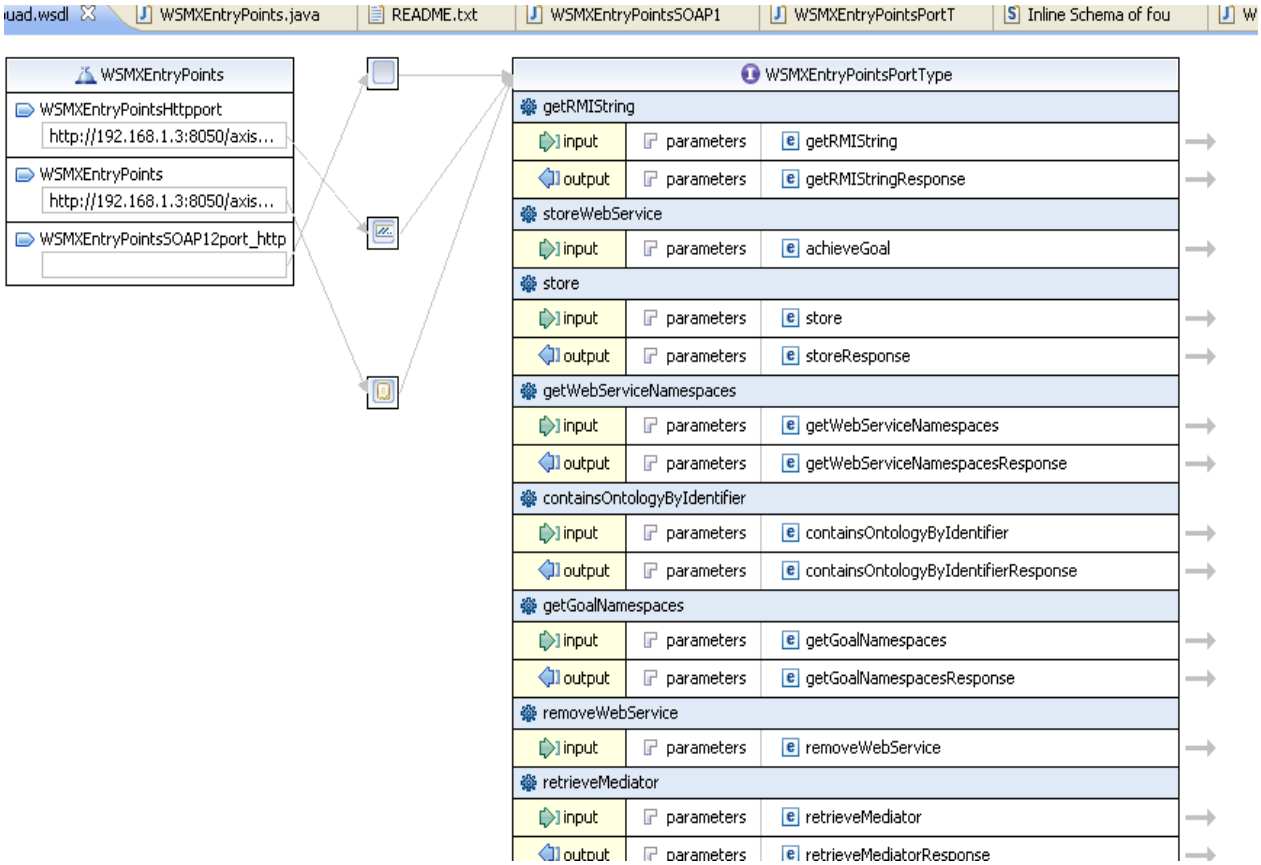


Figure 14. Description graphique du ficher WSDL de WSMXEntryPoints

Voici ci-dessous une concise description de quelques importantes méthodes du service WSMXEntryPoints :

- **DiscoverWebServices**(*String* ***wsmlGoal***) : Prend en paramètre un but (désirs du demandeur de services) écrit en WSML et qui retourne un ensemble de services web correspondants.
- **AchieveGoal**(*String* ***wsmlGoal***) : permet de faire les trois processus de découverte, de sélection et d'exécution des services Web qui correspondent au but donné en paramètre.

- **InvokeWebService**(*String* ***webService,*** *String* ***wsmlGoal***) : permet d'invoquer le service Web qui figure dans le premier paramètre. Le deuxième paramètre est un but d'utilisateur avec des instances pour l'exécution du service.
- **Store**(*String* ***wsmlEntity***) : permet d'enregistrer localement n'importe quelle entité définie en WSML.

Dans la version actuelle de WSMX le *WSMXEntryPoints* compte en tout 45 méthodes. Les 41 méthodes qui ne sont pas décrites ici permettent essentiellement de donner des informations complémentaires sur les différentes entités enregistrées (services Web, ontologies, buts et médiateurs).

Pour utiliser le service *WSMXEntryPoints* depuis notre code Java, nous avons utilisé un programme qui vient avec Axis qui se nomme « *WSDL2Java* ». Ce dernier permet de générer à partir d'un fichier WSDL d'un service des classes Java qu'on peut utiliser pour accéder directement aux méthodes de ce service.

5. Préparation d'un nœud WSMX

Après avoir installé la plateforme JXTA (configuré les pairs et les connecté au réseau) et implémenté notre prototype, il reste à ajuster les paramètres spécifiques du service « *WSMXEntryPoints* » de la plateforme WSMX d'un nouveau pair coopérant dans notre approche et à faire joindre ce dernier au groupe de nœuds WSMX.

Pour faire intégrer les paramètres du service « WSMXEntryPoints » (pour communiquer avec la plateforme WSMX), il suffit de récupérer le ficher de description WSDL de l'URL : « *http://Nom_de_la_machine:8050/axis/services/WSMXEntryPoints?wsdl* » et l'enregistrer dans le répertoire du prototype. Une fois en possession du fichier de description, une exécution du programme (*Axis*) *WSDL2Java* générera un package « *com.wsmx.client* ». Ce package contient des classes Java utilisées pour accéder à la plateforme WSMX du pair.

Cependant, la manière la plus simple pour faire joindre le pair au groupe d'appartenance des nœuds WSMX est de spécifier le nom du groupe dans les lignes suivantes :

```
WSMXGroupID = createPeerGroupID('jxta:uuid-De
l'annonce du groupe des nœuds WSMX') ;

WSMXGroup = createPeerGroup(WSMXID,
'WSMXPeerGroup', 'Experimentation Group') ;

joinGroup(WSMXGroup) ;
```

6. Transfert de données

Une fois que la découverte ait été faite sur les noeuds WSMX qui ont reçu le message de la requête, les résultats sont maintenant prêts pour être transmis au demandeur de services.

Comme JxtaCast ne supporte pas les communications en unicast, nous avons eu besoin de concevoir un mécanisme de communication spécifique d'un pair à un pair pour faire véhiculer les messages des résultats.

Les pipes bidirectionnels (Bidirectionnel Pipes) sont un des moyens possibles dans ce cas pour une communication en unicast entre deux pairs. Cependant, la spécification de la plateforme JXTA n'offre pas une gestion de tels moyens de communication. Un développeur doit créer et gérer un pipe bidirectionnel à partir de deux « Unicast pipes » et faire paramétrer les écouteurs sur chacune des deux extrémités (voir figure 14).

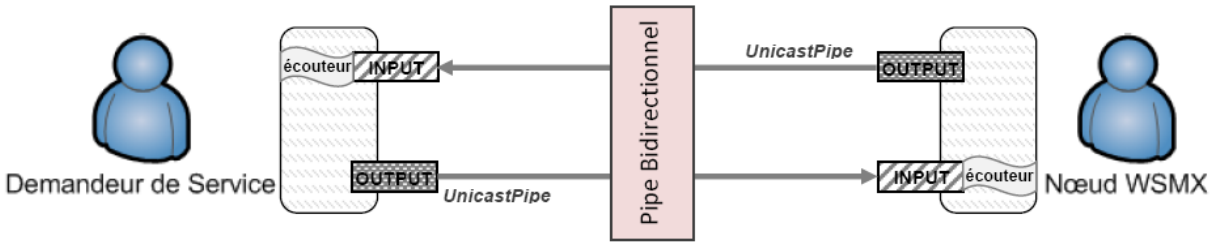


Figure 15. Pipe bidirectionnel JXTA

7. Conclusion

Dans la première partie de ce chapitre, nous avons présenté les différents outils techniques utilisés dans cette étape, ensuite, nous avons présenté l'implémentation et le déploiement du prototype pour notre approche proposée.

En examinant de près la manière dont nous avons implémenté notre prototype, un pair WSMX, en disposition de ce prototype n'a qu'à fournir un package correspondant à son service *WSMXEntryPoints* pour qu'il coopère dans le processus de découverte et de composition des SWS selon l'approche proposée.

Enfin, il faut mentionner que la phase d'implémentation nécessite une étape d'évaluation de performances de la solution proposée, en utilisant un simulateur des réseaux P2P comme PeerSim ou NS2.

Conclusion et perspectives

1. Conclusion

Dans ce mémoire, nous avons présenté une solution qui répond à la problématique de découverte et de composition des services Web sémantiques pour la plateforme WSMX. L'objectif principal est de proposer une approche pour la découverte des services Web et de concevoir une architecture permettant de concrétiser cette approche. Il est clairement démontré que la composition des services Web est fortement liée à l'opération de découverte considérée comme essentielle au bon fonctionnement des services Web.

Dans un premier temps, nous avons proposé une architecture basée sur le modèle hybride qui combine les avantages des structures centralisée et distribuée, les pairs du réseau (*nœuds WSMX*) ont été organisés en deux types de pairs : pairs locaux et pairs groupes, l'objectif de cette étape est de créer un espace collaboratif entre les pairs afin de répondre à une requête du demandeur du service Web tout en préservant la stabilité et la scalabilité du système.

La deuxième idée proposée dans cette stratégie est l'utilisation de la table de composition qui fournit une méthode distribuée pour découvrir les services Web composés dans un réseau P2P. Cette table se caractérise par la préservation des traces des différentes compositions déjà effectuées avec succès pour une éventuelle future réutilisation, ce qui permet d'améliorer le temps de découverte à travers la réutilisation des services Web déjà composés au lieu de lancer une nouvelles recherche sur le réseau.

Nous avons justifié notre choix de la plateforme WSMX comme notre banc d'essai en trois points. (1) WSMX est une implémentation de référence de l'approche WSMO qui offre un model complet et précis pour la description des services et des requêtes des utilisateurs. (2) WSMX couvre tout le cycle de vie d'un service, sert d'un annuaire pour ses propres services, et coopère avec ces semblables comme des systèmes distribués. (3) la plateforme WSMX offre la possibilité d'interagir avec d'autres systèmes qui ne sont pas des nœuds WSMX grâce à son module « *WSMX Adapters* », ce qui nous permet de simuler une coopération des systèmes non-WSMX en utilisant des nœuds WSMX.

Nous avons aussi justifié notre choix d'une approche P2P par le fait que les systèmes P2P ont prouvé leur efficacité comme des systèmes distribués pour l'échange d'informations et comme des solutions scalables pour plus de deux décennies. Pour toutes ces raisons, plusieurs travaux de recherche proposent la décentralisation de la publication et la recherche des Web services à travers la proposition de méthodes distribuées de découverte essentiellement sur des plateformes P2P.

2. Perspectives

Dans notre travail, nous avons pris en considération les problèmes de la sémantique, de la découverte et de la composition dynamique des services Web. Néanmoins, notre approche nécessite plus d'améliorations en terme de critères de qualité de service (*QoS*). L'enrichissement de la découverte par des critères de sélection permettrait d'améliorer le résultat de la recherche. De plus, il serait intéressant de rajouter un mécanisme de représentation de la qualité de service au niveau de la requête de découverte.

A cause de la difficulté de l'implémentation de cette solution dans un réseau à grande échelle, une étape de simulation est primordiale pour présenter ce type de réseau. Il est judicieux d'étudier l'utilisation de simulateurs des réseaux P2P comme NS2 ou PeerSim, afin de simuler le fonctionnement de notre framework dans un réseau P2P.

Nous envisageons aussi de faire intégrer des mécanismes de sécurité dans le prototype développé, en faisant appel aux mécanismes de sécurité de JXTA : *groupes sécurisés* et *pipes sécurisés*, pour sécuriser toutes les communications.

Comme perspectives à long terme, nous pouvons améliorer notre approche en fournissant plus de dynamisme aux pairs, c'est-à-dire, chaque pair est libre de joindre ou de quitter le réseau tout en gardant l'efficacité du système. De plus, nous voulons étudier la possibilité de regrouper les pairs suivant leurs contenus, cette proposition permet de résoudre des problèmes rencontrés dans notre solution.

Bibliographie

[ABE, 05] Marie-Hélène Abel «Ontologies pour le Web Sémantique et le e-learning», La Journée thématique « Web sémantique pour le e-learning », Du 30 mai à 03 juin 2005

[ALB, 97] P. Albers, Extension de la représentation pour la prise en des effets dépendant du context et des axiomes du domaine, Thèse de Doctorat, Université de Paul Sebatier de Toulouse, 1997.

[ALK, 08] Aniss Alkamari, Composition de services Web par appariement de signatures, Mémoire présenté comme exigence partielle de la maîtrise en informatique, Université du Québec à Montréal, Janvier 2008.

[ALT, 07] Altaf Hussain Bouk, Shanping Li and Nizamuddin Channa « Comparative Study of Semantic web Services » Journal of Information & Communication Technology Vol. 1, No. 1, (Spring 2007) 01-10, 2007.

[ARE, 06] Nerea Arenaza, Composition semi-automatique des Web services, Projet de Master, Ecole Polytechnique Fédérale de Lausane, Février 2006.

[ARR, 04] S. Arroyo and M. Stollberg « WSMO Primer. WSMO Deliverable D3.1, DERI Working Draft » Technical report, WSMO, 2004. http://www.wsmo.org/2004/d3/d3.1/.

[BA, 08] Cheikh BA, Composition de services Web avec PEWS approche par la théorie des traces, Thèse de Doctorat, Université François Rabelais Tours, 24 Novembre 2008.

[BAB, 03] Gilbert Babin, et Michel Leblanc, Les Web Services et leurs compact B to B, CIRANO, Centre Interuniversitaire de Recherche en Analyse des Organisations, Août 2003.

[BAT, 10] S. Batra, S. Bawa "Review of Machine Learning Approaches to Semantic Web Service Discovery", Journal of Advances in Information Technology (JAIT) vol. 1, no. 3, August 2010.

[BEN, 05] BENAYACHE Ahcene « construction d'une mémoire organisationnelle de formation et évaluation dans un contexte E-Learning : le projet MEMORAE », Thèse de Doctorat, l'université de technologie de Compiegne, 2005

[BER, 97] Berners-Lee Tim « Metadata Architecture » http://www.w3.org/DesignIssues/Metadata.html, Jan. 1997.

[BIA, 06] D. Bianchini, V.D Antonellis, M. Melchiori, D. Salvi, *"Peer-to-Peer Semanticbased Web Service Discovery: State of the Art"*, ESTEEM: Emergent Semantics and cooperaTion in multi-knowledgE EnvironMents, 2006. Disponible à www.dis.uniroma1.it/esteem/docs/deliverable%20soa/SoA_UNIBS.pdf

[BOR, 97] Borst W. N. «Construction of Engineering Ontologies », In Center for Telematica and Information Technology, University of Tweenty, Enschede, NL. 1997

[BOU, 03] A. Bouguettaya, A. Elmagarmid, et B. Medjahed, Semantic Web enabled composition of Web services, The VLDB Journal, Volume 12, pages 333-351, Novembre 2003.

[BOU, 04] Sabri Boutemedjet « Web Sémantique et e-Learning » Cours IFT6261, université de Montréal, 2004.

[BOU, 07] J. Bourdon, Multi-agent systems for the automatic composition of semantic Web services in dynamic environments, Mémoire de maîtrise, Ecole national Supérieur des mines en saint-Etienne, 2007.

[BRA, 09] M. Brahimi, L. Seinturier, M. Boufaïda: "*Multi-Agent Architecture for Developing Cooperative E-Business Applications*", International Journal of Information Systems and Supply Chain Management (IJISSCM), volume 2, N°4: pp.43-62, 2009.

[BRO, 96] Alan W. Brown, Component-Based Software Engineering : Selected Papers from the Software Engineering Institute, IEEE Computer Society Press, Los Alamitos, CA, USA, 1996.

[BRU 05] Bruijn J., Lausen H., Krummenacher R., Polleres A., Kifer M. and Fensel D. « Deliverable D16.1v0.2, The Service Web Modeling Language WSML » Final Draft, WSMO project, http://www.wsmo.org/TR/d16/d16.1/v0.2/20050320/, 2005.

[BRY, 04] Sean Brydon, et Beth Stearns, Designing Web Services with the J2EE 1.4 Platform JAXRPC, SOAP, and XML Technologies, édition Addison Wesley, le 9 Juin 2004.

[BUC, 06] Antonio Bucchiarone, et Stefania Gnesi. A survey on services composition languages and models, In Antonia Bertolino and Andrea Polini, editors, in Proceedings of International Workshop on Web Services Modeling and Testing (WS-MaTe2006), pages 51-63, Palermo, Sicily, Italy, le 9 Juine 2006.

[BUL, 03] T. Bultan, X. Fu, R. Hull, et J. Su, Conversation Specification : A New Approach to Design and Analysis of E-Service Composition, Proceedings of the 12th International World Wide Web Conference (WWW 2003), ACM, 2003.

[BUS, 04] Bussler C. et al., « WSMO » The Eleventh International Conference on Artificial Intelligence: Methodology, Systems, 2004.

[CAB, 05] L. Cabral and J. Domingue « Mediation of semantic Services Web in irs-iii » In First International Workshop on Mediation in Semantic Services Web (MEDIATE 2005), 2005.

[CAS, 01] F. Casati, M. Sayal, et M.-C Shan, Developping e-services for composing e-services, In Proceedings of 13th International Conference on Advanced Information Systems Engineering (CAiSE), Interlaken, Switzerland, June 2001.

[CHA, 07] Y. Charif, et N. Sabouret, Coordination in Introspective MultiAgent Systems, Proceeding of the International Conference on Intelligent Agent Technology (IAT'07), pages 412-415, Silicon Valley, California, USA. 2007.

[CHA, 10] Pierre Châtel. Une approche qualitative pour la prise de décision sous contraintes nonfonctionnelles dans le cadre d'une composition agile de services, Thèse de Doctorat de l'université Pierre et Marie Curie Paris 6, de l'Ecole Doctorale d'Informatique, Télécommunication et Electronique de Paris, 2010.

[CHE, 02] Z. Cheng, M. Singh, et M. Vouk, Composition constraints for semantic web services, In Proceedings of the WWW-2002 Workshop, 2002.

[CHE, 04] Issam Chebbi, CoopFlow une approche pour la coopération ascendante de workflows dans le cadre des entreprises virtuelles, Thèse de Doctorat, Institut National des Télécommunications dans le cadre de l'école doctorale SITEVRY en co-accréditation avec l'Université d'Évry Val d'Essonne, le 17 Avril 2007.

[CRU, 03] Crubezy M, Fensel D, Benjamins R, Wielinga B, Motta E, Musen M, Omelayenko B « Upml: The language and tool support For making the semantic Web alive » Technical report, MIT, 2003.

[DAV, 04] N.J. Davies, D. Fencel, and M. Richardson « The future of Services Web » BT Technology Journal, 22(1), Jan 2004.

[DOY, 05] Guillaume Doyen, « Supervision des réseaux et services pair à pair », dans sa thèse de doctorat soutenue à l'université de Henri Poincaré- Nancy 1 , pp. 29-46, 12 décembre 2005.

[EME, 04] F. Emekci, O.D. Sahin, D. Agrawal, and A. El Abbadi, "A Peer-to-Peer Framework for Web Service Discovery with Ranking", *In the Proc of the IEEE International Conference on Web Services (ICWS'04)*, California USA, 2004, pp. 192-199.

[ERI, 04] Eric van der Vlist « LE TRIPTYQUE SOAP/WSDL/UDDI » Web Services Convention, Juin 2004.

[ESS, 05] T. ESSAFI, N. DORTA and D. SERET, "A Scalable Peer-to-Peer Approach To Service Discovery Using Ontology", *In the Proc of 9th World Multiconference on Systemics, Cybernetics and Informatics.* Orlando, 2005.

[EUZ, 04] Jérôme Euzenat - Raphaël Troncy « Web sémantique et pratiques documentaires » ftp://ftp.inrialpes.fr/pub/exmo/publications/euzenat2004e.pdf, 2004.

[FEN, 02] D. Fensel et C. Bussler « The Service Web Modeling Framework » Electronic Commerce Research and applications, 1(2):113-137, 2002.

[GEO, 95] D. Georgakopoulos, M. Hornick, et A. Sheth, An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure, Distributed and Parallel Databases, An International Journal, Volume 3, 1995.

[GHA, 04] M. Ghallab, D. Nau, et P. Traverso, Automated Planning : Theory and Practice, edition Morgan Kaufmann, 2004.

[GHA, 11] Mohamed Gharzouli, Mahmoud Boufaida, « PM4SWS: A P2P Model for Semantic Web Services Discovery and Composition », Journal of Advances in Information Technology, vol. 2, no. 1, february 2011.

[GOM, 04] Asuncion Gomez-Pérez, Mariano Fernandez-Lopez et Oscar Corcho « Ontological Engineering: with examples from the areas of Knowlodge Management, e-commerce and the Semantic Web » Edition Springer, 2004.

[GOM, 99] Gomez Pérez A., Benjamins V.R. « Overview of Knowledge Sharing and Reuse Components: Ontologies and problem-Solving Methods ». Proceeding og the IJCAI-99, workshop on Ontologies and problem-Solving Methods (KRR5), Stockholm (Suède), pp.1.1-1.15. 1999.

[GRU, 93] Gruber T. «A Translation Approach to Portable Ontology Specifications », Knowledge Acquisition, 5(2): pp. 199-220. 1993.

[HEN, 02] J. Hendler, B. Parsia, et E. Sirin, Semi-automatic composition of Web services using semantic descriptions, In Proceedings Web Services : Modeling, Architecture and Infrastructure. Workshop in Conjunction with ICEIS2003, ICEIS Press, pages 17-24, Angers, France, 2002.

[HEN, 03] J. Hendler, D. Nau, B. Parsia, E. Sirin, et D. Wu, Automating DAML-S web services composition using SHOP2, In ISWC'03, 2003.

[HOL, 04] David Hollingsworth, The Workflow Reference Model 10 Years, édition Workflow Handbook, 2004.

[HU, 05] J. Hu, C. Guo, H. Wang, and P. Zou, "Web Services Peer-to-Peer Discovery Service for Automated Web Service Composition", Springer-Verlag Berlin Heidelberg (LNCS 3619), ICCNMC 2005, Zhangjiajie China, 2005, pp. 509-518.

[HUB, 03] Hubert Kadima & valérie Monfort « Les web services », Livre, Edition DUNOD 2003

[JIN, 10] Baohua Jin, Dongyao Zou, « Research of Web Service based on P2P », IEEE Second International Conference on Communication Software and Networks, 2010, pages 412-415.

[KEL, 03] Patrick Kellert et Farouk Toumani « les Services Web sémantiques » Research Report LIMOS/RR 03-15, 11 Juillet 2003.

[KOW, 05] R. Kowalczik, et I. Muller, Service composition through agent-based coalition formation, In Proceedings of WWWService composition with SemanticWeb Services, pages 34-43, Compiegne, France, September 2005.

[KUN, 03] P. Kungas, M. Matskin, et J. Rao, Application of linear logic to Web service composition, In Proceeding of the 1st International Conference on Web Services, Las Vegas, USA, June 2003.

[LAU, 02] Philippe Laublet, Chantal Reynaud, Jean Charlet « Sur quelques aspects du Web sémantique », http://www.lalic.paris4.sorbonne.fr/stic/articles/03-WebSemantique.pdf , 2002 (Consulter le 19/10/2006).

[MAN, 07] F. Mandreoli, A. M. Perdichizzi, and W. Penzo, "A P2Pbased Architecture for Semantic Web Service Automatic Composition", IEEE computer Society DOI 10.1109/DEXA2007, Regensburg Germany, 2007, pages. 429-433.

[MCL 02a] S. Mcilraith, et S. Narayanan, Simulation, verification and automated composition of Web services. Dans la 11 ème conference international du World Wide Web (WWW2002) Honolulu, Hawaii, Mai 2002.

[MCL, 02b] S. McIlraith, et T. Son, Adapting Golog for composition of semantic Web services, In Proceedings of the 8th International Conference on Knowledge Representation and Reasoning, Morgan Kaufmann Publishers, pages 482-493, Toulouse, France, April 2002.

[MEL, 04] Tarek Melliti, Interopérabilité des Services Web complexes, Thèse de Doctorat, Université Paris IX Dauphine, le 8 Décembre 2004.

[MET, 05] METEOR-S, « METEOR-S project deliverables » http://lsdis.cs.uga.edu/Projects/METEOR-S/, 2005.

[MOT, 03] Enrico Motta, John Domingue, Liliana Cabral, and Mauro Gaspari « IRS-II: A Framework and Infrastructure for Semantic Services Web » In Proceedings of the 2nd International Semantic Web Conference (ISWC 2003), 2003 http://kmi.open.ac.uk/people/domingue/papers/irs-iswc-03.pdf

[MRI 07] Michaël Mrissa « Médiation Sémantique Orientée Contexte pour la Composition de Services Web » Thèse de doctorat, Université Claude Bernard Lyon I, 2007.

[MRI, 07] Michael Mrissa, Médiation Sémantique Orientée Contexte pour la Composition de service Web, Thèse de Doctorat, Université Claude Bernard Lyon I, France, UFR Informatique, le 15 Novembre 2007.

[OWL, 04] The OWL services Coalition « OWL-S: Semantic Markup for Services Web » http://www.w3.org/Submission/OWL-S/ , 2004.

[PAT, 04] Patrick Kellert et Farouk Toumani « Les Web services sémantiques », Article 2004 Laboratoire LIMOS - UMR (6158) du CNRS ISIMA - Campus des Cezeaux - B.P. 125 63173 AUBIERE Cedex

[PON, 08] Julien Ponge. Model Based Analysis of Time-aware Web Services Interactions. Thèse de Doctorat de l'Université Blaise Pascal - Clermont-Ferrand II, dans le cadre de l'Ecole Doctorale des Sciences pour l'Ingénieur, France, 2008.

[POU, 07] Frédéric Pourraz, Diapason une approche formelle et centrée architecture pour la composition évolutive de services Web, Thèse de Doctorat, LISTIC : Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance, le 10 Décembre 2007.

[RAH, 05] Rahee Ghurbhurn « Introduction aux Web Services », Master Web Intelligence, 2005

[RAT+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network", In Proc of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, ACM SIGCOMM, California, USA, 2001, pp. 161-172.

[REG, 04] P. Regnier, Algorithmique de la Planification en Intelligence Artificielle, édition Cepadues, 2004.

[RIC, 00] Bruno Richard, « Les technologies pair à pair », Hewlett packard (Laboratoire Grenoble), 2000.

[ROW, 01] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems", In the Proc of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), Heidelberg, Germany, 2001.

[SAH, 05] O. D. Sahin, C. E. Gerede, D. Agrawal, A. El Abbadi, O. Ibarra, and J. Su, "SPiDeR: P2P-Based Web Service Discovery", *In the Proc of Third International Conference on Service-Oriented Computing (ICSOC)*, Amsterdam, The Netherlands, 2005, pp. 157-169.

[SEG, 04] A.E.F Seghrouchni, S. Haddad, T. Melitti, A. Suna, "*Interopérabilité des systèmes multi-agents à l'aide des services Web* ", les Journées Francophones sur les Systèmes Multi-Agents, Paris, France, pp. 91-104, 2004.

[SHA, 07] Omair Shafiq, Matthew Moran, Emilia Cimpian, Adrian Mocan, Michal Zaremba and Dieter Fensel « Investigating Semantic Service Web Execution environments: A comparison between WSMX and OWL-S tools » Second International Conference on Internet and Web Applications and Services, IEEE, 2007.

[SOF, 07] Web services, http://www.softeam.fr/technologies_web_services.php, 2007.

[SOT, 02] Juan Carlos Soto, "Project JXTA: Project JXTA: An Open P2P Applications Platform An Open P2P Applications Platform Introduction and Update". Disponible depuis : http://www.cs.rpi.edu/academics/courses/fall02/netprog/notes/jxta/jxta.pdf.

[SOY, 05] Olivier Soyez, « Stockage dans les systems pair à pair », dans sa thèse de doctorat soutenue à l'université de Picardie Jules verne d'Amiens, novembre 2005. Disponible depuis : http://oliviersoyez.free.fr.

[STO, 01] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", In Proceedings of the ACM SIGCOM, California, USA, 2001, pp.149-160.

[TEM, 07] Nacéra Temglit « Un modèle de composition des services Web sémantique » Mémoire de magister, université USTHB, 2007.

[VAN, 06] Kurt Vanmechelen, Jan Broeckhove, Gunther Stuer,T om Dhaene. "Jini and JXTA Based Lightweight Grids", (Chapitre 13 du livre "Engineering The Grid: Status and Perspective", pp 4-11 by American Scientific Publishers,2006,ISBN: 1-58883-038-1. Disponible depuis : www.comp.ua.ac.be/publications/files/Jini_JXTA_Chapter13.pdf.

[VAD, 11] G. Vadivelou, E.IIavarasan, S. Prasanna, "*Algorithm for Web Service Composition using Multi-Agents*", International Journal of Computer Applications (0975 – 8887),Volume 13– No.8, January 2011.

[W3C, 04] W3C, « Services Web », http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/ , 2004.

[WAT, 00] R.Waltinger,Web agents cooperating deductively, In Proceeding of FAABS 2000, Greenbelt, volume 1871 of Lecture Notes in Computer Sciences, Springer-Verlag, pages 250-262, April 2000.

[WSD, 05] WSDL-S, « Service Web Semantics - WSDL-S, Version 1.0 » W3C Member Submission, http://www.w3.org/Submission/WSDL-S/, 2005.

[ZAC, 06] Grégory Zacharewicz, Un environnement G-DEVS/HLA : application à la modélisation et simulation distribuée de Workflow, Thèse de Doctorat, Université de Paul Cezanne Aix- Marseille III, page 35-55, 2006.

Annexe A : les standards des services Web

1. Le protocole SOAP

Le protocole SOAP est l'un des produits du W3C, il assure des appels de procédures à distance (RPC). Il assure l'interaction entre les web services en permettant le transport de paquets de données sous format XML, ceci à l'aide du HTTP[13], SMTP[14] et POP[15].

Il comporte trois composantes principales qui sont : un framework de messagerie, un standard d'encodage et le mécanisme RPC. [HUB, 03]

1.1. Le framework de messagerie SOAP

Le framework de messagerie SOAP requiert que le message SOAP soit composé d'une enveloppe qui contient un Header et un Body. Le Header comporte les métas – données du message et le Body comporte le corps du message lui-même.

1.2. Encodage / sérialisation standard pour les objets

Le standard d'encodage / sérialisation permet l'encodage des objets dans des messages SOAP d'une manière standard ainsi que leurs décodage de manière standard au niveau du destinataire.

1.3 Invocation de service d'objets distant via SOAP RPC

SOAP permet à l'aide de XML, la réalisation d'un appel RPC. XML n'est pas utilisé pour transporter des documents mais pour véhiculer des appels de procédure et leur résultat.

Les différentes étapes d'invocation d'un objet distant sont illustrées dans le schéma suivant :

[13] HTTP : HyperText Transfer Protocol.

[14] SMTP : Simple Mail Transfer Protocol (Protocole simple de transfert de courrier), est un protocole de communication utilisé pour transférer le courrier électronique vers les serveurs de messagerie électronique.

[15] POP : Post Office Protocol (*le protocole du bureau de poste*), est un protocole qui permet de récupérer les courriers électroniques situés sur un serveur de messagerie électronique.

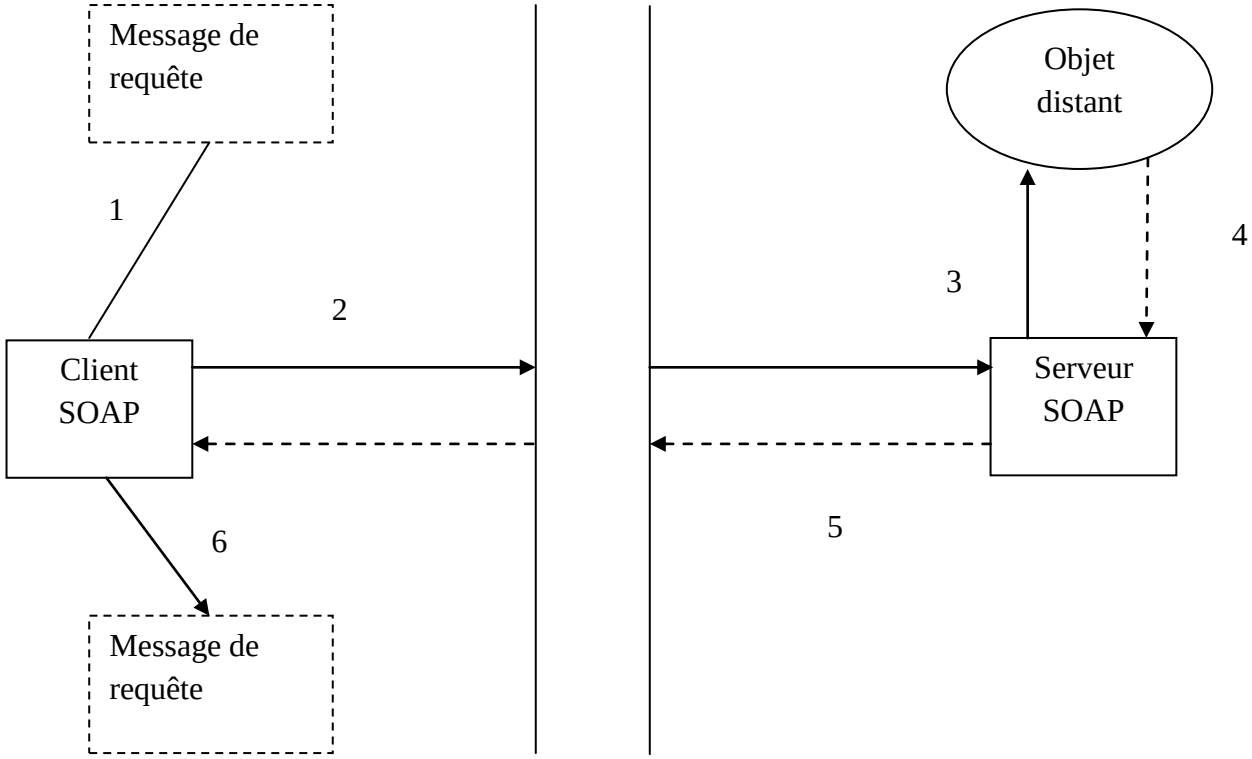


Figure 16. Etapes d'invocation d'objets distants avec SOAP. [HUB, 03]

1. Le client SOAP crée un document XML qui contient les informations nécessaires pour invoquer les services d'un objet distant. Ensuite ce document est inséré dans une enveloppe SOAP avant d'être transmis sous forme d'une requête HTTP.
2. Le message est transmis via une connexion HTTP.
3. Le message est reçu et analysé par le serveur SOAP, ensuite envoyé à l'objet distant.
4. L'objet fait le traitement de la requête et envoie la réponse au serveur SOAP.
5. La réponse est envoyée, sous forme d'un document SOAP, au client via le HTTP.
6. Le client reçoit la réponse, ouvre l'enveloppe et envoie le résultat au demandeur initial.

1.4. Structure d'un message SOAP

La structure générale d'un message SOAP est la suivante :

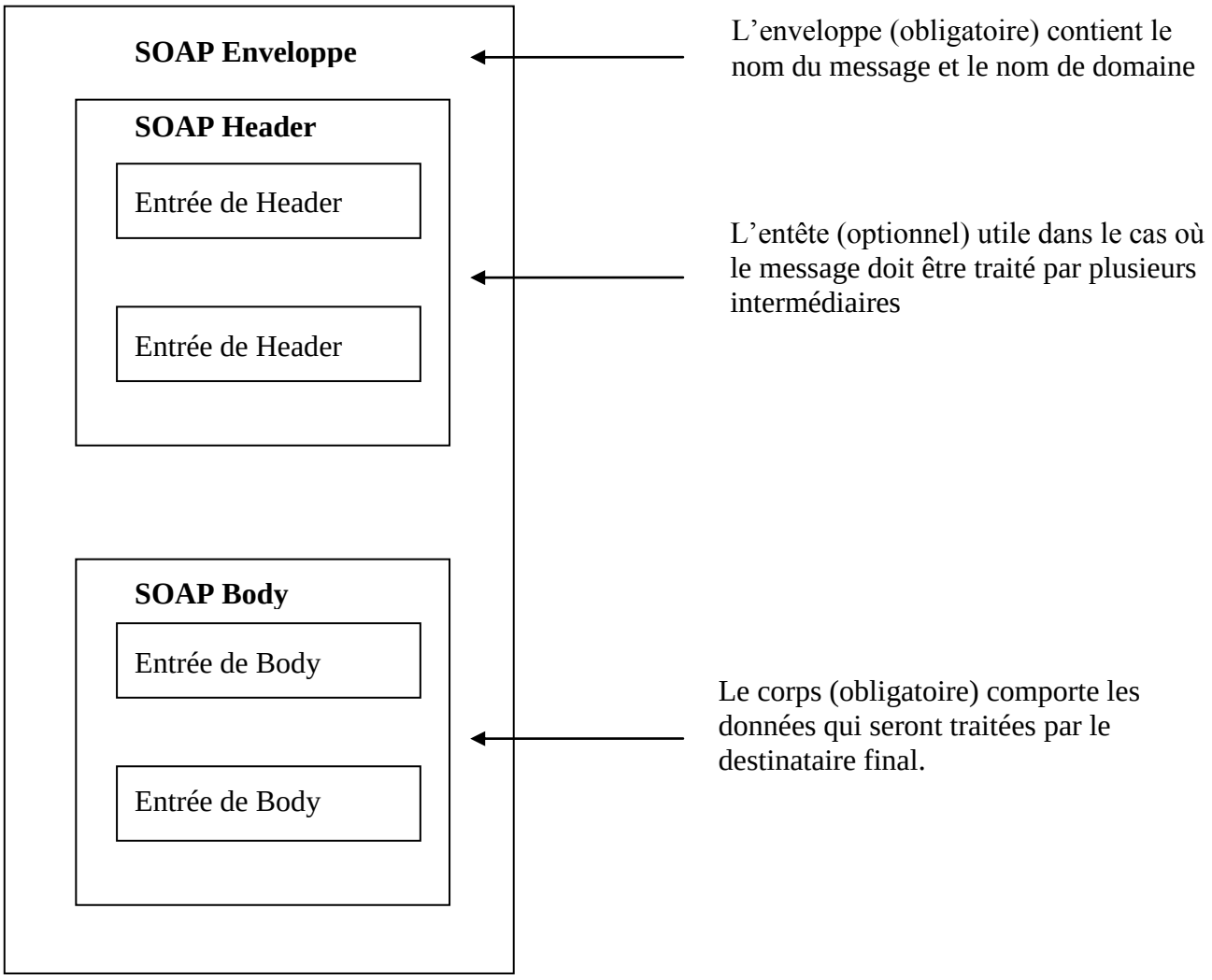


Figure 17. Structure d'un message SOAP. [HUB, 03]

2. Le langage WSDL

C'est un langage de description des web services en format XML, il repose essentiellement sur les mécanismes de SOAP, HTTP et MIME[16] pour l'invocation d'objets distants. Les services réseaux sont décris par une grammaire XML comme des ensembles de points finaux d'accès aux réseaux ou aux ports. WSDL est extensible afin de permettre la description de points finaux et leurs messages indépendamment des formats de messages ou protocoles réseaux utilisés réellement pour communiquer. [HUB, 03]

[16] MIME : Multipurpose Internet Mail Extensions, est un standard Internet qui étend le format de données des courriels pour supporter des textes en différents codage de caractères autres que l'ASCII.

Un port peut être défini par l'association d'une adresse réseau et une liaison réutilisable, un web service est un ensemble de ports. Une liaison réutilisable est le protocole concret et les spécifications de format de données pour un type de port particulier.

2.1. Structure d'un document WSDL

WSDL peut faire la description d'un web service à l'aide de six éléments principaux qui sont, [HUB, 03] :

- **types :** fournissent les définitions des types de données utilisées pour décrire les messages échangés.
- **message :** représente une définition abstraite de la donnée en cours de transmission. Un message comporte des parties logiques, chacune étant associée avec une définition dans un système de type.
- **portType :** est un ensemble d'opérations abstraites. Chaque opération se réfère à un message d'entrée et à des messages de sortie. Les portTypes sont utilisés pour définir les traitements offerts par un web service.
- **binding :** spécifie un protocole réel et les spécifications de format de données pour les opérations et les messages définis par un type de port donné.
- **port :** spécifie une adresse pour une liaison définissant un simple point terminal de communication.
- **service :** est utilisé pour agréger un ensemble de ports associés.

2.2. Liaison avec les protocoles de transport

Une liaison définie un format de message et les détails d'un protocole pour les opérations et les messages définis par un portType particulier. Une liaison spécifie exactement un protocole et non pas une information d'adresse. [HUB, 03]. Il peut y avoir n'importe quel nombre de liaisons pour un portType donné, à savoir :

- Liaison SOAP.
- Liaisons HTTP GET et POST
- Liaison MIME.

3. Le standard UDDI

L'annuaire UDDI permet aux utilisateurs de faire la publication la découverte et l'invocation des applications (des web services). En effet, UDDI peut contenir des informations sur les fournisseurs et les services qu'ils publient. L'inscription d'un fournisseur de services à l'annuaire UDDI lui permet de se présenter et présenter ses services, l'adoption de cet annuaire par les fournisseurs permet l'accélération des échanges surtout les échanges commerciaux de type B2B L'enregistrement des web services dans un annuaire UDDI s'effectue auprès d'un opérateur en accédant au site web de ce dernier à partir d'un navigateur ou d'un outil intégré à un environnement de développement. Des recherches précises peuvent

s'effectuer dans l'annuaire par catégorie de fournisseurs en utilisant des standards de taxinomie et d'identification de fournisseurs. [HUB, 03], [ERI, 04]

Les données capturées dans l'UDDI sont divisées en trois catégories [HUB, 03], [ERI, 04] :

- Pages blanches : le référentiel comporte des informations sur les fournisseurs de services telles que le nom et les coordonnées du fournisseur.
- Pages jaunes : le référentiel comporte des critères de catégorisation de services, les critères de catégorisation s'appuient sur des standards de classification de fournisseur, les services sont décrits par des documents au format WSDL. Un fournisseur peut disposer de plusieurs entrées dans l'annuaire pour l'ensemble des différents services et produits qu'il publie.
- Pages vertes : le référentiel comporte des informations techniques (WSDL) détaillées sur les services fournis telles que les informations sur les processus métier, les descriptions de services et les informations de liaison sur les services.

3.1. Mécanismes d'accès aux services fournis par UDDI

La communication (requêtes / réponses) avec un annuaire UDDI repose sur le protocole de transport SOAP.

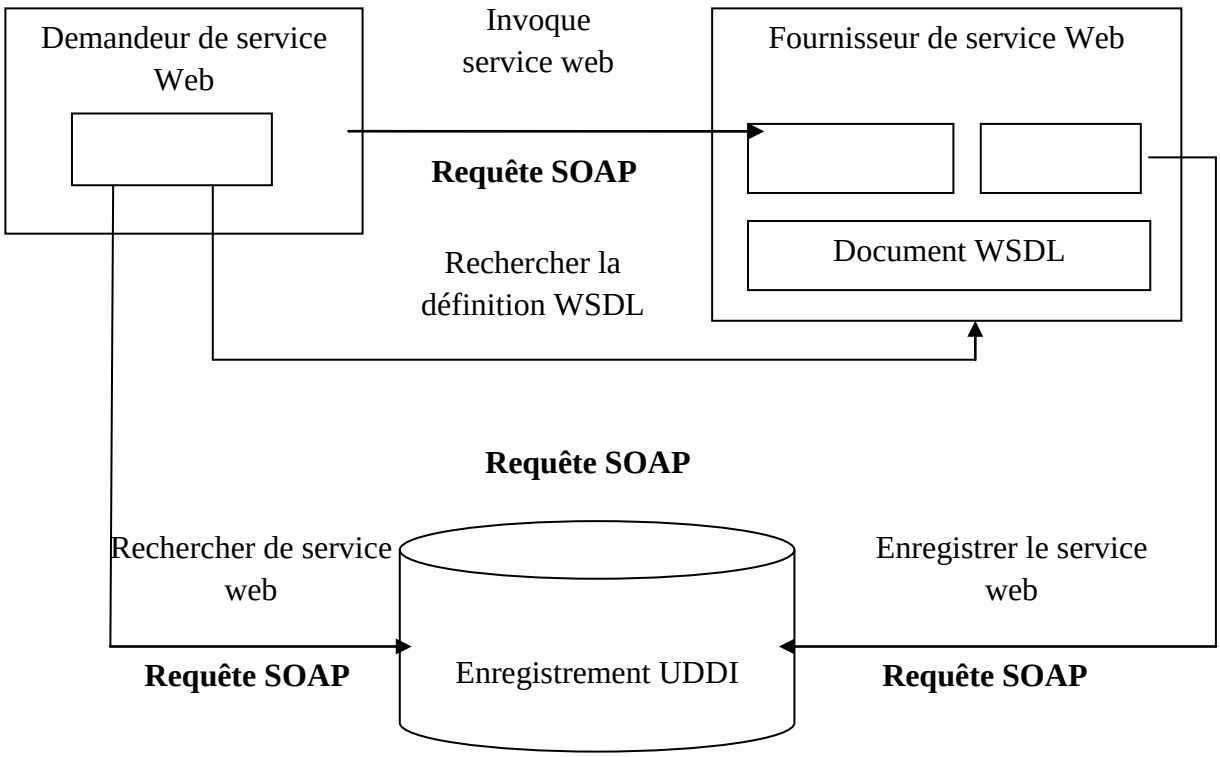


Figure 18. Mécanismes d'accès aux services de UDDI. [HUB, 03]

Un enregistrement UDDI est comparable au service DNS, il a deux types de clients : les fournisseurs de services et les utilisateurs de ces services. UDDI comporte des pages qui fournissent des informations sur les fournisseurs (nom, coordonnées,…etc.) et des pages qui comportent la description, au format WSDL, des web services. Un fournisseur peut disposer de plusieurs entrées dans l'annuaire pour l'ensemble de services qu'il propose, et des pages qui disposent d'informations techniques détaillées sur les produits proposés.

UDDI est un modèle de données permettant de décrire un web service, la définition d'une interface permettant de manipuler ce modèle de données et la mise à disposition de la communauté de UDDI sous forme d'un service universel gratuit est nécessaire.

3.2. Le modèle de données d'UDDI

Le modèle de données UDDI est défini sous forme de schéma W3C XML Schéma, ce schéma XML comporte cinq structures de données principales ; [HUB, 03], [ERI, 04] :

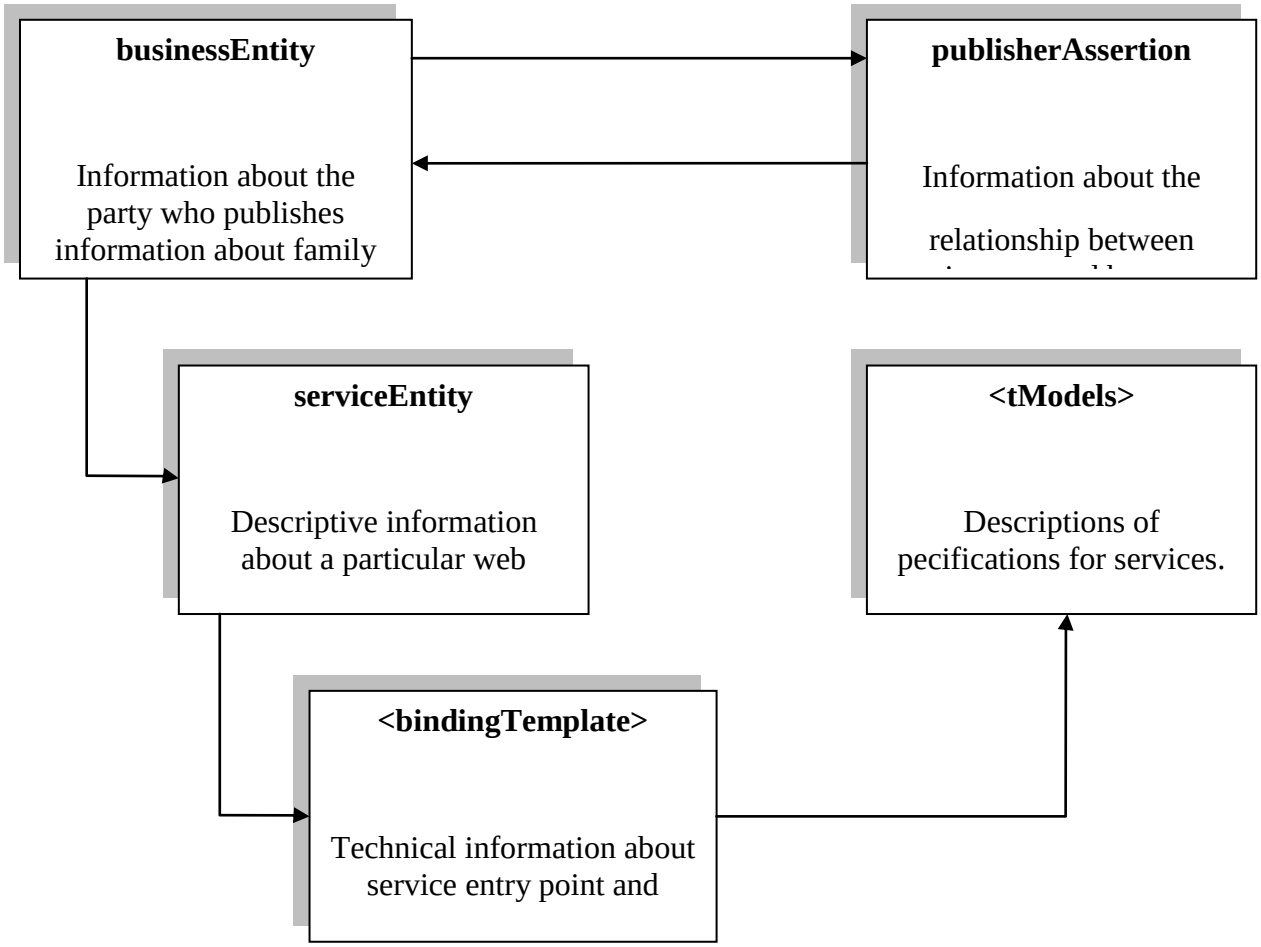


Figure 19. Modèle de données de l'annuaire UDDI**.** [RAH, 05]

- **BusinessEntity :** chaque businessEntity est identifiée par une «businessKey». Les «businessEntities» sont en quelque sorte les pages blanches d'un annuaire UDDI, elles comportent des informations sur les fournisseurs de services ayant publié des services dans l'annuaire. On y trouvera notamment le nom de l'organisation, ses adresses (physiques et web), des éléments de classification, une liste de contacts, ...etc.
- **ServiceEntity :** ce sont en quelque sorte les pages jaunes d'un annuaire UDDI, qui décrivent de manière non technique les services proposés par les différents fournisseurs. On y trouvera essentiellement le nom et la description textuelle des services ainsi qu'une référence à l'organisation proposant le service et un ou plusieurs «bindingTemplates».
- **BindingTemplate (coordonnées des services) :** ce sont des informations qui concernent le lieu d'hébergement du service, elles donnent les coordonnées des services. UDDI permet de décrire des web services HTTP, mais également des services invoqués par d'autres moyens (SMTP, FTP, fax, téléphone,...etc.). Elles contiennent notamment une description, la définition du « point d'accès » (suivant les cas, une URL, un numéro de téléphone, ...) et les éventuels « tModels » associés.
- **tModel (descriptions techniques des services) :** ce sont des informations qui concernent le mode d'accès au services, ce sont les descriptions techniques des services. UDDI n'impose aucun format pour ces descriptions qui peuvent être publiées sous n'importe quelle forme et notamment sous forme de documents textuels (XHTML par exemple). C'est à ce niveau que WSDL intervient comme le vocabulaire de choix pour publier des descriptions techniques de services.
- **publisherAssertion :** assertions contractuelles entre partenaires dans le cadre des échanges d'exécution d'un service.

3.3. L'interface d'UDDI

L'interface UDDI est définie sous forme de documents UDDI et implémentée sous forme de web service SOAP. Elle est composée des modules suivants :

- **Interrogation :** cette interface permet de rechercher des informations dans un répertoire UDDI et de lire les différents enregistrements enregistrés suivant le modèle de données UDDI.
- **Publication :** cette interface permet de publier des informations dans un répertoire UDDI conformément à son modèle de données.
- **Sécurité :** cette interface est utilisée pour obtenir et révoquer les jetons d'authentification nécessaires pour accéder aux enregistrements protégés dans un annuaire UDDI.
- **Contrôle d'accès et propriété :** cette interface permet de transférer la propriété d'informations (qui est à l'origine attribuée à l'utilisateur ayant publié ces informations) et de gérer les droits d'accès associés.
- **Abonnement :** cette interface permet à un client de s'abonner à un ensemble d'informations et d'être avertis lors des modifications de ces informations. Tous les

répertoires UDDI doivent gérer un avertissement par polling (le client interroge le serveur pour savoir si des modifications ont eu lieu sur les données auxquelles il est abonné). Une fonctionnalité optionnelle est également prévue permettant au client de communiquer au serveur la définition d'un web service sur lequel il souhaite être prévenu en cas de modification.

- **Réplication interne (noeuds d'un même annuaire) :** à côté des interfaces utilisateurs que nous venons de voir, UDDI définit également l'interface permettant de synchroniser les noeuds d'un même annuaire UDDI.
- **Réplication externe (interrogation, publication, abonnement) :** La réplication externe par duplication d'informations entre différents annuaires UDDI n'a pas donné lieu à la définition d'une interface spécifique mais se fait en utilisant les interfaces d'interrogation (pour la lecture dans un annuaire), publication (pour la publication dans un autre annuaire) et éventuellement abonnement (pour pouvoir propager les modifications ultérieures).

3.4. L'usage de l'annuaire UDDI

L'annuaire UDDI permet la description, la publication et la découverte des web services:

1. **La publication de services :** un web service peut être publié en publiant sa description, après sa production bien sur. Cette description peut être générée manuellement ou automatiquement à l'aide des outils qui peuvent générer des parties WSDL et de créer des entrées dans UDDI à partir de méta données par exemple. Un UDDI peut être de plusieurs types et leurs utilisations dépendent du domaine des services à publier, à savoir :

    - Noeud UDDI pour une application interne : les noeuds UDDI se trouvent derrière le firewall, avec ce type de noeud on a plus de contrôle sur l'enregistrement, l'accessibilité, la disponibilité et les spécifications de publication lors de la publication de service.
    - Noeud portail UDDI : il se trouve à l'extérieur du firewall du fournisseur de service ou entre les firewall. Sur ce type de noeud on publie des web services pour les partenaires externes tout en implémentant des mécanismes d'accés sélectifs selon le profil des utilisateurs.
    - Noeud catalogue du partenaire avec UDDI : les web services peuvent être publiés sur un noeud du catalogue du partenaire avec UDDI (deriére le firewall). Le partenaire est choisi avec une autorisation d'accés spécifique.
    - Noeud de place de marché UDDI : il s'agit de relations inter entreprises et de contrôle du partage de l'information entre systèmes d'information.

2. **La découverte de services :** c'est la recherche et la localisation d'un web service particulier dans un annuaire de services décrivant le nom du fournisseur, l'objectif de

chaque service,…etc. Donc il s'agit de l'acquisition des descriptions de web services et leur utilisation.

3. **L'invocation de services :** le client peut invoquer un service une fois sa description est reçue. En exploitant les informations de cette description, le client peut générer des requêtes SOAP pour invoquer le service.

4. Relation entre UDDI et WSDL

Le WSDL contient deux parties : la description de l'interface du service et la description de l'implémentation du service. La description de l'interface d'un service correspond aux informations techniques du service, c'est en quelque sorte une classe abstraite qui sera utilisée pour implémenter un ou plusieurs autres services. Elle regroupe : types, import, message, portType et binding. La description de l'implémentation du service correspond aux informations relatives à l'endroit où est publié le service. Elle regroupe : documentation, import, service et port.

UDDI permet de prendre en charge n'importe quel langage de description des web services, grâce à son model de donnée extensible. Une description WSDL d'un web service peut être traduite dans UDDI grâce à la combinaison de « BusinessService », « BindingTemplate » et de « tModel » tel que indiqué dans la figure ci – après :

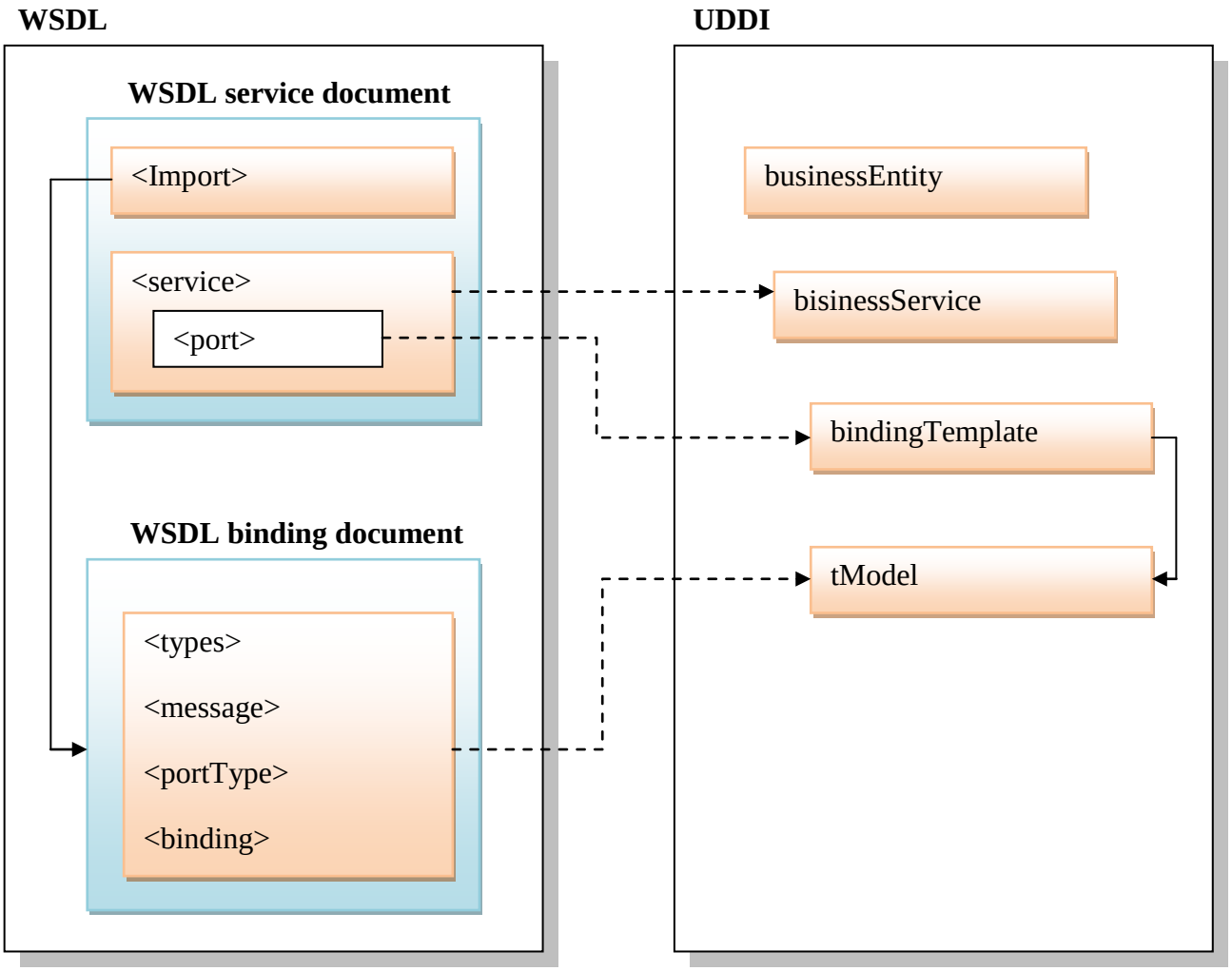


Figure 20. Relation entre UDDI et WDSL. [RAH, 05]

Le « port » du WSDL est représenté par « bindingTemplate » de UDDI. La relation entre « service » et ses ports en WSDL est exactement reflétée par la relation entre « businessService » et son « bindingTemplates » dans l'UDDI. Les éléments « type », « message », « portType » et « binding » du WSDL sont représentés par le tModel de UDDI.

En résumé : UDDI est un standard pour faciliter la collaboration entre partenaires dans le cadre d'échanges commerciaux, le coeur d'UDDI est un annuaire qui contient des informations techniques et administratives sur les fournisseurs et les web services qu'ils publient. Donc l'annuaire UDDI permet de publier et découvrir des informations qui concernent un fournisseur et ses web services.

Annexe B : Les langages du Web sémantique

1. URI (Uniform Ressource Identifier)

Une URI[17] est un identificateur d'une ressource mais surtout d'un concept, c'est une forme d'étiquette. C'est une chaîne de caractères formatée qui identifie avec certitude et de façon unique une ressource par un nom, une localisation ou une autre caractéristique. Un URI doit permettre d'identifier une ressource de manière permanente (même si cette ressource est déplacée, est supprimée ou cesse d'exister) [TEM, 07].

2. Méta-données

Un des grands principes du Web sémantique est qu'il est nécessaire d'associer aux ressources du Web des informations exploitables par des agents logiciels afin de favoriser l'exploitation de ces ressources. Une méta-données, « données sur les données » [BER, 97] fournit une information descriptive sur une donnée (une ressource) et doit permettre le traitement automatique des données sur laquelle elle porte. C'est une information graphique ou textuelle attachée et placée dans un document. Dans le contexte du Web sémantique, elle constitue un module fondamental et permet notamment de faciliter la recherche d'information. Les métadonnées font référence à des entités diverses : ensemble de documents, un document, un passage, une phrase, un terme,….

3. XML

XML (eXtensible Markup Language*)*, est un standard du W3C. Tout comme le HTML, il est basé sur les balises, seulement en XML toutes les balises doivent être fermées. Cette rigueur facilite le traitement automatique d'un document. De plus, C'est un *métalangage* : il permet de décrire un vocabulaire et une grammaire associée selon un certain formalisme.

L'intérêt premier de XML est de séparer le *fond* (le contenu) de la *forme* (le contenant ou la mise en page).

La structure d'un document XML est définie par une DTD (Document Type Definition). La DTD peut être écrite dans un document à part puis référencée dans le document XML ou peut être directement intégrée dans ce dernier. Dans l'exemple qui suit nous préférons intégrer la DTD en raison de sa simplicité. On peut également écrire un fichier XML Schéma pour définir une structure plus détaillée et surtout pour compléter les définitions de types qui sont très restreintes en DTD.

[17] http://websemantique.org/URI

Exemple :

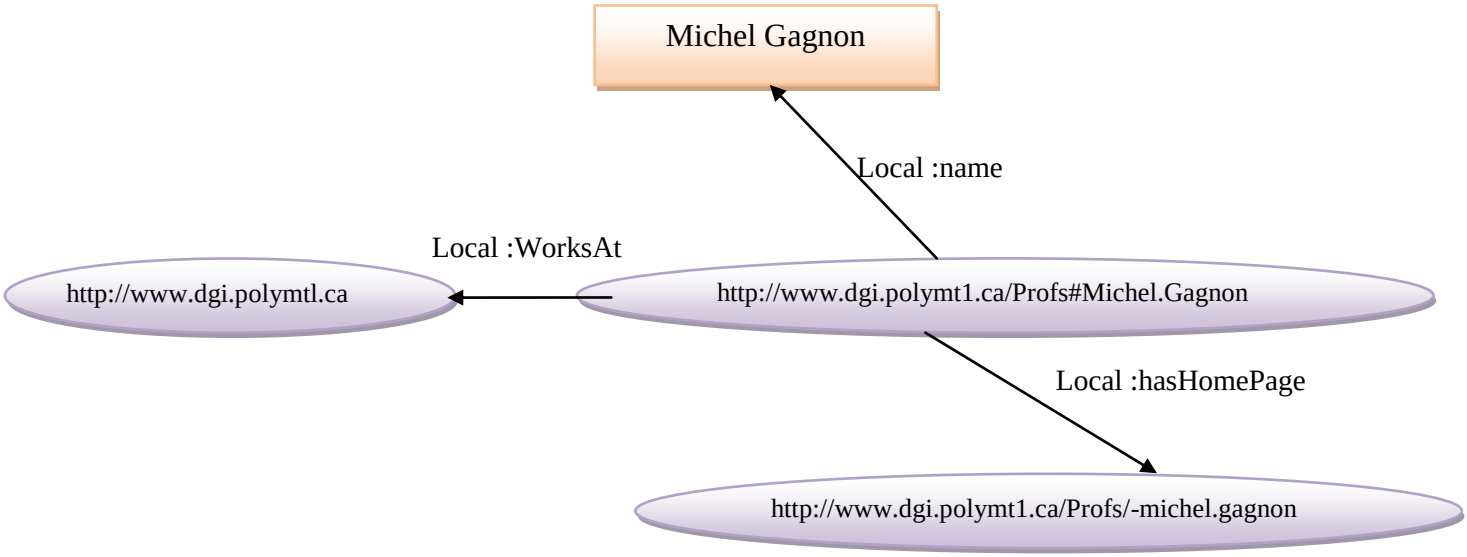
```
<?xml version="1.0" encoding="UTF-8"?>
<personne>
        <nom>Dupond</nom>
        <prenom>Jean</prenom>
        <naissance>
                <lieu>
                        <ville>Paris</ville>
                        <pays>rrance</pays>
                </lieu>
                <date>
                        <jour>14</jour>
                        <mois>7</mois>
                        <annee>1789</annee>
                </date>
        </naissance>
</personne>
```

Figure 21. Exemple d'un document XML

4. RDF et RDFS

RDF (Resource Description Framework) est un langage, recommandé par le W3C, fondé sur les notions de ressources et de relations entre ressources [EUZ, 04]. Il sera utilisé pour annoter des documents écrits dans des langages non structurés, ou comme une interface pour des documents écrits dans des langages ayant une sémantique équivalente (des bases de données, par exemple). Un document RDF est un ensemble de triplets de la forme <*sujet, prédicat, objet*>. Les éléments de ces triplets peuvent être des URIs (Universal Resource Identifiers) [LAU, 02], [BOU, 04], Les ressources peuvent être identifiées ou rester anonymes.

Cet ensemble de triplets peut être représenté de façon naturelle par un graphe (plus précisément un graphe orienté étiqueté), où les éléments apparaissant comme sujet dont les objets sont les sommets, et chaque triplet est représenté par un arc dont l'origine est son sujet et la destination son objet. Ce document sera codé en machine par un document RDF/XML, mais est souvent représenté sous une forme graphique.




```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:local="http://www.dgi.polymtl.ca/local">
<rdf:Description rdf:about="http://www.dgi.polymtl.ca/Profs#MichelGagnon">
 <local:hasHomePage resource="http://www.dgi.polymtl.ca/profs/~michel.gagnon"/>
 <local:worksAt resource="http://www.dgi.polymtl.ca"/>
 <local:name>Michel Gagnon</local:name>
</rdf:Description>
</rdf:RDF>
```

Figure 22. Exemple de graphe RDF

La *figure.22* présente une partie d'un document RDF. Les termes de la forme **http://...** Sont des URIs qui identifient des ressources définies de façon unique. Les objets d'un triplet qui sont des littéraux sont représentés dans un rectangle (pour cet exemple, Michel Gagnon).

RDF propose aussi certains mots-clés réservés, qui permettent de donner une sémantique particulière à des ressources. Ainsi, on peut représenter des ensembles d'objets (rdf:bag), des listes (rdf:sequence), des relations d'arité quelconque (rdf:value)… Ce ne sont cependant pas de réelles extensions du langage présenté ci-dessus, puisqu'une transformation permet d'exprimer cette « sémantique étendue » dans le langage de base : *R1* est une conséquence (sémantique étendue) de *R2* si et seulement si *réif(R1)* est une conséquence (au sens précédent) de *réif(R2)*.

RDFS (pour RDF Schéma) a pour but d'étendre le langage en décrivant plus précisément les ressources utilisées pour étiqueter les graphes. Pour cela, il fournit un mécanisme permettant de spécifier les classes dont les ressources seront des instances, comme les propriétés. RDFS s'écrit toujours à l'aide de triplets RDF, en définissant la sémantique de nouveaux mots-clés comme :

- <ex:Vehicule rdf:type rdfs:Class> la ressource <ex:Vehicule> a pour type <rdfs:Class>, et est donc une classe ;
- <sncf:TER8153 rdf:type ex:Vehicule> la ressource <sncf:TER8153> est une instance de la classe <ex:Vehicule> que nous avons définie ;
- <sncf:Train rdfs:subClassOf ex:Vehicule> la classe <sncf:Train> est une sousclasse de <ex:Vehicule>, toutes les instances de <sncf:Train> sont donc des instances de <ex:Vehicule>;

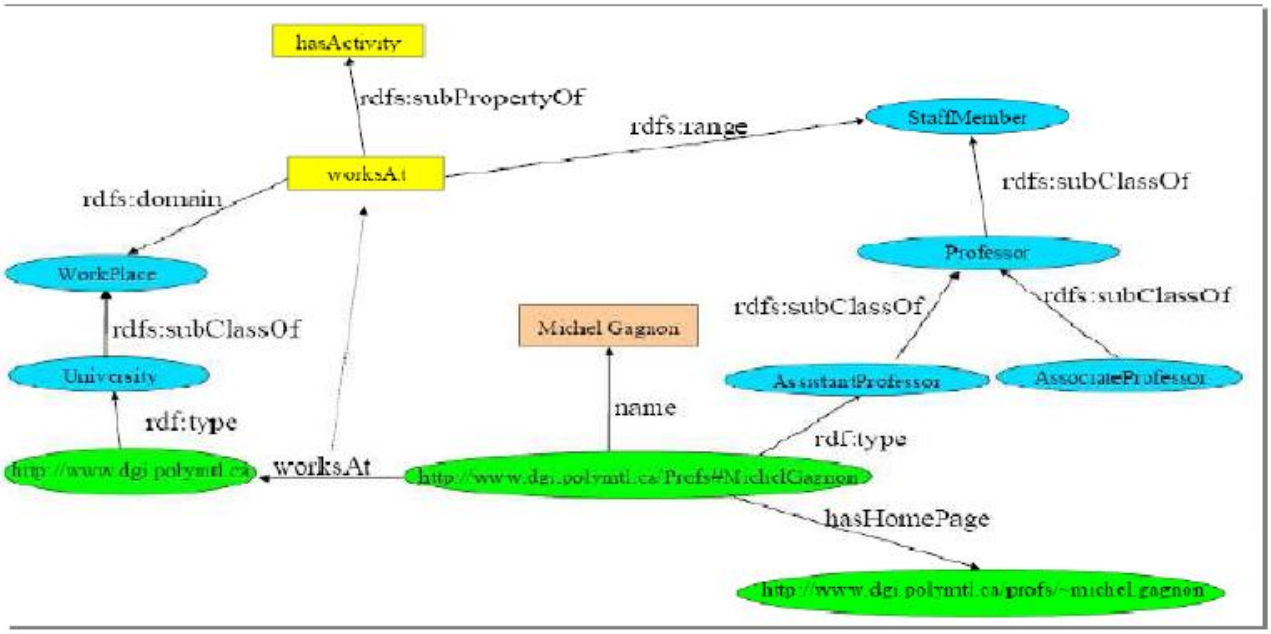


Figure 23. Exemple de graphe RDFS

Annexe C : La plateforme JXTA

1. La plate-forme JXTA

JXTA est une spécification en XML d'un ensemble de protocoles P2P génériques qui permettent à n'importe quel périphérique connecté au réseau (téléphones portables, PDA, PCs et serveurs) de communiquer et de collaborer en tant que pairs. Une des caractéristiques de ces protocoles est qu'ils sont indépendants des langages de programmation.

JXTA est une plateforme de développement d'applications P2P tout à fait originale par la façon dont elle perçoit le problème ainsi que par les mécanismes de mise en œuvre d'un réseau P2P qu'elle offre en ce situant à un niveau d'abstraction assez élevé.

Au moyen de la plate-forme JXTA, *Sun* offre aux développeurs et aux utilisateurs un ensemble de services de haut niveau permettant la création et l'utilisation d'applications P2P. Ces services et ces applications ont l'avantage d'être interopérables et multiplateformes.

JXTA est un ensemble des protocoles qui peuvent être implémentés dans tout langage de programmation. Pour l'instant, Sun fournit une implémentation de JXTA en Java, en C, et en C#, mais d'autres implémentations existent en Perl, en Python et en Smalltalk.

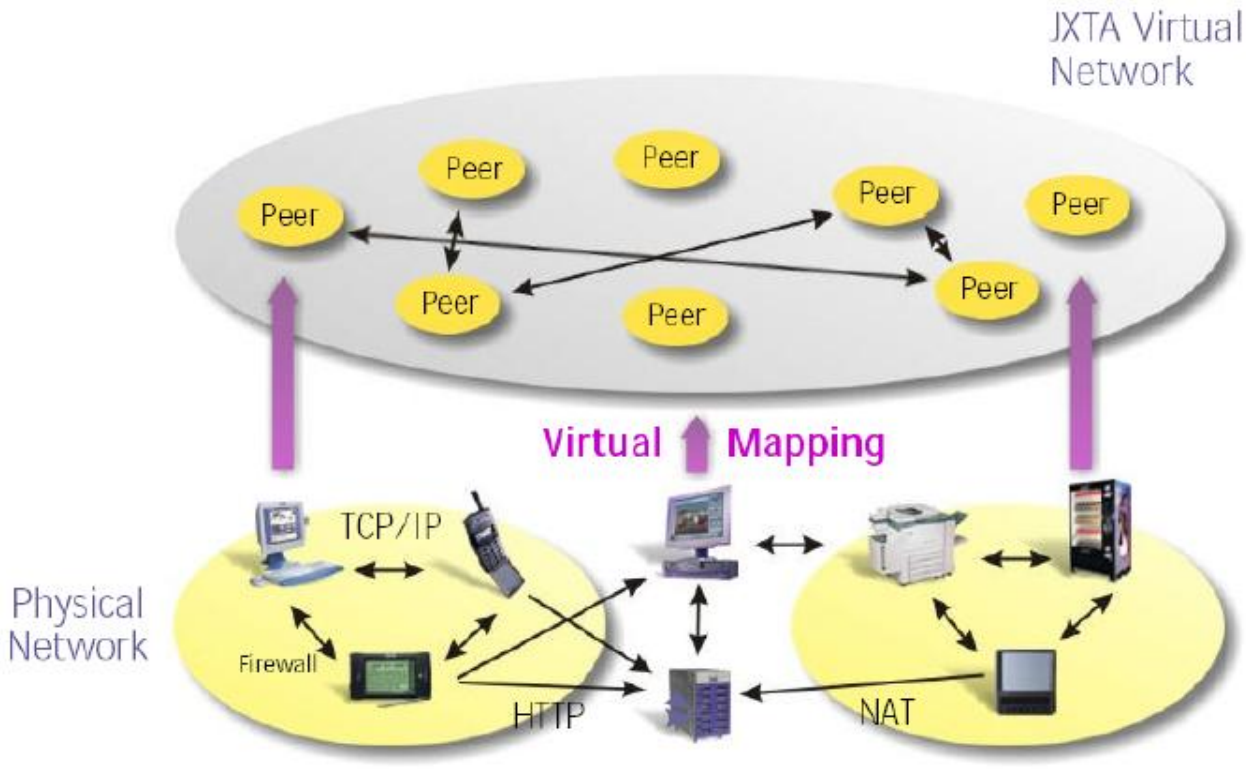


Figure 24. La vision générique (overlay) de la plate-forme JXTA [SOT, 02]

1.1. Objectifs de la plate-forme JXTA

La technologie JXTA cherche à surmonter les points faibles des différents systèmes P2P existants. Elle est désignée pour répondre aux objectifs suivants:

- **La factorisation :** C'est la mise en commun des fonctionnalités identiques par découpage en couches modifiables indépendantes les unes des autres. En effet, les systèmes pair-à-pair utilisent souvent des protocoles différents mais qui ont les mêmes fonctionnalités, comme la découverte de ressources, la communication entre pairs et la gestion des groupes. Ainsi, pour une amélioration de ces fonctionnalités, celles-ci doivent être découpées en couches indépendantes les unes des autres.
- **Interopérabilité :** Les différents systèmes élaborés sont pour l'instant incompatibles entre eux. Or, il serait souhaitable de pouvoir bénéficier des ponts d'échanges entre ces systèmes. Le développement de services spécialisés basés sur un même environnement générique est nécessaire afin de rendre ces services interopérables.
- **Indépendance vis-à-vis de la plate-forme d'accueil :** L'objectif d'un système pair à pair à grande échelle est d'être déployé sur un grand nombre de pairs. Il faut donc disposer d'un environnement générique et indépendant de la plateforme d'accueil. Une spécification précise et claire de l'ensemble des protocoles et de leurs formats à respecter lors de l'implémentation de l'environnement, est nécessaire. Ainsi, le choix du langage ne doit pas influencer l'interopérabilité des différentes implémentations existantes.
- **Ubiquité :** Le dernier objectif est l'ubiquité. Il matérialise le désir que la plateforme JXTA soit implantable sur n'importe quel périphérique doté d'un coeur digital « *digital heartbeat* ». Cette notion est très générale puisqu'elle comprend les PDA, les GSM, les routeurs réseaux, les ordinateurs de bureau, les serveurs de données (ou d'applications).

1.2. Architecture de la plate-forme JXTA

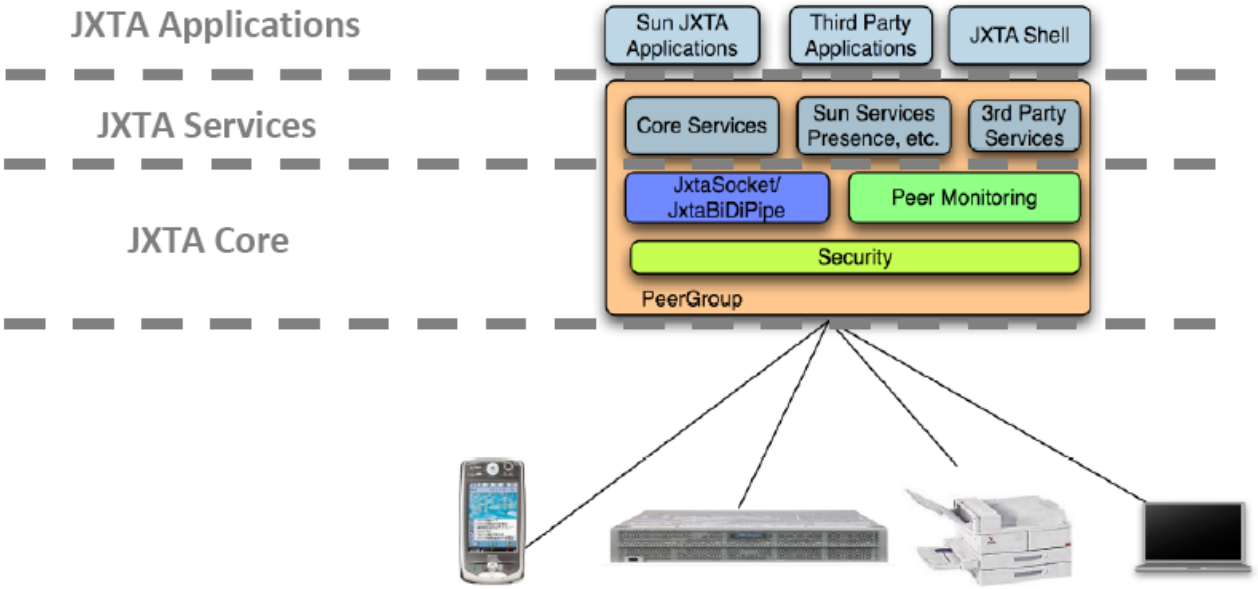


Figure 25. Architecture logique du framework JXTA

- **La couche noyau (Core)** Le noyau de JXTA encapsule les primitifs minimaux et essentiels qui sont communs à la gestion d'un réseau P2P. Il offre aux développeurs un ensemble de mécanismes de base pour les applications P2P permettant la découverte, le transport des messages (y compris de traverser les Firewalls et les NATs), la création des pairs, des groupes des pairs, et les primitifs de sécurité telles que l'anonymat et le cryptage.
- **La couche Service** Permet d'étendre les mécanismes de base offerts par la couche noyau et ainsi faciliter le développement d'applications de plus haut niveau. Les principaux services proposés dans cette couche sont des mécanismes de recherche et d'indexation **(JXTA Search)**, de partage de ressources **(JXTA cms)**, de stockage **(jxtaspaces)**, de traduction de protocoles **(JXTAxmlrpc,jxta-rmi,jxtasoap,…)**,etc.
- **La couche Applications** Au sein de cette couche, les applications seront développées sur la base des services disponibles dans la couche services. Exemples de ces applications : la messagerie instantanée P2P, partage de document et de ressource, la gestion et la livraison de contenu multimédia, les systèmes d'email P2P, et beaucoup d'autres.

La frontière entre les services et les applications n'est pas rigide. Une application d'un pair donné peut être vue comme service par un autre pair. Le système entier est conçu pour être

modulaire, permettant aux développeurs de sélectionner et choisir une collection des services et d'applications qui convient le plus à leurs besoins.

1.3. Composants d'un réseau JXTA

Un réseau JXTA consiste en un ensemble de pairs *(peers)*. Ces pairs peuvent s'autoorganiser dans des groupes des pairs *(peer groups)* fournissant un ensemble des services.

Conséquence de la nature de l'architecture adoptée par JXTA dans l'organisation des pairs et le flux des données qui est l'architecture hybride, il existe deux types de pairs JXTA, les simples pairs et les super pairs *(Super peers)*. Les super pairs jouent, en plus de leur rôle de simples pairs, le rôle d'annuaires[18] pour les pairs qui leurs sont directement connectés, comme ils gèrent périodiquement la liste des autres super pairs.

Les pairs JXTA annoncent leurs services sous forme de documents XML appelés annonces *(advertisements)*. Les annonces permettent aux autres pairs dans le réseau de savoir comment se connecter et interagir avec les pairs qui offrent ces services.

Les pairs JXTA utilisent les tubes *(Pipes[19])* pour envoyer des messages vers les autres pairs. Les pipes constituent un mécanisme de transfert asynchrone et unidirectionnel utilisé pour assurer la communication entre les pairs. Le message est un simple document XML qui constitue l'unité de base d'échanges entre pairs. Les tubes *(pipes)* sont reliés à des extrémités spécifiques *(Endpoints)*.

Quatre aspects essentiels distinguent JXTA des autres modèles des réseaux distribués :

- Utilisation des documents XML pour décrire les ressources disponibles dans le réseau;
- Abstraction des tubes par rapport aux pairs, et des pairs par rapport aux extrémités, sans dépendre d'un moyen de nommage centralisé tel que le DNS;
- Utilisation d'un plan d'adressage uniforme et plat (non hiérarchique) qui repose sur les Peer IDs;
- Une infrastructure décentralisée de recherche basée sur la table de hachage distribuée ***(SRDI[20])*** pour l'indexation des ressources.

18 Ce sont des annuaires des pairs et des ressources détenues par ces pairs.
19 A la base de la simple définition des pipes JXTA, d'autres types de pipe ont vu le jour tels que : Unicast pipe, Propagate pipe, SecureUnicast pipe, Bidirectionnal pipe…
20 **SRDI: S**hared **R**essource **D**istributed **I**ndex (la DHT de JXTA); www.di.unipi.it/~ricci/jxta-dht.pdf

Annexe D : Web Service Modeling eXecution environment (WSMX)

1. Présentation de WSMX

WSMX est une initiative menée par l'institue de recherche DERI10 et les projets européens (DIP[21], SEKT[22] et Knowledge Web[23]). WSMX est une implémentation de référence de WSMO et un environnement d'exécution dans lequel les descriptions sémantique des buts des utilisateurs et celles des services web des fournisseurs peuvent être utilisés pour découvrir, composer, faire la médiation, sélectionner et invoquer les services web qui correspondent aux besoins de l'utilisateur. Une caractéristique de WSMX est qu'il ne définie pas les services en terme de leurs détails de conception ou d'implémentation, mais seulement en terme des fonctionnalités attendues et leur comportent externe visible.

2. Architecture de WSMX

La plateforme WSMX est conçue selon les standards du SOA, et a une architecture basée sur des composants, où chaque composant fait une part d'une fonctionnalité. Dans cette section une brève description des composants (figure 26) de WSMX et leur rôle est donnée.

- ***Data & communication protocols Adapters:*** comme leur nom indique, ce sont des adaptateurs pour les applications qui veulent communiquer avec WSMX. Ce type de composants est considéré par les auteurs de WSMX comme externe, mais quand les interfaces des applications n'intègrent pas des API WSMO, ces adaptateurs sont nécessaires pour se connecter à l'interface du serveur WSMX. Ils traduisent n'importe quel format de message au format WSML.
- ***WSMT (Web Service Modeling Toolkit):*** est un environnement de développement qui regroupe un bon nombre d'outils qui facilitent la création des descriptions des ontologies, des services web, des buts et des médiateurs, graphiquement ou en utilisant des éditeurs WSML. Comme ils permettent de se connecter au serveur WSMX et faire des inférences à base des descriptions créées.
- ***Communication Manager :*** a deux rôles principaux, le premier est de fournir aux adaptateurs une interface qui leur permet de recevoir et d'envoyer des messages WSML. Le deuxième est de traduire les messages dans n'importe quel autre format requis (pour une éventuelle invocation par des entités externes et n'utilisant pas le format WSML, c'est l'opération du *«Lowering»*). En invoquant le *« Communication Manager »*, il est responsable de la traduction d'un message reçu d'une entité externe (qui utilise n'importe quelle représentation de données) en WSML. Cette traduction est souvent appelée *« Lifting »*.

[21] DIP: http://dip.semanticweb.org/

[22] SEKT : http://sekt.semanticweb.org/

[23] Knowledge Web : http://knowledgeweb.semanticweb.org/

- ***WSMX Manager :*** joue le rôle de coordinateur dans l'architecture, il gère le traitement de toutes les notifications en les passants aux composants appropriés.

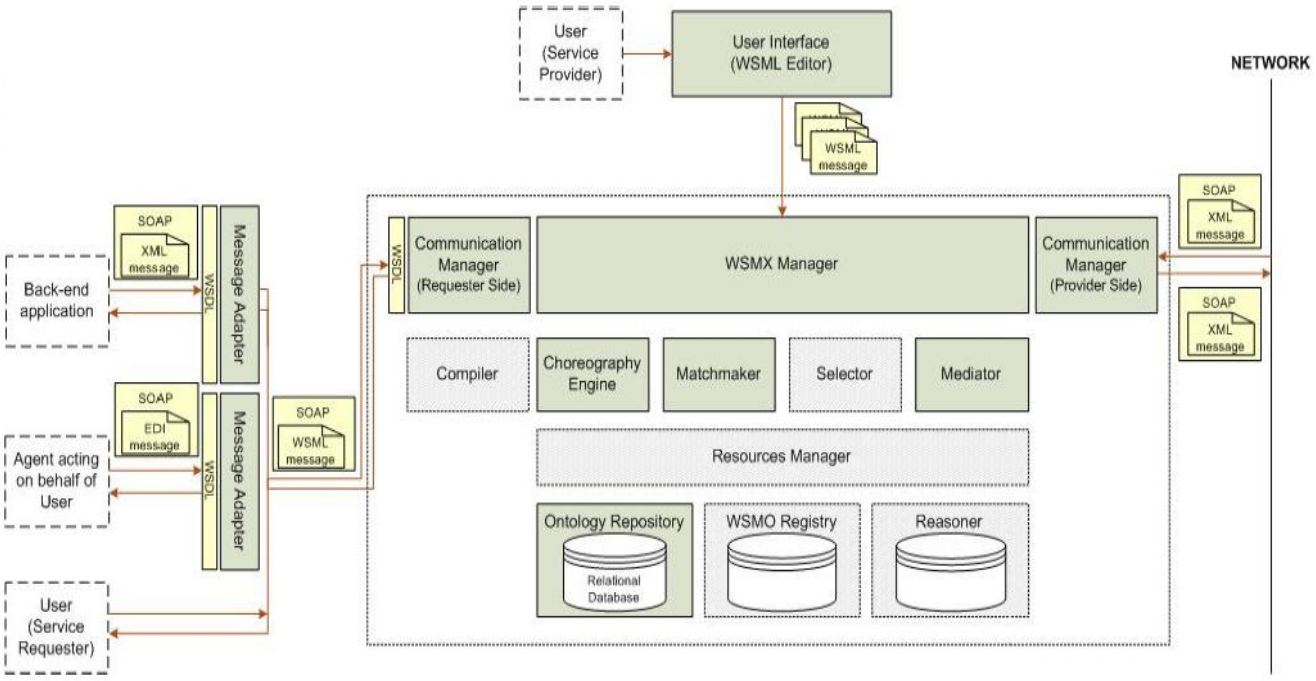


Figure 26. Architecture de WSMX

- ***Compiler :*** Considéré comme le composant essentiel, il est utilisé dans toutes les opérations. Comme il sert à valider les documents WSML.
- ***Matchmaker :*** Offre un ensemble des services web en faisant correspondre les « *Capapilities* » déjà enregistrées avec les buts de l'utilisateur. Il accepte comme entrées un ensemble de services web enregistrés et le but du demandeur. Il fournie en sortie un ensemble de services web qui correspondent au but décrit.
- ***Mediator :*** Offre une médiation pour les données communiquées. Il permet, si nécessaire, de trouver un médiateur approprié pour une requête donnée. Fait la correspondance entre une ou plusieurs ontologies sources vers une ou plusieurs ontologies de destinations, qui sont utilisés soit dans les opérations de découverte/sélection ou bien dans l'opération d'exécution des services web. Ce composant est basé essentiellement sur les « *ooMediator* » de la spécification WSMO.
- ***Choreography Engine :*** Définie les modèles des communications des services web, en faisant la médiation entre les modèles des communications du demandeur et ceux du fournisseur des services. Pour ce faire, il utilise des médiateurs pour compenser le manque (ou la non-correspondance) dans les modèles de

communication. Les règles de chorégraphie conçues, sont créées et enregistrer au même temps de leur conception.