Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

pour l'obtention du diplôme de Master en Informatique

Option: *Modèle Intelligent et Décision(M.I.D)*

Thème

Evaluation des techniques de codage d'ontologies sur les performances de la composition de services Web

Réalisé par :

- **M^elle DEHANE Aicha Djihad**
- **M^elle KEBIR Zohra**

Présenté le 03 Juillet 2012 devant le jury composé de MM.

- *BENAZZOUZ .M* *(Président)*
- *HADJILA .F* *(Encadreur)*
- *BERRAMDANE .D* *(Examinatrice)*
- *BENZIANE .Y* *(Examinateur)*

Année universitaire : 2011-2012

Remerciements

Nous tenons à remercier :

Allah le tous puissant.

Mr FETHALLAH HADJILA, notre encadreur, pour ses conseils, sa disponibilité et son encouragement qui nous ont permis de réaliser ce travail dans les meilleures conditions.

Les jurys pour leurs efforts et leur soin apporté à notre travail :

- *Mr. BENAZZOUZ.M*
- *Melle .BERRAMDANE .D*
- *Mr. BENZIANE.Y*

Aux enseignants de notre université et du département d'informatique.

Dédicace

Je dédie ce modeste travail à :

Mes parents

Mes frères

Ma chère amie BOUZIANE Houriya

Ma très chère amie Djihade

Dr. OMARI Mohammed

Département d'informatique d'ADRAR

Mes collègues à l'université d'ADRAR

Mes collègues en Master 2 MID.

Toutes Mes amies.

Zohra

Dédicace

Je dédie ce travail à

Mes très chers parents.

Mes très chères sœurs ainsi que leur maris et leur adorable garçons et princesses.

Mes très chers frères ainsi que leur maries et leur adorable garçons et princesses.

Tous mes amis, ainsi que toute la promotion du Master 2 en informatique MID 2011/2012 de l'université de Tlemcen.

Mes très chères amies HOURIA & ZOHRA

AICHA DJIHAD

Table des matières

Chapitre I : Les Services Web

Chapitre II : Les ontologies

Chapitre III : Les techniques d'encodage d'ontologies

Chapitre IV : Prototype

Liste des figures

Liste des Tableaux

Liste d'abréviations

API	Application Programming Interface
B2B	Business to Business
B2C	Business to Consumer
BVE	Bit Vector Encoding
CORBA	Common Object Request Broker Architecture
DCOM	Distributed Component Object Model
HTTP	HyperText Transfert Protocol
MTR	Matrice Transitive et Réflexive
OIL	Ontology Inference Layer
OWL	Web Ontology Language
OWL-S	Web Ontology Language for Services Web
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RMI	Remote Method Invocation
RPC	Remote Procedure call
SAWSDL	Semantic Annotations for WSDL and XML Schema
SE	Schéma d'Etiquetage
SHOE	Simple HTML Ontology Extensions
SOA	Service Oriented architecture
SOAP	Simple Object Access Protocol
SW	Service Web
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web consortium
WSMO	Web Service Modeling Ontology
WSDL	Web Service Description Language
WSDL-S	Web Service Description Language-Semantic
XML	eXtansible Markup Language

Introduction générale

Introduction générale

Contexte

L'interaction de différentes applications offertes par des entreprises et organisations en réseau a toujours été une affaire complexe, tant que ces applications peuvent être écrites dans des langages de programmation différents et exécutées sur des plates-formes hétérogènes, la standardisation de certains aspects de cette interaction est nécessaire. Plusieurs technologies permettent de résoudre ce problème de communication à travers un réseau. Parmi eux les « **Services Web** ».

Les **services Web** (en anglais *Web services*) représentent un mécanisme de communication entre applications distantes à travers le réseau internet indépendant. Telle que ces applications peuvent être vues comme un ensemble de services métiers, structurés et correctement décrits, dialoguant selon un standard international plutôt qu'un ensemble d'objets et de méthodes entremêlés.

Les services Web considérés comme une fonction accessible a distance par une requête qui exprime les besoins d'un utilisateur a pour entrée un ensemble de concepts indiqués dans une ontologie spécifique et donne comme sortie un autre ensemble de concepts.

Les services Web, tels qu'ils sont conçus, peuvent également se voir employés dans le cadre d'applications distribuées. En effet, en exploitant les standards ouverts du Web et en assurant un couplage faible des composants, la technologie service Web présente une approche flexible et universelle pour l'interopérabilité de systèmes hétérogènes. Ceci a pour effet de permettre une intégration des applications plus rapide, moins couteuse et avec des perspectives prometteuses d'évolution et de réutilisation pour les entreprises. Cette opération d'intégration est appelée composition.

La composition est une technique permettant d'assembler des services Web afin d'atteindre un objectif particulier, par l'intermédiaire de primitives de contrôle (boucle, test, traitement d'exception, etc.) et d'échange (envoi et réception de messages). Les services composants existent au préalable et peuvent ne pas être fournis par la même organisation.

Malgré que les services Web soient conçus pour résoudre les problèmes d'interopérabilité notamment l'interopérabilité technique, ils ne sont conçus initialement pour répondre aux exigences d'échange sémantique (interopérabilité sémantique). Par conséquent, les hétérogénéités sémantiques sont gérées de manière manuelle durant la phase de conception d'une composition.

Afin de pallier à ce problème, des solutions et des approches ont été proposées d'une part pour décrire la sémantique des services Web de manière formelle et compréhensible par les machines. Et d'une autre part pour résoudre les hétérogénéités et les conflits sémantiques entre ces services engagés dans la composition. Ces solutions se basent sur les ontologies (technologies proposés par la communauté du Web sémantique) qui permet d'expliciter la sémantique des fonctionnalités offertes par les services Web de façon plus formelle.

Problématique

Les ontologies sans doute sont capables de décrire la sémantique des services Web et les fonctionnalités offertes par ces services de manière explicite et compréhensible par les machines.

Dans le cadre de ce mémoire, nous traitons la problématique de l'interopérabilité sémantique au niveau des services Web engagés dans une composition et particulièrement au niveau de données (inputs et outputs) échangées entre les services Web à composer.

Plus précisément nous évaluons l'impact du codage des ontologies sur les performances des approches de composition.

Contribution

Nous aborderons dans notre travail, les hétérogénéités des données échangées entre les services Web. Nous nous focaliserons, particulièrement, sur les hétérogénéités sémantiques de ces données. Les méthodes de génération de compositions sémantiques utilisent toujours la notion de subsomption, et généralement cette très gourmande en termes de temps d'exécution, l'objet techniques de codages d'ontologies sur les performance de la génération des compositions,

En particulier nous allons comparer Schéma d'étiquetage, Fermeture transitive et réflexive et l'API Pellet.

Plan du mémoire

Ce manuscrit est composé de quatre chapitres et conclusion générale qui sont organisés comme suit :

Chapitre I : Ce chapitre est consacré à l'étude des services Web et leur composition pour mieux comprendre les concepts de base de cette technologie. Ce chapitre se divise en deux parties : nous présentons dans la première partie les services Web, leur architecture, leur modèle d'interaction et leurs standards de base. Dans la deuxième partie de ce chapitre, nous aborderons la composition des services Web, en suite quelques avantages et inconvénients.

Chapitre II *:* Présente un état de l'art sur les ontologies et les services Web sémantiques. Il se divise aussi en deux parties: la première partie est consacrée à étudier les ontologies et leurs apports. La deuxième partie présente quelques approches de réalisation des services Web sémantiques proposées dans la littérature. Une comparaison entre ces différentes approches est effectuée.

Chapitre III : Dans ce chapitre nous avons présenté les techniques d'encodage d'ontologie.

Chapitre IV: Ce dernier chapitre présente le prototype destiné à la composition de service Web.

Chapitre I

Les services Web

I. Introduction

De nos jours, les entreprises et les systèmes d'information expriment un grand besoin pour échanger des informations et des services. Ceci nécessite des langages communs de communication. Les efforts de standardisation de ces langages ont donne lieu a un nouveau domaine de recherche connu sous le nom de « protocoles B2B ». Une technologie émergeante dans ce domaine a permis de tracer quelques pistes intéressantes pour la communication entre entreprises. Cette technologie est celle de services Web.

Les services Web sont un paradigme naissant qui vise à la transposition des architectures par composant dans le cadre du Web. Un service Web est un composant logiciel qui offre des services à travers une interface standardisé. La particularité des services Web est l'utilisation de la technologie Internet comme infrastructure pour la communication entre eux.

Les fonctionnalités offertes par les services Web sont relativement simples, alors que les besoins des utilisateurs sont de plus en plus complexes, à tel point qu'un seul service Web ne peut pas les satisfaire. L'implication de la composition des services Web, dans le but de créer un nouveau service Web dit composite offre des nouvelles perspectives qui répondent aux exigences complexes des utilisateurs.

Dans cette section, nous présenterons l'architecture orientée services et sa principale réalisation : les services Web. Nous donnons les définitions des services Web et décrivons brièvement leurs principaux standards et technologies. Enfin, nous abordons le concept de composition des services Web en explicitant les deux méthodes de composition : l'orchestration et la chorégraphie.

II. Architecture orientée services

Ces dernières années ont vu l'émergence d'architectures orientées services (SOA) conçues pour faciliter la création, l'exposition, l'interconnexion et la réutilisation d'applications à base de services.

Plusieurs définitions attribués la notion d'architectures orientées services

L'architecture orientée services [OASIS, 2006] « is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains »

L'architecture orientée services [Eric et al., 2005] « SOA represents a model in which functionality is decomposed into small, distinct units (services), which can be distributed over a network and can be combined together and reused to create business applications. »

L'architecture orientée services [Erl, 2005] « est une architecture logicielle visant à mettre en place un système d'information constitué de services applicatifs indépendants et interconnectés »

L'architecture SOA permet la réutilisation des applications et des services existants, ainsi de nouveaux services peuvent être créés à partir d'une infrastructure informatique des systèmes déjà existante, en leur offrant une interopérabilité entre applications et technologies hétérogènes. L'objectif d'une architecture SOA est de décomposer une fonctionnalité en un ensemble de services et de décrire leurs interactions.

Dans une architecture SOA, comme le montre figure I.1, les fournisseurs de services publient (ou enregistrent) leurs services dans un annuaire de services (qui peut être publique ou local à un réseau d'entreprise par exemple). Cet annuaire est utilisé par les utilisateurs (i.e. les clients de services) pour trouver des services vérifiant certains critères ou correspondant à une certaine description. Si l'annuaire contient de tels services, il envoie au client les descriptions de ces services avec un contrat d'utilisation. Le client fait alors son choix, s'adresse au fournisseur et invoque le service Web.

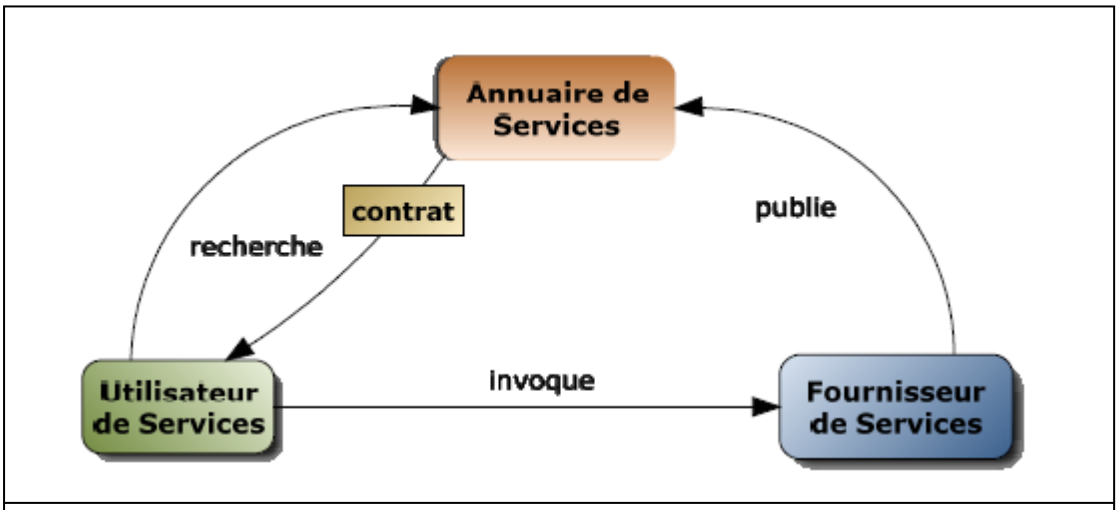


Figure I.1 : Le modèle fonctionnel d'une architecture orientée SOA.

En résumé les services Web s'articulent autour de trois acteurs principaux illustrés par le schéma précédant:

a. **Fournisseur de service :** Le fournisseur de service crée le service web, publie son interface et décrit le service en termes de mise en œuvre ainsi que des informations d'accès au service en WSDL dans un registre de services web UDDI et implémente le service dans un serveur d'applications tel qu'apache.
b. **Utilisateur de services :** C'est n'importe quel consommateur du service Web. Le demandeur utilise un service Web existant en ouvrant une connexion réseau et en envoyant une demande en XML (REST, XML-RPC, SOAP).
c. **Annuaire de services :** Le registre de service est un annuaire de services. Le registre fournit un endroit central où les programmeurs peuvent publier de nouveaux services ou en trouver.

Les interactions entre ces trois acteurs suivent plusieurs étapes :

- **La publication du service** : le fournisseur diffuse les descriptions de ses services Web dans l'annuaire.
- **La recherche du service** : le client cherche un service particulier, il s'adresse à un annuaire qui va lui fournir les descriptions et les URL des services demandés afin de lui permettre de les invoquer.
- **L'invocation du service** : une fois que le client récupère l'URL et la description du service, il les utilise pour l'invoquer auprès du fournisseur de services.

Dans une architecture SOA, les ressources d'un réseau sont rendues disponibles en tant que services indépendants, auxquels on peut accéder sans rien connaître de la façon dont ils sont réalisés. Cette architecture caractérisée par :

- un couplage faible entre les services: implique qu'un service n'appelle pas directement un autre service .En effet les interactions sont gérées par une fonction d'orchestration.
- la réutilisation d'un service et plus facile, du fait qu'il n'est pas directement lié aux autres services de l'architecture dans laquelle il évolue.
- L'indépendance par rapport aux aspects technologiques : est obtenue grâce aux contrats d'utilisation associés à chaque service. En effet, ces contrats sont indépendants de la plate-forme technique utilisée par le fournisseur du service.

- la découverte des services disponibles, afin d'être sûr que les utilisateurs potentiels découvriront le service dont ils ont besoin.
- La mise à l'échelle: est rendue possible grâce à la découverte et à l'invocation de nouveaux services lors de l'exécution.

III. Services Web

III.1. Historique

Le World Wide Web on s'accroit de manière exponentielle depuis les années suivantes [Monfort, 2003] :

- **En 1990** : Ce phénomène est dû en partie aux entreprise qui ont vu dans l'Internet l'outil par excellence pour ce faire connaitre. Mais également aux organismes ayant vocation à fournir un service publique d'information, de proximité ou non.
- **En 1993** : les entreprises et les états ont exploité l'Internet et en particulier les serveurs Web pour faire le commerce électronique ou bien les gouvernements électroniques.
- **En 1998** : les services Web prennent leur origine dans l'informatique distribuée et dans l'événement de Web.

III.2. Définition

Les services Web représentent un domaine de recherche jeune, plusieurs définitions des services Web ont été mises en avant par différents auteurs. Ci-dessous, nous citons quelques définitions généralement acceptées et fournies :

Selon [Justin, 2002]

« Un service web est une agrégation de fonctionnalités publiées pour être utilisées. Il utilise internet comme conduit pour réaliser une tâche. Il est semblable à un processus métier virtuel qui définit des interactions au niveau application »

IBM [Colan, 2003] donne la définition suivante des services Web:

« Les services web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus complexes. Une fois qu'un service Web est déployé, d'autres applications peuvent le découvrir et l'invoquer ».

Par le consortium W3C [W3C-WSA-Group, 2004] :

«Le consortium W3C définit un service Web comme étant une application ou un composant logiciel identifié par un URI, dont ses interfaces et ses liens peuvent être décrits en XML (eXtensible Markup Language), sa définition peut être découverte par d'autres services Web et il peut interagir directement avec d'autres services Web à travers le langage XML et en utilisant des protocoles Internet standards ».

En d'autres mots, les services Web sont des applications auto descriptives, modulaires et faiblement couplées fournissant un modèle simple de programmation et de déploiement d'applications. Ils reposent principalement sur des technologies basées sur XML pour la structure et le contenu de messages échangés entre services (SOAP), pour la description des services (WSDL) et pour la découverte des services (UDDI).

III.3. Caractéristiques des services Web

- **Web based** : les Web services sont basés sur les protocoles et les langages du Web, en particulier HTTP et XML.
- **Self-described, self-contained** : le cadre des Web services contient en lui-même toutes les informations nécessaires à l'utilisation des applications, sous la forme de trois fonctions : trouver, décrire et exécuter.
- **Modular**: les Web services fonctionnent de manière modulaire et non pas intégrée. Cela signifie qu'au lieu d'intégrer dans une seule application globale toutes les fonctionnalités, on crée (ou on récupère) plusieurs applications spécifiques qu'on fait intéropérer entre elles, et qui remplissent chacune une de ces fonctionnalités. Une fonctionnalité développée sous forme de Web services peut dorénavant être réutilisée et recombinée à une suite d'autres fonctionnalités pour composer une nouvelle application.

III.4. Intérêt des services Web

Les technologies des services Web peuvent être appliquées à toute sorte d'applications auxquelles elles offrent de considérables avantages en comparaison aux ancienne API propriétaires, aux implémentations spécifiques à une plate-forme et à quelques autres restrictions classique que l'on peut rencontrer (multi plate-forme, multi langage, disponible sur l'internet avec une information actualisée disponible en temps réel, …).

Les entreprises qui mettent à disposition leurs services Web permettent aux développeurs intéressés par ses fonctionnalités de les réutiliser sans avoir à les recoder.

Le principe des services Web permet d'avoir un partage des fonctionnalités et facilite grandement le développement, leur internet majeur et l'instauration :

- L'interopérabilité des applications.
- L'intégration des applications sur le Web.

III.5. Architecture des services Web

L'exposition et l'utilisation des services se fait dans un contexte particulier qui définit clairement les interactions entre le service et ses utilisateurs. Les services Web communiquent via un ensemble de technologies fondamentales qui partagent une architecture commune. Ils ont été conçus pour être réalisés sur de nombreux systèmes développés et déployés de façon indépendante. Les technologies utilisées par les services Web sont HTTP, WSDL, XML-RPC, SOAP et UDDI.

L'architecture standard d'un service Web est organisée en plusieurs couches. Chacune d'elles répond à des préoccupations fonctionnelles différentes telles que la publication, la description, la messagerie et le transport. Comme illustré sur la Figure I.2 [Hubert et al., 2003].

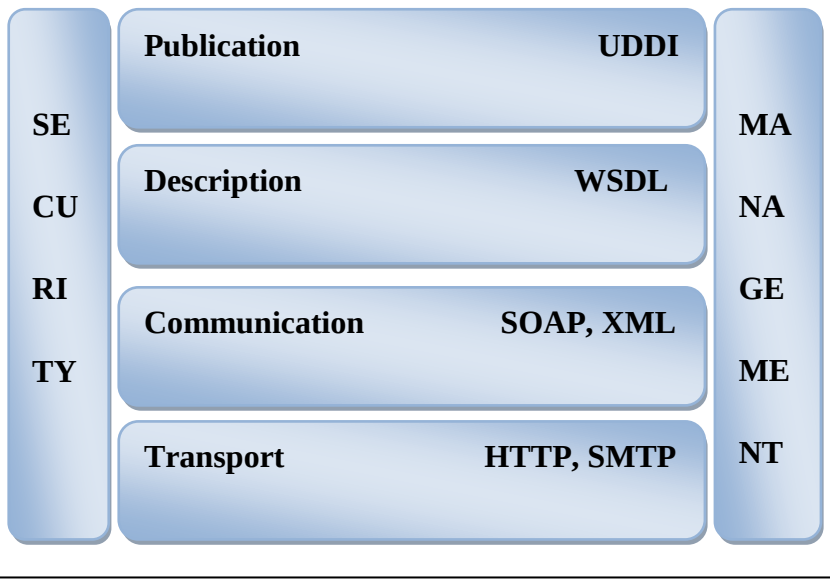


Figure I.2 : Architecture du service Web.

Les différentes couches de l'architecture d'un service Web s'interfacent avec des standards, comme suit:

- **La couche de publication:** repose sur le protocole UDDI (Universal Description, Discovery and Integration), qui assure le regroupement, le stockage et la diffusion des descriptions des services Web.
- **La couche description:** est prise en charge par le langage WSDL (Web Service Description Language) [Christensen et al., 2001], qui décrit les fonctionnalités fournies par le service Web, les messages reçus et envoyés pour chaque fonctionnalité, ainsi que le protocole utilisé pour la communication.
- **La couche communication:** la couche de communication des messages propose différents mécanismes liés à l'acheminement des messages (format de communication des messages, adressage, routage...etc.). Cette couche utilise des protocoles reposants sur le langage XML, car sa syntaxe unique résout les conflits syntaxiques lors de l'encodage des données. Actuellement, SOAP (Simple Object Access Protocol) est le protocole le plus utilisé pour cette couche.
- **La couche transport:** le protocole le plus utilisé dans cette couche est l'HTTP (Hyper Text Transfert Protocol). Cependant, d'autres protocoles peuvent être utilisés, tels que le SMTP (Simple Mail Transfer Protocol) ou le FTP (File Transfer Protocol), permettant ainsi aux services Web de rester indépendants du mode de transport utilisé.

III.6. Standards de services Web

Une caractéristique qui a permis un grand succès de la technologie des services web est qu'elle est construite sur des technologies standards, nous présentons dans cette section les trois standards de base, à savoir, SOAP (Simple Object Access Protocol), WSDL (Service web Description Language) et UDDI (Universal Description, Discovery and Integration) .Ces trois standards sont basés sur le langage XML (eXtansible Markup Language).

III.6.1. SOAP

Les communications entre les différentes entités impliquées dans le dialogue avec le service web se font par l'intermédiaire du protocole SOAP (*Simple Object Access Protocol*). Ce protocole est normalisé par le W3C.

Principe

SOAP, est un standard du Consortium W3C définissant un protocole de transmission de messages permettant la normalisation des échanges de données. Il présente un ensemble de règles pour structurer des messages, qui peuvent être utilisées dans de simples transmissions unidirectionnelles, mais il est particulièrement utile pour exécuter des dialogues requête-réponse RPC (Remote Procedure Call) en utilisant http comme protocole de communication, mais aussi les protocoles SMTP (Simple Mail Transport Protocol) et POP (Post Office Protocol) [Mitra et al., 2003].

SOAP assure l'interopérabilité entre composants tout en restant indépendant des systèmes d'exploitation et des langages de programmation, donc, théoriquement, les clients et serveurs de ces dialogues peuvent fonctionner sur n'importe quelle plate-forme et être écrits dans n'importe quel langage à partir du moment où ils peuvent formuler et comprendre des messages SOAP. Il représente donc un composant de base pour développer des applications distribuées, qui exploitent des fonctionnalités publiées comme services par des intranets ou Internet.

SOAP utilise principalement les deux standards HTTP et XML :

- HTTP comme protocole de transport des messages SOAP. Il constitue un bon moyen de transport en raison de sa popularité sur le web.
- XML pour structurer les requêtes et les réponses, indiquer les paramètres des méthodes, les valeurs de retours, et les éventuelles erreurs de traitements.

Structure du message SOAP

Les messages échangés lors de l'utilisation du protocole SOAP sont basés sur le langage XML. Ils sont composés de deux parties, l'en-tête de protocole de transport et l'enveloppe SOAP (la Figure I.3).

Figure I.3 : Structure de message SOAP.

1. **L'en-tête du protocole de transport:** qui dépend de protocole de transport utilisé, par exemple si le protocole HTTP est utilisé, l'en-tête contient :
   - La version de protocole HTTP utilisée.
   - La date de génération de message SOAP.
   - Le type d'encodage du contenu (généralement de type XML).
2. **L'enveloppe SOAP :** la partie principale d'un message SOAP est l'enveloppe (symbolisée par la balise enveloppe), cette dernière est subdivisée en deux sous-parties: la partie en-tête (Header) et la partie corps du message (Body).
   - **L'en-tête du message SOAP:** est optionnelle et extensible. Les balises XML qui permettent de symboliser cette partie sont: **<env:Header>** et **</env:Header>.** Ces balises peuvent être complétées par des attributs permettant de définir le domaine de noms du service Web. En fait, l'en-tête permet principalement d'ajouter des informations sur le comportement des différents nœuds intermédiaires, lors de traitement du message.

Un nœud étant un intermédiaire SOAP, incluant le récepteur et l'émetteur SOAP, désignable depuis un message SOAP. Son rôle est de traiter l'en-tête, ensuite de transférer le résultat (le message SOAP modifié) à un autre intermédiaire (qui peut être le récepteur final).

Les principaux attributs des éléments qui forment le bloc d'en-tête sont :

env:role: permet d'indiquer à quel nœud la fonction décrite est destinée.

env:mustUnderstand: c'est une valeur booléenne, elle permet de préciser que le traitement devient obligatoire pour un nœud intermédiaire, par exemple, pour un calcul très long, il peut être utile d'envoyer un e-mail à chaque étape.

env:relay: cet attribut permet de relayer un message à un autre nœud si le premier nœud n'est pas capable de le traiter.

- **Le corps du message SOAP:** l'élément corps de message SOAP contient des données spécifiques à l'application. Les balises symbolisant cette partie sont: **<env:Body>**et **</env:Body>.** Les données doivent donc être sérialisées selon l'encodage XML. En plus, des données de cette partie peuvent transporter un type spécial comme: les messages d'erreurs (SOAP Fault).

III.6.2.UDDI

UDDI (Universel Description, Discovery and Integration) est un standard défini par OASIS. Il définit la structure d'un annuaire de services Web, et la structure de gestion de services (publication, localisation, découverte) Sous forme de répertoire, il permet de stocker les informations nécessaires pour retrouver et accéder à un service, telles que les informations techniques et l'adresse des services Web, le nom de la personne/société qui gère un service donné, la description des fonctionnalités [Clement et al., 2004].

Un service d'annuaire UDDI est un service Web qui gère les méta-données des services, l'information sur les fournisseurs de services et les implémentations des services. Afin de trouver un service Web, il est possible d'utiliser un annuaire UDDI en précisant des exigences concernant service requis. On cherche le service par son nom et/ou par des mots clés.

Consultation de l'annuaire

L'annuaire UDDI se concentre sur le processus de découverte de l'architecture orientée services (SOA), et utilise des technologies standards telles que XML, SOAP et WSDL qui permettent de simplifier la collaboration entre partenaires dans le cadre des échanges commerciaux. L'accès au référentiel s'effectue de différentes manières.

Les informations sur un service publié dans un annuaire UDDI se présentent sous trois facettes comme illustre dans la figure I.4 [Hubert et al., 2003] :

- Les pages blanches comprennent la liste des entreprises ainsi que des informations associées à ces dernières (coordonnées, description de l'entreprise, identifiants...).
- Les pages jaunes recensent les services Web de chacune des entreprises sous le standard WSDL.
- Les pages vertes fournissent des informations techniques précises sur les services fournis.

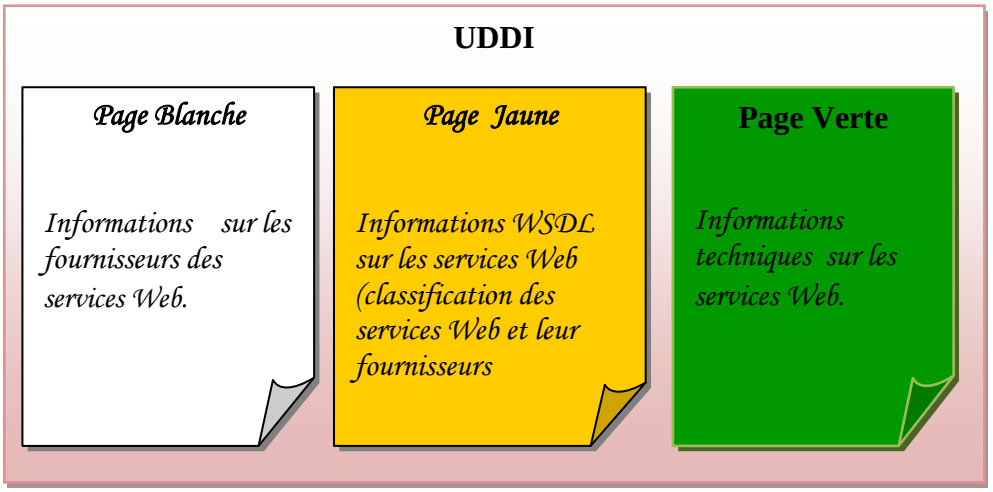


Figure I.4 : les trois facettes de l'annuaire UDDI [Hubert et al., 2003].

Un annuaire UDDI est conçu pour être interrogé par des messages SOAP et afin de pouvoir stocker et fournir des informations permettant l'accès aux documents WSDL (Web Services Description Language) décrivant les protocoles et les formats de messages nécessaires pour interagir avec les services Web répertoriés dans l'annuaire ,Les outils de recherche disponibles sont basés sur des mots-clés et ne prennent pas en considération les relations entre les services Web et les caractéristiques sémantiques de chaque service Web, forçant l'utilisateur à recommencer la recherche depuis le début en utilisant de nouveaux termes clés.

Les spécifications UDDI inclue notamment :

- des API SOAP qui permettent l'interrogation et la publication d'informations,
- la représentation XML du modèle de données de l'annuaire et des formats de message SOAP
- des définitions de l'interface WSDL de SOAP,
- des définitions d'APIs de différents modèles techniques qui facilitent le travail des systèmes d'identification et de catégorisation des enregistrements UDDI.

Un registre UDDI se compose de quatre types de structures de données [Arnaud, 2005] : le **businessEntity**, le **businessService**, le **bindingTemplate** et la **tModel**. Cette répartition par type fournit des partitions simples pour faciliter la localisation rapide et la compréhension des différentes informations qui constituent un enregistrement.

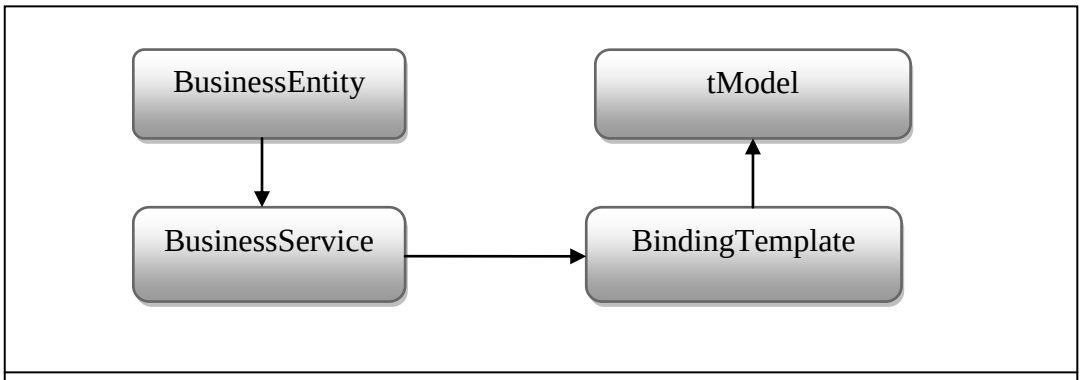


Figure I.5 : Structures de données de l'annuaire UDDI [Arnaud, 2005].

- **BusinessEntity (entité d'affaires)**

Les « businessEntities » sont en quelque sorte les pages blanches d'un annuaire UDDI. Elles décrivent les organisations ayant publié des services dans le répertoire. On y trouve notamment le nom de l'organisation, ses adresses (physiques et Web), des éléments de classification, une liste de contacts ainsi que d'autres informations.

- **BusinessService (service d'affaires)**

Les « businessServices » sont en quelque sorte les pages jaunes d'un annuaire UDDI. Elles décrivent de manière non technique les services proposés par les différentes organisations. On y trouve essentiellement le nom et la description textuelle des services ainsi qu'une référence à l'organisation proposant le service et un ou plusieurs « bindingTemplate ».

- **BindingTemplate (modèle de rattachement)**

UDDI permet de décrire des services Web utilisant HTTP, mais également des services invoqués par d'autres moyens (SMTP, FTP...). Les « bindingTemplates » donnent les coordonnées des services. Ce sont les pages vertes de l'annuaire UDDI. Ils contiennent notamment une description, la définition du **point d'accès** (une URL) et les éventuels «tModels » associés.

- **tModel (index)**

Les « tModels » 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.

III.6.3.WSDL

Le besoin d'une description claire de la communication entre services Web a abouti à des normalisations aux niveaux des messages échangés et des protocoles. Une grammaire XML bien structurée était proposée pour décrire les services Web et paramétrer les échanges de messages. Un langage de description s'est basé sur cette grammaire et est devenu le standard des services Web, c'est le "Web Service Description Language" (WSDL).

Principes du standard

WSDL [Christensen et al., 2001], est un standard du W3C, qui permet de définir une syntaxe XML pour décrire les méthodes et les paramètres des services Web invocables par le biais de messages au format SOAP. Il permet de définir qu'est-ce qu'un service Web est capable de faire, son emplacement et comment l'invoquer.

Eléments de WSDL

Le standard offre une description à deux parties : une première partie de description abstraite et une deuxième partie Les éléments de description représentent des concepts du modèle WSDL.

Les éléments de la partie abstraite décrivent principalement le service en termes de messages (envoyés et reçus). Ils sont indépendamment décrits par un schéma XML de description concrète.

Au niveau d'une balise XML notée "types", la partie abstraite inclut des éléments notés "operation" dont chacune associe un ou plusieurs messages à un modèle d'échange de messages. Le modèle spécifie le nombre et l'ordre ainsi que la cible et la source abstraites des messages. Les opérations sont groupées dans un élément noté "interface" ou aussi portType" sans préciser le moyen (le protocole) d'échange de messages.

Techniquement, la structure d'un élément "binding" ressemble à la structure d'un élément "interface/portType" mais avec des détails en plus. La spécification WSDL offre deux types de "binding" pour l'envoi de messages entre client et services Web :

- soit intégrés dans des messages du protocole SOAP transmis par exemple en HTTP,
- soit directement par exemple dans des messages du protocole HTTP.

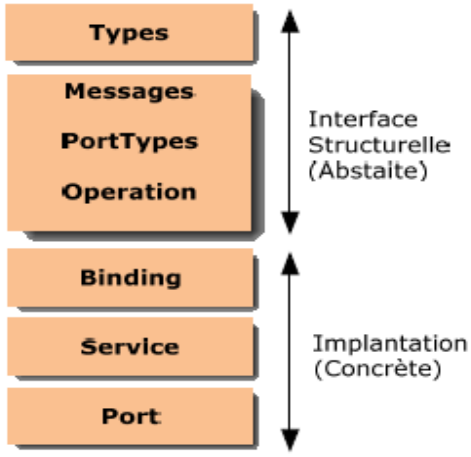


Figure I.6 : Structure d'une description WSDL.

En résume un fichier WSDL contient donc sept éléments.

- **Types** : fournit la définition de types de données utilisés pour décrire les messages échangés.
- **Messages** : représente une définition abstraire (noms et types) des données en cours de transmission.
- **PortTypes** : décrit un ensemble d'opérations. Chaque opération a zéro ou un message en entrée, zéro ou plusieurs messages de sortie ou d'erreurs.

- **Binding** : spécifie une liaison entre un <portType> et un protocole concret (SOAP, HTTP...).
- **Service** : indique les adresses de port de chaque liaison.
- **Port** : représente un point d'accès de services défini par une adresse réseau et une liaison.
- **Opération** : c'est la description d'une action exposée dans le port.

IV. Quelques domaines d'application de services Web

- Les services Web peuvent être utiles dans la plupart des scénarios applicatifs lorsque la communication peut être établit sur un modèle bidirectionnel (requête/réponse).
- L'application des services Web est multiple, autant dans les domaines du B2B, B2C que pour les domaines de gestion de stock, etc.
- B2C (Business to Consumer) : qualifie une application, un site internet destiné au grand public.
- B2B (Business to Business) : qualifie une application, un site internet destiné au commerce professionnel à professionnel.

V. Composition de services Web

La composition de services Web est la plus importante fonctionnalité assurée par une architecture SOA. Celle-ci offre un environnement homogène pour la composition dans la mesure où toutes les parties de la composition sont des services idéalement décrits de la même façon et communiquant par les mêmes standards d'échange de messages.

Dans cette section nous allons présenter le concept de composition de services Web ainsi que les deux approches de composition l'orchestration et la chorégraphie.

V.1. Définition de composition de services Web

Une composition de services Web est constituée de plusieurs services qui interagissent les uns avec les autres [Aît-Bachir, 2008], afin d'offrir de nouvelles fonctionnalités qu'un seul service ne pourrait pas les offrir.

La composition permet de combiner des services pour former un nouveau service dit composé ou composite. L'exécution d'un service composé implique des interactions avec des services partenaires en faisant appel à leurs fonctionnalités. Le but de la composition est avant tout la réutilisation de services (simples ou composés) et de préférence sans aucune modification de ces derniers.

V.2. Les types de composition de services Web

D'après [Dumez, 2010] et [Lopez-velasco, 2008], Nous pouvons distinguer deux types de composition de services Web :

- **Orchestration de services Web :**

L'orchestration de services Web résulte un nouveau service Web dit service Web composé, qui peut être défini comme l'agrégation de plusieurs autres services Web atomiques ou composés. Ce service composé contrôle la collaboration entre les services Web engagés dans la composition, tel qu'un chef d'orchestre [Dumez, 2010].

L'orchestration de services Web exige de définir l'enchaînement des services Web selon un canevas prédéfini, et de les exécuter selon un script d'orchestration. Ces derniers (le canevas et le script) décrivent les interactions entre services Web en identifiant les messages, et en spécifiant la logique et les séquences d'invocation. Le module exécutant le script d'orchestration de services Web est appelé un moteur d'orchestration. Ce dernier est une entité logicielle qui joue le rôle d'intermédiaire entre les services, en les appelants suivant le script d'orchestration [Lopez-velasco, 2008].

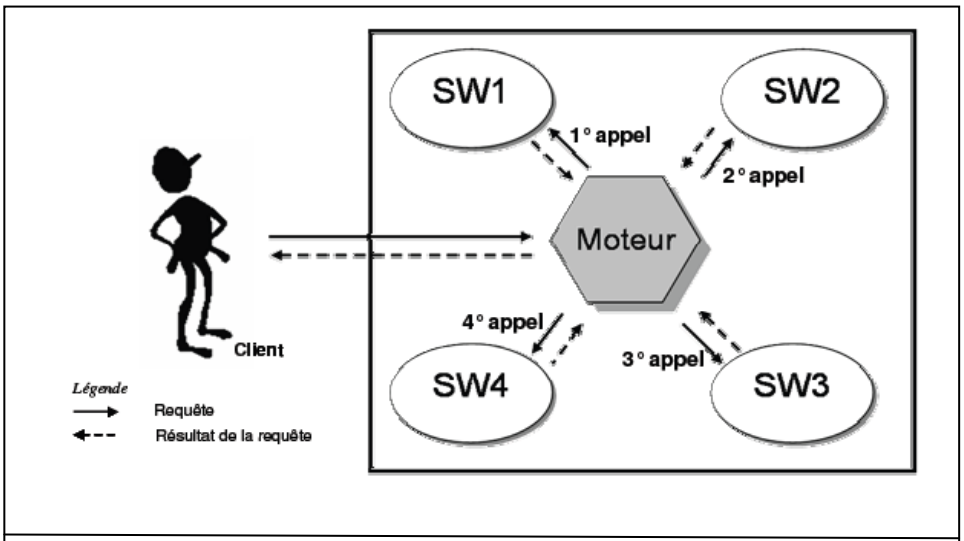


Figure I.7 : Vue générale de l'orchestration [Lopez-velasco, 2008].

- **Chorégraphie de services Web :**

La chorégraphie de services Web est une généralisation de l'orchestration qui consiste à concevoir une coordination décentralisée des services Web. Dans une chorégraphie, les interactions de type pair-à-pair (P2P) sont décrites dans un langage de description de chorégraphie (CDL). Les services suivent alors le scénario global de composition sans point de contrôle central [Dumez, 2010].

La chorégraphie est aussi appelée composition dynamique. En effet, l'exécution n'est pas régie de manière statique comme dans une composition de type orchestration. Dans une chorégraphie à chaque pas de l'exécution, le service Web choisit le service Web qui lui succède et implémente ainsi une partie de la chorégraphie. La composition de type chorégraphie n'est pas connue, ni décrite à l'avance [Lopez-velasco, 2008].

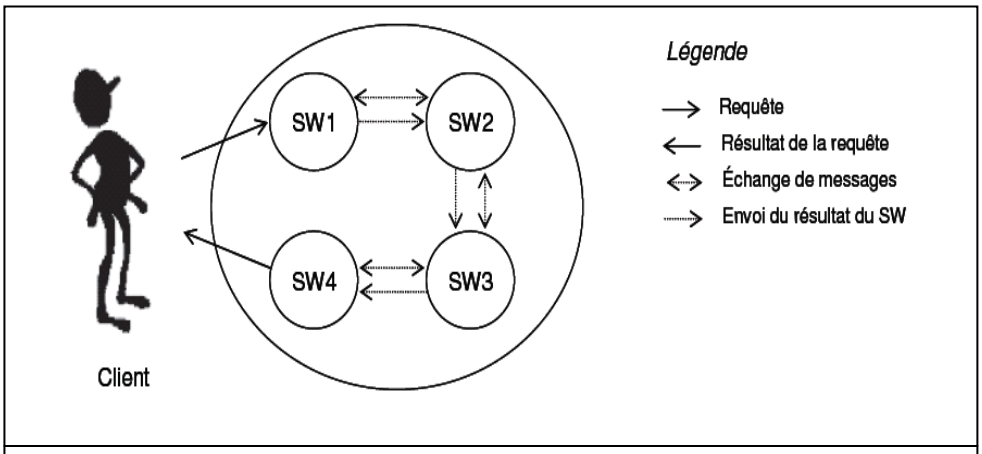


Figure I .8: vue générale de la chorégraphie [Lopez-velasco, 2008].

VI. Avantages et inconvénients des services Web

VI.1.Avantages

L'utilisation des services web engendre plusieurs avantages dont on peut citer :

- **L'interopérabilité** : C'est la capacité des services web d'interagir avec d'autres composantes logicielles via des éléments XML et utilisant des protocoles de l'Internet.

- **La simplicité** : Les services web réduisent la complexité des branchements entre les participants. Cela se fait en ne créant la fonctionnalité qu'une seule fois plutôt qu'en obligeant tous les fournisseurs à reproduire la même fonctionnalité à chacun des clients selon le protocole de communication supporté.
- **Une composante logicielle légèrement couplée** : L'architecture modulaire des services Web, combinée au faible couplage des interfaces associées, permet l'utilisation et la réutilisation de services qui peuvent facilement être recombinés à différents autres applications.
- **L'hétérogénéité** : Les services web permettent d'ignorer l'hétérogénéité entre les différentes applications. En effet, ils décrivent comment transmettre un message (standardisé) entre deux applications, sans imposer comment construire ce message.
- **Auto-descriptivité** : Les services web ont la particularité d'être auto-descriptifs, c'est à dire capables de fournir des informations permettant de comprendre comment les manipuler. La capacité des services à se décrire par eux-mêmes permet d'envisager l'automatisation de l'intégration de services.

VI.2. Inconvénients

- Les services Web ont de faibles performances par rapport aux autres approches de l'informatique répartie telles que le RMI, CORBA, ou DCOM.
- En l'utilisation du protocole http, les services Web peuvent contourner les mesures de sécurité miss en place à travers les firewalls.
- La sémantique n'est pas prise en charge de façon efficace car le WSDL décrit les services de manière syntaxique.
- Les transferts reposent sur le XML, ce qui pose un problème sur la taille des fichiers échangés. Les fichiers XML sont le plus souvent de très gros fichiers et ceci entrainera une lourdeur considérable.
- Ils ne sont pas sécurisés à100 %.

VII. Conclusion

A travers ce que nous avons présenté dans ce chapitre, il ressort que les services Web sont le résultat de la collaboration exceptionnelle des joueurs majeurs des technologies de l'information qui se sont entendus sur un certain nombre de protocoles et d'approches qui favoriseront l'interopérabilité entre les plates formes ,les systèmes d'exploitation, les langages et les programmes.

A fin que les services Web répondent aux exigences complexes des utilisateurs, une composition de ces services est nécessaire. Elle permet de créer, d'exécuter, de maintenir des services qui reposent sur d'autres services et ceci de façon efficace.

La composition des services web peut se faire de deux manières différentes qui sont l'orchestration et la chorégraphie, telle que la différence majeure entre l'orchestration et la chorégraphie est que l'orchestration offre une vision centralisée de la composition, alors que la chorégraphie offre une vision globale et plus collaborative de la coordination.

Après ce panorama des technologies et standards qui constituent la base du service Web, le chapitre suivant est consacré à étudier les ontologies afin de décrire la sémantique des services Web.

Chapitre II

Les ontologies

I. Introduction

L'utilisation de connaissances a pour but primordiale de permettre un dialogue, une coopération entre le système et les utilisateurs. Pour cela, le système doit avoir accès non seulement aux termes utilisés par l'être humain mais également à la sémantique qui leur est associée, afin qu'une communication efficace soit possible. En premier lieu, les ontologies sont devenues très populaires, en occupant une place de choix dans le domaine de l'ingénierie des connaissances et ont prouvé leur efficacité pour la représentation des connaissances.

Dans le cadre de services Web, la technologie d'ontologie est utilisée pour décrire sémantiquement les services Web d'une part, et d'une autre part la représentation sémantique des données échangées entre les services Web est nécessaire pour les rendre explicites et compréhensibles par les machines. La combinaison des deux technologies, de services web et de l'ontologie a donné naissance de *services Web sémantiques.*

Les services Web sémantiques sont des services Web simples enrichis par des descriptions supplémentaires définies dans des ontologies, explicites et compréhensibles par les machines. Les services Web sémantiques ont pour objectif de faciliter l'automatisation des taches issues des services Web telle que la découverte, la sélection, la composition et l'invocation de services Web.

Dans ce chapitre nous nous intéressons aux ontologies et services Web sémantiques. Nous présentons dans ce chapitre les ontologies, les concepts de base liés aux ontologies, ainsi que leurs approches de construction, enfin une étude de services Web sémantique est effectuée.

II. Les ontologies

Les ontologies constituent un enjeu stratégique dans la représentation et la modélisation des connaissances. Elles sont considérées comme un axe central pour diverses communautés particulièrement dans l'ingénierie des connaissances, la recherche d'information, le e-commerce et le web sémantique. Elles définissent les primitives indispensables pour la représentation des systèmes, ainsi que leur sémantique dans un contexte particulier.

Dans cette première partie du chapitre, nous nous attachons à décrire les différentes définitions du concept d'ontologie, ainsi que les différents éléments constituant l'ontologie. Un rapide aperçu des formalismes de représentation d'ontologies est ensuite donné. Puis, nous passons en revue les différentes étapes intervenant dans la construction des ontologies. Un résumé des principaux langages utilisés est présenté.

II.1. Définition d'ontologie

Le mot ontologie trouve sa racine du grec onto (le participe présent du verbe être) qui est l'étude de l'être en tant qu'être et logos qui signifie discours. Historiquement, l'ontologie est un concept philosophique. Il désigne la science de l'être en général. Plus tard, l'ontologie est apparue en pleine lumière dans le domaine de l'intelligence artificielle, afin de résoudre les problèmes de modélisation des connaissances. Ceci a engendré de nombreuses définitions dans la littérature.

En résumé, Gruber [Gruber, 1993] introduit la notion d'ontologie comme "une spécification explicite d'une conceptualisation". Cette définition a été légèrement modifiée par Borst [Borst, 1997]. Une combinaison des deux définitions peut être résumée ainsi : « une spécification explicite et formelle d'une conceptualisation partagée ». Cette définition s'explique ainsi [Studer, 1998] : explicite signifie que le « type des concepts et les contraintes sur leurs utilisations sont explicitement définies », formelle se réfère au fait que la spécification doit être lisible par une machine, partagée se rapporte à la notion selon laquelle une ontologie « capture la connaissance consensuelle, qui n'est pas propre à un individu mais validée par un groupe », conceptualisation se réfère à « un modèle abstrait d'un certain phénomène du monde reposant sur l'identification des concepts pertinents de ce phénomène ».

II.2. Eléments constitutifs de l'ontologie

Les ontologies permettent de représenter les connaissances et de les manipuler automatiquement tout en gardant leur sémantique. Ces connaissances sont véhiculés à l'aide de certain nombre de composantes sont principalement des concepts, des relations, des fonctions, des axiomes et instances [Gruber, 1993].

- **Les concepts,** aussi appelés termes ou classes de l'ontologie. Selon Uschlod et Grüninger [Uschold et Grüninger, 1996], un concept peut représenter un objet matériel, une notion ou une idée, ils constituent les objets de base manipulés par les ontologies et correspondent aux abstractions pertinentes du domaine du problème.
- **Les relations,** traduisent les interactions existent entre les concepts. Des relations comme les relations de spécification (subsomption), de composition (meronymie), d'instanciation, etc. sont définies entre les concepts.
- **Les fonctions,** sont des cas particuliers de relations dans lesquelles le $n^{ième}$ élément de la relation est défini de manière unique à partir des $(n-1)^{ième}$ éléments précédents.
- **Les axiomes,** permettent de modéliser des assertions toujours vraies à propos des abstractions du domaine traduites par l'ontologie, ils permettent aussi de combiner des concepts, des relations, des fonctions pour définir des règles d'inférences. ces axiomes peuvent intervenir dans la déduction ainsi que la définition des relations et de concepts.
- **Les instances,** (encore appelées individus) constituent la définition extensionnelle de l'ontologie. Ils représentent des éléments singuliers véhiculant les connaissances à propos du domaine du problème.

II.2.1. Propriétés des concepts

Un concept est généralement défini par un ou plusieurs termes, une intention et une extension. Le terme (le nom) correspond à l'identité du concept, l'intention (la notion) du concept contient la sémantique de concepts exprimée en termes de propriétés, d'attributs, de règles et de contraintes. L'extension (ensemble d'objets) du concept regroupe les objets manipulés à travers le concept ; ces objets sont appelés instances du concept [Bachimont, 2000].

Un concept est caractérisé par un ensemble de propriétés qui peuvent porter aussi bien sur l'extension que sur l'intention. Les plus intéressantes sont [Furst, 2002] :

- **La généricité :** un concept est générique s'il n'admet pas d'extension.
- **L'identité :** un concept porte une propriété d'identité si cette propriété permet de différencier deux instances de ce concept. Par exemple, dans un système de gestion de fichier, un nom désigne d'une manière unique un fichier ou un répertoire. Le nom est une identité du fichier ou répertoire.
- **La rigidité :** un concept est rigide s'il ne peut pas être une instance d'autres concepts. Par exemple, l'être vivant est un concept rigide, mais un " être humain " n'est pas un concept rigide, car l'humain est une instance du concept " être vivant ".
- **L'anti-rigidité :** un concept est anti-rigide s'il peut être une instance pour d'autres concepts. Comme le cas de l'être humain dans l'exemple précédent.

En plus de ces propriétés, d'autres propriétés inter-concepts portent sur des propriétés extrinsèques aux concepts, elles sont principalement [Furst, 2002] :

- **L'équivalence :** deux concepts sont équivalents s'ils ont la même extension.
- **La disjonction :** deux concepts sont disjoints ou incompatibles si leurs extensions sont disjointes.
- **La dépendance :** un concept est dépendant d'un autre concept si les instances du premier sont dépendantes des instances de l'autre.

II.2.2. Propriétés de relations

Comme les concepts, les relations sont aussi menées des propriétés. Elles sont principalement les propriétés intrinsèques, les propriétés interrelations et les propriétés liant une relation et un concept [Furst, 2002] et [Izza, 2006] :

Les propriétés intrinsèques à une relation permettent de décrire une relation. Nous distinguons :

- Les propriétés algébriques : la symétrie, la réflexivité et la transitivité.
- La cardinalité : le nombre de participations possibles d'une instance d'un concept dans une relation.

Les propriétés interrelations portant sur plusieurs relations, qui peuvent être :

- L'incompatibilité : deux relations sont incompatibles si elles ne peuvent pas lier les mêmes instances d'un concept.

- L'inverse : deux relations sont inverses l'une de l'autre si la liaison est faites dans les deux sens.

Les propriétés liant une relation et un concept sont :

- Le lien relationnel : il existe un lien relationnel entre une relation R et deux concepts C1 et C2 si, pour tout couple d'instances des concepts C1 et C2, il existe une relation de type R qui lie les deux instances de C1 et C2.
- La restriction de relation : pour tout concept de type C1 et pour toute relation de type R liant C1, les autres concepts liés par la relation sont d'un type imposé.

II.3. Typologie des ontologies

Les ontologies peuvent être classifiées selon plusieurs critères. D'après [Psyché et al., 2003] quatre catégories sont citées en fonction de (Cf. la figure II.1):

- L'objet de conceptualisation,
- Le niveau de détail,
- Le niveau de complétude,
- Le niveau de formalisme de représentation.

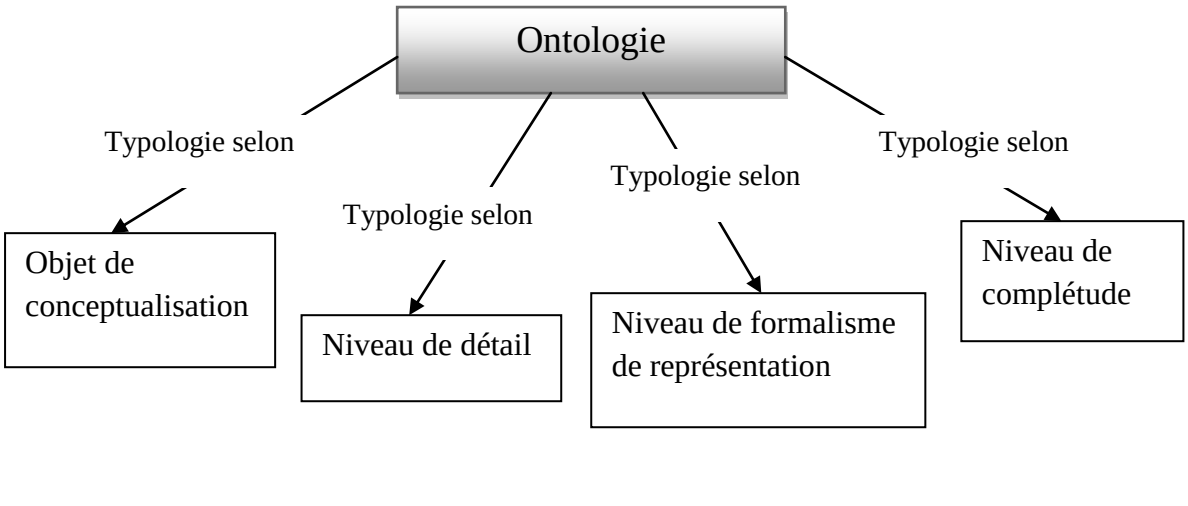


Figure II.1 : Typologies d'ontologies [Psyché et al., 2003].

Une autre classification donnée par Guarino [Guarino, 1998] définir aussi quatre types d'ontologies, à savoir (Cf. la figure II.2) :

- Les ontologies supérieures ou de Haut niveau,
- Les ontologies de domaine,
- Les ontologies de tâche,
- Les ontologies d'application.

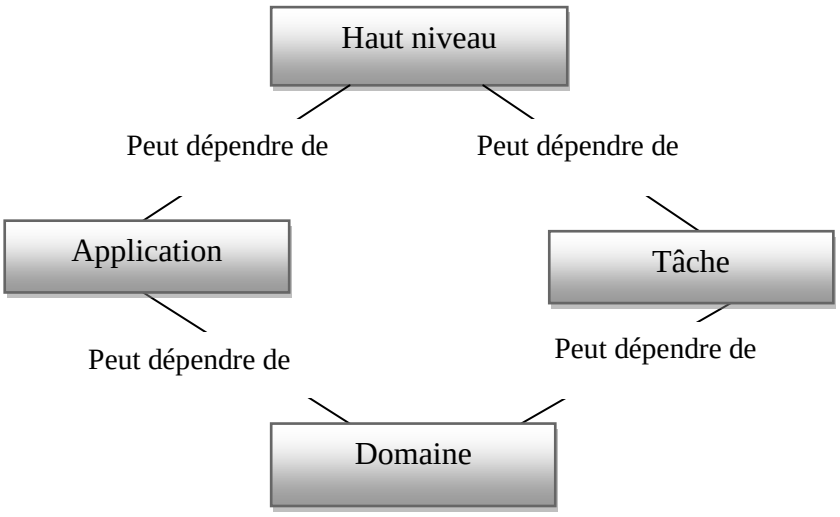


Figure II.2 : Typologies d'ontologies [Guarino, 1998].

II.4. Construction des ontologies

La construction des ontologies est une tâche très complexe, impliquant plusieurs acteurs, diverses connaissances et formalismes. Afin de maîtriser cette complexité, la mise en place d'une méthodologie peut se révéler nécessaire. L'utilisation d'une méthodologie a pour objectif de rationaliser et d'optimiser le processus de développement dans le sens où il permet de réduire les délais et les coûts ainsi que l'amélioration de la qualité du processus et du produit de développement.

À l'heure actuelle, nous pouvons recenser dans la littérature une multitude de méthodologies de construction [Bala, 2007]. Cependant, il n'existe pas encore de consensus en matière de normes de construction. Et par conséquent, il n'existe pas encore de méthodes universellement reconnue pour la construction d'ontologies [Uschold et al., 1996]. Ceci relève beaucoup plus du savoir-faire que de l'ingénierie. En dépit de cela, de nombreux critères et principes permettant de guider la construction d'ontologies ont été proposés. Certains sont décrits dans les prochaines sections.

II.4.1. Cycle de développement ontologique

Le développement des ontologies doit s'appuyer sur les mêmes principes que ceux appliqués en génie logiciel. Elles doivent avoir, donc, un cycle de vie qui nécessite d'être spécifié.

Quelque soit les méthodologies de construction utilisées, le cycle de vie ontologique comprend les étapes suivantes [Furst, 2002] et [Izza, 2006]:

- Evaluation des besoins.
- Construction (normalisation, formalisation et population).
- Diffusion.
- Utilisation (Cf. la Figure II.3).

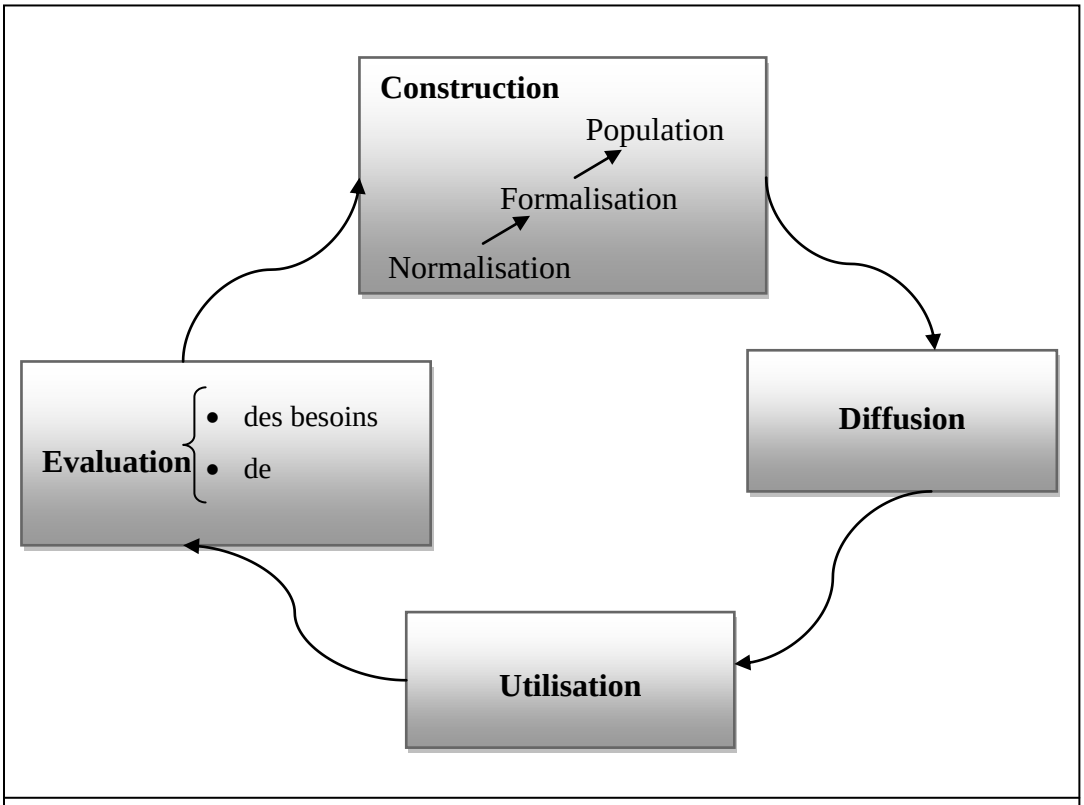


Figure II.3 : Cycle de vie d'une ontologie.

La première étape est celle de l'évaluation. Elle touche les besoins afin d'identifier l'objectif, la portée et les limitations de l'ontologie à construire, et touche aussi l'ontologie pour faire un jugement technique de cette dernière. La deuxième est de construction, elle s'agit de l'étape la plus longue et la plus difficile, contient elle-même trois sous-étapes, la normalisation, la formalisation et la population. La normalisation sert à définir les concepts du domaine et les relations entre eux, de manière à ne pas être ambiguës. La formalisation consiste à représenter l'ontologie dans un langage formel, la formalisation peut être de différents degrés (très informel, semi-informel, semi-formel ou rigoureusement formel). Enfin l'étape optionnelle de population sert à associer des instances à l'otologie crée. En suite l'étape de diffusion permette de mettre à disposition l'ontologie à ses utilisateurs. Enfin l'utilisation correspond à l'exploitation par ces utilisateurs des informations fournies.

II.4.2. Principes pour la construction des ontologies

Plusieurs principes ont été proposés dans la littérature afin de mieux guider la construction des ontologies. Il s'agit essentiellement, des principes de Gruber [Gruber, 1993] :

- **Clarté et objectivité :** l'ontologie devrait fournir des définitions claires et objectives pour les termes, indépendantes de tout choix d'implémentation.
- **Cohérence :** les raisonnements construits à partir des axiomes d'une ontologie ne doivent pas aboutir à des contradictions ;
- **Extensibilité:** c'est à dire la possibilité d'étendre l'ontologie sans modification.
- **Le biais d'encodage minimum :** la spécification de l'ontologie doit être aussi indépendante que possible d'un métalangage particulier de représentation.
- **L'engagement ontologique minimal :** l'objectif est de permettre la spécialisation des spécifications d'une ontologie donnée selon des besoins réels.

II.4.3. Méthodologies de construction des ontologies

Une méthodologie étant considérée comme ensemble de principes de construction systématiquement reliés, appliqués avec succès par un auteur (ou plusieurs) dans la construction d'ontologies. Selon [Bala, 2007], il existe une trentaine de méthodologies de développement d'ontologies.

Ces méthodologies permettent la construction d'ontologies :

- à partir du début,
- par intégration ou fusion avec d'autres ontologies,
- par réingénierie,
- par construction collaborative,
- et l'évaluation des ontologies construites.

La méthodologie proposée par Uschold [Uschold et al., 1996] est considérées comme étant la plus populaire, elle est composée de quatre principales étapes sont : identifier le but et la portée de l'ontologie, construire l'ontologie, évaluer l'ontologie et documenter l'ontologie.

II.4.4. Formalismes de représentation des ontologies

Les ontologies ont besoin d'être représentées formellement. Plus encore, elles doivent représenter l'aspect sémantique des relations liant les concepts. A cet effet, de nombreux formalismes ont été développés. Nous nous concentrons sur les plus utilisés [Lortal, 2002].

- **Les réseaux sémantiques**

Un réseau sémantique est une représentation graphique d'une conceptualisation d'une (ou plusieurs) connaissance humaine [Quillian, 1968]. Il est représenté sous la forme d'un graphe qui encode les connaissances ainsi que leurs propriétés. Les nœuds du graphe représentent des objets (concepts, situations, événements, etc.) et les arcs expriment des relations entre ces objets. Ces relations peuvent être des liens " sorte - de " exprimant la relation d'inclusion ou des liens " est-un " représentant la relation d'appartenance. De nombreuses études [Woods, 1975], [Brachman, 1977] ont montré que ce type de graphe manque de précision sémantique et mène à des confusions entre les relations et aussi entre les classes et individus. Elles ont mené à la définition de nouveaux formalismes tels que les frames, les logiques de description et les graphes conceptuels.

- **Les frames**

Les frames [Minsky, 1975] sont présentés comme étant une structure de données capable de représenter des objets structurés et également une ontologie d'après Gruber [Gruber, 1993]. Un frame représente donc une classe ou un objet. Les frames sont organisés dans une hiérarchie suivant un lien de spécification.

Les composants du frame sont appelés « slots », ils sont considérés comme des attributs de la structure. Chaque attribut peut prendre ses valeurs parmi un ensemble de facettes (facets) [Kifer et al., 1995]. L'intérêt des frames est qu'ils permettent de représenter la façon de penser d'experts en fournissant une représentation structurée et concise des relations utiles [Fikes, 1985]. L'information peut être partagée entre plusieurs frames grâce à l'héritage.

- **Les graphes conceptuels**

Les graphes conceptuels ont été mis au point par John F. Sowa pour modéliser une ontologie de haut niveau [SOWA, 2000]. Ils utilisent une notation à base de graphes. Ils ont été définis comme un langage pivot entre le langage naturel et la logique du premier ordre. Ils visent à formaliser les relations entre prédicats et arguments dans une phrase. Ils sont composés de deux types de nœuds étiquetés : les nœuds concepts et les nœuds relations. Les nœuds concepts et relations sont respectivement typés par des types de nœuds et des types de relations, organisés suivant un ordre partiel.

- **Les logiques de description**

Les logiques de description issues des frames reposent sur trois notions de base : les concepts représentant des classes (ensemble d'objets), les rôles (relations liant deux objets) et les individus (objets représentant les classes qu'ils instancient). Pour décrire ces éléments, deux structures sont utilisées : la T-BOX et la A-BOX. La T-BOX (boîte terminologique) comprend la description des concepts et des rôles. Cette description est structurée à l'aide du lien hiérarchique sorte-de. La A-BOX (boîte assertionnelle) est constituée des individus, de leur description et des règles qui leur sont attachés. Les logiques de description sont plus flexibles que les frames et reposent sur une sémantique et une syntaxe rigoureuses [Baader, 1991].

Langages de présentation des ontologies orientés web

De nombreux langages informatiques basés sur la logique de descriptions, plus ou moins récents, spécialisés dans la création et la manipulation des ontologies sont apparus à partir des années 1990. Nous nous concentrons dans cette section sur les langages orientés Web Sémantique. La raison de ce choix est que ces langages sont ou ont été pour la plupart recommandés par le W3C.

- **RDF**

RDF (Resource Description Framework) [Lassila, 1999] permet d'encoder, d'échanger et de réutiliser des méta-données structurées. Il a été créé pour gérer les méta-données de documents XML mais peut également être utilisé pour des ontologies. Il permet de définir des ressources avec des propriétés et des états. **RDF-Schéma** définit les relations entre ces ressources.

Cependant, Le pouvoir sémantique de ces deux langages est limité car les axiomes ne peuvent pas être directement décrits. Le type des relations (symétrique, transitive, …etc.) ne peut être spécifié. En conséquence, un nouveau langage, OWL (Web Ontology Language), est apparu.

- **OWL**

OWL (Web Ontology Language) [Deborah et al., 04], est apparu plus tard. C'est une expression XML fondé sur une syntaxe RDF. Il fournit les moyens pour définir des ontologies Web structurées. Il se différencie du couple RDF/RDFS par le fait que c'est un langage d'ontologies, contrairement à RDF. Si RDF et RDFS apportent à l'utilisateur la capacité de décrire des classes et des propriétés, OWL intègre, en plus, des constructeurs de comparaison des propriétés et des classes: identité, équivalence, cardinalité, symétrie, transitivité, disjonction, etc. Ainsi, OWL offre aux machines une plus grande capacité d'interprétation du contenu Web que RDF et RDFS, grâce à un vocabulaire plus large et à une vraie sémantique formelle.

- **OIL**

OIL (Ontology Inference Layer) [Decker, 2000] est à la fois un langage de représentation et d'échange pour les ontologies. Il combine les primitives des langages reposant sur les frames avec une sémantique formelle et des possibilités de raisonnement issues de la logique de description. Pour être utilisé sur le Web, il repose sur les standards RDF(S) et XML. La description de l'ontologie est divisée en trois couches : la couche objet (instances concrètes), la couche de premier méta-niveau (définition de l'ontologie) et la couche de second méta-niveau (définition des caractéristiques de l'ontologie).

- **SHOE**

SHOE (Simple HTML Ontology Extensions) [Luke, 2000] est une extension de HTML qui permet de rajouter de la sémantique dans ce type de documents. Il permet de définir des primitives pour spécifier et étendre les ontologies et annoter les documents Web. Chaque page déclare quelle ontologie elle utilise. L'inconvénient de ce langage est que les annotations des documents sont stockées à leur niveau et ne peuvent être centralisées.

- **DAML+OIL**

DAML+OIL [Horrocks, 2001] a été proposé par le W3C pour représenter des méta-données et des ontologies. DAML a été transformé en DAML+OIL en intégrant certaines propriétés d'OIL [Decker, 2000]. Il repose sur RDF et RDF schéma et fournit en plus des primitives plus riches issues de la logique de description. Les frames définis dans OIL ont été pour la plupart supprimées et remplacées par les assertions faites à l'aide d'un ensemble limité d'axiomes.

II.4.5. Environnements et outils de développement d'ontologies

Les éditeurs d'ontologie constituent des outils nécessaires aidant à la construction des ontologies. Il existe différents éditeurs d'ontologie, les plus connus sont :

ONTOLINGUA : le serveur Ontolingua est le plus connu des environnements de construction des ontologies. Il consiste en un ensemble d'outils et de services qui supportent la construction en coopération des ontologies, par des groupes séparés géographiquement. Le langage ontologique utilisé dans Ontolingua est KIF [Genesereth, 2005].

PROTEGE : est un environnement graphique de développement d'ontologie le plus utilisé pour construire les ontologies avec multiples langages. C'est un logiciel libre d'utilisation et support le modèle de frames qui contient des classes, des slots (attributs) et des facettes (valeurs des propriétés et contraintes) ainsi que des instances des classes et des propriétés pour permettre le contrôle et la visualisation d'ontologies. Il regroupe aujourd'hui une communauté d'utilisateurs assez importante et constitue une référence pour beaucoup d'autres outils. Protégé est un éditeur ontologique pour les différents langages à savoir RDF, RDFS et OWL.

III. Services Web sémantiques

Les services Web sémantiques sont des services Web simples dont la description WSDL est augmentée par des descriptions supplémentaires. Ces descriptions sont explicitées dans des ontologies à part, afin que d'autres applications ou d'autres services Web puissent comprendre la sémantique des ces derniers qui renforce l'interopérabilité.

Dans cette partie, nous abordons la technologie de services Web sémantiques, en suite nous présentons, proposées dans la littérature, quelques approches de réalisation des services Web sémantiques avec une brève comparaison entre ces différentes approches présentées.

III.1. Définition de services Web sémantiques

Les services Web sémantiques sont des services Web décrits de telle sorte qu'un client (applications ou autres services Web) peut interpréter leurs fonctionnalités offertes. Pour permettre cela, la description syntaxique du service Web doit être augmentée en information sémantique et exploitable par machine (application ou service) et cela fait par la technologie d'ontologique.

III.2. Approches pour la réalisation des services Web sémantiques

Les approches de réalisation des services Web sémantique selon [Mrissa, 2007] se divisent en deux catégories, la première catégorie consiste à développer un langage complet qui décrit le service Web et sa description sémantique dans un seul bloc, on peut citer dans cette catégorie les approches OWL-S [Martin et al., 2004] et WSMO [Lausen, 2005]. La deuxième catégorie consiste à annoter les langages existants avec la description sémantique parmi les approches qui s'inscrivent dans cette catégorie WSDL-S [Akkiraju et al., 2005] et SAWSDL [Farrell et Lausen, 2007]. Nous présentons dans la suite que les quatre approches suivantes OWL-S, WSMO, WSDL-S et SAWSDL.

III.2.1. OWL-S (Web Ontology Language for Services Web)

OWL-S (Web Ontology Language for Services Web) [Martin et al, 2004] fournit une description sémantique des services Web. Cette ontologie de haut niveau indique qu'une ressource est liée à un service, ce dernier est constitué de trois classes ServiceProfile, ServiceModel et ServiceGrounding comme illustré sur la Figure II.4.

- ServiceProfile : exprime ce que le service Web propose.
- ServiceModel : exprime le fonctionnement du service Web.
- ServiceGrounding : exprime comment le service Web peut être utilisé.

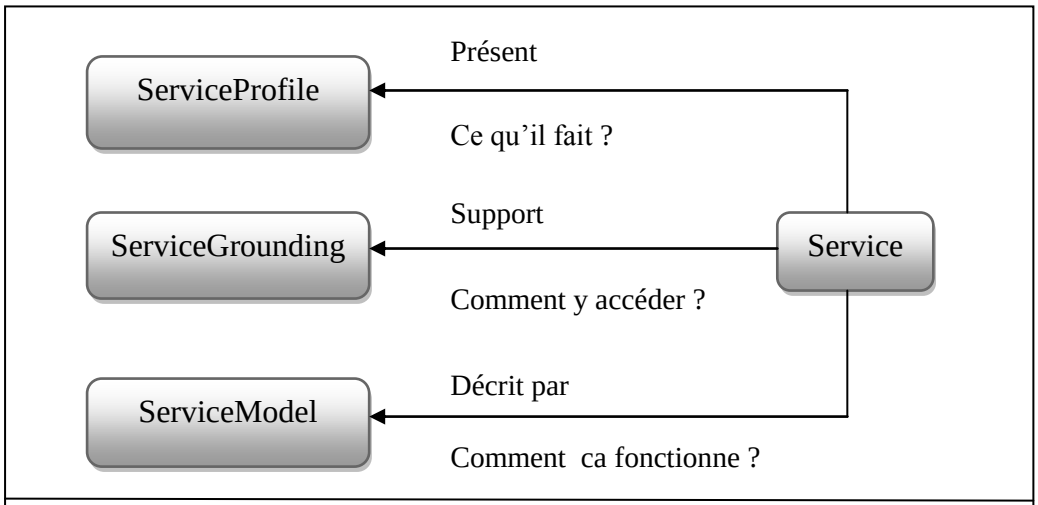


Figure II.4 : Les classes de l'ontologie OWLS [Martin et al, 2004].

III.2.2.WSMO (Web Service Modeling Ontology)

WSMO (Web Service Modeling Ontology) [Lausen et al, 2005] est une architecture conceptuelle visant à expliciter la sémantique des services Web. Elle est organisée en quatre éléments principaux :

- **Les services Web:** sont définis comme des entités qui fournissent une fonctionnalité. Une description est associée à chaque service, dans le but 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 sont une vue orientée utilisateur du processus d'utilisation des services Web, ils sont une entité à part entière dans le modèle WSMO. Un objectif décrit la fonctionnalité, les entrées, les sorties, les préconditions et les postconditions d'un service Web.

- **Les médiateurs:** sont utilisés pour résoudre de nombreuses incompatibilités, telles que les incompatibilités de données dans le cas où les services Web utilisent différents 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.
- **Les ontologies:** fournissent la terminologie de référence aux autres éléments des WSMO, afin de spécifier le vocabulaire du domaine de connaissance d'une manière interopérable par les machines.

III.2.3.WSDL-S (Web Service Description Language-Semantic)

WSDL-S (Web Service Description Language-Semantic) [Akkiraju et al., 2005] est un langage WSDL augmenté de sémantique, cette sémantique est ajoutée en deux étapes, La première consiste à faire référence, dans la partie définition WSDL à une ontologie dédiée au service à publier; La deuxième consiste à annoter les opérations de définition WSDL de sémantique.

WSDL-S distingue quatre modèles sémantiques [Kopecký, 2005] à savoir:

- InputSemantic: le sens des paramètres d'entrée.
- OutputSemantic: le sens des paramètres de sortie.
- Precondition: un ensemble d'états sémantiques qui doivent être vrais afin d'invoquer une opération avec succès.
- Effect: un ensemble d'états sémantiques qui doivent être vrais après qu'une opération réalise son exécution.

III.2.4. SAWSDL (Semantic Annotations for WSDL and XML Schema)

SAWSDL (Semantic Annotations for WSDL and XML Schema), est la suite de WSDL-S [Farrell et Lausen, 2007]. Il a été conçu pour spécifier les éléments de WSDL à l'aide des ontologies de manière plus standardisée. Le mécanisme d'an notation de SAWSDL est décrit de manière indépendante de tout langage de représentation d'ontologies.

L'approche SAWSDL permet d'ajouter facilement une description sémantique au langage WSDL. L'avantage de cette initiative par rapport au WSDL-S est la standardisation ainsi qu'elle permet d'annoter les WSDL existants sans trop alourdir le fichier de description initiale.

III.3. Comparaison des approches

Nous résumons, dans le Tableau II.1, les principales caractéristiques des principales approches de la réalisation des services Web sémantiques présentées précédemment. Les critères de comparaison sont [Mahfoud, 2011] :

- Niveau d'abstraction: qui permet de préciser le niveau d'abstraction (conceptuel, technique) associée à l'approche.
- Niveau d'ouverture: qui permet de représenter le degré de standardisation permis par l'approche.
- Description sémantique: qui permet de préciser la manière avec laquelle les services Web sont décrits sémantiquement.

Les critères	**OWL-S**	**WSMO**	**WSDL-S**	**SAWSDL**
Niveau d'abstraction	Conceptuel (basée sur une ontologie générique de services Web)	conceptuel (basée sur MOF)	conceptuel (basé sur l'annotation sémantique de services Web) WSDL)	conceptuel (basé sur l'annotation sémantique de WSDL et BPEL)
Niveau (Standardisation)	très forte (basé sur OWL, WSDL)	Faible	forte (basé sur OWL, WSDL, mais WSDL-S n'est pas standard)	forte (basé sur OWL, WSDL et BPEL)
Description Sémantique	Ontologie générique OWL-S	Ontologie WSMO	Annotations des Fichiers WSDL	Annotations des fichiers WSDL et BPEL

Tableau II.1 : principales caractéristiques des approches de réalisation des services Web sémantiques.

IV. Conclusion

Le Web sémantique propose une nouvelle approche pour la réalisation des solutions à services en utilisant un modèle plus riche d'informations : les ontologies.

Une ontologie est définie comme spécification d'une conceptualisation dans un domaine de connaissances. Elle est définie par un ensemble de concepts reliés par des relations spécifiques. La terminologie définie par une ontologie est ensuite utilisée pour la représentation des ressources ou de l'information que les services du domaine manipulent.

Les ontologies ont pour but d'ajouter l'aspect sémantique aux services Web afin d'être les données échangées entre ces services lors d'une composition explicitement et correctement interpréter et exploitable par les machines.

Chapitre III

Les techniques de codage d'ontologies

I. Introduction

Les ontologies ont prouvés leur efficacité pour représenter les connaissances modélisant un domaine précis. En effet, elles sont généralement lourdes et volumineux. Par conséquent, le langage OWL n'est pas scalable pour supporter la subsomption.

Pour remédier à ce problème, les chercheurs font appel aux techniques d'encodage d'ontologies. Les techniques d'encodage des ontologies permettent de minimiser le temps d'exécution des opérations suivantes :

- La recherche du plus petit subsumant de 02 concepts;
- La recherche du plus grand spécialisant de 02 concepts;
- Test de subsomption.

De plus, ces techniques d'encodage assurent deux objectifs en même temps : l'économie de l'espace mémoire (pas besoin de stocker l'ontologie initiale), et l'amélioration des performances: le test de subsomption peut être fait en temps constant ou linéaire.

Dans ce chapitre, nous nous intéressons aux techniques d'encodage d'ontologies, et particulièrement le schéma d'étiquetage, la fermeture transitive et réflexive de la matrice d'adjacence et l'encodage par vecteur de bits.

II. Les méthodes d'encodage d'ontologies

Comme l'ontologie est une hiérarchie de concepts, elle est considérée comme un graphe, où les sommets sont les concepts et les arcs sont représentent les relations entre ces concepts. De ce fait les méthodes d'encodage appliquées sur les arbres, peuvent être adaptées aux ontologies.

II.1. Schéma d'étiquetage

D'après [Christophides et al., 2003], l'une des méthodes fournissant un codage efficace d'ontologies (en termes de temps et d'espace) est le schéma d'étiquetage. La notion de schéma d'´etiquetage, introduite par Breuer [Breuer, 1966] permet d'obtenir une représentation de graphe beaucoup plus locale. L'idée est d'affecter à chaque sommet une étiquette de telle sorte que l'on puisse savoir si deux sommets sont adjacents en regardant uniquement leurs étiquettes.

Principe de schéma d'étiquetage

Cette méthode est conçue pour le codage des arbres (pas d'héritage multiple). Elle associe à chaque type x une paire d'entiers notés : d(x), f(x) en utilisant la recherche en profondeur d'abord.

Les grands lignes de l'algorithme d'encodage a base de schéma d'étiquetage

Entrée : Ontologie sous forme de matrice d'adjacence, label=0 ;

Sortie : les concepts étiqueté C = [f, d]

Fonction étiquetage (concept , label, ontologie)

Début

d(concept)= label+1 ; ** d : début de l'intervalle de concept**

**effectué le parcours en profondeur d'abord **

Pour tout concept j sous type de «concept » ***faire***

étiquetage (j, label, ontologie) ;

finpour

f(concept)=label+1 ; ** f: fin de l'intervalle de concept**

fin

Figure III.1 : Pseudo Algorithme de schéma d'étiquetage.

Le test de subsumption pour cette méthode est définit par une fonction binaire donnée par la formule suivante :

$$subsume(C1, C2) = \begin{cases} 1 & si\ l(C2)\ \leq l(C2)\ et\ r(C1) \leq r(C2) \\ 0 & sinon \end{cases}$$

Telle que C1 et C2 sont des concepts, $l(C_i)$ et $r(C_i)$ est l'étiquette de concept C_i. Donc, il suffit de vérifier l'inclusion des intervalles associés aux deux concepts.

Exemple :

Pour une meilleure compréhension de fonctionnement de schéma d'étiquetage, nous allons présentés un exemple illustratif. L'ontologie est représentée par la figure III.1

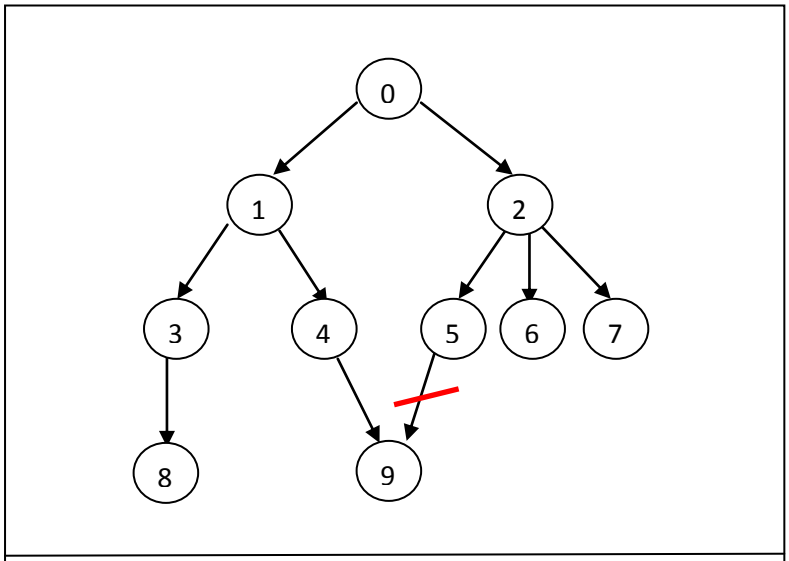


Figure III.2 : L'arbre représentant l'ontologie.

Le parcours profondeur d'abord à donné la liste des concepts : {0, 1, 3, 8, 4, 9, 2, 5, 6, 7}, et les étiquettes données aux concepts sont cités dans le tableau III.1 :

Concept	Etiquette
0	[1,20]
1	[2,11]
2	[12,19] [8,9]
3	[3,6]
4	[7,10]
5	[13,14] [8,9]
6	[15,16]
7	[17,18]
8	[4,5]
9	[8,9]

Tableau III.1 : Les intervalles associés à chaque concept.

Pour le concept 9 est considéré comme sous type de concept 4, après on ajout son intervalle aux concepts 5,2 et 1.

Le résultat de subsumption entre les concepts 0 et 1 est 1 car l'intervalle de concept 1 est inclut dans l'intervalle de concept 0.

II.2. Fermeture transitive et réflexive de la matrice d'adjacence

Avant de définir la fermeture transitive et réflexive de la matrice d'adjacence, nous définissons la matrice d'adjacence représentant un graphe *G=(X, U), X* est l'ensemble de sommets de G et *U* l'ensemble des arcs, est une matrice carrée M (concept x concept) telle que :

$$M(i,j) = \begin{cases} 1 \; si \;\; i \; est \; supertype \; j \\ 0 \qquad\qquad\qquad\qquad sinon \end{cases}$$

Principe

La fermeture transitive et réflexive d'un graphe orienté permette de déterminer les chemins de plus grand longueur, ce qui nous aider à tester la subsumption d'un concept par un autre. Elle est déduite a partir de la matrice d'adjacence équivalente et aussi une matrice carrée (concept x concept), dont :

$$M^*(i,j) = \begin{cases} 1 \; si \;\; i = \; j \;\; ou \; i \; est \; supertype \; direct \; ou \; indirect \; de \; j \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad sinon \end{cases}$$

Le test de subsumption pour cette méthode est définit par une fonction binaire donnée par la formule suivante :

$$subsume\ (C1, C2) = \begin{cases} 1 & si \;\; M * [C1, C2] = 1 \\ 0 & \qquad\qquad sinon \end{cases}$$

Le pseudo code de l'algorithme d'encodage d'ontologie a base de fermeture transitive et réflexive de la matrice d'adjacence est présenté par la figure suivante.

Entrée : Ontologie sous forme de matrice d'adjacence M

Sortie : matrice transitive et réflexive de M note M*

Fonction M*(M)

```
Début
    Pour i=1 jusqu'à nombre_de_concepts faire
        Pour j=1 jusqu'à nombre_de_concepts faire
            Si M [i, j] = 1 alors
                Pour k=1 jusqu'à nombre_de_concepts faire
                    Si M [k, i]=1 alors
                        M* [k, j] = 1 ;
                    Sinon
                        M*[i, j] = 0 ;
                    finsi
                Fin pour
            Sinon
                Si i = j alors
                    M*[i,j] = 1 ;
                finsi
            finsi
        finpour
    finpour
Fin
```

Figure III.3 : Pseudo algorithme de fermeture transitive et réflexive.

Exemple :

Considérons l'ontologie représentée par la figure III.4, la matrice d'adjacence M, l'application de la fermeture donne la matrice M* :

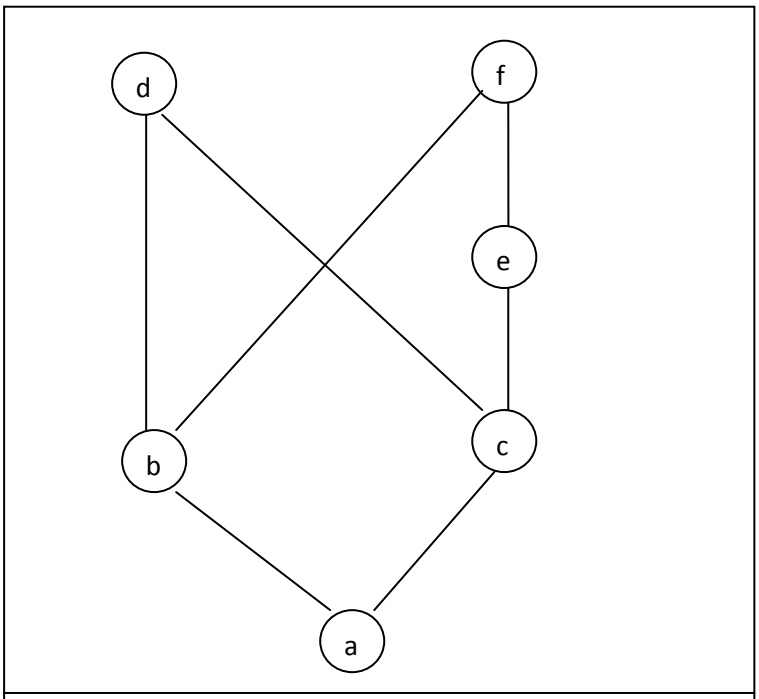


Figure III.4 : L'arbre de l'ontologie.

Le tableau suivant représent la matrice d'adjacence de l'ontologie précédente.

	a	B	c	D	E	f
a	0	0	0	0	0	0
b	1	0	0	0	0	0
c	1	0	0	0	0	0
d	0	1	1	0	0	0
e	0	0	1	0	0	0
f	0	1	0	0	1	0

Tableau III.2 : Matrice d'adjacence M de l'exemple.

Et la fermeture transitive et réflexive de l'ontologie est donc :

	a	B	c	D	E	f
a	1	0	0	0	0	0
b	1	1	0	0	0	0
c	1	0	1	0	0	0
d	1	1	1	1	0	0
e	1	0	1	0	1	0
f	1	1	1	0	1	1

Tableau III.3 : M* fermeture transitive et réflexive de la matrice M.

Pour tester la subsumption entre deux concepts i et j, il suffit de lire M*[i, j] par exemple : le concept b subsume le concept a car M*[b, a]=1.

II.3. Encodage par vecteur de bit (Bit vector encoding)

La méthode de bit vector encoding BVE, étiquète aussi les concepts mais cette fois par une séquence de bits (0 et 1). Formellement, Le Bit Vector encoding est une fonction φ qui associe à chaque type x de l'ordre partiel P(X, ≤) un sous ensemble de S= {1,2,….K} tel que, l'inclusion des concepts coïncide avec l'inclusion des sous ensembles. La fonction φ définit comme suit :

$$\varphi : \begin{matrix} X \rightarrow 2^S \\ x \rightarrow \varphi(x) \end{matrix}$$

La taille de l'encodage φ est le cardinal de S : |S|, et les éléments de S sont appelés genes ou couleurs. On distingue des encodages top down et bottom up.

Principe

Plusieurs algorithmes et idées ont été proposées par les chercheurs pour trouver la séquence de bits a affecté aux concepts. Les opérateurs AND(&) et OR (|) sont utilisé pour tester la subsumption entre deux concepts A et B :

B est sous type de A si : $\varphi(A)\ \&\ \varphi(B) = \varphi(B)\ ou\ \varphi(B)\ |\ \varphi(A) = \varphi(A)$.

Des approches pour déterminer un code de bit sont apparues, parmi les quelles nous citons:

- **Première approche [Wirth, 1988]**

Permette d'associer a chaque concept un vecteur de bits de taille n bits, où n est le nombre de concepts de l'ontologie.

Un « 1 » bit a une certaine position unique identifié le concept dans le code $\varphi(concept)$, et chaque concept hérite les bits identifiants son ancestor (ou descendant) pour l'encodage top down (ou bottom up). D'une manière plus formelle, l'étiquette de concept A dans φ est $\varphi(A) = \{b1, b2, \dots, bn\}$, bi = 1 si le i^{ème} concept est soit A ou c'est un ancestor (descendant) B de A. sinon bi = 0.

Exemple

L'ontologie illustrée par la figure III.5 est constitué de huit concepts, alors, le nombre de bits nécessaire pour coder chaque concept est huit, les codes de bits sont associés aux concepts sur la figure aussi.

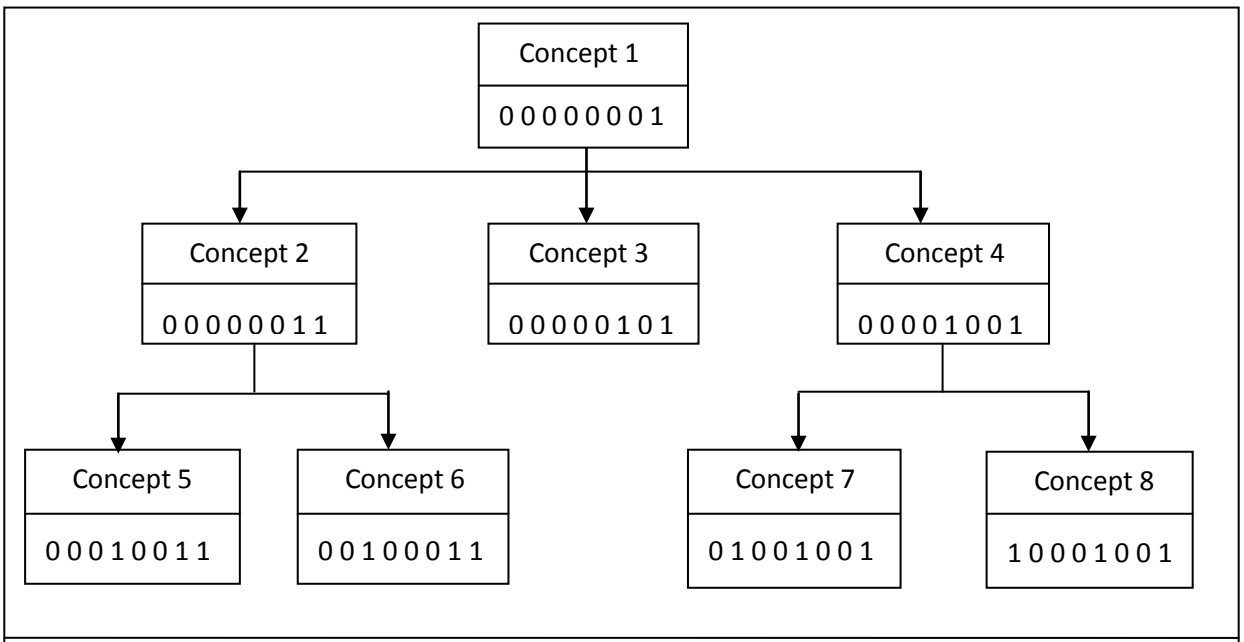


Figure III.5 : L'arbre de l'ontologie avec le code de bit pour chaque concept.

Par conséquent, ce mécanisme d'encodage ne marche pas pour les ontologies qui possèdent un grand nombre de concepts. Pour résoudre ce problème, l'approche polychotomique permette de trouver un code de bit le plus réduit.

- **Polychotomique [Raynauld et al., 2001]**

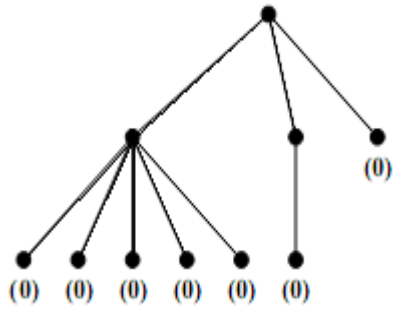
La taille du BVE est égale au poids maximal des différentes chaines possibles associées à l'arbre, Sachant que le poids d'une chaine c est:

$$Poids(c) = \sum_{0<i<p} sp(deg(xi))$$

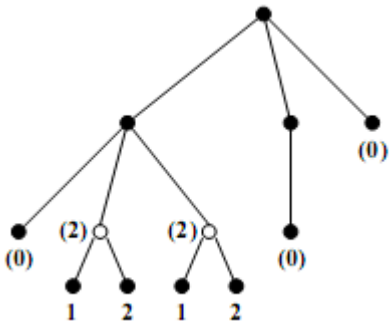
Il introduit deux heuristiques pour construire les codes des concepts qui sont :

- Si l'arbre est vertical de n concept alors la taille du code est n-1
- Et si l'arbre est horizontal de n concept alors la taille du code est égale à $\log_2$ (n)

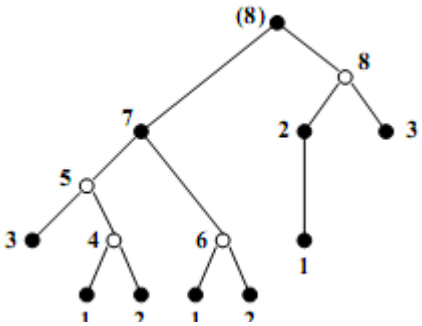
Les poids des concepts sont modifiés de manière récursive de feuilles jusqu'à la racine. La figure III.6 illustre un exemple.



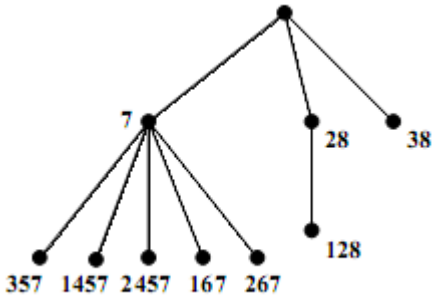
a) initialisation



b) modification des poids



c) modification de tout les poids, le poids de la racine est (8) donc 8 bits est suffisants pour l'encodage.



d) l'ontologie finale avec les codes de chaque concept.

Figure III.6 : Les étapes de construction du code selon [Raynaud et al., 2001].

III. Comparaison de méthodes d'encodage

Une comparaison de méthodes d'encodage d'ontologies peut être faite en comparant ces techniques en termes de test de subsumption et l'algorithme d'obtention de codage comme le montre le tableau III.3.

	Test de subsomption	L'algo d'obtention de codage	Stockage d'ontologie
Schéma d'étiquetage	- Constant si l'ontologie est un arbre - Linéaire si l'ontologie est un graphe.	- Exponentielle	- Constant si l'ontologie est un arbre (très faible). - Linéaire si l'ontologie est un graphe (moyen).
Fermeture transitive & réflexive	- Constant	- Polynomial	- Polynomial (grand)
Bit vector encoding	- Pseudo constant	- Polynomial	- Logarithmique (faible)

Tableau III.4 : Comparaison de techniques d'encodage d'ontologie.

IV. Conclusion

Les méthodes d'encodage de sommets des graphes et arbres sont bien adaptées aux ontologies. Elles permettent de vérifier le test de subsumption d'une manière simple et en temps constant (généralement).

Théoriquement, ces techniques d'encodage ont prouvées leur efficacité pour le test de subsumption et par extension le grand spécialisant et le petit subsumant.

Dans le chapitre suivant, nous allons montrer l'efficacité de ces méthodes d'un coté pratique, pour le test de subsumption utilisé dans le cadre de composition de services Web sémantiques.

Chapitre IV

Prototype

I. Introduction

La plupart des mécanismes de raisonnement des services Web sémantiques se basent sur l'utilisation des ontologies, qui offrent un moyen efficace pour décrire d'une façon sémantique les fonctionnalités des services Web issus d'une composition.

L'utilisation de l'ontologie telle qu'elle produit de nombreux problèmes telle que la performance en terme de temps d'exécution et d'espace mémoire (Stockage de l'ontologie). Pour résoudre ces problèmes, nous proposons d'utiliser les techniques d'encodage d'ontologies.

L'objectif de ce chapitre est de comparer l'influence des techniques de codage sur la composition de services Web, et en particulier leur temps d'exécution. L'encodage de l'ontologie est fait par des méthodes qui sont particulièrement « le schéma d'étiquetage », « la fermeture transitive et réflexive de la matrice d'adjacences », ces méthodes sont comparées au raisonneur Pellet.

Dans ce dernier chapitre nous allons voir les différentes étapes suivies durant la réalisation de notre application, Nous commencerons d'abord par décrire l'environnement dans le quel nous avons réalisé le prototype, puis nous présenterons les paramètres de l'application qui sont l'ontologie ainsi que la base de services, puis nous allons parler du processus de développement logiciel utilisé, en suit, nous présentons les différents diagrammes de conception ainsi que l'IHM de l'application, et en finissons par une discussion des résultats que nous avons obtenus.

II. Outils et environnement de développement

Nous avons développé notre application sur une machine Intel Core2duo, avec une vitesse de 1.6 GHZ, doté d'une capacité mémoire de 1GB de RAM sous Windows 7.

Avant de commencer l'implémentation de notre application, nous allons tout d'abord spécifier les outils utilisés qui nous ont semblé être un bon choix vu les avantages qu'ils offrent.

- **Java**

L'application est de développé en utilisant le langage de programmation Java. Java est un langage de programmation à usage général, évolué et orienté objet. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être le choix préféré pour le développement de notre application.

- **NetBeans**

Nous avons écrit notre application en Netbeans version 6.8, le choix de Netbeans était fondamental puisqu'il est un logiciel permettant principalement le développement en java. il fournit un environnement standard de développement pour créer des interfaces très puissants. NetBeans est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (common development and Distribution license).

- **L'API JDOM**

JDOM est une API open source Java, vue comme un modèle de documents objets dont le but est de représenter et manipuler un document XML de manière intuitive pour un développeur Java sans requérir une connaissance pointue de XML. JDOM propose aussi une intégration de SAX, DOM, XSLT et XPath.

Un document XML est encapsulé dans un objet de type Document. Les éléments d'un document sont encapsulés dans des classes dédiées : Element, Attribute, Text, Processing Instruction, Namespace, Comment, DocType, EntityRef, CDATA. JDOM permet aussi de vérifier que les données contenues dans les éléments respectent la norme XML.

JDOM propose des réponses à certaines faiblesses de SAX et DOM. La simplicité d'utilisation de JDOM lui permet d'être une API dont l'utilisation est assez répandue.

- **JFreeChart**

JFreeChart est une bibliothèque open source qui permet d'afficher des données statistiques sous la forme de graphiques. Elle possède plusieurs formats dont le camembert, les barres ou les lignes et propose de nombreuses options de configuration pour personnaliser le rendu des graphiques. Elle peut s'utiliser dans des applications standalones ou des applications web et permet également d'exporter le graphique sous la forme d'une image.

Les données utilisées dans le graphique sont encapsulées dans un objet de type Dataset. Il existe plusieurs sous-types de cette classe en fonction du type de graphique souhaité.

- **.L'API Pellet**

Cette api permet le raisonnement sur les ontologies formalisées avec owl, et en particulier elle offre des mécanismes d'inférences basés sur les logiques de description.

III. Présentation de la base

Notre base est constituée des éléments suivants : Une liste de n services, une ontologie domaine.

L'Ontologie : L'ontologie est représentée sous forme de document OWL, contenant l'ensemble de concepts ainsi que l'hiérarchie de ces concepts comme illustré dans la figure suivante.

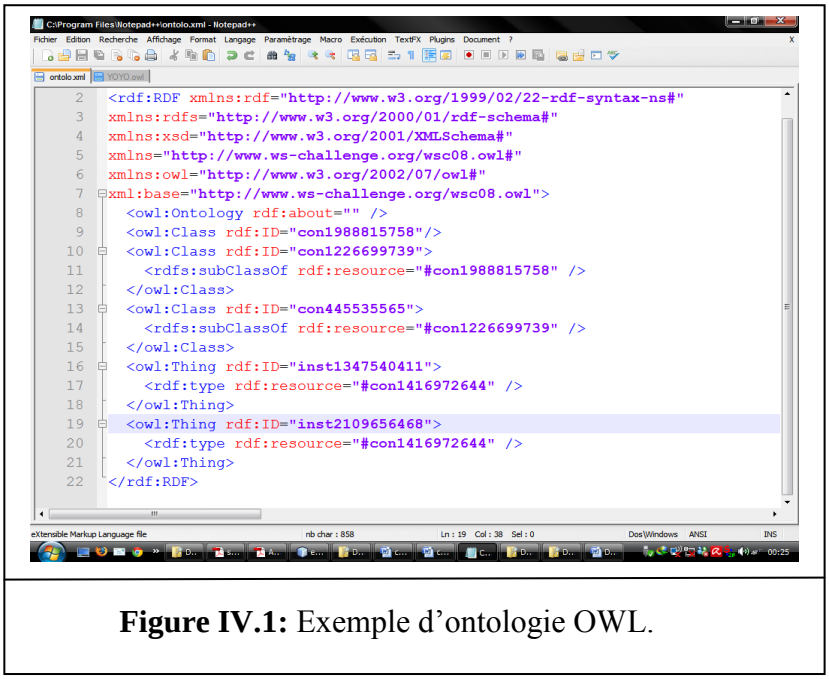


Figure IV.1: Exemple d'ontologie OWL.

Base de services : la base de services est fournie sous forme d'un document XML regroupant la liste des services Web, telle que chaque service possède un input et un output qui sont des concepts de l'ontologie.

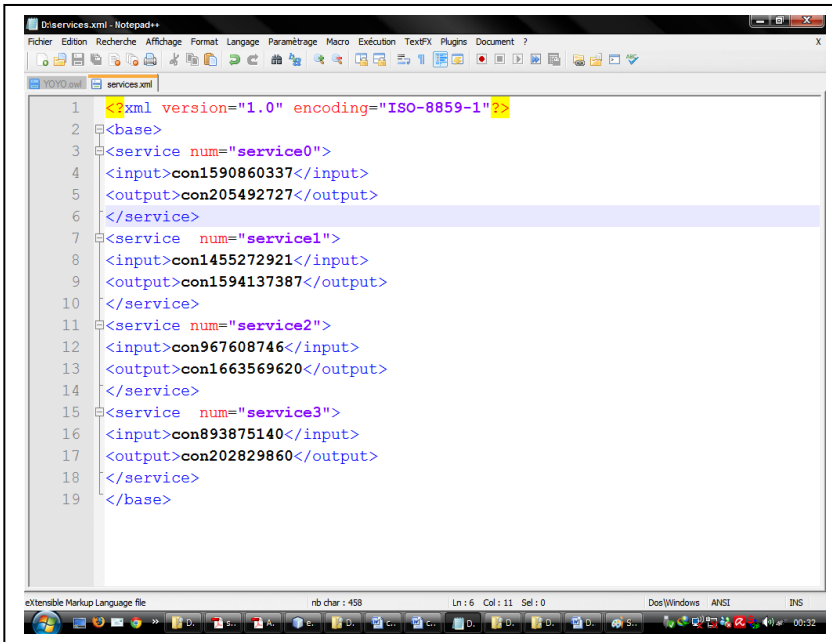


Figure IV.2 : Exemple de base de services XML.

IV. Processus de développement logiciel

Un processus définit une séquence d'étapes, en parties ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. [Rocques, 2007]

En d'autre terme, c'est les différentes opérations réalisées à fin d'élaborer le produit logiciel. Dans notre cas nous avons opté pour un processus unifié, ce choix est justifié par les principes sur les quelles se base ce processus.

IV.1. Processus unifié

Un processus unifié est un processus de développement logiciel construit sur UML, il est itératif et incrémentale centré sur l'architecture, conduit par les cas d'utilisations et piloté par les risques [Rocques, 2007].

- **itératif et incrémentale,** au sens que la réalisation du produit se fait en plusieurs itérations où chaque itération aboutit à livrer une partie de produit (un module) et l'ajoute aux différentes parties déjà construit (incrémentation) en gardant un produit homogène.
- **Centré sur l'architecture**, car il impose le respect des décisions d'architecture a chaque état de instruction du modèle.

- **Conduit par les cas d'utilisations**, ce qui veut dire que la conception est réalisée conformément aux attentes des utilisateurs du produit.
- **Piloté par les risques,** dans le cadre, où les différentes sources d'échec du produit doivent être ecratées.une source importante d'échec d'un produit logiciel est son inadéquation aux attentes de l'utilisateur.

La gestion du processus est organisée suivant les quartes phases suivantes : initialisation, élaboration, construction, transition.

IV.2. Modélisation avec UML

UML ce définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue.

UML unifie à la fois les notations est les concepts orientés objet. Il ne s'agit pas d'une simple notation, mais les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d'un langage. UML permet de modéliser de manière claire et précise la structure et le comportement d'un système indépendamment de toute méthode où de tout langage de programmation. UML est à présent un standard défini par l'Object Management Group (OMG).

IV.2.1. Diagramme de cas d'utilisation

Un diagramme de cas d'utilisation permet de structurer les besoins des utilisateurs et les objectifs correspondant d'un système.il permet aussi d'identifier les possibilités d'interactions entre le système et les acteurs (intervenants extérieurs au système). Il parte du principe que les objectifs du système sont tout motivés.

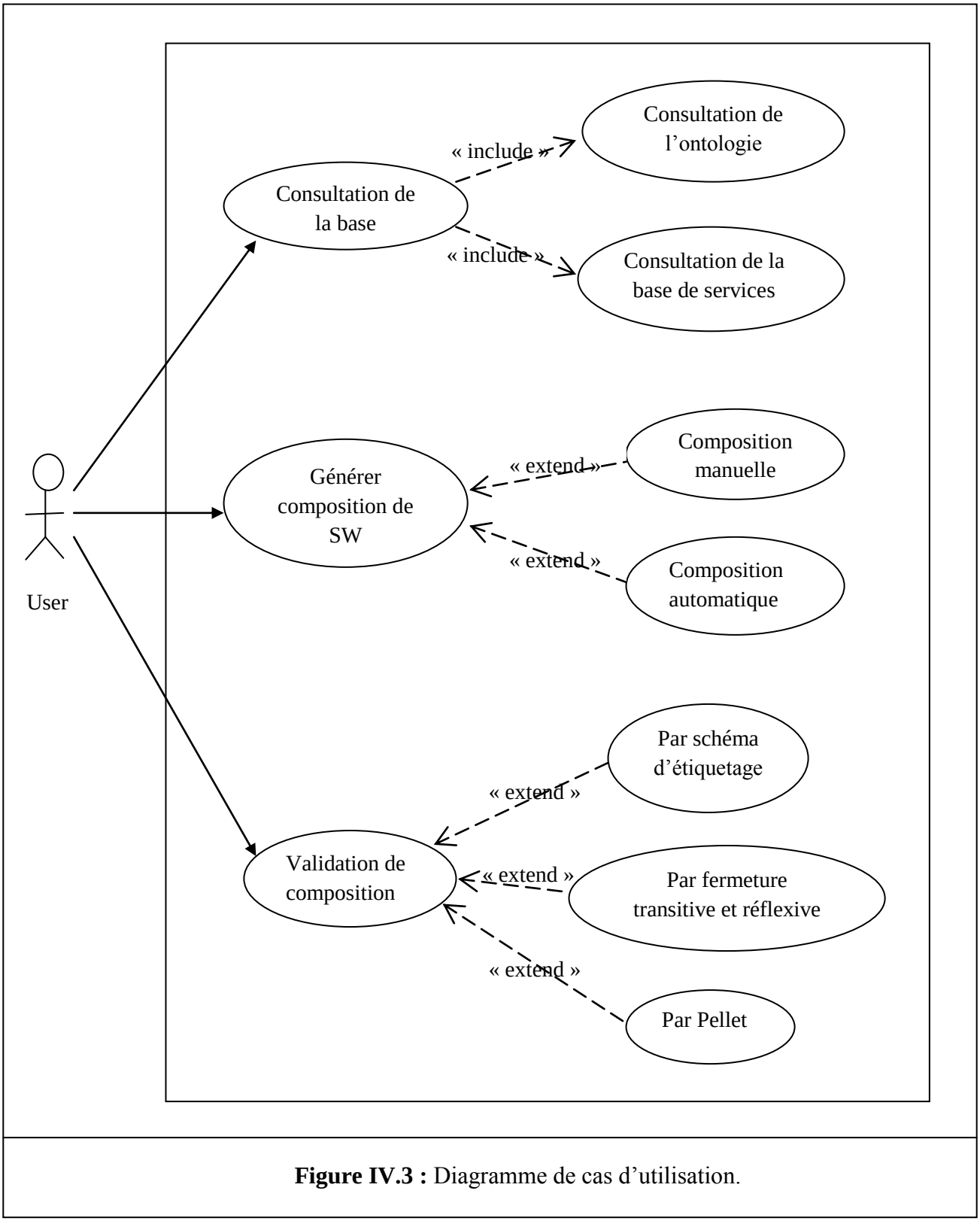


Figure IV.3 : Diagramme de cas d'utilisation.

IV.2.2. Diagramme de séquence

Un diagramme de séquence permet de représenter des collaborations entre objets selon un point de vue temporel. Les objets communiquent en échangeant des messages représentés sous forme de flèches. Il peut servir à illustré un cas d'utilisation. Le diagramme de séquence de cas « composition manuelle » est représenté dans la figure IV.4.

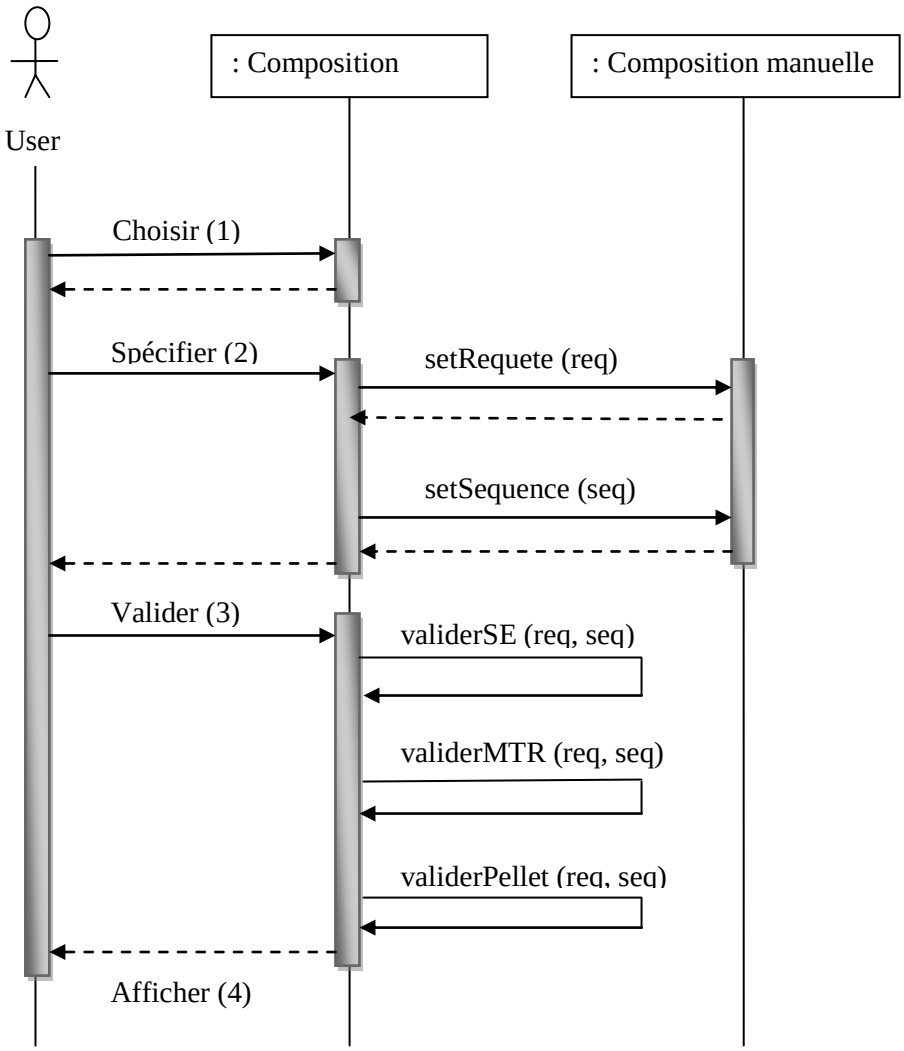


Figure IV.4 : Diagramme de séquence du cas « composition manuelle ».

La figure suivante illustre le diagramme de séquence correspondant au cas « consultation de base ». (1) : l'utilisateur ici choisir de consulter la base, une boite de dialogue est affichée qui lui permette de lire l'ontologie ainsi que la base de services (2) et (3). Puis l'user lance le codage de l'ontologie (4) par les deux techniques d'encodage.

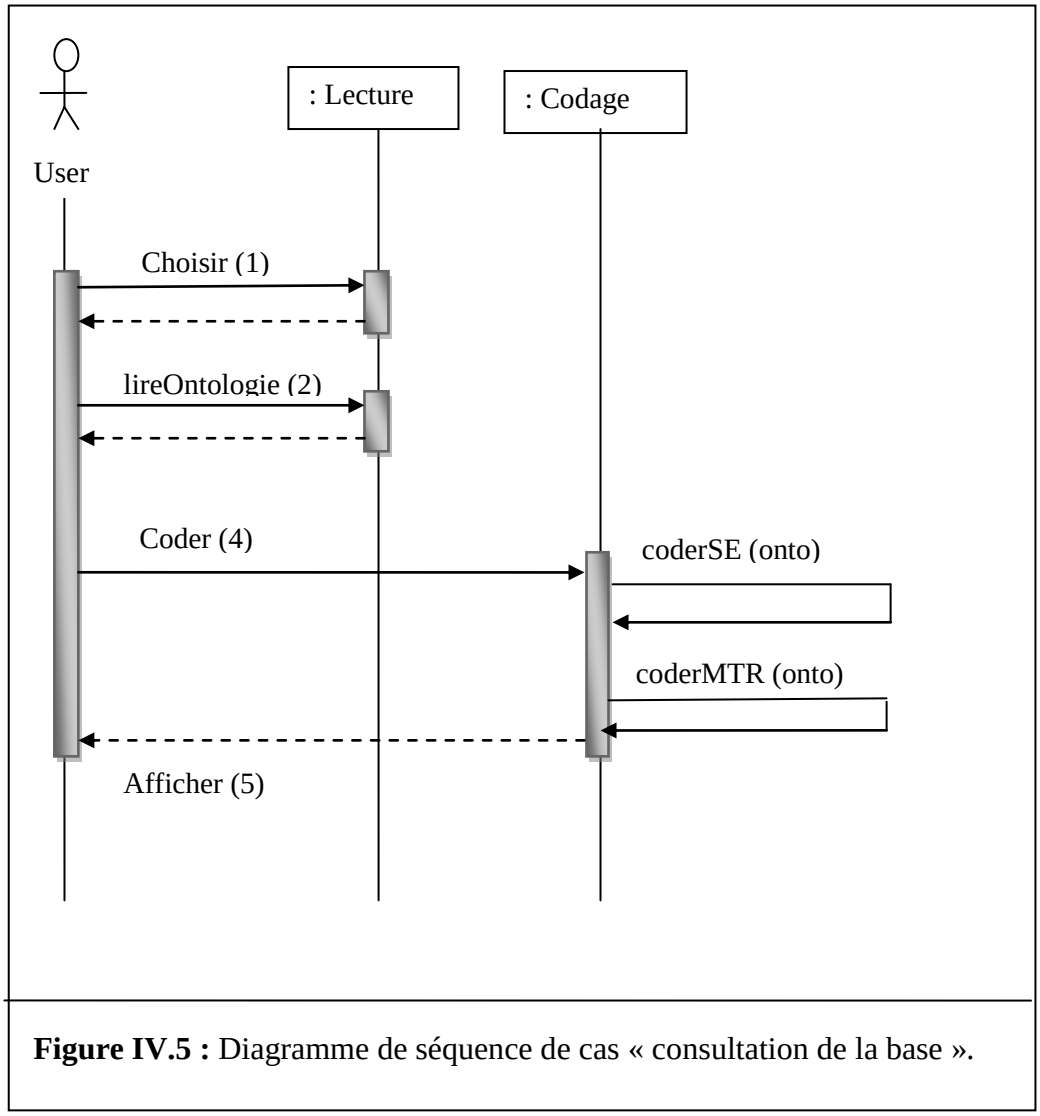


Figure IV.5 : Diagramme de séquence de cas « consultation de la base ».

IV.2.3. Diagramme de classe

Un diagramme de classe permet de représenter les classes intervenant dans le système.il constitue un élément très important de la modélisation et permettre de définir quelle seront les composantes du système final. Il permet de structuré le travail de développement de manière très efficace. La figure IV.6 représente le diagramme de classes de notre application.

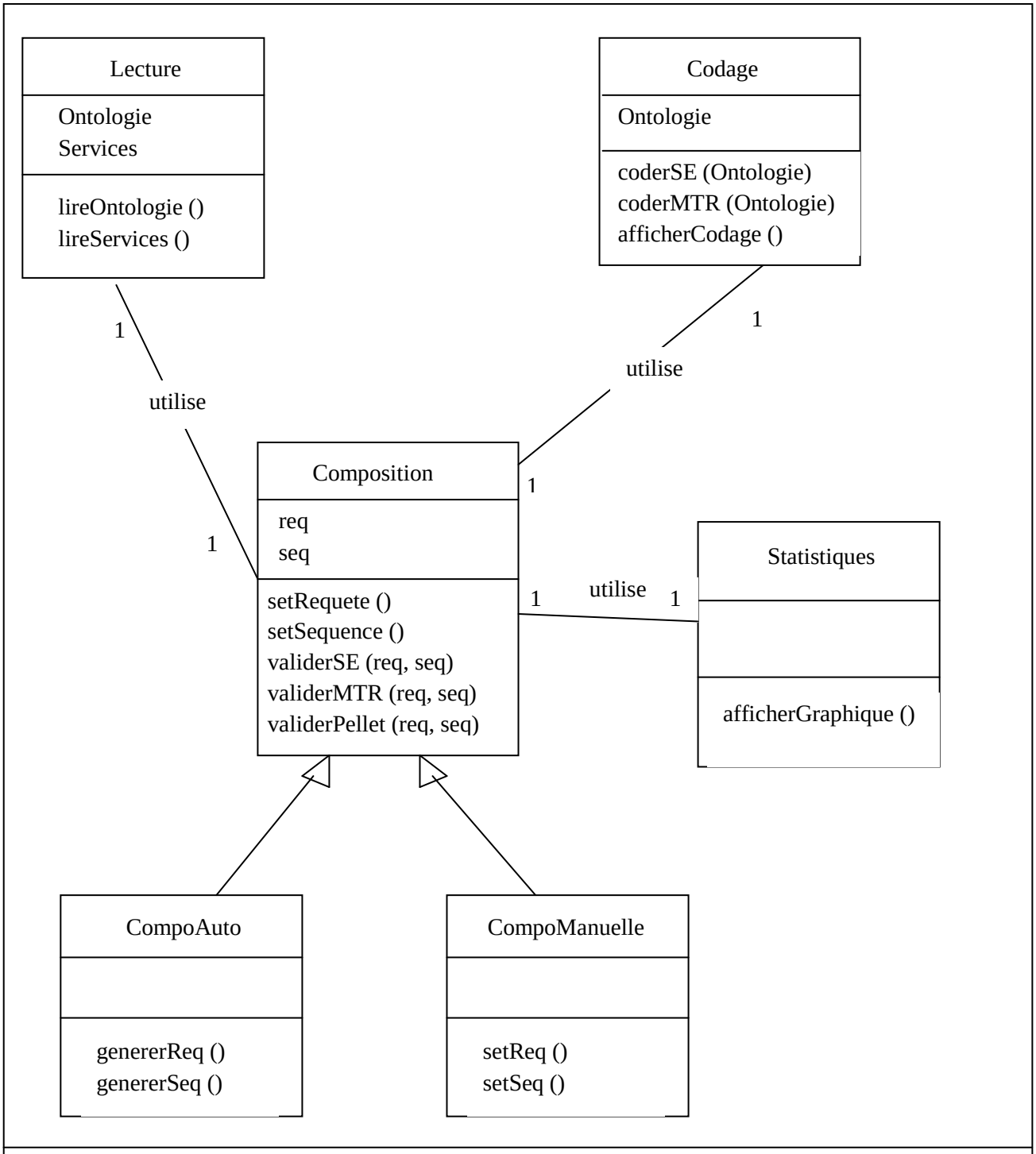


Figure IV.6 : Diagramme de classes.

V. Les Algorithmes proposées

V.1. Fermeture transitive et réflexive de la matrice d'adjacence

- parcourir la base d'ontologie.
- créer la matrice d'adjacence **M** qui représente l'ontologie(les lignes et les colonnes de la matrice sont les concepts de l'ontologie).
- case (i, j) est mise a 1 s'il y a une relation superclasse entre i et j.
- construire la fermeture transitive et réflexive de M notée M* telle que :

  $M^* = \bigcup_{i=0}^{n} M^i$
- test de subsumption (c1, c2) est définit comme suit :

$$\begin{cases} Vrais & Si\ M^*(c1, c2) = 1 \\ Faux & Sinon \end{cases}$$

V.2. Schéma d'étiquetage

- parcourir la base d'ontologie.
- créer la matrice d'adjacence **M** qui représente l'ontologie(les lignes et les colonnes de la matrice sont les concepts de l'ontologie).
- Effectuer une recherche en profondeur d'abord.
- Etiqueter chaque concept par un intervalle d'entier.
- test de subsumption (c1, c2) est définit comme suit :

$$\begin{cases} Vrais & Si\ E(c1) \supset E(c2) \\ Faux & Sinon \end{cases}$$

VI. Présentation de l'IHM

L'interface homme/machine représente l'élément clé dans l'utilisation de tout système informatique.les interfaces de notre système sont conçues de manière à être simples, naturelles, compréhensible et d'utilisation faciles.

Fenêtre principale

La fenêtre principale constitue une barre de menu composée de :

- Fichier: qui permet de charger l'ontologie ainsi que la base de services à composer.

- Composition: contient deux MenuItems la première Item permet à l'utilisateur de faire une composition manuelle en spécifiant la requête et la séquence de service à composer.la deuxième permet de générer une composition automatique.
- Statistiques.

Figure IV.7 : Fenêtre Principale de composition de SW.

Chargement de la base d'ontologie:

Après le choix de l'ontologie et la base de services l'IHM suivante consiste à aider l'utilisateur à l'ouvrir et la coder.

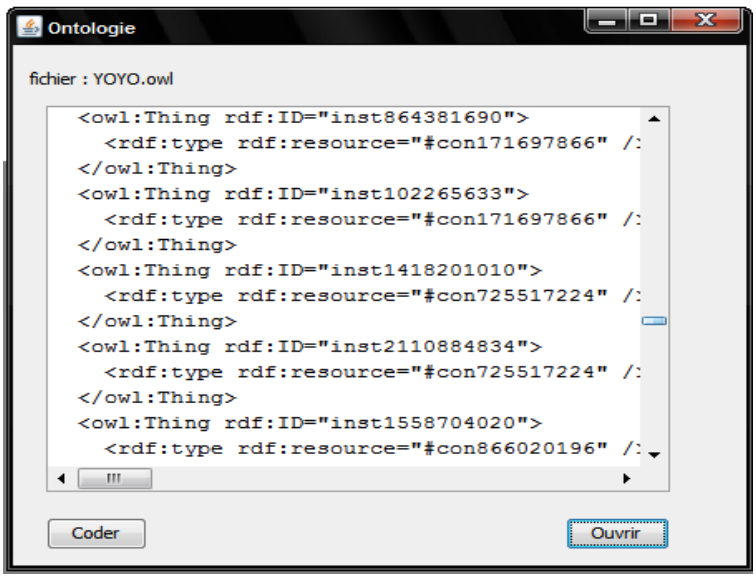


Figure IV.8 : Chargement de l'ontologie.

Codage de l'ontologie

L'IHM suivante consiste à afficher le résultat de codage selon les deux techniques.

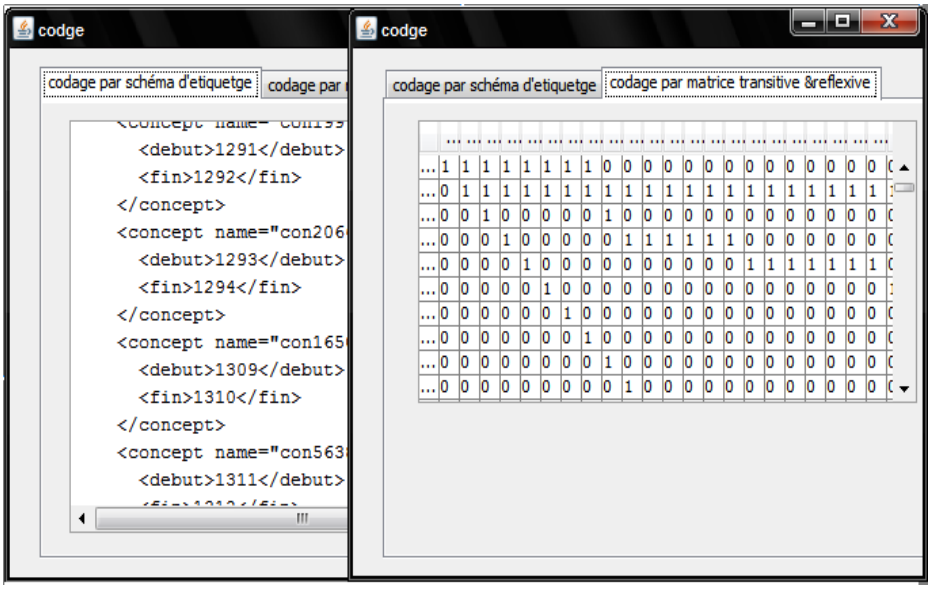


Figure IV.9 : Codage de l'ontologie.

Génération de composition

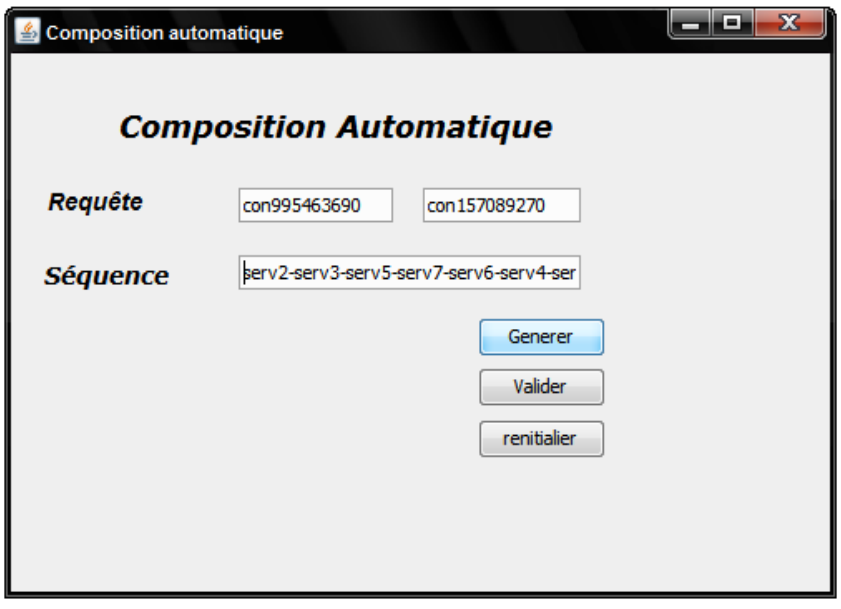


Figure IV.10 : Génération de composition.

Dès que la requête et la séquence sont générer le résultat de validation de composition selon les techniques d'encodage et représenté dans la figure suivante :

Figure IV.11 : Résultat de validation de composition.

VII. Expérimentations

Nous avons présenté dans ce chapitre deux techniques d'encodage d'ontologie et les comparer avec le raisonneur pellet, en termes de temps d'exécution de teste de subsomption.

La première adopte la technique de schéma d'étiquetage, et la deuxième adopte la technique fermeture transitive et réflexive, et la troisième adopte pellet.

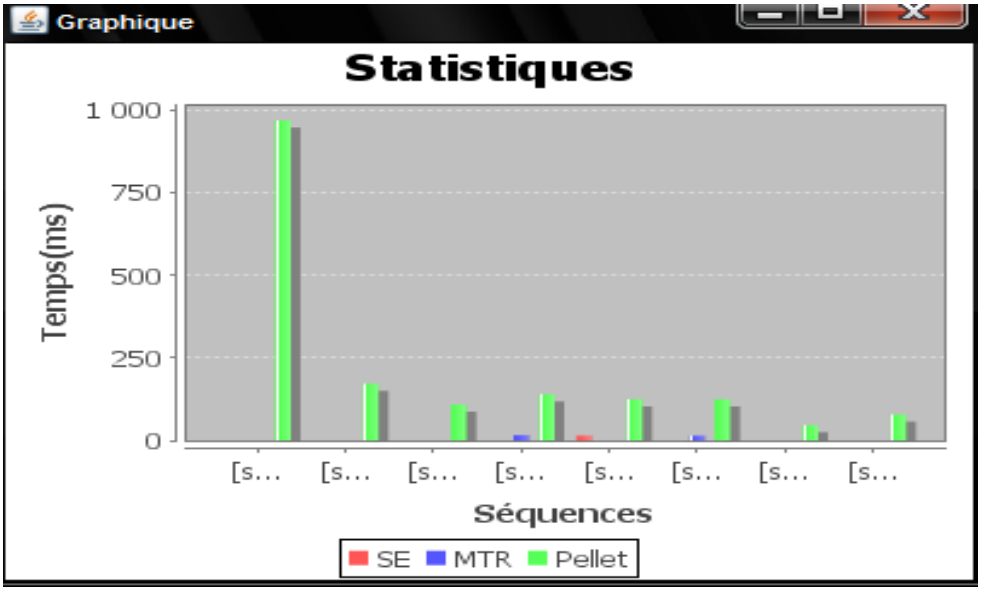


Figure IV.12 : Temps d'exécution des deux techniques SE & MTR et Pellet.

D'après les résultats obtenus, nous concluons que :

- La fermeture transitive de la matrice d'adjacence est meilleure que le schéma d'étiquetage, car le test de subsumption est plus simple pour la fermeture transitive.
- Il n'y a pas de grande différence entre le schéma d'étiquetage et la fermeture transitive, car le test de subsumption est simple et presque le même pour les deux.
- Les deux méthodes : schéma d'étiquetage et fermeture transitive sont meilleurs que Pellet, car Pellet est basé sur les logiques de description et les axiomes pour tester la subsumption.

VIII. Conclusion

Nous avons présenté dans ce chapitre un système de composition de services Web, en se basant sur l'utilisation directe de l'ontologie, et les techniques d'encodage de l'ontologie qui visent l'amélioration de performances du test de subsumption en termes de temps d'exécution.

Les algorithmes d'obtention du schéma d'étiquetage et de la fermeture transitive possèdent des complexités exponentielles et polynomiales (respectivement). Mais ils ne sont utilisés que hors ligne, (on les utilise pas pendant la subsomption). Les résultats obtenus sont acceptables et encourageant des future travaux sur cet axe de recherche.

Conclusion générale

Conclusion générale

Aujourd'hui, l'interopérabilité est devenue un domaine de recherche fondamental des systèmes d'information distribués et hétérogènes. Les services Web sont considérés comme une solution potentielle aux problèmes d'interopérabilités. Ils définissent un nouveau paradigme de développement des interactions entre des applications distribuées de manière à ce qu'elles restent indépendantes des environnements et des plateformes d'exécution d'une part et aussi des choix des langages de développement et technologies d'implémentations utilisés d'une autre part. La composition de services Web en particulier permet de combiner plusieurs fonctionnalités des services Web afin de répondre aux exigences qu'un seul service ne peut satisfaire.

Malgré les avantages indéniables qu'apportent les services Web au niveau des problèmes d'interopérabilité, leur composition est souvent confrontée à plusieurs hétérogénéités sémantiques des données échangées entre service Web. Afin de remédier à cette lacune, plusieurs approches ont été proposées, pour l'enrichissement de la description des services Web par la sémantique, généralement explicitée dans des ontologies.

Dans ce mémoire, nous avons présenté un travail basé sur l'encodage des ontologies pour optimiser le temps d'exécution du test de subsumption, et comme domaine d'application nous avons adopté la technologie des services Web, et plus précisément la composition de services Web sémantiques.

La réalisation de prototype nous a permis de valider et de confirmer notre contribution qui consiste à coder l'ontologie.

Pour ce la nous avons réalisé trois méthodes pour implémenter la subsomption au niveau de la composition de service. La premier utilisé la fermeture transitive et réflexive de la matrice d'adjacence, la deuxième utilise le schéma d'étiquetage, la troisième utilise directement l'API pellet(l'ontologie sans codage).

Les résultats montrent la supériorité de la technique de fermeture transitive et réflexive par apport aux autres méthodes, et de même ils confirment la supériorité de schéma d'étiquetage par apport à l'utilisation directe des API de subsomption.

Tout travail est amené à être amélioré, en ce sens, notre application peut encore évoluer et se voir améliorer. Pour une continuation de notre travail, plusieurs perspectives peuvent être envisagées :

- Implémenter d'autres techniques d'encodage par exemple le Bits vector encoding comme la version dichotomique, polychotomique et le comparer avec les deux méthodes implémentées (schéma d'étiquetage et fermeture transitive et réflexive de la matrice d'adjacence).
- Prise en charge des critères de qualité de service Web QOS dans la composition de services.
- Nous proposons d'utiliser le contexte de données dans une approche de découverte pour aider les concepteurs à rejeter les services Web, qui ne sont pas pertinents par rapport aux besoins de la composition.

.

Références bibliographiques

Références bibliographiques

[Akkiraju et al., 2005] : R. Akkiraju, J. Farrell, J. A. Miller, M. Nagarajan, M. Schmidt, A. Sheth, K. Verma, Web Service Semantics - WSDL-S, rapport technique, IBM, http://www.w3.org/Submission/WSDL-S/, 2005.

[Arnaud, 2005] : Arnaud VEZAIN, Les service web - présentation générale, rapport technique, Association HERMES, Février 2005.

[Aît-Bachir, 2008] : A. Aît-Bachir, Archi Med, un canevas pour la détection et la résolution des incompatibilités des conversations entre services web, thèse de doctorat, Université Joseph Fourier-Grenoble 1, 2008.

[Baader, 1991]: F. Baader, B. Hollunder, A Terminological Knowledge Representation System with Complete Inference Algorithms, In Proceedings of the Workshop on Processing Declarative Knowledge, 1991.

[Bachimont, 2000] : B. Bachimont, Engagement Sémantique et Engagement Ontologique : Conception et Réalisation D'ontologies En Ingénierie Des Connaissances, chapter 19, pages 305–324, Eyrolles, 2000.

[Bala, 2007] : M. Bala, Interopérabilité Sémantique des Systèmes d'Information Distribués, thèse de magister, Institut National de Formation en Informatique INI Alger, 2007.

[Borst, 1997]: P. Borst, Construction of Engineering Ontologies for Knowledge Sharing and Reuse, Ph.D Dissertation, Tweente University, 1997.

[Brachman, 1977]: R.J. Brachman, What's in a concept: structured foundation for semantic networks, International Journal of Man-Machine Studies 9, pp 127-152, 1977.

[Breuer, 1966] : M. A. Breuer, Coding the vertices of a graph, IEEE Trans. Inf. Th., 12, 148–153, 1966.

[Christensen et al., 2001] : E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, Web Services Description Language (WSDL) 1.1, rapport téchnique,W3C, http://www.w3.org/TR/wsdl, 2001.

[Christophides et al., 2003]: Vassilis Christophides, Michel Scholl, Sotirios Tourtounis, On Labeling Schemes for the Semantic Web, 2003.

[Clement et al., 2004] : Clement L., Hately A., von Riegen C. et Rogers T., UDDI v.3.0.2, OASIS Specification, http://uddi.org/pubs/uddi_v3, 2004.

[Colan, 2003]: IBM.BPEL4WS (version1.1), http://www.ibm.com /developerworks/library/ws-bpel, 2003.

[Deborah et al., 2004] : F. Deborah, P. McGuinness, H M. Frank , Owl web ontology language, rapport technique, W3C, http://www.w3.org/TR/owl-features/, 2004.

[Decker, 2000] : S. Decker, M. Erdmann, D. Fensel, I. Horrocks, M. Klein, F. van Harmelen, Oil in a nutshell, In Proceedings of the 12th European Knowledge Acquisition Workshop (EKAW'00), 2000.

[Dumez, 2010] : C. Dumez, Approche dirigée par les modèles pour la spécification, la vérification formelle et la mise en œuvre de services Web composés, thèse de doctorat Université de Technologie de Belfort-Montbéliard, 2010.

[Eric et al., 2005] : Newcomer, Eric; Lomow, Greg, Understanding SOA with Web Services, Addison Wesley, 2005.

[Erl, 2005] : Thomas Erl, Service-oriented Architecture : Concepts, Technology, and Design, Prentice Hall, 2005.

[Farrell et Lausen, 2007] : J. Farrell et H. Lausen, SAWSDL: Semantic Annotations for WSDL and XML Schema, rapport technique, W3C, http://www.w3.org/TR/sawsdl/, 2007.

[Fikes, 1985] : R. Fikes, T. Kehler, The Role of Frame-Based Representation in Reasoning, Communications of the ACM (CACM), 28(9), pp 904-920, 1985.

[Furst, 2002] : F. Furst. L'ingénierie ontologique, rapport technique, Institut de recherche en Informatique de Nantes, 2002.

[Genesereth, 2005] : Michael R. Genesereth, KIF: Knowledge Interchange Format, rapport technique, site web: http://logic.stanford.edu/kif/dpans.html, 2005.

[Guarino, 1998] : N. Guarino, Some ontological principles for designing upper level lexical resources, In Proceedings of the 1st International Conference on Language Resources and Evaluation, 1998.

[Horrocks, 2001] : I. Horrocks, F. van Harmelen, P.F. Patel-Schneider, Reference description of the DAML+OIL (March2001) ontology markup language, http://www.daml.org/2001/03/reference.html, 2001.

[Hubert et al., 2003] : K. Hubert , M. Valérie , LES WEB SERVICES, Edition DUNOD, 2003.

[Izza, 2006] : S. Izza, Integration des systèmes d'information industriels, Ecole Nationale Supérieure des Mines de Saint-Etienne, thèse de doctorat, 2006.

[Justin, 2002] : O'Sullivan Justin, Edmond David, Ter Hofstede Arthur, What's in a service Distrib. Parallel Databases ?, 12(2-3): 117–133, 2002.

[Kifer et al., 1995] : M. Kifer, G. Lausen, J. Wu., Logical foundation of Object-Oriented and Frames-Based language journal of the ACM , 42(4) : pp 741-843, 1995.

[Kopecký, 2005] : J. Kopecký, Aligning WSMO and WSDL-S, rapport technique, http://www.wsmo.org/TR/d30/v0.1/20050805/, 2005.

[Lassila, 1999] : O. Lassila, R. R. Swick, Resource description framework (rdf) model and syntax, w3c, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, 1999.

[Lausen et al., 2005] : H. Lausen, A. Polleres, D. Roman, Web Service Modeling Ontology (WSMO), rapport technique, http://www.w3.org/Submission/WSMO/, 2005.

[Lopez-velasco, 2008] : C. Lopez-velasco, Sélection et composition de services Web pour la génération d'applications adaptées au contexte d'utilisation, thèse de doctorat, université JOSEPH FOUR IER, 2008.

[Lortal, 2002] : Lortal GAËLLE, État de l'art ontologies et intégration/fusion d'ontologies, 2002.

[Luke, 2000] : S. Luke, J. Hein, Shoe 1.01 proposed specification, SHOE project, April 2000, http://www.cs.umd.edu/projects/plus/SHOE/spec.html, 2000.

[Mahfoud, 2011] : Mahfoud Sami, Interopérabilité sémantique des données échangées entre les services Web, engagés dans une composition (Vers une approche de médiation de données basée sur le contexte), mémoire de Magistère, université M'hamed Bougara-Boumerdès, 2011.

[Martin et al., 2004] : D. Martin, M. Bursteinal, J. Hobbs et al, Owl-s : Semantic markup for Web services, rapport technique, W3C, http://www.w3.org/Submission/OWL-S/, 2004.

[Minsky, 1975] : M. Minsky, A framework for representing knowledge, In Psychology of Computer Vision, P.H. Winston (Ed), pp 211-277, 1975.

[Mitra et al., 2003] : Mitra N. et Lafon Y, SOAP Version 1.2 Part 0: Primer (Second Edition), W3C, http://www.w3.org/TR/soap12-part0/, 2003.

[Monfort, 2003] : MONFORT .V, & KADIMA.H, les services Web: techniques, démarches et outils, Dunod, 2003.

[Mrissa, 2007] : M. Mrissa, Médiation Sémantique Orientée Contexte pour la Composition de Services Web, Université Claude Bernard Lyon I, thèse de doctorat ,2007.

[OASIS, 2006] : OASIS Reference Model for Service Oriented Architecture, Committee Specification, 2006.

[Psyché et al., 2003] : V.Psyché, O.Mendes et J.Bourdeau. Apport de l'ingénierie ontologique aux environnements de formation à distance. Revue STICEF, 10, 2003.

[Quillian, 1968] : M.R. Quillian, Semantic memory, In Semantic Information Processing, pp 216-270, MIT press, Cambridge, 1968.

[Raynaud et al., 2001] : Raynaud .O, Thi erry .E, A quasi optimal bit-vector encoding of tree hierarchies, Application to efficient type inclusion tests, 2001.

[Rocques, 2007] : Rocques P. et Vallées F., « UML 2 en action, de l'analyse des besoins à la conception », Eyrolles, 2007.

[Studer, 1998] : R. Studer, R. Benjamins, D. Fensel, Knowledge Engineering: Principles and Methods, Data and Knowledge Engineering, 25(1-2) pp 161-197, 1998.

[SOWA, 2000] : John SOWA. Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks/Cole, August 2000.

[Uschold, 1996]: Mike Uschold et Michael Grüninger. Ontologies: principles, methods, and applications. Knowledge Engineering Review, 11(2):93–155, 1996.

[W3C-WSA-Group, 2004]: W3C-WSA-Group, W3C Web Service Architecture Group, Web Services Architecture. http://www.w3.org/TR/ws-arch, 2004.

[Wirth, 1988] : N. Wirth, Type extension, ACM Trans, On Prog. Languges and Systems, 10 (2) : 204-214, 1988.

[Woods, 1975] : W.A. Woods, What's in a link: Foundation for Semantic Networks, In Representation and Understanding; Studies in Cognitive Science, D.G. Bobrow, A. Collins (Eds.), Academic Press, pp 35-82, 1975.

Résumé

La composition de services Web est l'une des motivations principales de l'architecture SOA. En effet, l'obtention de telles compositions est très gourmande en termes de temps, surtout lorsqu'on prendre en compte la sémantique (la subsumption).

L'objectif de ce travail est de proposer un ensemble de techniques d'encodage d'ontologie pour améliorer la performance du test de subsumption dans une composition de services Web. Les résultats obtenus sont très satisfaisants, et confirment l'utilité de ces approches dans la réduction de temps de réponse.

Mots clés :

Services Web, composition de services Web, Ontologie, test de subsumption, encodage d'ontologie, schéma d'étiquetage, fermeture transitive et réflexive de la matrice d'adjacence

Abstract

The SOA architecture is A new paradigm for developing distributed applications. It is mainly based on a set of principals such as loosely coupling and composition capability.

In this work, we propose a set of encoding techniques in order to reduce the execution time of the composition's generation. The obtained results are very encouraging and can serve for further ameliorations.

Keywords:

Web services, Web services composition, Ontology, subsumption test, Ontology encoding, labeling scheme, transitive and reflexive closing of contingence matrix.

ملخص:

تعد عملية تركيب خدمات الويب من أهم تحديات SOA .في الواقع الحصول على هاته التركيبة يتطلب الكثير من الوقت خاصة عندما نأخذ بعين الاعتبار دلالية المعلومات المتبادلة بين هذه الخدمات (التعميم) .

الهدف من هذا العمل هو اقتراح مجموعة من تقنيات لترميز الانطولوجيا من اجل تقليل وقت تنفيذ اختبارالتعميم .النتائج المتحصل عليها مرضية للغاية، وتشجع على العمل أكثر في هذا الميدان.

الكلمات المفتاحية :

خدمات الويب، تركيب خدمات الويب، الانطولوجيا، اختبار التعميم، ترميز الانطولوجيا.