République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Djilali BOUNAÂMA Khemis Miliana

Faculté des Sciences et de la Technologie

Département de Mathématiques et d'Informatique

Mémoire Présenté

Pour l'obtention de diplôme de

Master en Informatique

Spécialité: ingénierie du logiciel

Titre :

Composition des Services Web

Réalisé par : Remadelia Abdellah
Matallah Imane

Soutenu publiquement le :…. /…../…….

devant le jury composé de:

Mr A.Khalfi .Président

M.F.Boudali .Encadreur

M. H.Hachichi .Examinateur1

Mr N.Azzouza .Examinateur2

Année Universitaire 2015/2016

Remerciements

Nous remercions **Allah** le tout puissant, qui nous a donné la foi, la force et la patience pour aller jusqu'au bout de ce travail.

Au terme de ce travail de fin d'études nous tenons à exprimer notre gratitude et nos remerciements pour toutes les personnes qui ont contribués à sa réalisation.

Nous tenons tout d'abord à remercier notre promotrice **Mme Boudali Fatiha** pour son aide, ses conseils, son encouragement et sa disponibilité dans ce mémoire.

Ainsi que tous nos professeurs qui nous ont enseignés durant nos études à la faculté des sciences et techniques département mathématique et informatique.

Nos profonds remerciements pour les membres de jury qui ont acceptés d'évaluer ce travail.

A la fin nous tenons à remercier tous nos collègues d'études particulièrement notre promotion.

Sommaire

Chapitre II : La Composition des Services Web

Chapitre III : Conception du Système

Chapitre IV : Mise en œuvre du système

Conclusion générale

Références bibliographiques

Liste des figures

Résumé

Un web service peut être défini comme un programme autonome qui s'exécute sur le web. Les web services sont décrits par des descriptions WSDL, qui sont enregistrées dans des registres UDDI afin de faciliter leurs recherches (découverte) par la suite.

Ces derniers sont des applications accessibles sur Internet réalisant chacune une tâche spécifique. Pour fournir une solution à une tâche complexe, on peut regrouper des services web pour n'en former qu'un seul, on parle alors de composition de services web.

Créer des compositions de services (des services composites) signifie ordonner les invocations aux opérations, router les messages, modifier les paramètres et gérer les exceptions. Donc la composition des services web requiert plusieurs tâches : la description, la découverte des services, l'orchestration (l'exécution), la surveillance, et la coordination de l'échange des différents messages (informations).

L'objectif de ce travail est de développer un système qui vise à composer des services web. La prise en compte des informations sur les fonctionnalités et les caractéristiques des services pour développer le processus de composition est l'une des caractéristiques essentielles du système de composition.

Il y'a plusieurs approches pour la composition des services web. Ce mémoire est basé sur l'approche du web sémantique (OWL-S) en utilisant les ontologies pour sauvegarder les informations du système.

Mots clé : Composition, service web, ontologie, OWL-S.

Introduction générale

Introduction générale

Les dernières décennies ont été marquées par le développement rapide des systèmes d'information distribués, et particulièrement par la démocratisation de l'accès à l'internet. Cette évolution du monde informatique a entrainé le développement de nouveaux paradigmes d'interaction entre applications. Notamment, l'architecture orientée service (Service Oriented Architecture, ou SOA) qui a été mise en avant afin de permettre des interactions entre applications distantes. Cette architecture est construite autour de la notion de service, qui est matérialisé par un composant logiciel assurant une fonctionnalité particulière et accessible via son interface. Un service est toujours accompagné d'une description fournissant aux applications les informations nécessaires à son utilisation.

Les Web services, tels qu'ils sont présentés, sont conceptuellement limités à des fonctionnalités relativement simples qui sont modélisés par une collection d'opérations. Toutefois, pour certains types d'applications, il est nécessaire de combiner un ensemble de Web services (Web services simples) en web services plus complexes (Web services agrégés ou composites) afin de répondre à des exigences plus complexes.

L'objectif de la composition de service est de créer de nouvelles fonctionnalités en combinant des fonctionnalités offertes par d'autres services existants, composés ou non en vue d'apporter une valeur ajoutée. Un Web service est dit composite lorsque son exécution implique des interactions avec d'autres Web services, et des échanges de messages entre eux afin de faire appel à leurs fonctionnalités. La composition de Web services spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction. Il y'a plusieurs approches pour réaliser la composition des services web, mais dans notre travail on a utilisée l'approches du web sémantique notamment l'ontologie et OWL-S, qui constitue une prolongation, et une sorte de révolution de fond du Web actuel qui permet une définition non ambiguë de l'information, pour favoriser une meilleure coopération entre humain et machine. Il permet de s'ouvrir à de nouvelles possibilités d'automatisation d'une grande quantité d'information sur le Web. Notre document se compose de deux parties :

Partie 1 : « L'état de L'art » cette partie se comporte de deux chapitres :

Le premier chapitre est consacré aux notions fondamentales des Web services où nous avons défini l'architecture SOA et les standards de description, de publication et d'invocation de Web services,son fonctionnement et son architecture.

Dans **le deuxième chapitre**, nous allonsprésenter la composition des services web, ses types, quelque langages et approches proposées dans la littérature pour la résolution du problème de composition.

Partie 2 : « Conception et Réalisation » cette partie est aussi représentée par deux chapitres :

Le troisième chapitre représente la conception de notre système, son architecture et ces fonctionnalités principales.

Le quatrième chapitre traite la mise en œuvre du système, son environnement de développement (les outils utilisés) et l'implémentation de notre système.

Nous terminerons notre mémoire par **une conclusion générale** et nous énoncerons quelques perspectives de recherche.

ChapitreI

Les Service Web

I. Les Service Web

I.1. Introduction

La technologie des services Web (SW) est un moyen rapide de distribution de l'information entre clients, fournisseurs, partenaires commerciaux et leurs différentes plates-formes. Les services Web sont basés sur le modèle SOA (Architecture Orientée Services). D'autres technologies telles que RMI, DCOM et CORBA ont précédemment adopté ce style architectural mais ont généralement échoué en raison de la diversité des plates-formes utilisées dans les organisations et aussi parce que leur usage n'était pas adapté à Internet (problème de passage à travers des Firewalls, etc.) d'où la lenteur, voire l'absence de réponses sur ce réseau. Les applications réparties fondées sur ces technologies offrent des solutions caractérisées par un couplage fort entre les objets. Les solutions proposées par les services Web, permettent néanmoins un couplage moins fort. De plus, l'utilisation des technologies standards du Web tels que l'HTTP et XML par les SWfacilite le développement d'applications réparties sur Internet, et permet d'avoir des applications très faiblement couplées. L'intégration est sans doute le facteur essentiel qui favorise l'utilisation des SW.

Dans la suite de ce chapitre, nous présenterons tout d‘abord le concept des services web ainsi que son fonctionnement, Architecture en couches, et les Principaux standards utilisé dans les services Web.

I.2. Architecture Orientée Services :

L'architecture orientée services (SOA) est une architecture logicielle s'appuyant surun ensemble de composants simples appelés **Services Web**. Son objectif est de décomposer une fonctionnalité complexe en un ensemble de fonctionnalités basiques, fournies par des services Web et de décrire finement le schémad'interaction entre ces services Web. SOA implique quatre entités d'interaction : lesservices Web, les fournisseurs de services, les annuaires de service et les demandeurs des services auxquels nous rajoutons les attributs. [1]

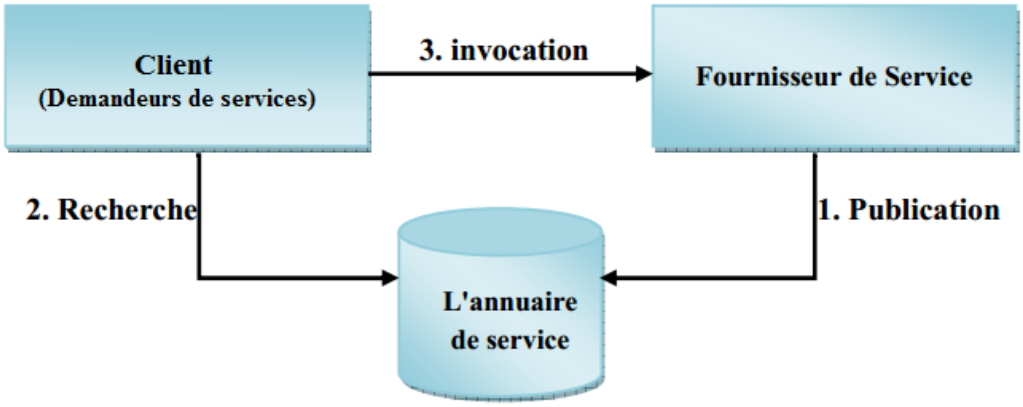


Figure 1 : Modèle Fonctionnel de l'architecture SOA. [1]

I.3. Définition des Services Web :

Un service Web selon W3Cest une application ou un composant logiciel conçu pour permettre une interaction interopérable de machine à machine via un réseau. Identifié par un URI (Uniform Resource Identifier), il dispose d'une interface décrite dans un format exploitable par des machines. Il peut également interagir directement avec d'autres services Web à travers le langage XML et en utilisant des protocoles Internet (dans la plupart des cas, le protocole utilisé est SOAP). [2]

Les services Web sont décrits dans un langage standard appelé WSDL, Cette descriptionspécifie la fonctionnalité du service, les types de données employés dans les messages, le protocole de transport utilisé ainsi que des informations pour localiser le service.
Grâce à la description des services, les demandeurs s'abstraient des techniques d'implémentation, comme par exemple les langages de programmation utilisés ou leurs plateformes d'exécution.

Les services Web sont **normalisés** car ils utilisent les standards XML et HTTP pour transférer des données, ils sont aussi compatibles avec de nombreux autres environnements de développement. Ce qui les rend indépendants des plates-formes. C'est dans ce contexte qu'un intérêt très particulier a été attribué à la conception des services Web puisqu'ils permettent aux entreprises d'offrir des applications accessibles à distance par d'autres entreprises. Cela s'explique par le fait que les services Web n'imposent pas de modèles de programmation spécifiques.

En d'autres termes, les services Web ne sont pas concernés par la façon dont les messages ont produits ou consommés des programmes.
Cela permet aux vendeurs d'outils dedéveloppement d'offrir différentes méthodes et interfaces de programmation au-dessus de n'importe quel langage de programmation.

Les services Web représentent donc la façon la plus efficace de partager des méthodes et des fonctionnalités. De plus, ils réduisent le temps de réalisation en permettant de tirer directement parti de services existants.

D'après l'explication précédente un service Web possède les caractéristiques suivantes :

- Il est accessible via le réseau
- Il dispose d'une interface publique (ensemble d'opérations) décrite en XML,
  Ses descriptions (fonctionnalités) sont stockées dans un annuaire.
- Il communique en utilisant des messages XML, ces messages sont transportés par des protocoles Internet (généralement HTTP, mais rien n'empêche d'utiliser d'autres protocoles de transfert tels que : SMTP, FTP, BEEP...). [3]

I.4. Fonctionnement et Architecture des services Web :

Les services Web reprennent la plupart des idées et des principes du Web (HTTP, XML), et les appliquent à des interactions entre machines. Comme pour le World Wide Web, 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.

Avant de procéder à l'architecture et le fonctionnement des services web, nous devons en savoir plus sur les technologies utilisées par les services Web [3] :

XML-RPC :

XML-RPC est un protocole simple utilisant XML pour effectuer des messages RPC. Les requêtes sont écrites en XML et envoyées via HTTP POST. Les requêtes sont intégrées dans le corps de la réponse HTTP. XML-RPC est indépendant de la plate-forme, ce qui lui permet de communiquer avec diverses applications.

SOAP :

SOAP (Simple Object Access Protocol) est un protocole standard de communication. C'est l'épine dorsale du système d'interopérabilité. SOAP est un protocole décrit en XML et standardisé par le W3C. Il se présente comme une enveloppe pouvant être signée et pouvant contenir des données ou des pièces jointes.

Il circule sur le protocole HTTP et permet d'effectuer des appels de méthodes à distance.

(C'est une légère simplification qui sera expliqué plus loin dansI.5.2).

WSDL :

WSDL (Web Services Description Language) est un langage de description standard. C'est l'interface présentée aux utilisateurs. Il indique comment utiliser le service Web et comment interagir avec lui. WSDL est basé sur XML et permet de décrire de façon précise les détails concernant le service Web tels que les protocoles, les ports utilisés, les opérations pouvant être effectuées, les formats des messages d'entrée et de sortie et les exceptions pouvant être envoyées. (C'est une légère simplification qui sera expliqué plus loin dansI.5.3).

UDDI :

UDDI (Universal Description, Discovery and Integration) est un annuaire de services. Il fournit l'infrastructure de base pour la publication et la découverte des services Web. UDDI permet aux fournisseurs de présenter leurs services Web aux clients.

Les informations qu'il contient peuvent être séparées en trois types :

- les pages blanches qui incluent l'adresse, le contact et les identifiants relatifs au service Web.
- les pages jaunes qui identifient les secteurs d'affaires relatifs au service Web ;
- les pages vertes qui donnent les informations techniques.

(C'est une légère simplification qui sera expliqué plus loin dansI.5.4).

I.4.1. Fonctionnement des services Web

Le fonctionnement des services Web s'articule autour de trois acteurs principaux illustrés dans la figure suivante [3] :

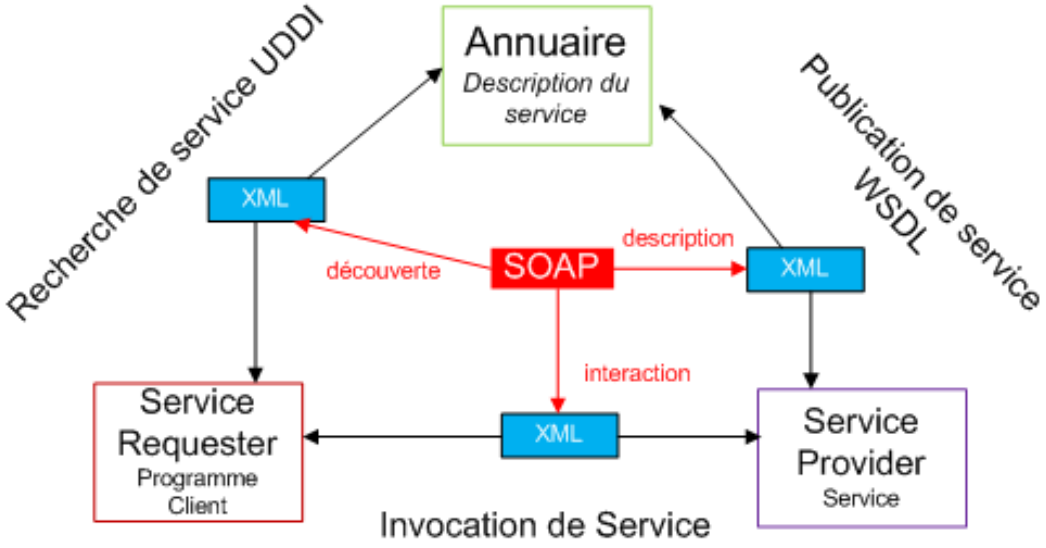


Figure 2 : Fonctionnement d'un Service Web

Service provider service : Le fournisseur de service met en application le service Web et le rend disponible sur Internet.

Service requester programme client : 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).

Annuaire service registry : 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 [3] :

- 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 l'utilise pour l'invoquer auprès du fournisseur de services.

I.4.2. Architecture en couches des services Web :

Les services Web emploient un ensemble de technologies qui ont été conçues afin de respecter une structure en couches sans être dépendante de façon excessive de la pile des protocoles. Cette structure est formée de quatre couches majeures :

Découverte de services	**UDDI**	**La publication et la découverte des services web sont assurées par le biais du référentiel UDDI, un référentiel UDDI est un catalogue de services Web.**
Description de services	WSDL	La description d'un service Web se fait en utilisant le langage WSDL. il expose l'interface du service.
Communication	SOAP	SOAP prévoit la couche de communication basée sur XML pour accéder à des services Web.
Transport	HTTP	Le transport de messages SOAP est assuré par le standard HTTP.

Table 1 : Description en couche des services Web

I.4.2.1. Couche transport :

Cette couche est responsable du transport des messages XML échangés entre les applications. Actuellement, cette couche inclut HTTP, SMTP, FTP, et de nouveaux protocoles tels que BEEP.

I.4.2.2. Couche communication :

Cette couche est responsable du formatage des données échangées de sorte que les messages peuvent être compris à chaque extrémité. Actuellement, deux styles architecturaux totalement différents sont utilisés pour ces échanges de données. Nous avons d'un côté l'architecture orientée opérations distribuées (protocoles RPC) basée sur XML et qui comprend XML-RPC et SOAP et de l'autre côté une architecture orientée ressources Web, REST (Représentationnel State Transfer) qui se base uniquement sur le bon usage des principes du Web (en particulier, le protocole HTTP).

I.4.2.3. Couche description de service :

Cette couche est responsable de la description de l'interface publique du service Web. Le langage utilisé pour décrire un service Web est le WSDL qui est la notation standard basée sur XML pour construire la description de l'interface d'un service. Cette spécification définit une grammaire XML pour décrire les services Web comme des ensembles de points finaux de communication (ports) à travers lesquels on effectue l'échange de messages.

I.4.2.4. Couche publication de service :

Cette couche est chargée de centraliser les services dans un registre commun, et de simplifier les fonctionnalités de recherche et de publication des services Web. Actuellement, la découverte des services est assurée par un annuaire UDDI (Universal Description, Discrovery, and Integration).

I.5. Principaux standards des services Web

Une caractéristique qui a permis un grand succès de la technologie des services Web et qui est construite sur un certain nombre de technologies standards de l'industrie. En commençant par XML, comme langage de base commun ensuite les trois principaux standards qui sont : SOAP, WSDL et UDDI que nous détaillerons ci-dessous.

I.5.1. XML (eXtensibleMarkupLanguage) :

I.5.1.1. *Définition :*

XML et un langage de balisage qui permet la description et la structuration des données, Ce métalangage est particulièrement performant pour l 'échange et le stockage de tous types de données et de leurs structures. [4]

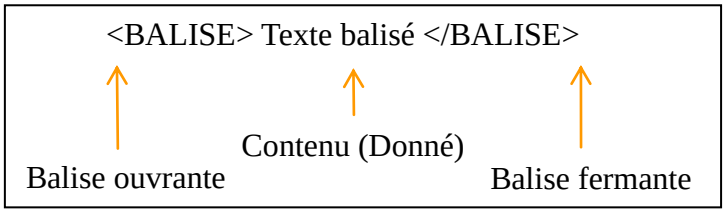


Figure 3 : Exemple d'une balise XML

I.5.1.2. *Structure d'un document XML*

Un document XML est un arbre composé d'un ensemble d'éléments structurés sous forme de Balises. Cette structuration hiérarchisée ouvre la voie au traitement automatique du document pour l'extraction d'information de sa structure aussi bien que de soncontenu.

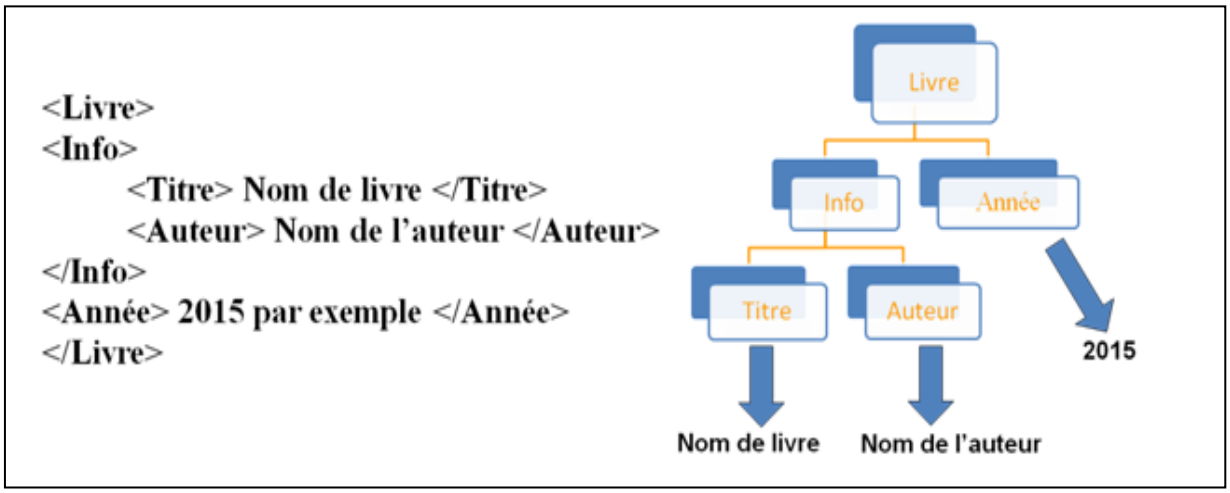


Figure 4 : Structure hiérarchisée d'un document XML

Il existe une panoplie d'outils permettant l'exploitation des documents XML. Parmi celles-ci

Nous pouvons citer :

• Xpath : pour définir la manière d'adresser des parties d'un document XML.

• Xquery : s'intéresse à la structure logique abstraite d'un document XML.

• XSLT : pour définir la présentation de document XML

Un ensemble de standards basés sur XML et traitants des différents aspects inhérents au déploiement des services Web s'est développé et se développe encore. Ces spécifications émanant de constructeurs et consortium divers permettent de construire des services Web, d'assurer l'interopérabilité entre composants applicatifs et de garantir des conditions opérationnelles d'exploitation réelles de manière à:

• Assurer le fonctionnement et le déploiement des services Web: SOAP, WSDL, UDDI…

• Effectuer l'échange de données dans un contexte de commerce interentreprises: xCBL RosettaNet, …Etc.

• Assurer l'intégration de processus collaboratifs sur le Web (Workflow): BPML, XLANG, BPEL …

• Fournir des mécanismes de sécurité: XKMS … [5].

I.5.2. SOAP (Simple Object Access Protocol):

I.5.2.1. Définition :

SOAP est un standard proposé par W3C [6]. Il représente un protocole à la fois simple et léger destiné à l'échange d'informations. Ce dernier est de type RPC (RemoteProcedure Call) d'échange de messages entre les Web services, standard ouvert et entièrement basé sur le langage XML pour assurer l'interopérabilité entre composants et les appels de procédures distantes (RPC).

Ce protocole ne requiert aucune plate-forme spécifique, ni d'exigence particulière en ce qui concerne l'implémentation de l'application. SOAP peut cependant reposer sur d'autres protocoles de transport comme par exemple SMTP ou JMS (Java Message Service) mais HTTP est le plus populaire. Un message SOAP est un document XML encapsulé dans une enveloppe permettant d'organiser les informations d'une manière à être échangées entre partenaires.

Il existe deux types de messages SOAP :

• Request : il inclut le nom de la méthode à invoquer et la liste de ses paramètres.

• Response: il inclut le résultat de la méthode invoquée dans le message Request.

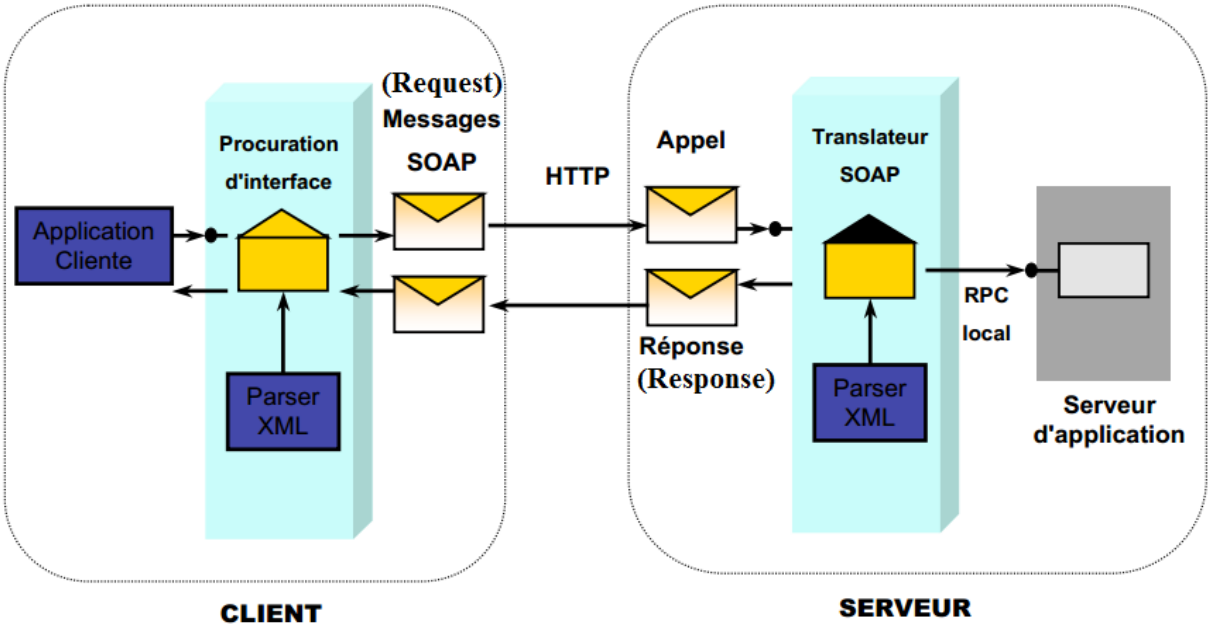


Figure 5 : Exemple d'utilisation message SOAP dans une application client/serveur [7]

I.5.2.2. Structure d'un message SOAP [7] :

Envelope: C'est lui qui contient le message et ses différents sous-blocs. Il s'agit du bloc racine XML. Il peut contenir un attribut encodingStyledont la valeur est un URL vers un fichier de typage XML qui décrira les types applicables au message SOAP.

Header : C'est un bloc optionnel qui contient des informations de l'en-tête sur le message. S'il est présent, ce bloc doit toujours se trouver avant le bloc Body à l'intérieur du bloc Envelope.

Body : C'est le bloc qui contient le corps du message. Il doit absolument être présentde manière unique dans chaque message et être contenu dans le bloc Envelope.

SOAP ne définit pas comment est structuré le contenu de ce bloc. Cependant, ildéfinit le bloc qui peut s'y trouver.

Fault: Ce bloc est la seule structure définie par SOAP dans le bloc Body. Il sert àreporter des erreurs lors du traitement du message, ou lors de son transport. Il nepeut apparaître qu'une seule fois par message. Sa présence n'est pas obligatoire.

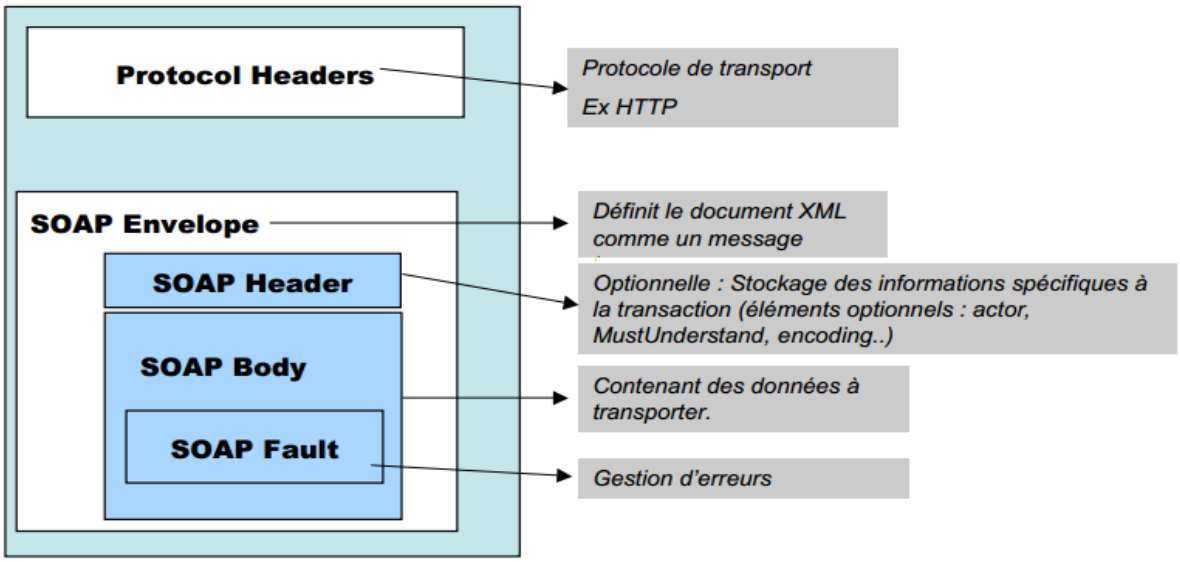


Figure 6 : Structure d'un message SOAP

I.5.2.3. Objectif du SOAP [8] :

- Offrir un format de message pour les communications entre partenaires, décrivant comment les informations à échanger sont structurées dans un document XML.
- Définir les conventions nécessaires à l'appel de procédures à distance pour invoquer un service par un client, en envoyant un message SOAP, et comment le service répond par un autre message SOAP.
- Décrire comment un message SOAP doit être transporté au-dessus de HTTP ou SMTP.

Il s'agit, donc, d'un mécanisme fondamental pour assurer l'interaction et le dialogue entre applications distribuées dans un environnement services Web, du moment que clients et fournisseurs peuvent formuler, transmettre et comprendre les messages SOAP échangés[9].

I.5.3. WSDL (Web Service Description Language) :

I.5.3.1. Définition :

WSDL [10] décrit les services Web et particulièrement leur interface dans le format XML. En plus de la spécification des opérations offertes par le service, WSDL décrit les mécanismes pour l'accès aux services Web (Protocole de communication) et sa localisation (URI), afin de préciser où envoyer les messages SOAP. Un document WSDL décrit essentiellement le nom de la méthode utilisée, son nombre de paramètres, et leur type, ce qu'un service Web offre, où il est hébergé et comment on peut l'invoquer.

I.5.3.2. Les objectifs de WSDL:

- Décrire les interfaces des services en précisant les opérations offertes et leur signature (Paramètres d'Entrée/Sorties et types). Ces interfaces constituent les contrats que les Clients doivent respecter pour pouvoir interagir avec le service.
- Préciser le (s) protocole (s) d'accès au service (HTTP, SMTP…).
- Fournir la localisation du service où il peut être invoqué (URI).
- Définir une description de la sémantique du service par la fourniture des informations Permettant, éventuellement, aux développeurs de traiter la sémantique des services. [5]

I.5.3.3. Spécification d'un service Web avec WSDL

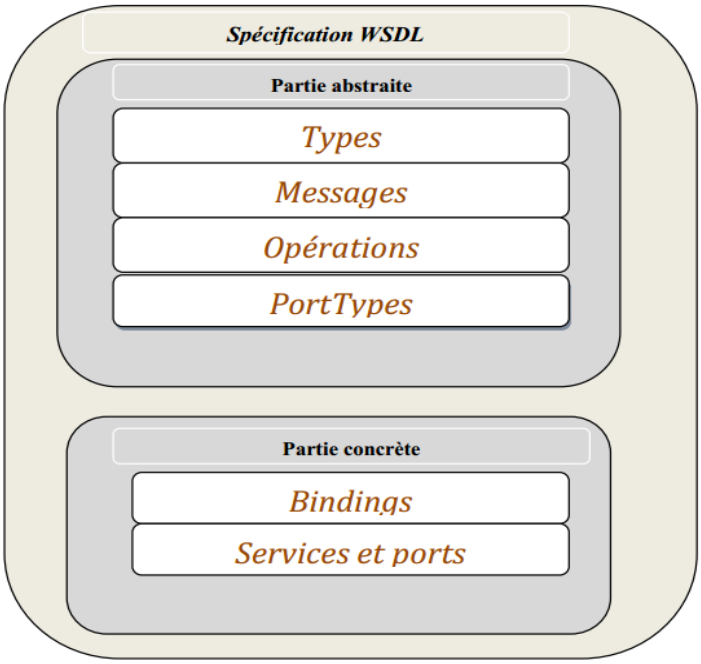


Figure 7 : Spécification d'un service Web avec WSDL

Dans une spécification WSDL, on décrit chaque service Web nouvellement créé par Sept éléments [3] :

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.

Types port (PortTypes) : décrit un ensemble d'opérations. Chaque opération à 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. Ainsi, la spécification WSDL joue un rôle important dans l'interopérabilité des services Web. Moyennant cette spécification, WSDL permet aux services de définir ce qui est nécessaire à leur invocation et à la réception de leur résultat. La spécification WSDL sépare la définition abstraite du service (échange de messages) de ses mécanismes de liaison (définition des protocoles applicatifs). Cette dernière caractéristique permet au service d'interagir même si l'application a été modifiée ce qui est un point important pour assurer l'interopérabilité des services.

I.5.4. UDDI (Universal Description, Discovery and Integration):

I.5.4.1. Définition :

L'annuaire des services UDDI [11] est un standard pour la publication et la découverte des informations sur les services Web. La spécification UDDI est une initiative lancée par ARIBA, Microsoft et IBM. Cette spécification n'est pas gérée par le W3C mais par le groupe OASIS. La spécification UDDI vise à créer une plate-forme indépendante, un espace de travail (Framework) ouvert pour la description, la découverte et l'intégration des services des entreprises.

I.5.4.2. La découverte et la Consultation de l'annuaire :

La découverte exploite les informations liées aux services Web dans le but de localiser des services capables de répondre à une requête particulière avec la meilleure adéquation possible. Dans le processus de découverte, la mise en correspondance consiste à rechercher les similitudes potentielles entre la requête et les services Web publiés dans les annuaires. Elle s'opère par une comparaison des propriétés de la requête avec les propriétés des services Web disponibles. Plus particulièrement, les services recherchés doivent produire les mêmes buts que le service demandé et exiger les même entrées.

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 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. [5]

I.5.4.3. Scénario classique d'utilisation :

Les entreprises publient les descriptions de leurs services Web en UDDI, sous la forme de fichiers WSDL. Ainsi, les clients peuvent plus facilement rechercher les services Web dont ils ont besoin en interrogeant le registre UDDI.

Lorsqu'un client trouve une description de service Web qui lui convient, il télécharge son fichier WSDL depuis le registre UDDI. Ensuite, à partir des informations inscrites dans le fichier WSDL, notamment la référence vers le service Web, le client peut invoquer le service Web et lui demander d'exécuter certaines de ses fonctionnalités.

Le scénario classique d'utilisation d'UDDI est illustré ci-dessous. L'entreprise B a publié le service Web S, et l'entreprise A est client de ce service :[3]

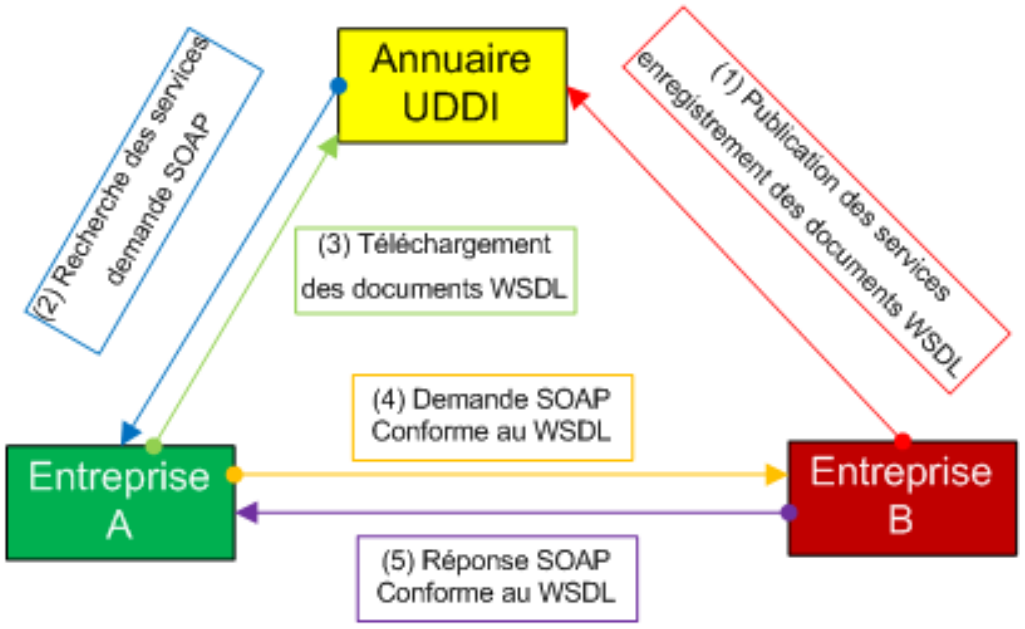


Figure 8 : Schéma générale de l'annuaire UDDI

I.5.4.4. Structures de données UDDI

Un registre UDDI se compose de quatre types de structures de données, 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.

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 « business Services » 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.

I.5.4.5. Les objectifs de l'annuaire UDDI :

- Offrir un cadre pour la description et la découverte des services Web, en fournissant des structures de données et des API (Application Programming Interfaces), pour la publication des descriptions des services dans le registre et pour la découverte de ces publications pour les clients.
- Soutenir les développeurs dans la recherche des informations concernant le service afin qu'ils puissent réaliser des clients pour interagir avec ces services.
- Permettre les liens (Binding) dynamiques en offrant aux clients la possibilité d'interroger le registre UDDI et récupérer les références des services (description WSDL..) qui les intéressent. [5]

I.6. Conclusion :

Les services Web regroupent tout un ensemble de technologies bâties sur des standards (SOAP, WSDL, UDDI, XML). Ils permettent de créer des composants logiciels distribués, de décrire leur interface et de les utiliser indépendamment de la plate-forme sur laquelle ils sont implémentés. La recherche dans ce domaine est très active et s'intéresse entre autres à la composition, l'orchestration, la sécurité et la sémantique desservices Web...etc.

Dans le chapitre suivant nous aborderons le conceptde la composition des services Web.

ChapitreII

La Composition des Services Web

II. La Composition des Services Web

II.1. Introduction

La composition de services est considérée comme l'une des motivations les plus importantes du paradigme SOA, en effet si une application ou un client requièrent des fonctionnalités, et qu'aucun service n'est seul apte à les fournir, il devrait être possible de combiner ou de composer des services existants afin de répondre aux besoins de cette application ou de ce client. C'est ce que l'on appelle la composition de services [12].

La composition de services vise à faire inter-opérer, interagir et coordonner plusieurs services pour la réalisation d'un but. Malgré les efforts de recherche et de développement autour de la problématique de la composition des services, elle reste une tâche hautement complexe et pose un certain nombre de défis. Sa complexité provient généralement des sources suivantes :

- L'augmentation dramatique du nombre des services web sur le web rend très difficile la recherche et la sélection des services web pouvant répondre à un besoin donné.
- Les services sont créés et mis à jour de façon hautement dynamique.
- Les services web sont d'habitude développés par différentes organisations qui utilisent différents modèles conceptuels pour décrire les caractéristiques des services web.
- La modularité constitue une caractéristique importante des services Web, par conséquent, les services Web composites doivent garder récursivement les mêmes caractéristiques que les services Web basiques à savoir auto-descriptifs, interopérables, et facilement intégrables [12].

Dans ce chapitre nous présenterons le concept de composition des services Web. Nous citerons les principaux types de compositions: orchestration/chorégraphie, statique/dynamique.

Nous mettrons en relief le fait qu'une composition peut être manuelle ou (semi) automatique.

Ensuite nous étudierons les langages de composition et pour finirnous citons quelques travaux qui proposent des approches de composition.

II.2. Définition :

La composition de services est le mécanisme qui permet l'intégration des services dans une application. Le résultat de la composition des services peut être un nouveau service, appelé service composite.

La composition de services est aujourd'hui un sujetde grand intérêt autant pour le monde de la recherche que pour le monde industriel. De nombreuses recherches visent à développer des modèles de composition de services et à fournir lesoutils nécessaires pour la composition de services [5].

La réalisation d'une application par composition des services comporte plusieurs étapes qui permettent le passage incrémental d'une spécification abstraite vers une composition concrète de services, c'est à dire une composition prête à être exécutée.

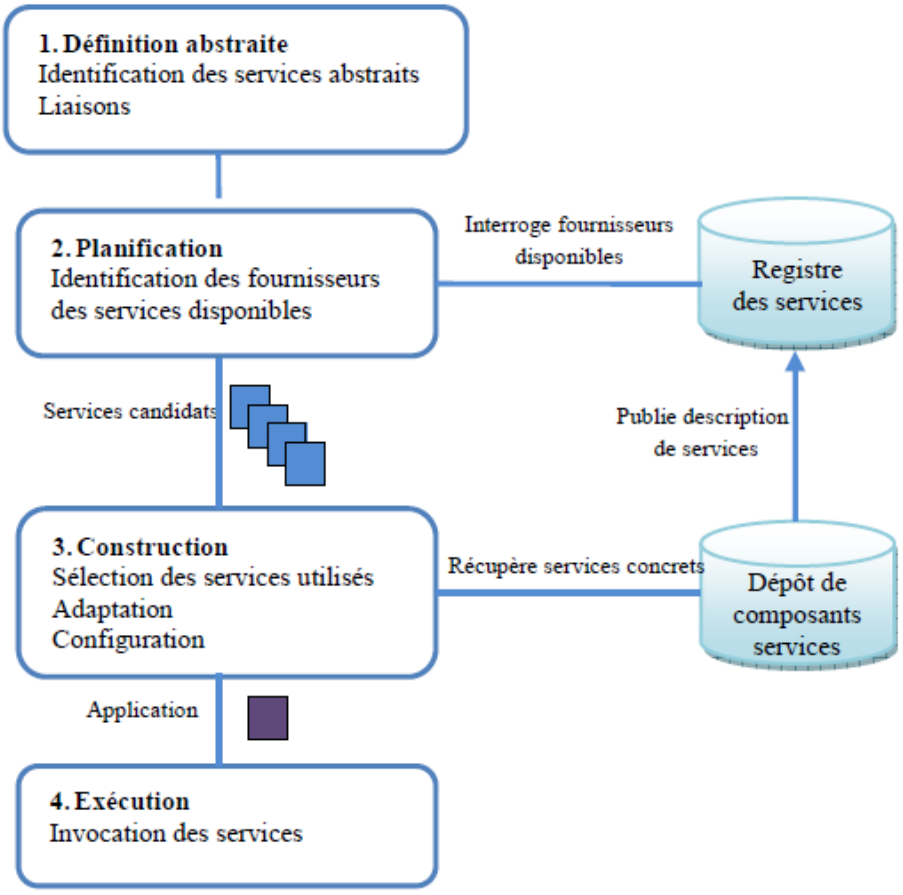


Figure 9 : Etapes de réalisation d'une composition de services.

La définition abstraite de la composition des services a comme point de départ l'identification des fonctionnalités qui doivent être remplies par l'application ainsi créée. Elle

demande l'identification des fonctionnalités abstraites que les différents participants doivent fournir ainsi que celle des interactions qui auront lieu entre ces participants.

La planification identifie les fournisseurs mettant à disposition des services compatibles avec les besoins fonctionnels identifiés au préalable. Les fournisseurs ainsi identifiés représentent des candidats pour la réalisation de la composition concrète de services.

La construction sélectionne (selon une stratégie donnée) les fournisseurs qui mettent à disposition les fonctionnalités nécessaires parmi les candidats précédemment identifiés.
Cette étape prépare pour l'exécution les services concrets correspondants : les services sont configurés et d'éventuelles adaptations sont réalisées. Dans le cas d'un service composite, la description du service ainsi réalisée est publiée dans un ou plusieurs registres de services.

La phase d'exécution de l'application ainsi obtenue réalise l'invocation des services préparés au préalable.

II.3. Cycle de vie d'une composition de Services Web :

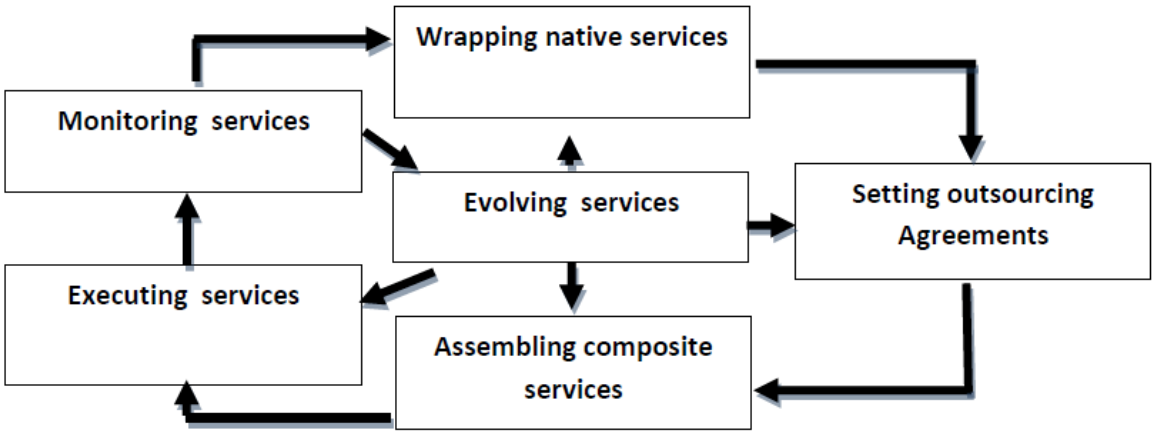


Figure 10 : cycle de vie de d'une composition de services Web

Le cycle de vie d'une composition des services Web est définit à partir de six activités :

L'encapsulation de services natifs (Wrapping services) : Cette première activité permet de s'assurer que tout service peut être appelé lors d'une composition, indépendamment de son modèle de données, de son format de message, et de son protocole d'interaction.

L'établissement d'accord d'externalisation (Setting outsourcing agréments) : Cette seconde activité consiste à négocier, établir, et appliquer des obligations contractuelles entre les services.

L'assemblage de services composants (Assembling composite services) : Cette activité permet de spécifier, à un haut niveau d'abstraction, l'ensemble des services à composer afin d'atteindre l'objectif attendu. Cet assemblage comporte une phase d'identification des services et

de spécification de leurs interactions conformément aux descriptions et aux accords entre services.

L'exécution de services composants (Executing services) : Cette activité consiste en l'exécution des spécifications de la composition précédemment définies.

Lecontrôle de l'exécution de services composites (Monitoring services) : La phase de contrôle permet de superviser l'exécution de la composition en vérifiant, par exemple, l'accès aux services, les changements de statut, les échanges de messages. Ce contrôle permet de détecter des violations de contrats, de mesurer les performances des services appelés et de prédire des exceptions.

L'évolutivité des services (Evolving services) : Cette dernière phase permet de faire évoluer la composition en modifiant les altérations de l'organisation de services, en utilisant de nouveaux services, ou en prenant en compte les retours de la phase de contrôle.[12]

II.4. Types de composition de services Web

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

II.4.1. L'orchestration :

L'orchestration est définie comme un ensemble de processus exécutés dans un ordre prédéfini afin de répondre à un but. Ce type de composition permet de centraliser l'invocation des services Web composants. Chaque service est décrit en termes d'actions internes. Les contrats entre deux services sont constitués selon le processus à exécuter. [15]

L'orchestration des services Web consiste en la programmation d'un moteur qui appelle un ensemble de services Web selon un processus prédéfini. Ce moteur définit le processus dans son ensemble et appelle les services Web (tant internes qu'externes à l'organisation) selon l'ordre des tâches d'exécution. (La Figure 11) illustre l'exécution du moteur (lui-même un service Web – Service Web Moteur) permise par l'enchaînement de l'exécution de deux autres services Web (le Service Web 1 puis le Service Web 2). Cet enchaînement est possible via un opérateur d'ordonnancement (représenté par le losange dans la Figure 11). L'exécution de la composition repose sur l'appel du Service Web 1, puis sur l'appel du Service Web 2, réalisés tous deux par le Service Web Moteur.[14]

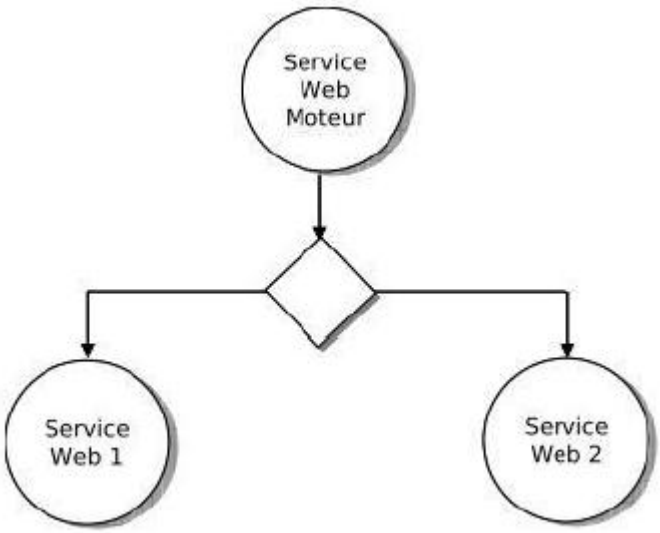


Figure 11 : Illustration de l'orchestration, d'après [16]

En d'autres termes, 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 moteur d'orchestration est une entité logicielle qui joue le rôle d'intermédiaire entre les services en les appelants suivant le script d'orchestration.

II.4.2. La chorégraphie :

La chorégraphie permet de décrire la composition comme un moyen d'atteindre un but commun en utilisant un ensemble de services Web. La collaboration entre chaque service Web de la collection (faisant partie de la composition) est décrite par des flots de contrôle. Pour concevoir une chorégraphie, les interactions entre les différents services doivent être décrites. La logique de contrôle est supervisée par chacun des services intervenant dans la composition. L'exécution du processus est alors distribuée. [14]

La description de chaque service Web intervenant dans la chorégraphie inclut la description de sa participation dans le processus. De ce fait, ces services peuvent collaborer à l'aide de messages échangés afin de savoir si tel ou tel service peut aider dans l'exécution de la requête. Chaque service Web peut communiquer avec un autre service Web par l'intermédiaire d'échange de messages. (La Figure 12) représente un protocole d'initiation de collaboration entre deux services dans le cadre d'une chorégraphie. Dans cet exemple, le Service Web 1 demande l'exécution d'une méthode du Service Web 2 par l'intermédiaire d'un envoi de message

(Requête de service). Cette requête est acceptée par le service Web 2. Ce dernier envoie un message d'acceptation au Service Web 1(Acceptation). Le Service Web 1 accepte le service proposé par le Service Web 2 en lui envoyant un message (Service admis) accordé (Acceptation) par ce second service. Une fois ces messages échangés le Service Web 1 peut invoquer le Service Web 2 dans le cadre de la composition. [13]

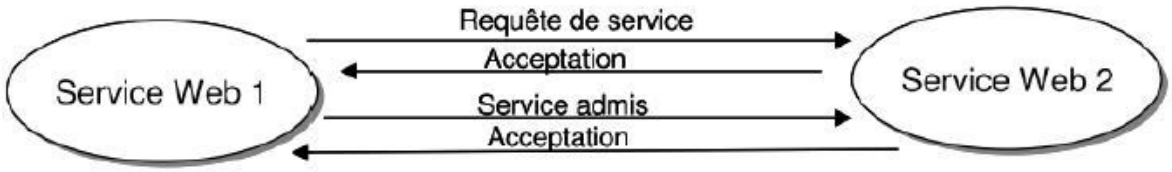


Figure 12 : L'illustration de la chorégraphie

La chorégraphie est aussi appelée composition dynamique [15]. 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 (i.e. à chaque étape de la composition), un 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.

L'idée de la composition de services Web consiste à définir comment les services Web vont être rassemblés selon certains règles, pour atteindre le but demandé par un utilisateur. Une fois que, la description de la composition est réalisée, il est possible de savoir facilement quels services web appartiendront à cette composition. Les solutions proposées peuvent être classifiées selon deux axes :

II.4.3. Selon le degré d'automatisation :

En fonction du degré de participation de l'utilisateur dans la définition du schéma de composition, ces propositions sont : manuelles, semi-automatiques ou automatiques [13] :

II.4.3.1. La composition manuelle :

La composition manuelle des services Web suppose que l'utilisateur gère la composition à la main via un éditeur de texte et sans l'aide d'outils dédiés.

II.4.3.2. La composition semi-automatique:

Les techniques de composition semi-automatiques sont un pas en avant en comparaison avec la composition manuelle, dans le sens qu'elles font des suggestions sémantiques pour aider à la sélection des services Web dans le processus de composition.

II.4.3.3. La composition automatique :

La composition totalement automatisée prend en charge tout le processus de composition et le réalise automatiquement, sans qu'aucune intervention de l'utilisateur ne soit requise.

II.4.4. Statique/dynamique

On pourra dire qu'une autre approche est statique ou dynamique on se basant sur le fait que la sélection des services web et la gestion du flot soient a priori réaliser ou non. [16]

II.4.4.1. La composition statique des services web :

Dans cette approche, les services web à composer sont choisis à l'heure de faire l'architecture et le design. Les composants sont choisis et reliés ensemble, avant d'être compilés et déployés .Ceci peut marcher en autant que l'environnement des services web, les partenaires d'affaires, ainsi que les dits composants changent peu ou pas du tout. Microsoft Biztalk et Bea Weblogic sont deux exemples de moteurs de composition statique de services web. Si les fournisseurs de services proposent d'autres services ou changent les anciens services, des incohérences peuvent être causées, ce qui demanderait un changement de l'architecture du logiciel, voire de la définition du processus et engendrait l'obligation de faire une nouvelle conception du système. Dans ce cas, la composition statique des services web est considérée trop restrictive: les composants doivent s'adapter automatiquement aux changements imprévisibles [12].

II.4.4.2. La composition dynamique des services web :

Dans ce type de composition, les services Web à composer sont déterminés lors de l'exécution de la requête d'un client. Ils peuvent être déterminés selon les contraintes de chaque client, la disponibilité des services web, ….etc. La composition dynamique apparaît la plus intéressante d'une part, elle promet d'être capable de faire face à un environnement très dynamique dans lequel des services apparaissent et disparaissent rapidement. D'autre part, elle permet de mieux satisfaire les besoins de chaque client en minimisant son intervention.

La composition des services web implique : la découverte des « «bons » services à composer selon des contraintes et des besoins, la dynamité et la flexibilité dans la composition de service web. [16]

II.5. Langagesde composition des services Web :

De nombreux industrie (tels qu'IBM ou Microsoft) et consortium (tel que le W3C) travaillent afin de mettre en œuvre un langage de composition de services Web standard. Dans

cette partie, nous présenterons quelques standards ayant émergé dans le domaine de la composition des services Web [5] [17] :

- **BPML (Business Process Modeling Language):**C'est un métalangage de modélisation des processus collaboratifs qu'engage l'entreprise avec ses partenaires sur Internet. Il prend en compte aussi les aspects les plus complexes de la gestion et de la coordination de ces processus, en abordant les transactions, la sécurité et la liaison dynamique avec les services Web en cours d'exécution.
- **XLANG :** XLANG de Microsoft, permet de décrire formellement les processus d'entreprises comme des interactions entre participants. La description complète d'un processus consiste aussi bien à définir le comportement de chaque participant, qu'à décrire la manière dont l'ensemble des participants interagit pour produire le processus complet...
- **WSFL(Web Service Flow Language) :**C'est une approche IBM qui permet d'intégrer la composition de services Web comme un processus métier statique. C'est un langage basé XML permettant de décrire des orchestrations de services Web.
- **BPEL4WS (Business ProcessExecutionLanguage For Web Services)** :BPEL4WS est un langage de composition qui décrit les interactions entre les services Web qui composent un processus métier, proposé par Microsoft, IBM, BEA et d'autres grands acteurs du monde de l'informatique et de l'Internetpour palier le problème de composition des SW. Il a remplacé les précédents langages XLANG (Microsoft) et WSFL (IBM). BPEL4WS permet la coordination des services Web dans un processus métier.
- **WSCL (Web Service Conversation Language) :** WSCL a été proposé par le consortium W3C, il permetde définir les interfaces abstraites des services Web, c.-à-d. les conversations au niveau métier soutenues par un service Web. Il spécifie les documents XML à échanger et leur ordre.
- **WSCI (Web Service Choregraphy Interface) :** C'est un langage reposant sur XML et qui décrit le flux de messages échangés par un service Web participant à une chorégraphie.WSCI fonctionne en conjonction avec WSDL pour construire l'interface dynamique du service Web en réutilisant les opérations définies pour l'interface statique.

II.6. Les différentes approches de la composition automatique

Différentes approches ont été proposées pour composer les services web, la plupart des recherches dans ce domaine peuvent se regrouper selon trois axes :

II.6.1. Le Workflow:

Plusieurs auteurs traitent la composition de services en utilisant les workflow. La définition de la composition de services correspond alors à un ensemble de services atomiques (ou composés) sur lesquels on effectue un contrôle du flux. D'autre part, un workflow a également besoin de définir un flux d'activités.

Un workflow est une abstraction d'un processus de type business. Il est composé d'un nombre d'échelons logiques (aussi appelés tâches ou activités), de dépendances entre tâches, de règles et de participants. Dans ce dernier, une tâche peut représenter une activité humaine ou un système (logiciel). Il est nécessaire d'associer une tâche à un service quand il est appliqué aux services web.

Une composition d'un workflow implique la sélection de tâches appropriées à des fonctionnalités désirées, nous devant prendre en compte les connections entre ces tâches (flux de contrôle et de données). Les workflows qui gèrent les services web (aussi appelé d'E-Services) sont appelés des E-Workflows [19].

Les compositions de services qui utilisent cette approche sont normalement basées sur les workflows manuels, aussi appelé statiques. Dans le workflow manuel, l'utilisateur doit définir l'ensemble des tâches et des dépendances parmi les données. Chaque tâche contient des requêtes qui permettent de chercher des services concrets pour la satisfaire, puis l'exécuter. Dans ce cas, seules la sélection et la liaison du service sont faites automatiquement.

Du coté des compositions dynamiques, le modèle du processus ainsi que la sélection du service sont faits automatiquement, une composition automatique demande des workflows capables de reconnaitre les services web correspondants à chaque tâche, mais aussi de trouver d'autres services au cas où ceux-ci soient indisponibles.

Parmi les travaux qui investissent dans ce type de techniques de composition on peut évoquer

Les travaux ci-dessous :

- ***Fabio Casati and Ming-Chien Shan*** [5], les auteurs redéfinissent la plate-forme de composition de services EFlow et proposent un prototype de langage de définition de services composés (Composite Service DéfinitionLanguage : CSDL). Une intéressante

fonctionnalité de CSDL est la distinction entre les services et les opérations des services. Les auteurs définissent donc les nœuds de services et des nœuds d'opérations, les premiers faisant référence aux services, les seconds s'adressant directement à une méthode particulière d'un service. Selon les auteurs, CSDL fournit les fonctionnalités adaptatives et dynamiques qui correspondent à l'évolution rapide des entreprises et des environnements informatiques et technologiques dans lesquels les services Web sont utilisés.

- **PolymorphicProcess Model (PPM) :** les auteurs dans [20] utilisent une méthode qui combine le modèle de plan statique avec le modèle de plan dynamique. Les auteurs décrivent la composition de services Web à l'aide d'activités. PPM comporte deux types d'activités, les activités génériques, proposées et exécutées par le moteur de PPM et les activités spécifiques à l'application qui doivent être définies par l'utilisateur. PPM choisit de séparer les implémentations de ces activités, réalisées par les services Web, et les interfaces de ces activités, qui sont une représentation desservices Web. Les interfaces des activités sont modélisées comme des machines à états, qui incluent des états et des opérations permettant la transition d'états, et sont également décrites par leurs entrées/ sorties. Les activités spécifiques à l'application sont des abstractions du comportement des services Web. La réalisation des activités par PPM est effectuée en liant les opérations des activités avec des opérations de services Web spécifiques.
- ***Laukkanen et Helin*** [5] identifient deux solutions possibles pour composer dynamiquementdes processus métier : remplacer un service Web dans un processus métier existant par un autre service ayant des fonctionnalités similaires, ou définirun nouveau workflowà partir des services Web disponibles. Les auteurs proposent. Services Web sémantique et composition de servicesd'utiliser les descriptions sémantiques des services Web pour pouvoir comparer lesfonctionnalités en utilisant les notions de pré conditions et d'effets de OWL-S. Lasolution proposée peut être résumée ainsi :
    1. Identifier les fonctionnalités requises
    2. Trouver les services adaptés grâce à leur description sémantique
    3. Créer ou modifier le workflow
    4. Exécuter le workflow et vérifier son exécution.

II.6.2. Composition orientée intelligence artificielle :

La composition dynamique de services Web par des techniques d'intelligence artificielle, et plus particulièrement par des techniques de planification, est l'une des voies qui semblent

prometteuses [21]. En effet, les concepts d'OWL-S sont très fortement inspirés de ceux de la planification, les prés-conditions présentent les conditions qui doivent être satisfaites pourgarantir la bonne exécution d'un service Web. Les effets sont le résultat du succès del'exécution d'un service.

Dans ce qui suit, nous présenterons quelques axes de recherche actuels concernant la composition par planification et par d'autres techniques d'intelligence artificielle :

- **Calcul situationnel (situation calculus) :*McIlraith et al.*** [22] proposent d'adapter et d'étendre le langage Golog pour laconstruction automatique de services Web. *Golog*est un langage de programmation logique qui sert à représenter des changements ou évolutions en termes de situations, d'actions et d'objets. Le problème de la composition de services Web est abordé de la façon suivante : la requête de l'utilisateur et les contraintes des services sont représentées en termes de prédicats du premier ordre dans le langage de calcul situationnel. Les services sont transformés en actions (atomiques ou complexes) dans le même langage.Puis, à l'aide de règles de déduction et de contraintes, des modèles sont ainsi générés etsont instanciés à l'exécution à partir des préférences utilisateur.
- **Theoremproving :** ***Waltinger*** [23] a élaboré une approche pour la composition de services par preuve de théorèmes. Cette approche est basée sur la déduction automatique et la synthèse de programmes. Les services disponibles et les requêtes utilisateur sont traduits dans un langage du premier ordre. Puis, des preuves sont produites à partir d'une preuve de théorèmes. La composition est obtenue à partir de preuves particulières. Dans le même axe, ***Rao et al*** [24] ont introduit une méthode de composition automatique de services Web sémantiques en utilisant des preuves de théorèmes de la logique linéaire.
- **Planification à Base de règles** : ***(Medjahed et al. 2003)*** [25], une technique de rule-based planning est utiliséepourengendrer des services composites à partir de descriptions déclaratives de hautniveau. Cette méthode utilise des règles de compossibilité pour déterminer dans quelle mesure deux services sont composables. L'approche proposée se déroule en quatrephases : une phase de spécification de haut niveau de la composition désirée en utilisantle langage CSSL (Composite Service Spécification Langage). La phase de correspondance utilise des règles de compossibilité pour générer des plans conformes aux spécifications du service demandeur. Dans la phase de sélection, si plus d'un plan est généré, la sélection est effectuée par rapport à des paramètres de qualité de la composition. Dans la phase de génération, une description détaillée du service composite est automatiquement générée et présentée au demandeur.

La principale contribution de cette approche est la notion de règles de compossibilité. Les règles de compossibilité considèrent les propriétés syntaxiques et sémantiques des services web. Les règles syntaxiques incluent des règles pour les types d'opérations possibles et pour les liaisons protocolaires entre les services (les bindings). Les règles sémantiques incluent des règles concernant la compatibilité des messages échangés, la compatibilité des domaines sémantiques des services, mais également des règles de qualités de la composition [23]

- **Approche par les fonctions :**Afin de permettre une composition dynamique, Les auteurs basés surl'approchefonctionnelleontproposé un système de programmationfonctionnelle (FS4WSC) qui permet de composer les services Webavec l'étude de ladescription sémantique de ces services, où ils ont défini un formalisme quipermet de raisonner sur des structures de données, garantir la sémantique de larequête et des services Web et est muni des constructeurs pour composer lesservices. L'objectif recherché à travers l'utilisation de la sémantique est de permettreaux machines d'interpréter les données traitées et de saisir leurs significations demanière automatique.[26].

II.6.3. Approches basée sur le web sémantique :

Avant de passer aux approches on doit définir les services web sémantiques et le langage OWL-S :

II.6.3.1. Les services Web sémantiques :

L'idée consiste à faire converger le Web sémantique et les services Web pour pouvoir proposer des services et des procédés qui prennent en compte la connaissance que l'on peut avoir d'un service ou d'un procédé afin de pouvoir profiter des travauxsur le raisonnement à partir de connaissances qui ont été mis en avant par le Websémantique. Les premiers travaux sur les services web sémantiques proposent une approche basée sur l'utilisation de langages de descriptions de la famillede DAML. Cette approche permet de capturer les données et métadonnées associées à un service. Ces données spécifient les propriétés et capacités du service ainsi que lesprés requis et les conséquences de son exécution.Le langage de description de servicesWeb sémantiques de référence est le langageOWL-S, une extension du langage OWL, proposé par le groupe de recherche sur leWeb sémantique de la DARPAà l'origine du langage DAML.

II.6.3.2. Le langage OWL-S :

Appelé DAML-S dans ses premières versions, le langage OWL-S basé sur DAML+OIL a pour objectif d'ajouter des descriptions sémantiques aux servicesWeb (en plus de leur description syntaxique WSDL). Alors que les efforts industrielssont actuellement focalisés sur la standardisation des mécanismes d'enregistrementet de découverte de services, sur l'interopérabilité des services indépendamment de laplateforme, ainsi que l'échange de types de documents syntaxiquement bien formésentre services Web.

OWL-S a pour objectif de fournir une plus grande expressivitéenpermettant la description des caractéristiques des services afin de pouvoir raisonnerdessus dans le but de découvrir, invoquer, composer et gérer les services web de façonla plus automatisée possible.

Un service décrit avec OWL-S est une instance de l'ontologie OWL-S illustré dans la figure suivante : [18]

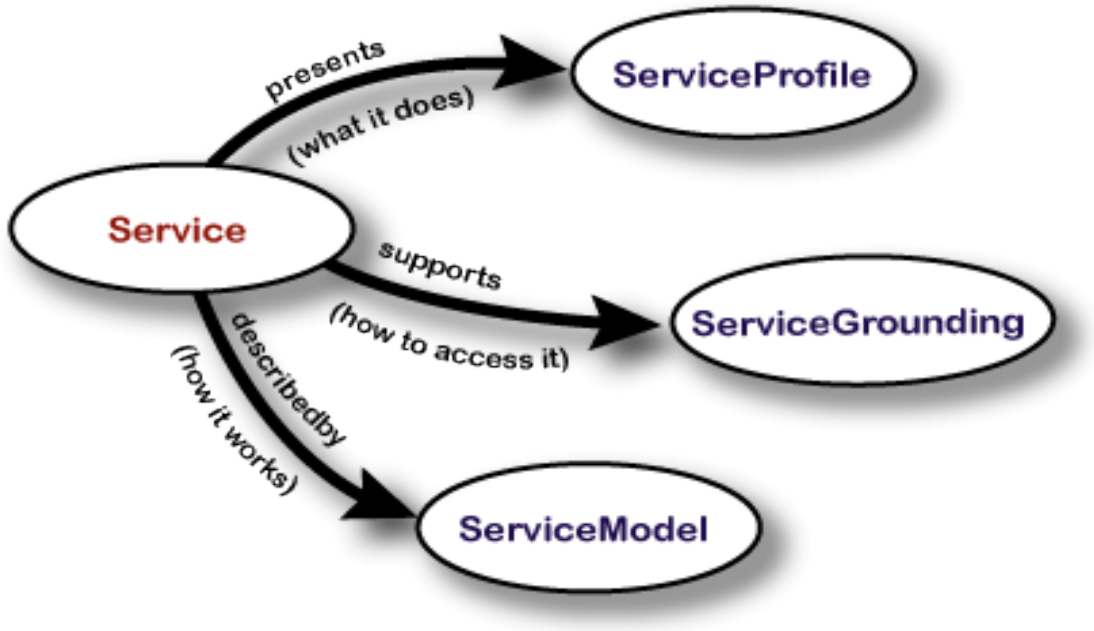


Figure 13 : schéma représente OWL-S.

- Cette approche basée sur le Framwork qui assure le suivi de toutes les demandes de composition de service par chaque utilisateur. Il stocke le journal de toutes les demandes de services composites uniques pour l'utilisateur spécifique sous User- service log. Services énumérés sous CompoServRegistry sont triés à l'aide User-service log. services composites fréquemment utilisés sont indiqués en haut de la liste qui fait l'interface utilisateurs plus convivial.

Les Auteur de l'article [27] ont utilisés le OWL-S pour la composition des services web, Le système est construit en Java en utilisant l'éditeur Eclipse et exécuté sur GlassFishServer, Les descriptions OWL-S de services Web sont construits en utilisant Protégé ontologie Editor. Pour Fusionner les ontologies ils ont utilisés PROMPT Plug-In de Protégé.

- Dans cette approche, les auteurs ont proposés des services e-learning flux de travail (workflow) composant l'architecture et les algorithmes pertinents pour l'appariement et la comparaison des flux de e-learning pour les apprenants ayant différents styles d'apprentissage. Ils ont proposé une logique basée appariement (matching) hybride et des algorithmes composant qui utilise OWL-S profil et le processus d'ontologies pour la composition du flux de travail dynamique de services Web e-learning. [28]
- **[**Abdaladhem A, Jacques P]ont proposés une approche qui utilise le domaine d'informations et sémantiques pour définir un environnement spécifique , leur principal objectif est d'offrir aux utilisateurs non-experts qui ont tendance de ne pas savoir à l'avance comment parvenir leurs objectifs,ils ont utilisés les descriptions et les ontologies sémantiques à définir un mécanisme assistant pour obtenir des solutions optimisées pour composer des services Web semi-automatique où la composition est progressivement générée par l'utilisation déclaratif orientée générique du Plan de la composition. Ils ont proposés à base sémantique Framework qui offre une flexibilité pour intégrer les services et renforce la collaboration d'humain-ordinateur paradigme .Pour atteindre leur objectif, ils ont proposés aux utilisateurs des modèles de processus génériques (template), Ces modèles sont orientés vers les utilisateurs des objectifs. [29]

II.7. Conclusion

La composition des services web est un point crucial qui a un grand impact sur plusieurs domaines de recherches. De nombreux efforts ont été fournis afin de permettre une composition utilisable et acceptable de services Web. L'objectif de la composition de services est de répondre, justement, aux exigences des clients en offrant des fonctionnalités à valeur ajoutée par articulations de fonctionnalités déjà offertes par d'autres services Web. La composition de services implique la capacité de sélectionner, d'articuler et de faire inter opérer des services existants, pour réaliser cette activité. Contrairement aux Business Process «traditionnels» qui sont exécutés d'une manière prévisible et répétitive dans un environnement statique, les services Web composés s'exécutent dans un environnement ouvert et versatile, où les services offerts sont diversifiés et évolutifs et où la forte compétition engendrée par la multitude de fournisseurs oblige les entreprises à mieux adapter leurs services pour répondre aux besoins croissants des clients et à moindre coût. Cependant, pour permettre une telle composition, des abstractions, des outils et des environnements sont nécessaires. Ils permettent de sélectionner les services impliqués dans la composition, de spécifier l'ordre des opérations à invoquer, de gérer et de contrôler le flux de données à travers les services à composer, ainsi que la gestion des transactions et des situations d'exception.

Après avoir présentéla première partie ***« l'état de l'art »***qui représente les différents chapitres des services web et la composition des services, nous présentons par la suite une deuxième partie « conception et réalisation ». Dans cette dernière nous traitons la conception de notre système ainsi que l'implémentation de la mise en œuvre

ChapitreIII

Conception du système

III. Conception du Système

III.1. Introduction

La composition des services web a pour objectif de déterminer une combinaison de services en fonction d'une requête d'un client. Du côté du client cette composition semblera un unique service. La composition sera transparente au client même si cette composition sera la combinaison de plusieurs services web. Alors dans ce chapitre nous allons décrire les différentes étapes de la conception de notre système de composition de services web. Tout d'abord on définit les services web et les ontologies utilisées et L'OWL-S, ensuite nous présenterons l'architecture du système proposé avec ses différentes tâches (découverte, sélection, composition et exécution).

III.2. Objectif

L'objectif de notre travail est de concevoir un prototype d'un système de composition des services web.Ce prototype assurera plusieurs tâches : la découverte des services, la sélection, la composition et l'exécution. Pour atteindre cet objectif, après une étude comparative des différentes approches de composition des services web, nous avons opté pour l'approche sémantique.Cette approche permet de rendre l'information plus accessible aux machines. En d'autres termes, elle permet de décrire les comportements et les données manipulés par les web services, elle permet aussi de répondre aux limites de WSDL par l'ajout d'une couche sémantique.

III.3. Description des services web utilisés

Pour assurer la fiabilité de notre systèmenous avions besoin de créer des services web et leurs descriptions sémantiques afin de faire destests durant la réalisation de chaque tâche du système nous détaillons dans ce qui suit les exemples que nous avons utilisés.

- **Service « RechercheLivres »**

Le service RechercheLivresfait la recherche des livres, il suffit d'entrer un titre pour avoir tous les détails du livre recherché (Prix, Auteur, Date_pub, Description, Sujet),

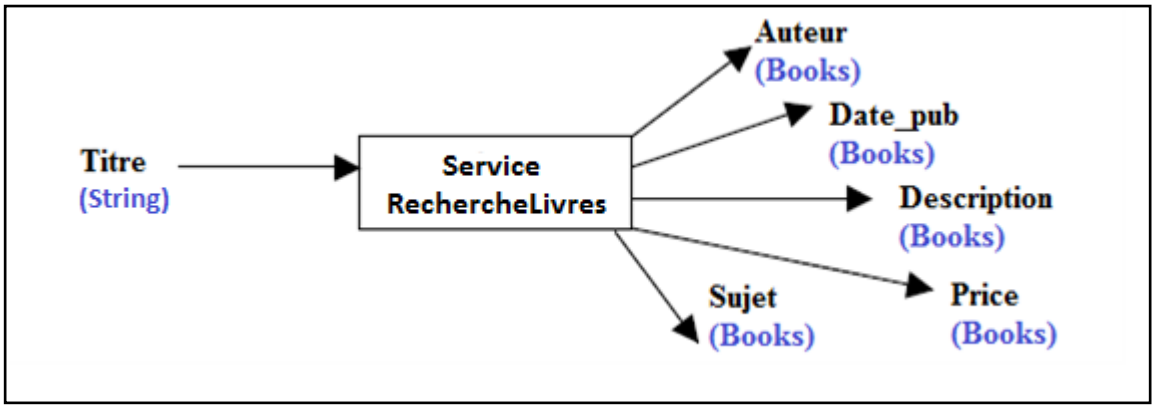


Figure 14 : Service RechercheLivres

Le service recherche à partir d'une Ontologie où tous les détails des livres sont instancie comme montre la figure suivante :

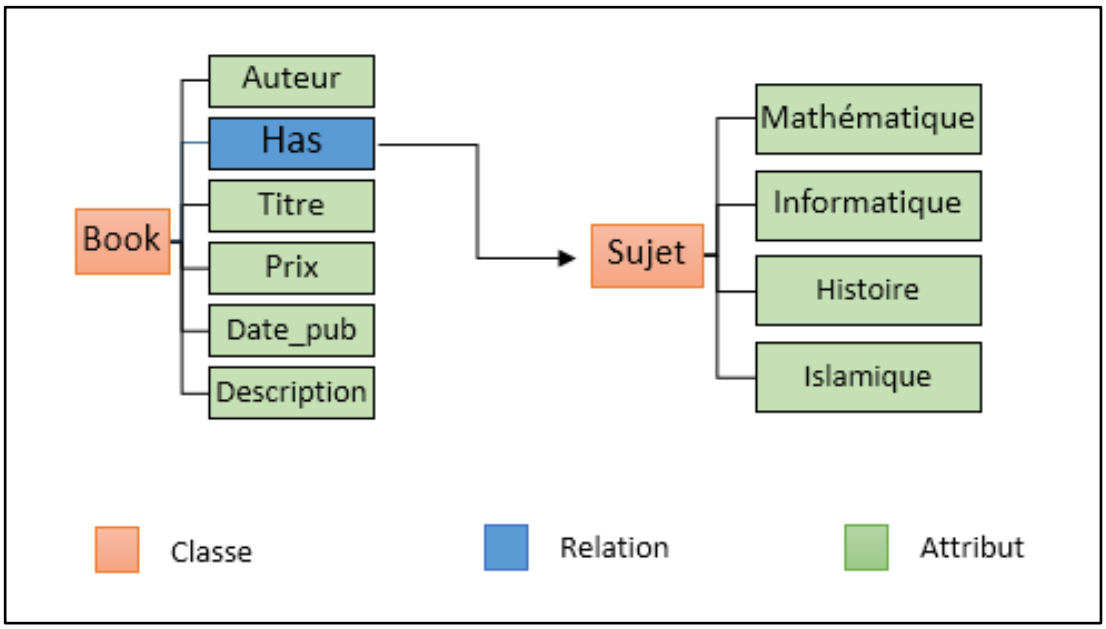


Figure 15 : la hiérarchie de l'ontologie RechercheLivres

Un service décrit avec OWL-S est une instance de l'ontologie OWL-S. Nous créons donc des instances des classes Service, Profile, Process et Grounding.(Voirla Figure 16)

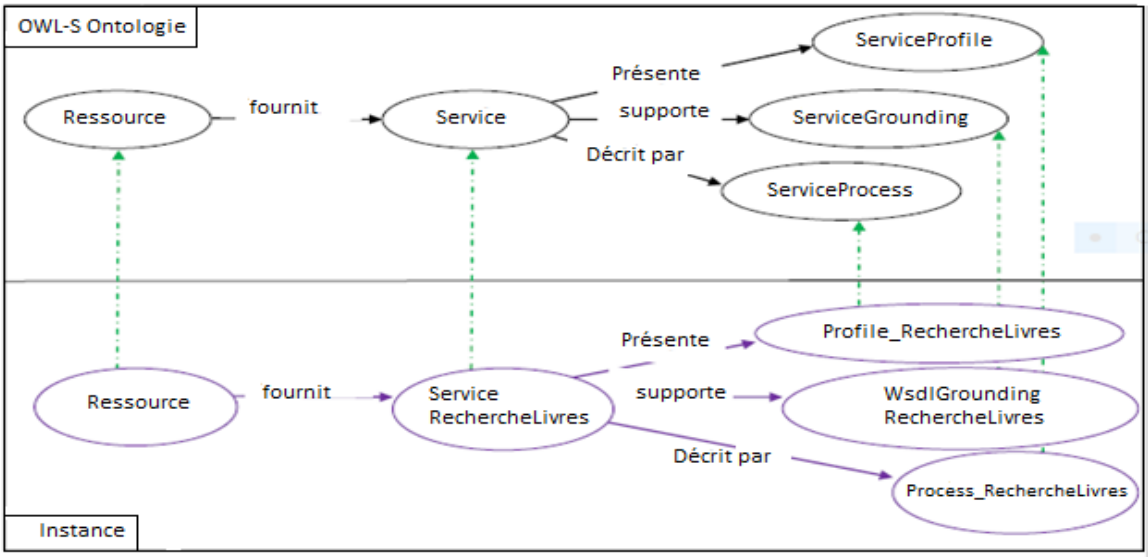


Figure 16: Exempled'instances OWL-S pour le service RechercheLivres

- **Service hôtel**

Ce service cherche tous les hôtels qui se trouve dans une ville donnée, il a comme entrée la ville et le nombre de chambre et la date et comme sortie tous les détails sur les hôtels de cette ville

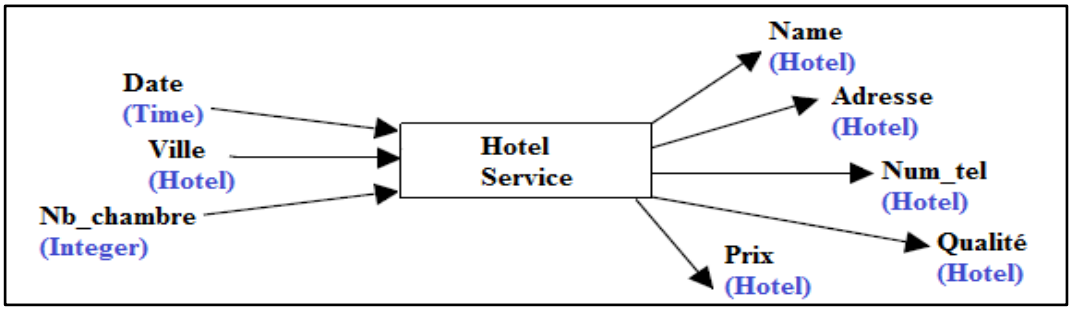


Figure 17 : Service Hôtel

L'hiérarchique de l'ontologie Hôtel est présenté dans la figure suivante :

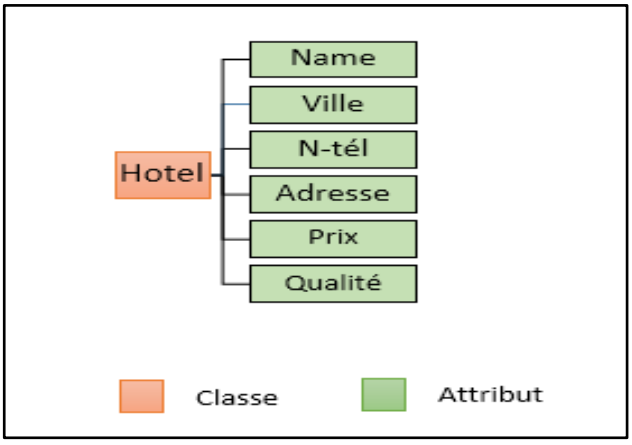


Figure 18 : la hiérarchie de l'ontologie Hôtel

- **Service de paiement**

Ce service permet de faire le paiement en ligne, il suffit d'introduire (Nom, Prénom, Num_Carte_Crédit, Prix), et nous aurons comme résultats une confirmation du paiement.

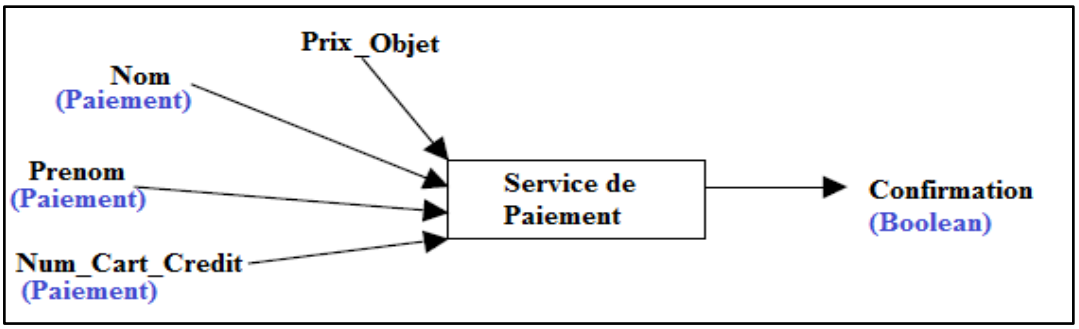


Figure 19 : Service Paiement

La figure suivante présente l'hiérarchique de l'ontologie du Paiement :

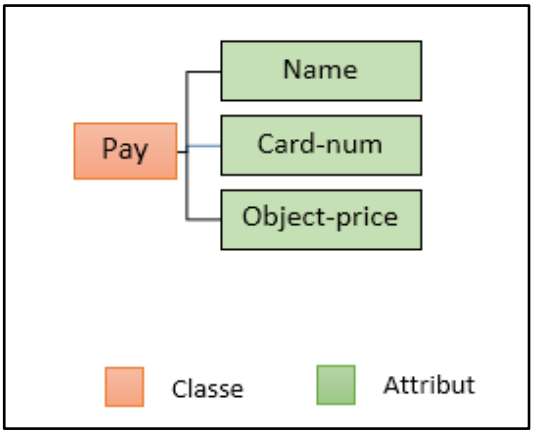


Figure 20 : la hiérarchie de l'ontologie Paiement

- **Service vol**

Dans ce service l'utilisateur peut faire une réservation en introduisant (ville_depart,ville _d'arrivé, date et heure) pour avoir en sortie tous les détails du vol (prix, nombre de passages), la figure suivante représente le service vol :

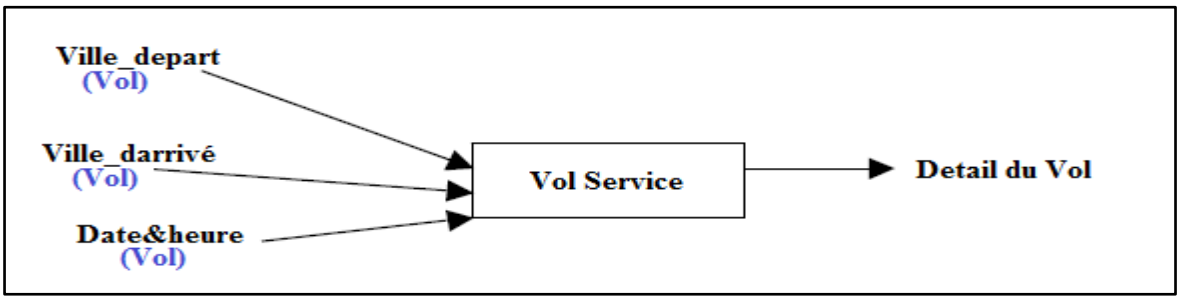


Figure 21 : Service Vol

L'hiérarchique de l'ontologie Vol est présenté dans la figure suivante :

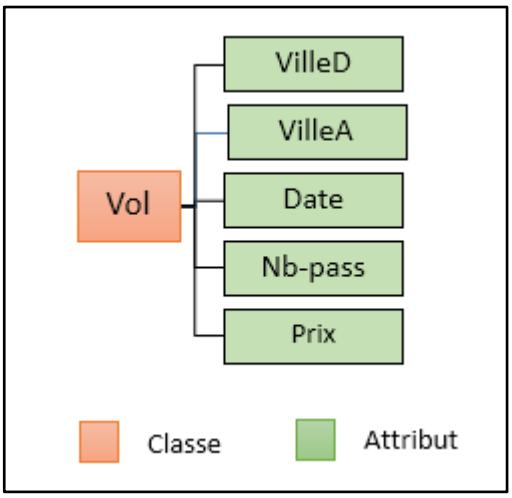


Figure 22 : la hiérarchie de l'ontologie Vol

- **Service location de voiture**

Le service de location permet de faire la recherche des différentes agence de location qui existent dans une ville donnée donc la ville est l'entrée et on aura comme sortie (Nom _agence, Adresse, Num_tel, voitures_dispo, leurs prix).

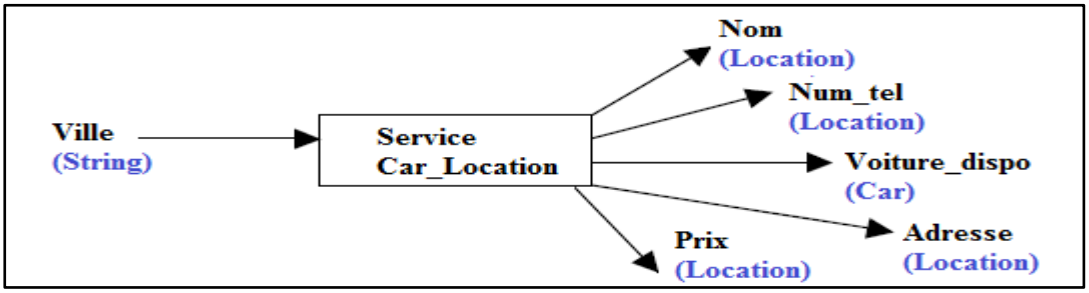


Figure 23 : Service Car-location

La figure suivante présente l'hiérarchique de l'ontologie location de voiture :

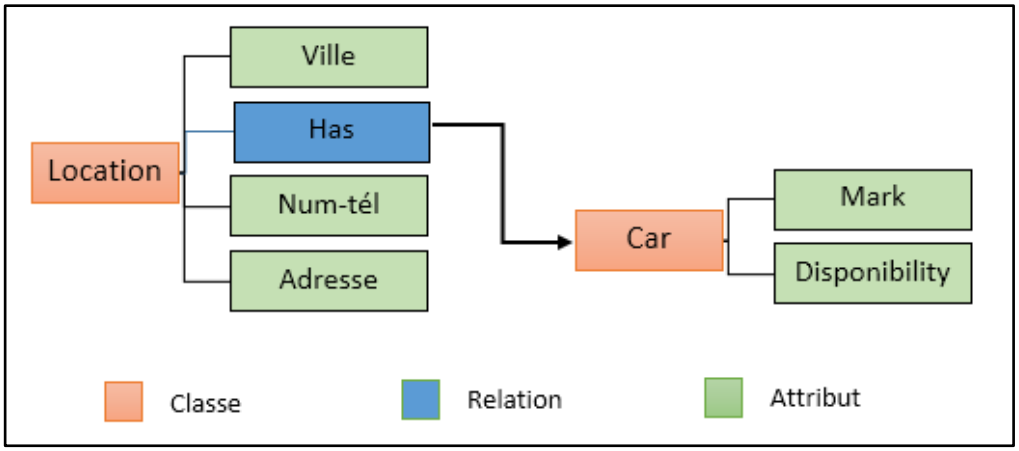


Figure 24 : la hiérarchie de l'ontologie Car-location

III.4. Présentation du système

La composition s'est avéré être essentiellement un processus qui prenait beaucoup de temps. En effet, une composition n'est pas simplement un regroupement quelconque de services web, mais un ensemble dont les tâches sont ordonnées en fonction des relations reliant ses services web. Donc notre système est constitué de quatreprincipales tâches : la tâchedécouverte, la tâche de sélection, la tâche de composition, la tâche d'exécution. La collaboration de ces tâches permet d'atteindre l'objectif global de notre système qui est la composition des services web et leursinvocations.

La figure suivante illustre une vue globale du système et les interactions entre les différentes tâches

III.4.1. Architecture du système

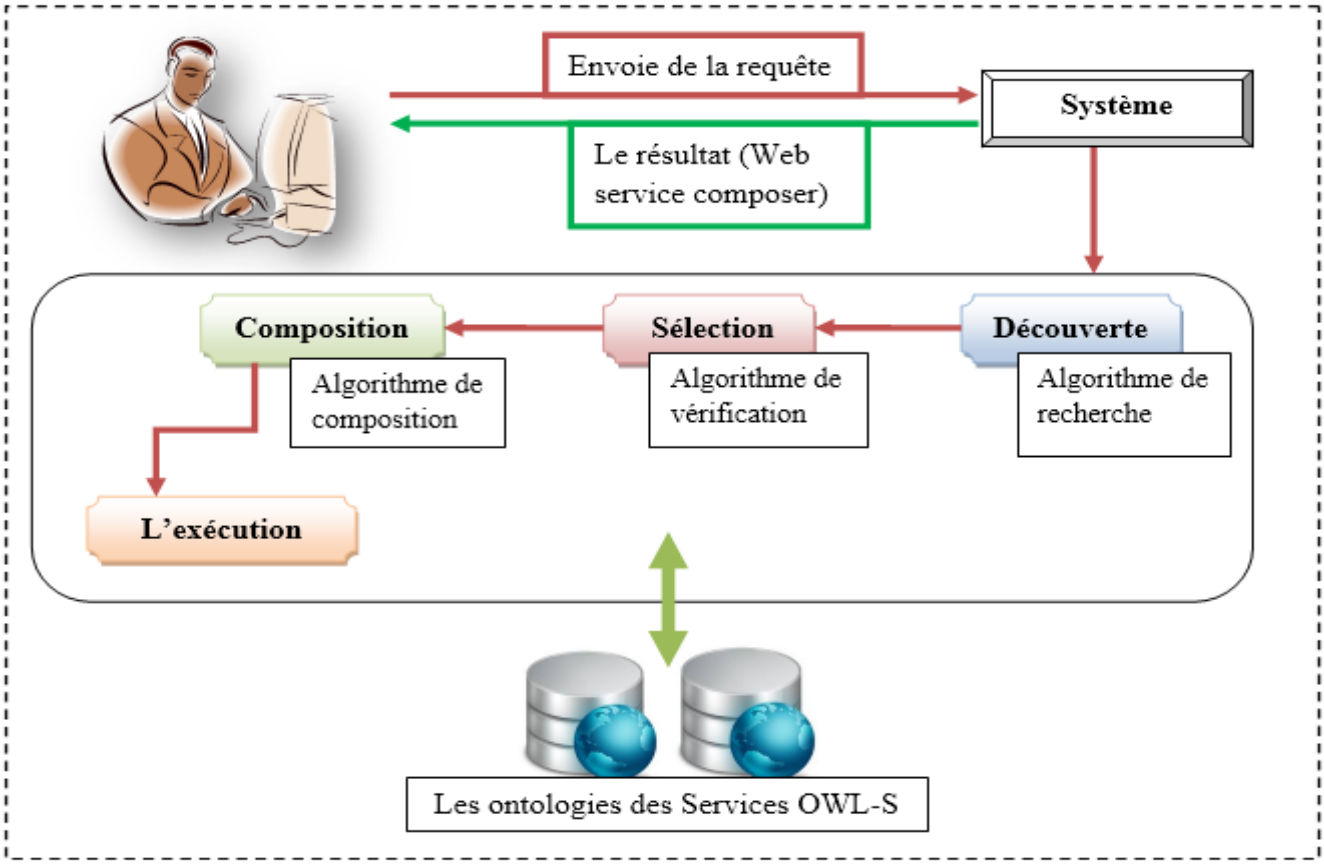


Figure 25 : Architecture générale du système

III.4.2. Diagramme de cas d'utilisation :

Le diagramme de cas d'utilisations de la figure ci-dessous permet de représenter les différentes interactions.

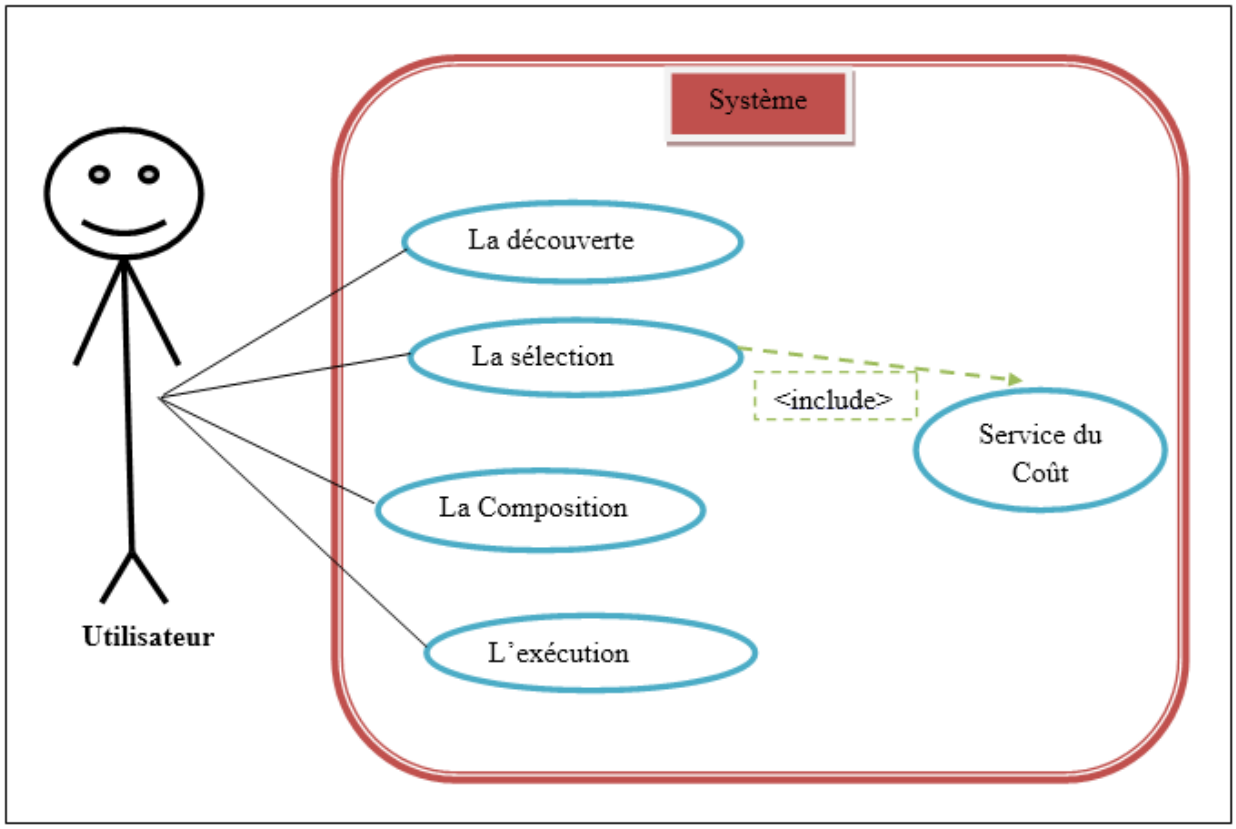


Figure 26 : Diagramme de cas d'utilisation

Le système proposera à l'utilisateur quatre fonctionnalités principales :

- La découverte des services web.
- La sélection des services web.
- La composition automatique des services web.
- L'exécution des services web composés.

Les utilisateurs qui représentent le rôle humain dans notre système, interagissent avec ce dernier via une interface web afin d'exécuter les fonctionnalités offertes par ce système.

III.4.3. Découverte :

III.4.3.1. Description:

Cette tâche assure la recherche des services qui répondent le plus, à la requête spécifiée par l'utilisateur, qui est basée sur les entrées et sorties des services publiés dans l'ontologie OWL-S, avec celles définies dans la requête.Pour optimiser la recherche, la tâche découverte interagit avec les requêtes afin d'adapter les résultats de la découverte aux attente des utilisateurs.

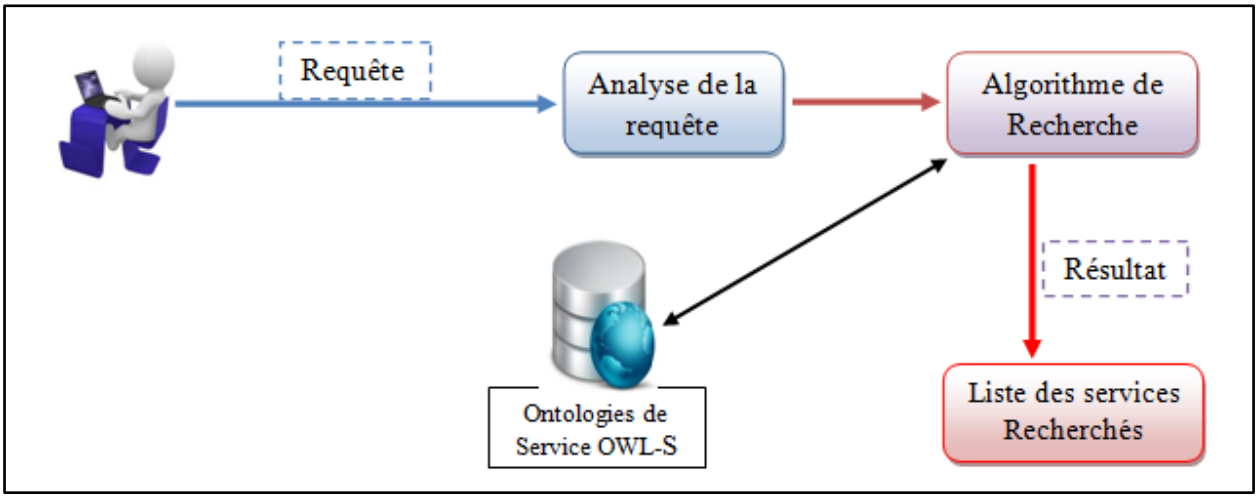


Figure 27 : Tâche Découverte

III.4.3.2. Fonctionnement

Cette tâche assure la fonctionnalité de recherche indépendamment des autres tâches. Elle reçoit en entrée une requête qui décrit les critères de recherches à l'aide de l'algorithme de recherche, et retourne la liste des services adéquats.

- **Algorithme de recherche :**

```
ProcedureRecherche (String requête, input userInput, output userOutput)
Début
    Pour chaque(ServiceS de ListeService)Faire
        Listechild := S.getchildren() ;
        Pourchaque(child de Listechild)Faire
            Si (child.Name = 'ServiceName') Faire
                Sname := child.getvalue() ;
            FSi
            Si (child.Name = 'HasInput') Faire
                SIn := child.getvalue() ;
            FSi
            Si (child.Name = 'HasOutput') Faire
                SOut := child.getvalue() ;
            FSi
        FPour
        Si ((SnameContient requête) ET (Sin = userInput) ET (SOut = userOutput)) Faire
            AfficherServiceS ;
        FSi
    FPour
Fin.
```

III.4.4. Sélection

III.4.4.1. Description

Après la découverte des services qui existent dans notre ontologie, l'utilisateur peut sélectionner les services pour faire une compositionentre eux .la figure ci-dessous présente l'architecture de cette tâche.

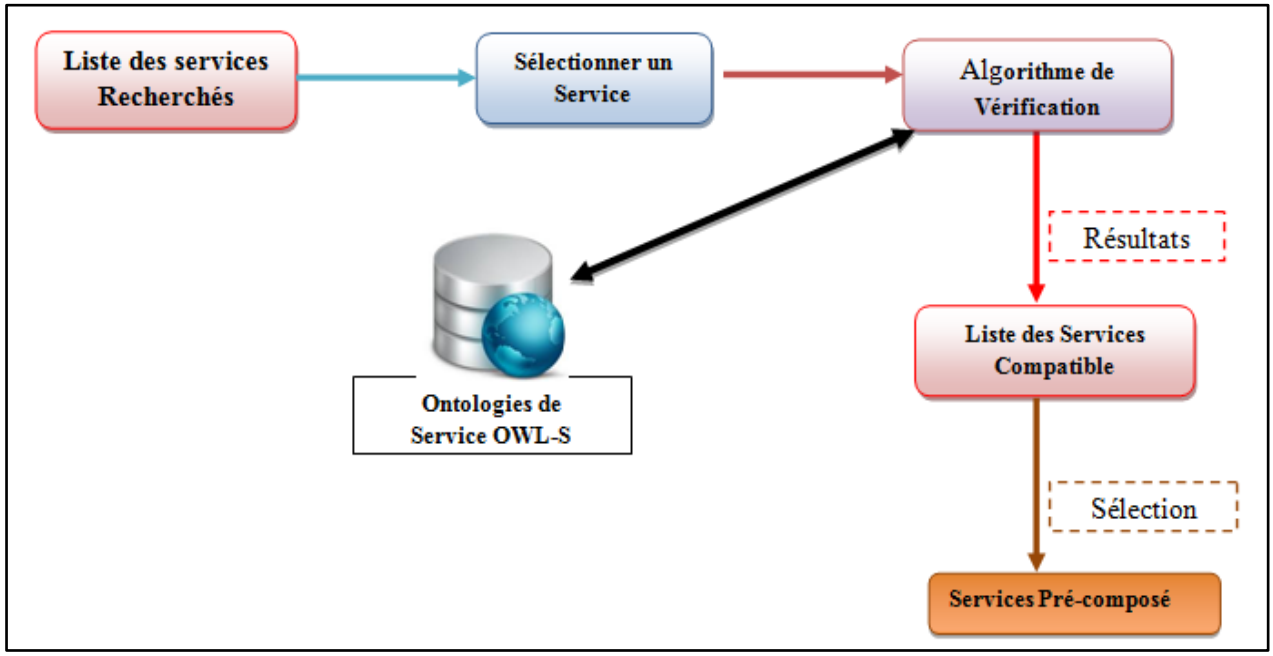


Figure 27 : Tâche Sélection

III.4.4.2. Fonctionnement

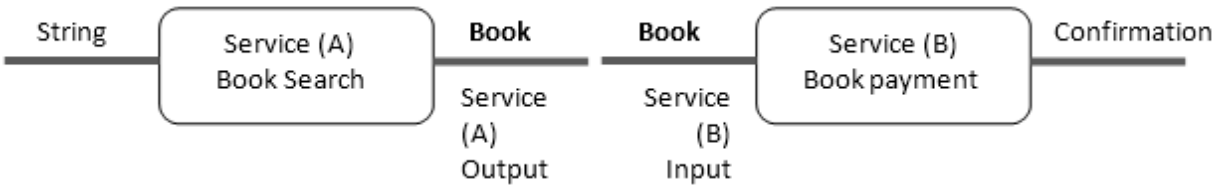
Cette tâche permet de sélectionner des services pour faire la composition entre eux. Donc on a développél'algorithme de vérification qui exploite les descriptions OWL-S pour avoir des services compatible et avoircomme résultat des services pré-composé.

- **Algorithme de vérification**

Un ensemble de services Web est dit compatible si les services peuvent collaborer « Correctement », La collaboration se réalise par l'échange d'un ensemble de messages

Les blocages qui peuvent surgir sont principalement dus à l'hétérogénéité desservices au niveau des types de données.

Pour garantir une forte composition la sortie du premier service (Output) doit être de même type que l'entrée du deuxième (Input) ce principe est illustré dans la figure suivante :



Cet algorithme est basé sur la comparaison (matching) des types des « entrées/sorties » des services afin de déterminer s'ils sont composables ou pas.

```
Procedurevérification (Service S, ListeService)
  Entier Cpt ;
Début
        Liste output := S.getOutputTypes ;
        Pour chaque (Service de ListeService) Faire
        Liste input := Service.getInputTypes ;
                Cpt := 0 ;
                Pourchaque (input de Listinput) Faire
                        Si (input Existe inListeoutput)Faire
                                Cpt ++ ;
                        FSi
                FPour
                Si (Cpt = Listoutput_Length) Faire
                 Afficher Service ;
                FSi
        FPour
Fin.
```

L'algorithme ci-dessus nous permet de déterminer la compatibilité des services web afin d'offrir aux utilisateurs toutes les compositions possibles, et pour faciliter la sélection à l'utilisateuron a introduit la notion du Coût pour déterminer la composition optimale.

III.4.4.3. Coût :

C'est une moyenne basée sur le tarif et le temps d'exécution des services pour donner a l'utilisateur un avis sur une composition

On calcule le coût suivant cette méthode :

On calcule d'abord le temps d'exécution (T_ex) de tous les services d'une composition

$$T\_ex_{(c)} = \sum_{s=1}^{n} T\_ex_{(s(i))}$$

(C) : Composition
(S) : Service

Ensuite on calcule le Tarif (Tr) de tous les services d'une composition :

$$Tr_{(c)} = \sum_{s=1}^{n} Tr_{(s(i))}$$

Puis on calcule le Tarif globale (Tr_g) et le temps d'exécution globale (T_ex_g)de toutes les compositions possibles :

$$T\_ex\_g = \sum_{s=1}^{n} T\_ex_{(c(i))}$$

$$Tr\_g = \sum_{s=1}^{n} Tr_{(c(i))}$$

Et pour finir on divise le tarif de chaque composition sur le Tarif globale, et le temps d'exécution de chaque composition sur le temps d'exécution globale pour avoir la moyenne de chaque composition par rapport aux autres compositions possibles

$$M.T\_ex_{(c(i))} = \frac{T\_ex_{(c(i))}}{T\_ex\_g}$$

$$M.Tr_{(c(i))} = \frac{Tr_{(c(i))}}{Tr\_g}$$

M : moyenne

Le Coût est la somme des deux moyennes obtenues dans le calcul précédent et puisqu'on à utiliser le temps d'exécution et le tarif dans le calcul, La Composition optimale correspond au Coût le plus bas dans la liste des composition possibles.

III.4.5. Composition:

III.4.5.1. Description :

La composition est la fonctionnalité majeur de notre système, elle consiste à composer les services présélectionnés dans le module précédent (sélection) afin d'obtenir un service composite optimisé et exécutable qui répond aux attentes de l'utilisateur.

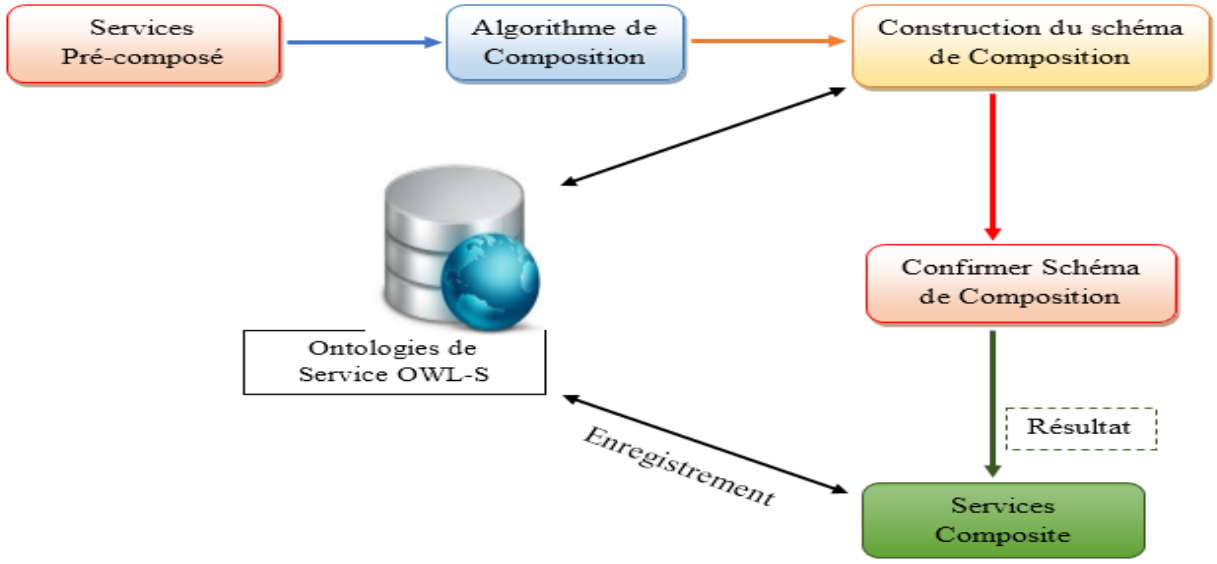


Figure 28 : Tâche Composition

III.4.5.2. Fonctionnement

Le développement d'un algorithme de composition permet de nous faire la composition entre les services web. Sa principale tâche est la construction d'un schéma de composition qui répond le plus aux besoins spécifiés par l'utilisateur, il prend en entrée la requête d'utilisateur, et génère si c'est possible un service composite en sortie.

III.4.6. Exécution:

III.4.6.1. Description :

Cette fonctionnalité permettra à l'utilisateur d'exécuter ou d'invoquer un service composée résultant de l'opération précédente (Composition).

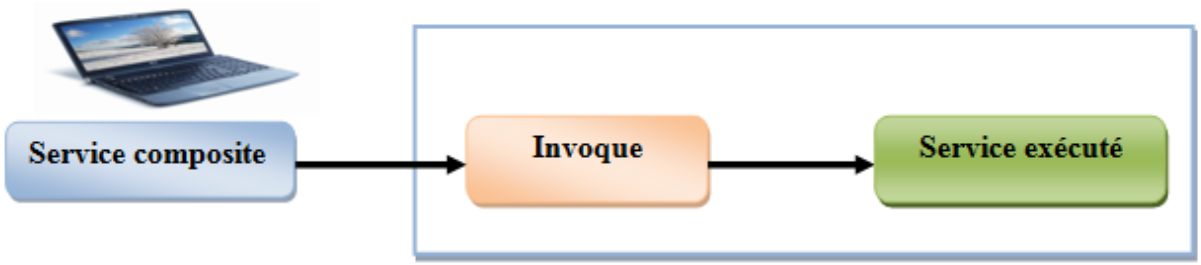


Figure 29 : Tâche exécution

III.5. Diagramme de séquence

La figure ci-dessousreprésente le diagramme de séquence de toutes les tâches de notre système

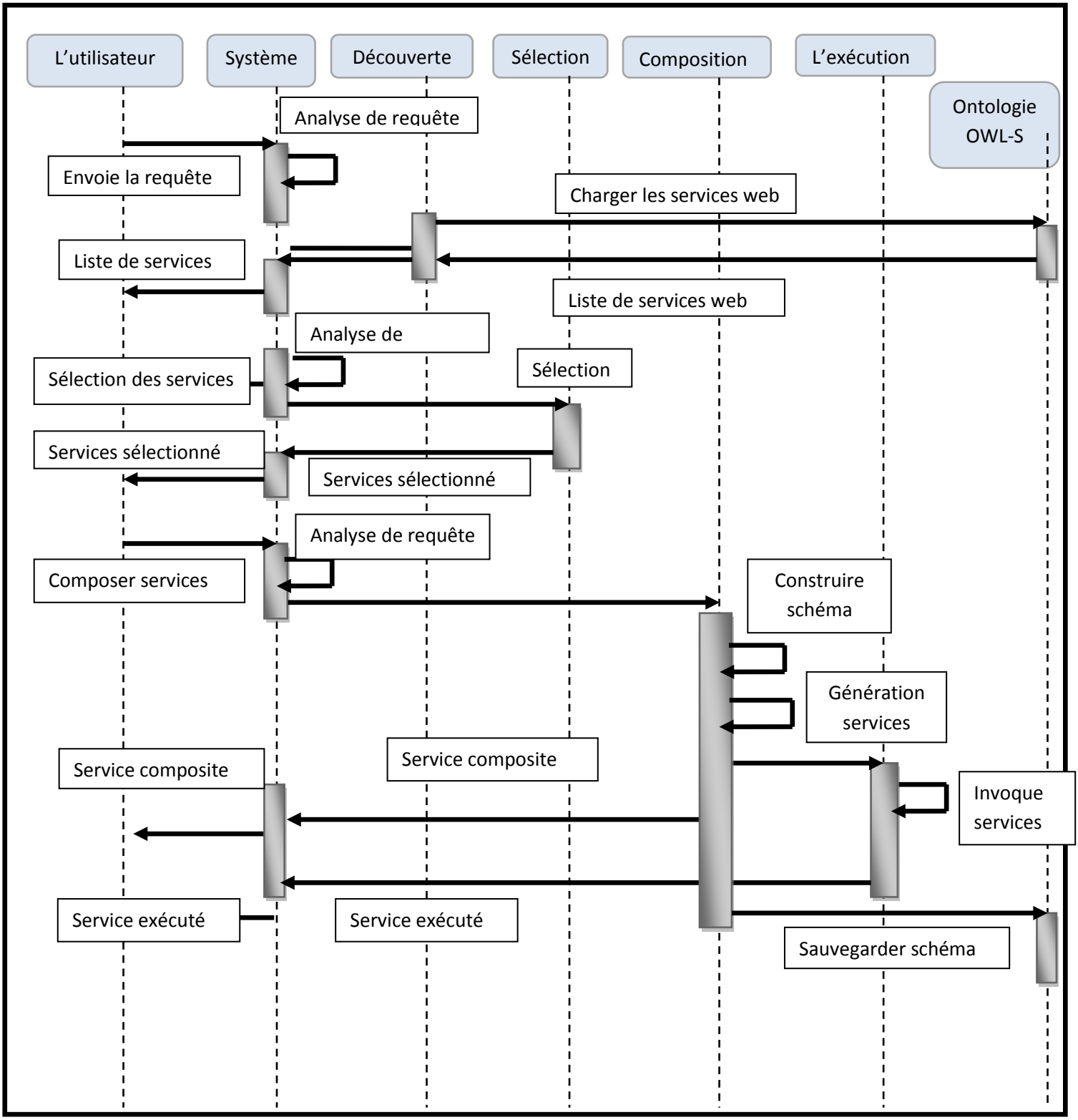


Figure 31 : Diagramme de séquence de toutes les tâches.

III.6. Conclusion

Après avoir présenté dans ce chapitre les différentes tâches qui composent notre système, à savoir le module de composition, de découverte et d'exécution, nous nous proposons de concrétiser notre système, en réalisant les différentes ontologies et les différentes fonctionnalités (découverte, sélection, composition et l'exécution).

Dans le chapitre qui suit, nous traiterons de l'implémentation et la mise en œuvre de notre système ainsi que des technologies et outils utilisés.

ChapitreIV

Mise en œuvre du système

IV. Mise en œuvre du système

IV.1. Introduction

Après la présentation de l'architecture de notre système dans le chapitre précédent, nous allons dans ce qui suit, décrire les différents aspects techniques liés à l'implémentation et le déploiement de notre système.

Pour mener à bien notre travail, nous avons fait appel à une panoplie d'outils et de langages de développement permettant la réalisation et la mise en œuvre de notre application.

Nous tâcherons ensuite de décrire les différentes tâches composants notre application et les différentes ontologies utilisées, ainsi que la manière avec laquelle s'effectue son déploiement au sein du serveur d'application.

Enfin, nous étalerons les différentes fonctionnalités offertes par notre système. Cela sera illustré par des prises d'écrans permettant d'expliquer au mieux le fonctionnement de notre application.

IV.2. Environnements de développement

Protégé 3.2.1

Protégé est un système auteur pour la création d'ontologies. Il a été créé à l'université Stanford et est très populaire dans le domaine du Web sémantique et au niveau de la recherche en informatique.Protégé est développé en Java. Il est gratuit et son code source est publié sous une licence libre (la *Mozilla Public License*).Protégé peut lire et sauvegarder des ontologies dans la plupart des formats d'ontologies : RDF, RDFS, OWL, OWL-S … .
Il possède plusieurs concurrents tels que Hozo, Onto Edit et Swoop. Il est reconnu pour sa capacité à travailler sur des ontologies de grandes dimensions.

Langage JAVA

Notre choix du langage de programmation s'est porté sur le langage JAVA et cela pour diverses raisons :

- JAVA est un langage orienté objet simple ce qui réduit les risques d'incohérence.
- Il est portable. Il peut être utilisé sous Windows, sur Macintosh et sur d'autres plates-formes sans aucune modification. C'est donc un langage multiplateforme, ce qui permet aux développeurs d'écrire un code qu'ils peuvent exécuter dans tous les environnements.

- Il possède une riche bibliothèque de classes comprenant des fonctions diverses telles que les fonctions standards, le système de gestion de fichiers, les fonctions multimédia et beaucoup d'autres fonctionnalités.
- Il existe une API (Interface de programmation d'applications) JAVA fournie avec l'éditeur d'ontologies Protégé ce qui permet d'accéder à l'ontologie à partir de notre application.

Glassfish4.1

GlassFish est un serveur d'applications Open Source Java EE 5 et désormais Java EE 7 avec la version 4.1 qui sert de socle au produit Oracle GlassFish Server2 (anciennement Sun Java System Application Server3 de Sun Microsystems). Sa partie Toplink persistence4provient d'Oracle. C'est la réponse aux développeurs Java désireux d'accéder aux sources et de contribuer au développement des serveurs d'applications de nouvelle génération.

JavaFX

JavaFX est une technologie créée par Sun Microsystems qui appartient désormais à Oracle, à la suite du rachat de Sun Microsystems par Oracle le 20 avril 2009.
Avec l'apparition de Java 8 en mars 2014, JavaFX devient l'outil de création d'interface graphique ('GUI toolkit') officiel du langage Java, pour toutes les sortes d'application (applications mobiles, applications sur poste de travail, applications Web...), le développement de son prédécesseur Swing étant abandonné (sauf pour les corrections de bogues).

Notre choix s'est porté sur le JavaFX pour diverses raisons :

- JavaFX contient des outils très divers, notamment pour les médias audio et vidéo, le graphisme 2D et 3D, la programmation Web, la programmation multi-fils etc.
- Le SDK de JavaFX étant désormais intégré au JDK standard Java SE, il n'y a pas besoin de réaliser d'installation spécifique pour JavaFX.
- Des projets libres complètent JavaFX en fournissant des composants de haute qualité absents de JavaFX proprement dit

L'API OWL-S

OWL-S API fournit une API Java pour lire, créer et exécuter des descriptions de services OWL-S (anciennement connu sous le nom DAML-S). L'API prend en charge les différentes versions du langage OWL-S. L'API offre un moteur d'exécution qui peut invoquer des AtomicProcesses qui ont des WSDL et les CompositeProcesses qui utilisent des séquences de contrôle, des constructions non-ordonnées. L'exécution des processus qui s'appuie sur les

conditionnels tels qu'If-then-else et RepeatUntil, n'est pas prise en charge dans l'implémentation par défaut. Mais, cette application peut être étendue pour traiter ces constructions si l'application qui utilise les descriptions OWL-S a une syntaxe personnalisée et une procédure d'évaluation pour les conditions.

IV.3. Implémentation du système :

Cette partie décrit les détails d'implémentation de notre prototype. Nous commencerons d'abord par la présentation de l'architecture de déploiement du système, puis on effectuera quelques tests pour enfin avoir nos résultats.

IV.3.1. Déploiement du système :

IV.3.1.1. *Organisation du projet de l'application*

Notre système est développé sur Eclipse, ce dernier impose une structure bien définie pour l'organisation de l'application comme le montre la figure suivante :

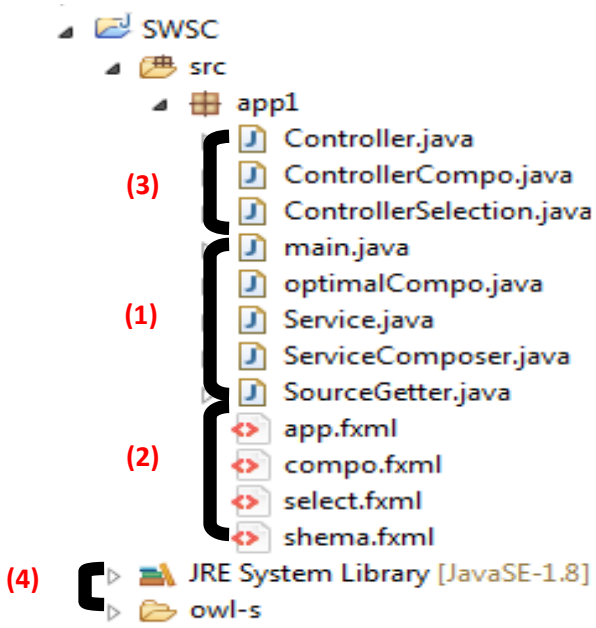


Figure 32 : Organisation de l'application.

Le système est structuré en quatre parties :

1. Les classes des tâches du système
2. Les interfaces décrites en JavaFX
3. Les contrôleurs qui relient les différentes tâches avec l'interface.
4. Le répertoire des ontologies des services (les descriptions des services web)

IV.3.1.2. *Description des classes des tâches du système :*

SourceGetter :

C'est la classe chargée d'extraire les informations à partir des descriptions des services webqui se trouvent dans le répertoire « owls » (4) figure (31) pour les utilisés dans les tâches du système (découverte, sélection).

Les méthodes principales de la classe **« SourceGetter »** sont :

- **allService() :**c'est la méthode qui charge toutes les descriptions des services à partir du répertoire « owls » .
- **CompareONOWL() :** cette méthode retourne une liste des services selon la requête entré par l'utilisateur elle s'occupe donc de la tâche découverte.
- **equivalentServices() :** cette méthode s'occupe de retourner les services compatibles à composer avec un service en entrée (tâche sélection)

optimalCompo :

Cette classe calcule le Coût des compositions obtenues par la méthode « equivalentServices » pour retourner la composition optimale et les compositions à éviter.

ServiceComposer :

Le rôle de cette classe est de retourner un service composé à partir des services obtenu par la tâche de sélection, elle s'occupe donc de la tâche composition.

IV.3.2. Architecture de déploiement du système :

La figure suivante représente le déploiement de notre système ainsi que les interactions entre les différentsEnvironnements de développement utilisés dans la réalisation du système :

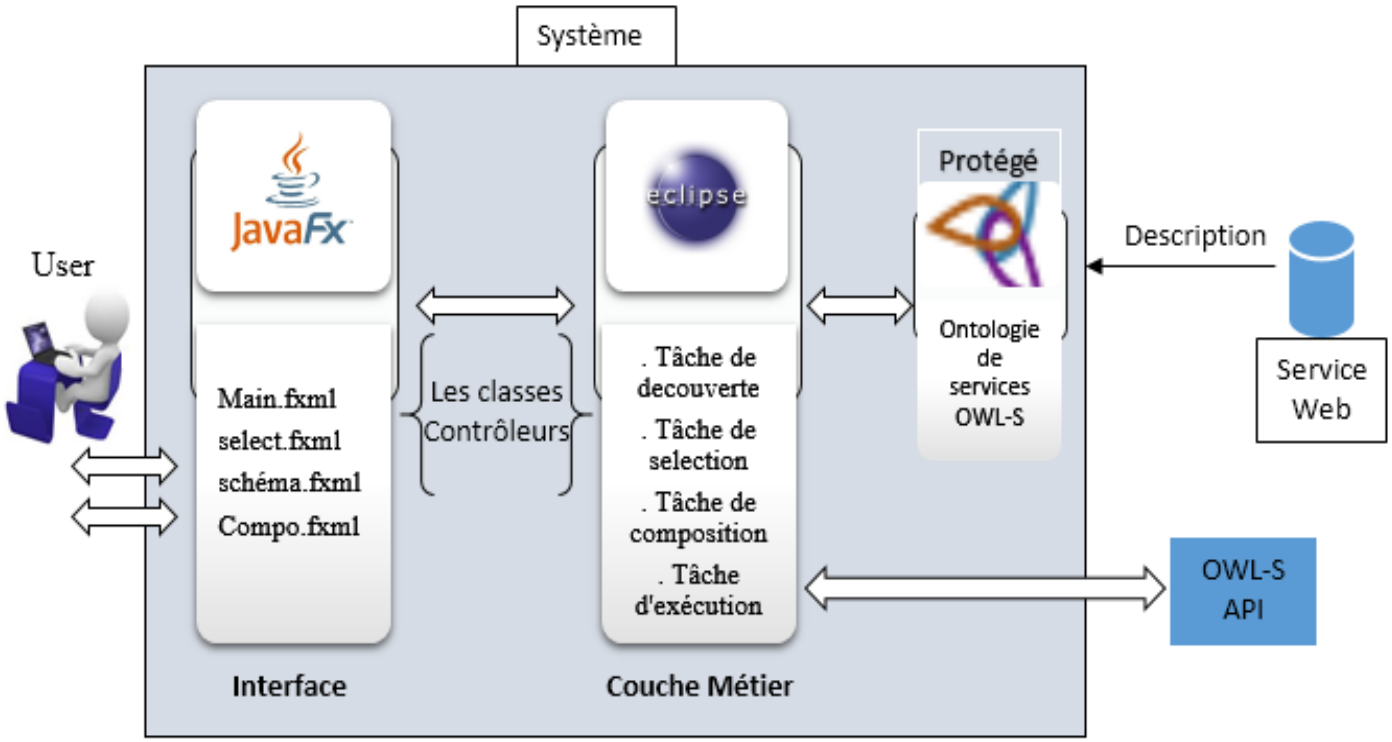


Figure 33 : L'architecture de déploiement du système

IV.4. Tests et résultats :

Maintenant on fait des tests sur l'application, et nous allons montrer via les prises d'écrans les différentes étapes nécessaires à la réalisation de la composition des services web. Les résultats vont afficher dans une interface simple qui facilite à l'utilisateur de bien exploité les fonctionnalités de notre application, qui sont :

IV.4.1. Découverte des services :

Nous avons déjà expliqués cette phase dans la partie conception, c'est la première étape de la composition qui consiste à faire la recherche des services web et répondre à la requête de l'utilisateur. On peut aussi découvrir les services web à l'aide de la recherche avancé (détailles sur les services).

La figure ci-dessous représente les résultats de la recherche simple des services web selon la requête introduite :

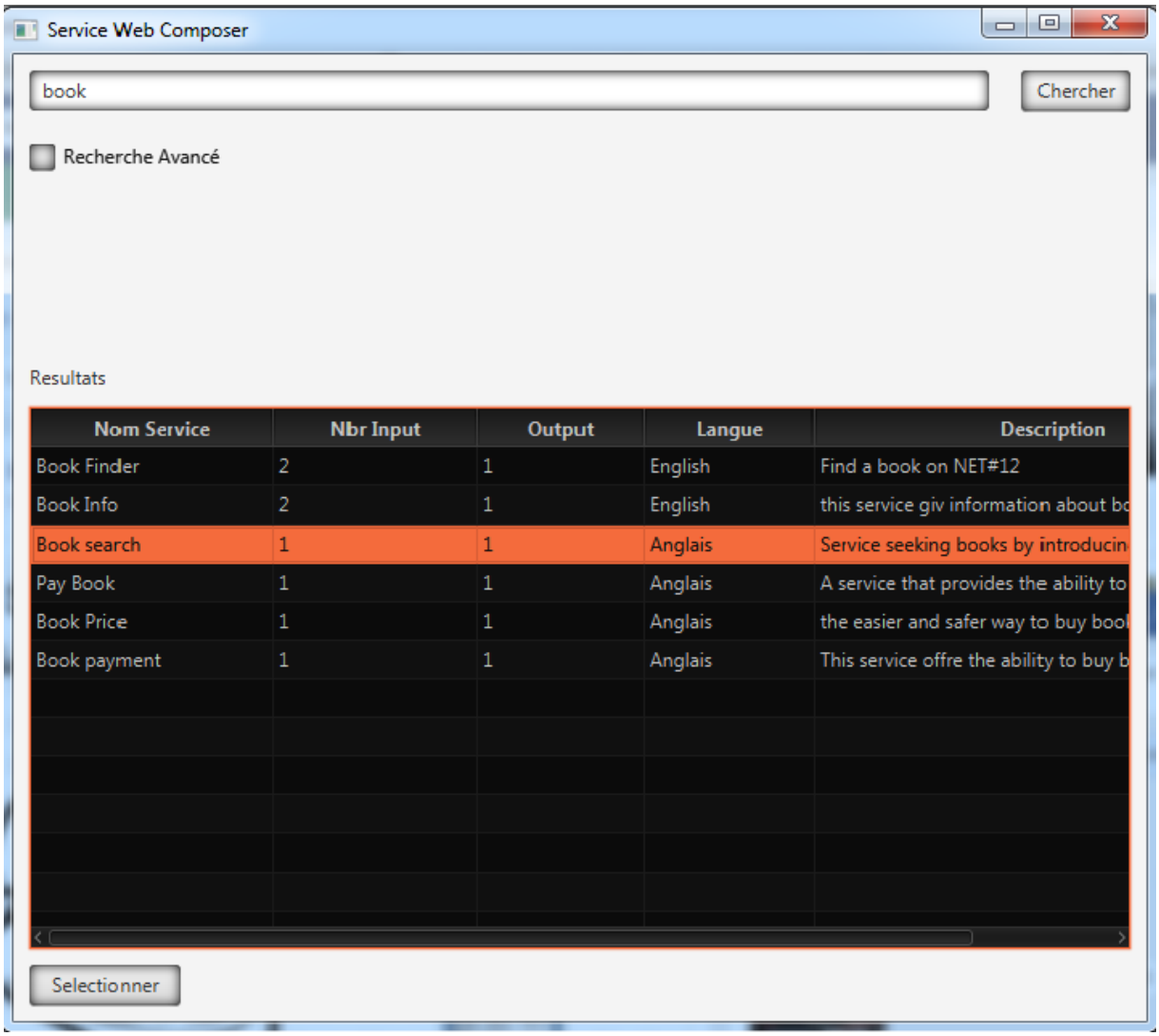


Figure 34 : L'interface de la découverte

La figure suivante montre des résultats plus détaillés sur les services, il suffit de cocher l'option « recherche avancée ». Et de spécifier les Input, les Output et le langage du service

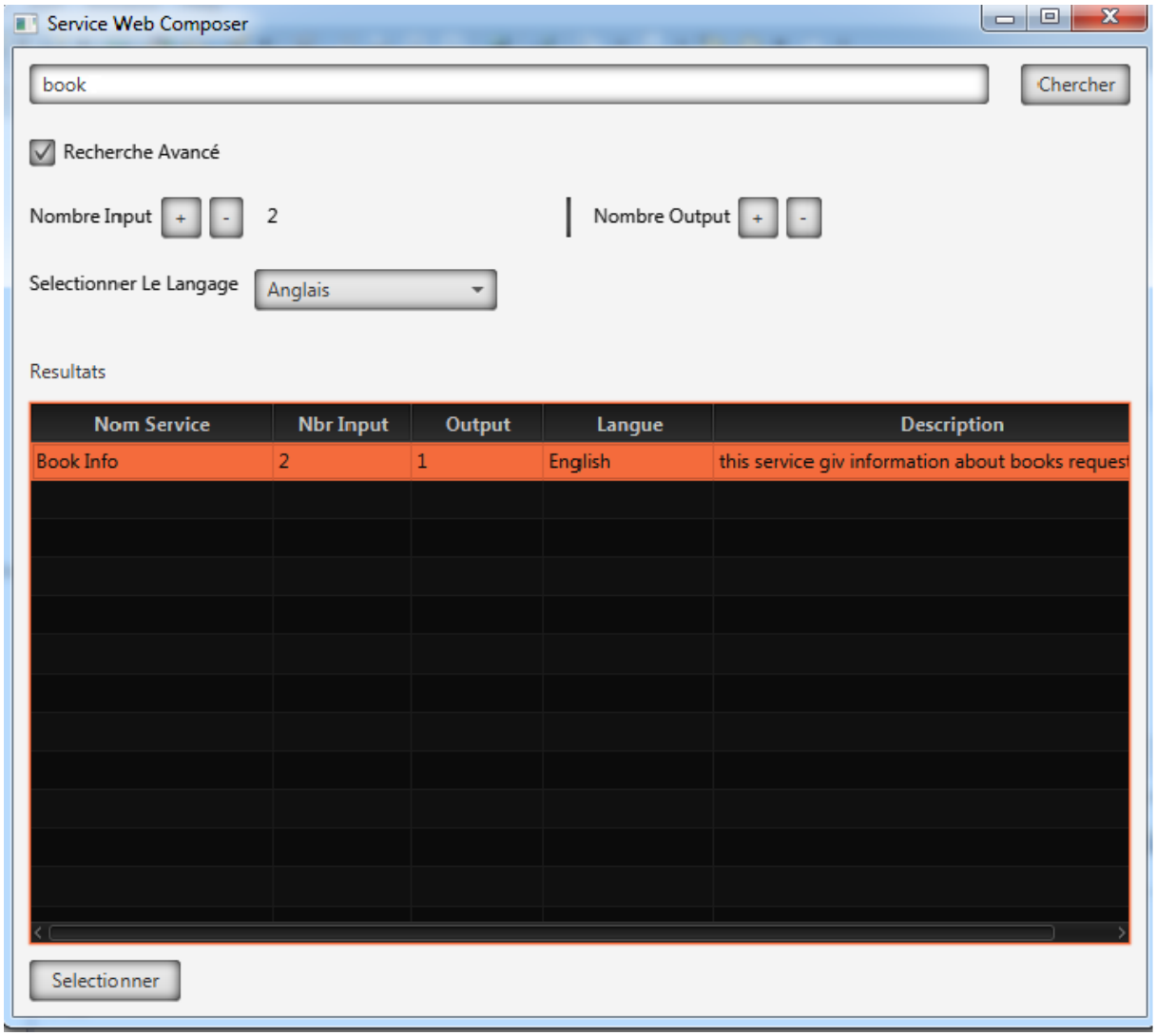


Figure 35 : Recherche Avancée

IV.4.2. Sélection des services :

Après la découverte des services web, la phase sélection c'est la deuxième étape qui permet à l'utilisateur de sélectionner une composition qui a comme service initiale le service choisi dans la tâche découverte parmi les compositions possibles que l'on affiche.

L'interface ci-dessous montre le résultat final, la composition optimale encadrée en vert l'utilisateur peut la choisir comme il peut choisir une autre composition

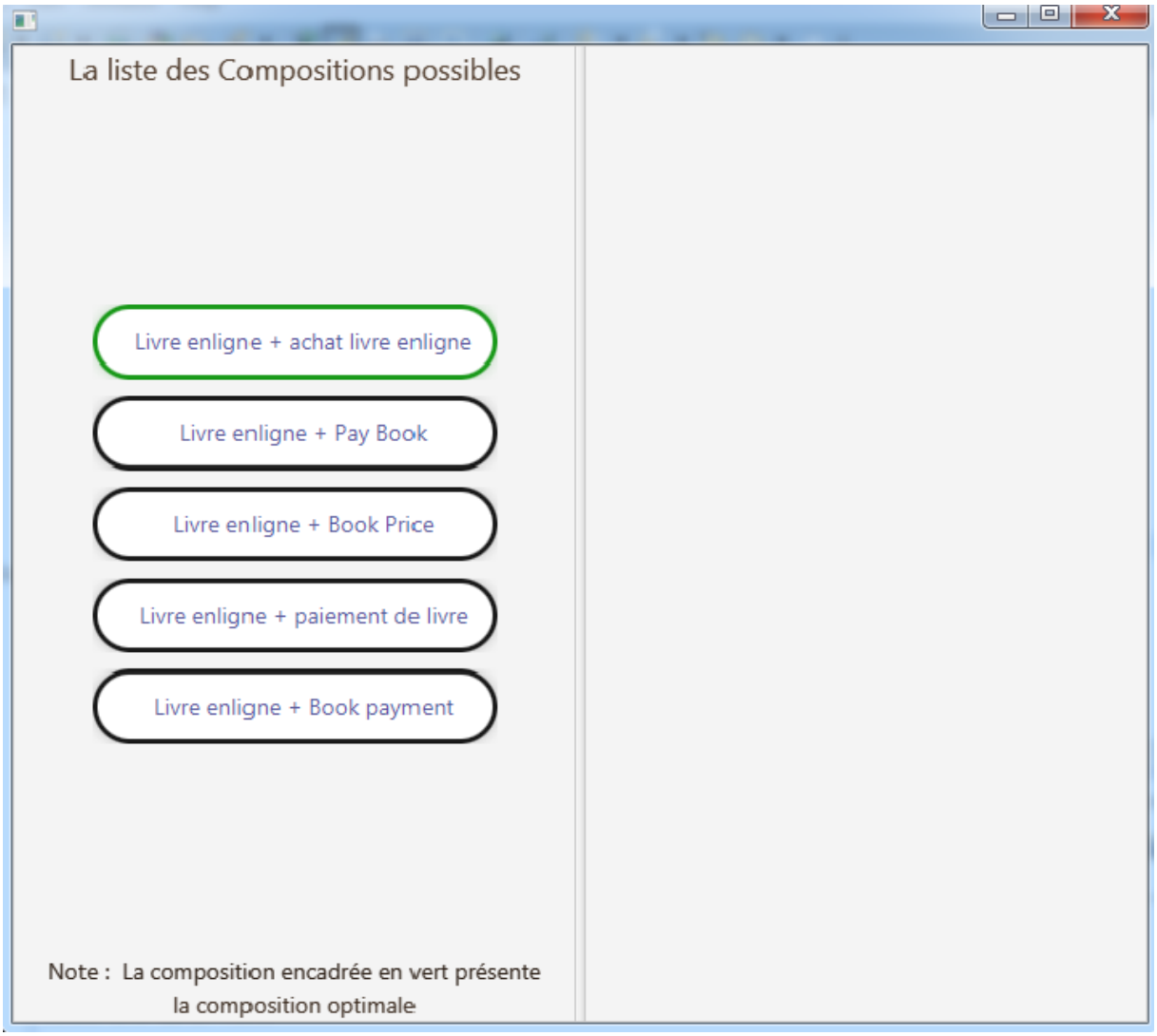


Figure 36 : La liste des compositions possibles (Sélection)

En cas où il n'existe pas des compositions possibles l'interface guide l'utilisateur vers une nouvelle découverte

IV.4.3. Schéma de composition :

C'est l'avant dernière étape du processus de composition qui nous montre la construction du schéma de la composition à partir de la composition choisi dans la tâche précédente, le schéma de composition apparait une fois, et l'utilisateur sélectionne une composition.

La figure suivante représente un exemple de schéma d'une composition choisie :

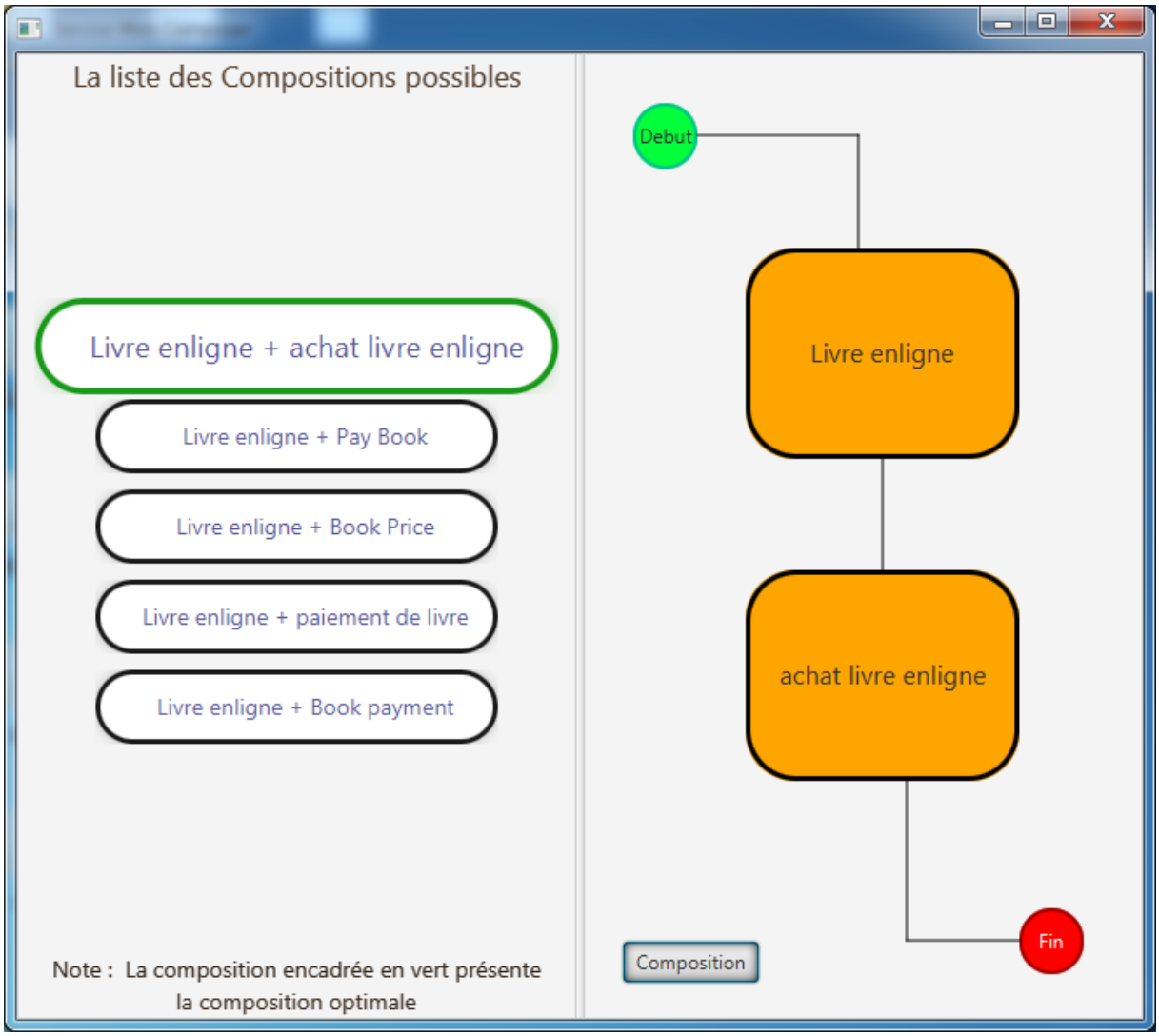


Figure 37 : Schéma de Composition

IV.4.4. La composition :

Pour confirmer une composition il faut cliquer sur **Composition** depuis son schéma de composition dans la figure précédente pour ouvrir une nouvelle fenêtre qui nous aide à **Enregistrer** cette composition en entrant un nom pour le fichier de description d'une composition, un nom du service composite et sa description, comme la montre sur la figure suivante :

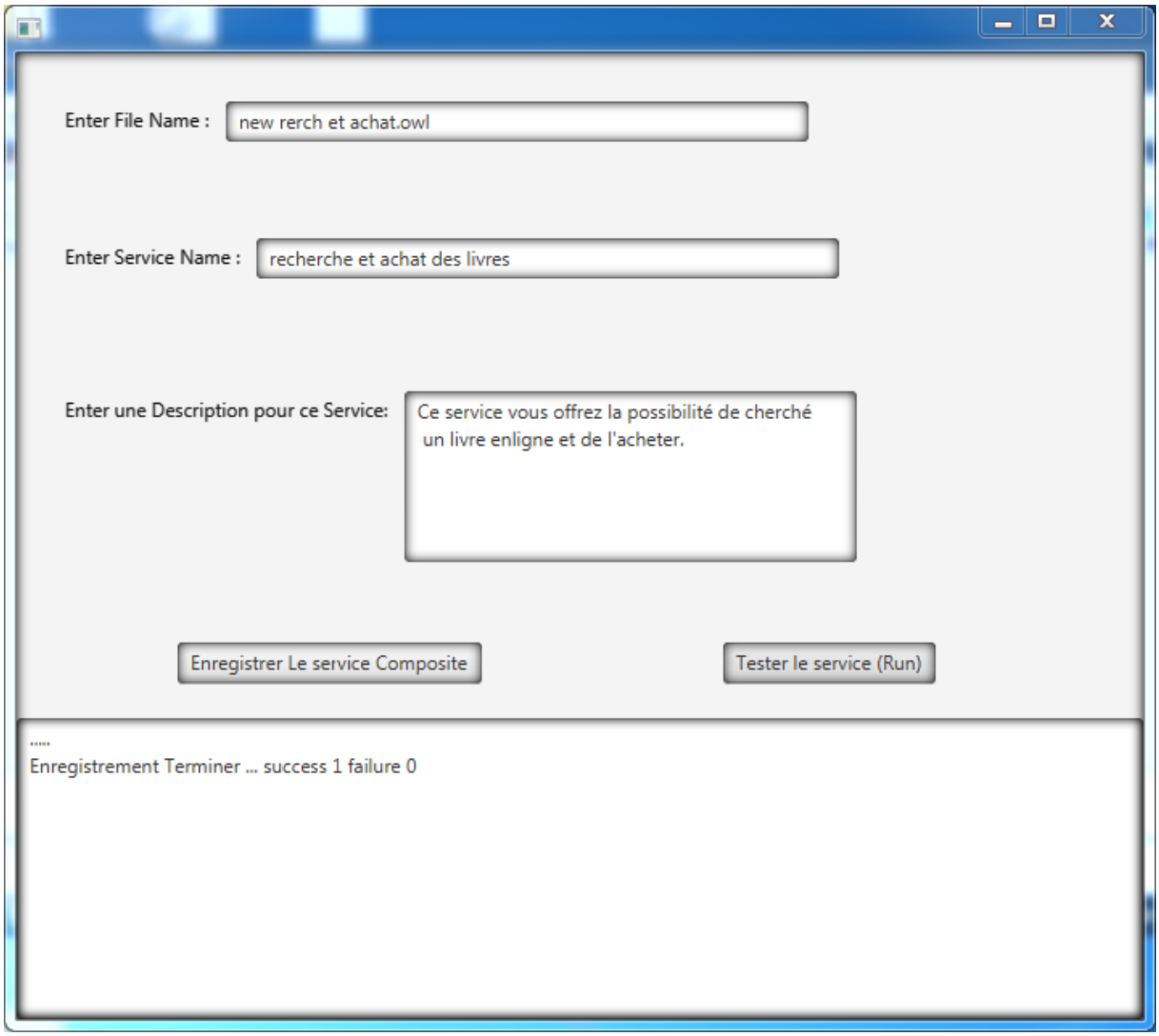


Figure 38 : Composer et Enregistré

IV.5. Conclusion

A travers ce chapitre, Nous avons réalisé un système de composition des services web, cet objectif est concrètement atteint par la description sémantique des services (owl-s). Au cours de cette dernière étape de notre travail, nous avons introduit les différents outils et technologies utilisés dans la réalisation et le déploiement puis on a décrit les principales classes et méthodes prenant part dans notre système. Et pour terminer nous avons créé des services web avec leurs descriptions sémantique pour les tester sur notre système afin de voir les résultats.

En conclusion, les résultats donnent de la satisfaction puisqu'ils répondent aux objectifs du travail demandé.

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 utilisées 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.

A la lumière de notre travail nous avons étudiés l'architecture des services web qui basé sur des standards Internet tels que XML et HTTPqui permettent une communication facile et fluide entre les différents acteurs et clients. Nous avons présentés la composition des services web ainsi les différentes approches.

Les approches de composition des services web varient, on peut les classer en deux catégories : l'approche statique qui est adoptée principalement par le monde industriel et l'approche dynamique, qui est sujet de recherche du monde académique. Après avoir étudiés les travaux existants et les différentes approches de composition de services web, nous avons proposé et réalisé notre modèle de composition de services basé sur le web sémantique (ontologies OWLS).

Les avantages qu'offre notre système :

- On peut exploiter les fonctionnalités du système (découverte, sélection, composition), et cela grâce à des algorithmes qu'on a élaboré.
- Pour avoir composé les services web, en calculant le cout (sa moyenne basée sur le tarif et le temps d'exécution) qui facilité à l'utilisateur de choisir la bonne composition (optimale).
- la découverte peut être une recherche simple ou une recherche avancée qui nous permet d'obtenir plus d'informations sur les services recherché.
- notre système propose plusieurs composition possible des services après la tache de sélection et sont classés de la composition optimale jusqu'en la composition faible selon leur coût.
- La composition de notre système est présentée sous forme d'un diagramme bien clair à l'utilisateur.

Les perspectives :

Afin de compléter le travail présenté dans ce rapport, on peut envisager quelques améliorations et travaux futurs :

- L'exécution des descriptions des services web.
- L'intégration de plusieurs services composite.
- La prise en charge de la réparation des services. En effet quand un service est inaccessible ou en panne, le système doit pouvoir le remplacer par un autre service au moment de l'exécution.
- Interface pour l'intégration personnalisé des descriptions des services dans notre système afin d'ajouter le temps d'exécution et le tarif dans les descriptions à partir du notre application.

Références bibliographiques

Références bibliographiques

[1] : Appui Management de l'Economie. Programme de coopération MEDA. Développement d'applicatifs métiers pour le MTP. "Mise en œuvre des services web avec JAX-WS".

[2] : World Wide Web Consortium " Organisme de normalisation des standarts du web ".

[3] : « http://openclassrooms.com/coures/les-services-web ».

[4] :« http://www.w3.org/XML ».

[5] : Khellaf Radia, "vérification de la compatibilité des services web pour une composition"thése magister informatique Université constantine2 2014.

[6] : « http://www.w3.org/TR/#tr_SOAP ».

[7] :Y.Belaid, "Service web (SOAP)" Urbanisation des SI-NFE107.

[8] : « http://www.w3.org/2003/06/SOAP12-pressrelease.html.fr ».

[9] : « http://igm.univ-mlv.fr/~dr/XPOSE2005/rouvio_webservices/soap.html ».

[10] : « http://www.w3.org/TR/wsdl ».

[11] : « http://www.Uddi.xml.org ».

[12] : Amina Bekkouche, "composition des services web sémantique à base d'algorithmesgénétiques" these magister informatiqueUniversitéTlemcen 2012.

[13] : Julien Guitton "planification multi-agent pour la composition dynamique de service web" Master 2 Université Joseph Fourien 2006.

[14] : Barros A., Dumas, M., Oaks, P. Standards for Web Service Choreography and Orchestration: Status and Perspectives. In: Procs of the 3 rd International Conference the Business Process Management (BPM 2005), 1 st International Workshop on Web Service Choreography and Orchestration for Business Process Management, Nancy, France, Sept. 2005, pp.1-15.

[15] : Peltz, C. Web Services Orchestration and Choreography. IEEE Computer, 2003, vol.36, n°10, pp.46-52.

[16] : ChouirefZahira "un système de programmation fonctionnelle pour la composition des services web" thèse de magister Université de Boumerdes 2008.

[17] : Daha.H, Lifa.S "Une approche formelle pour la composition des services web" Master académique université EL-Oued 2015.

[18] : Martin, D. Burstein, M. Hobbs, J. Lassila, O. McDermott, D.McIlraith, S., Narayanan, S.Paolucci, M.Parsia, B. Payne, T.Sirin, E. Srinivasan, N. and Sicara, K. Owl-s : Semantic markup for webservices. Tech. rep., France Telecom, MINDL Maryland, NIST, Nokia, 2004.

[19] : J. Cardoso and A. Sheth. Semantic e-workflow composition. Technical Report 02-004,LSDIS Lab., Computer Science Department, Univesity of Georgia, Athens, USA, 2002.

[20] : Hans Schuster, DimitriosGeorgakopoulos, AndrzejCichocki, and Donald Baker.Modeling and composing service-based and reference process-based multi-enterprise processes. In Proceedings of 12th International Conference on Advanced Information System Engineering, Stockholm, Sweden, June 2000.Springer-Verlag.

[21] : Rao, J., and Su, X. A survey of automated web service composition methods. In Proceedings of Semantic Web Services and Web Process Composition, First International Workshop (San Diego, CA, July 2004).

[22] : McIlraith, S., and Son, T. Adapting Golog for composition of semantic web services. In Proceedings of the 8th International Conference on Knowledge Representation and Reasoning (KR '02) (Toulouse, France, April 2002), Morgan Kaufmann Publishers, pp. 482–493.

[23] : Hemam née HioualOuassila« composition sémantique des services web dans un contexe d'ebXML »these de Doctorat Université de Constantine 2011.

[24] : Rao, J., Kungas, P., and Matskin, M. Application of linear logic to web service composition. In Proceeding of the 1st International Conference on Web Services (Las Vegas, USA, June 2003).

[25]: B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid. Composing web services on the semantic web. The VLDB Journal, 12 :333–351, 2003.

[26] : Chouirefzahira, Belkhirabdelkader « un systéme de programmation fonctionnelle pour la composition de services web » Université Boumerdes Laboratoire des Systémes informatique USTHB **RIST**.vol 18 N°1 2010.

[27] : DebajyotiMukhopadhyay, ArchanaChougule « A Framework for Semi-automated Web Service Composition in Semantic Web » *Department of Information TechnologyMaharashtra Institute of Technology Pune 411038, India*2013.

[28] :M. Farida Begam and GopinathGanapathy « OntologyBasedDynamic e-Learning Flow Composition of Learning Web Services » School of Computer Science, Engineering and Applications, BharathidasanUniversity, Tiruchirappalli-23, Tamilnadu, India 2014.

[29] : AbdaladhemAlbreshne, Jacques Pasquier « Semantic-BasedSemi-Automatic Web Service Composition » *Computer Department, Fribourg UniversitySwitzerland*2010.