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

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option : modèle d'intelligence et décision *(M.I.D)*

Thème

Découverte de service web sémantique à base de fusion probabiliste

Réalisé par :

- **Melle. Ahmed Dadda wassila**
- **Mme. Mekki nassira**

Présenté le 1 Juin 2016 devant le jury composé de MM.

- *Mr. BENAZZOUZ. M* *(Président)*
- *Mr. HADJILA. F* *(Encadreur)*
- *Mme. CHAOUCH RAMDAN. L* *(Examinateur)*
- *Mr. BENMOUNA. Y* *(Examinateur)*

Année universitaire : 2015-2016

REMERCIMENTS

Avant de commencer nous remercions le Dieu le tout puissant de nous avoir donné la volonté et la force pour achever ce travail.

Nous exprimons nos remerciements à notre encadreur : Mr FETHALLAH HADJILA, pour l'assistance qu'il nous a témoignée, pour sa disponibilité, pour ces orientations et conseils sans lesquels ce travail ne verra pas le jour, qu'il trouve ici l'expression de notre gratitude.

Nous remercions aussi Mr BENNAZZOUZ.M
« Président du jury ».
Mme. CHAOUCH RAMDANE. L. Et
Mr BENMOUNA. Y. pour avoir fait l'insigne honneur d'accepter de jurer notre mémoire.

Nous ne saurons oublier de remercier nos parents pour leur contribution, leur soutien et leur patience, nos proches, nos amis et toutes les personnes
Qui nous ont aidées par leur soutien permanent
De près ou de loin de nos études.

DEDICACE

Je dédie ce modeste travail à ceux qui me sont chers

Mon père

Qui m'a toujours poussé et motivé dans mes études, sans lui je ne serai pas devenue ce que je suis maintenant.

Ma chère mère

Parce qu'il est impossible de trouver les mots à la hauteur de l'amour et le soutient que vous m'avait toujours témoigné tous en long de ma vie et ma scolarité.

A mon mari

Qui n'a jamais cessé de me soutenir et de m'apporter son aide, je ne le remercierai jamais assez pour tout ce qu'il a fait pour moi ;

A mes enfants

Ma véritable source de bonheur, ***MANEL*** *et* ***WALID,*** *je vous aime très fort.*

A mes sœurs et mes frères

Ma véritable source de bonheur qui n'ont jamais cessé de croire en moi.

A toute la famille MEKKI.

A toutes mes collègues de travail et mes amies.

NACERA

DEDICACE

A mes très chers parents

Je vous dois ce que je suis aujourd'hui grâce à votre amour, à votre patience et vos innombrables sacrifices.

Que ce modeste travail, soit pour vous une petite compensation et reconnaissance envers ce que vous avez fait d'incroyable pour moi.

Que dieu, le tout puissant, vous préserve et vous procure santé et longue vie afin que je puisse à mon tour vous combler.

A mes très chères sœurs MALIKA, FATIMA ZOHRA

A mes très chers amis

MERIEM ET BOUTHAINA *En témoignage de l'amitié sincère qui nous a liée et des bons moments passés ensemble. Je vous dédie ce travail en vous souhaitant un avenir radieux et plein de bonnes promesses.*

En souvenir de nos éclats de rire, des bons moments et des nuits blanches. En souvenir de tout ce qu'on a vécu ensemble. J'espère de tout mon cœur que notre amitié durera éternellement.

A SAMIA, WAFAA, MALEK, MERIEM_FARAH, SALMAFATEN, SARA, SAID, AMINE, AMINE, KAMEL, KAMEL, NABILA, KASSEM, BENAOUDA, Aucune dédicace ne serait exprimée assez profondément ce que Vous avez fait pour moi.

WASSILA

Table des matières

Introduction générale

Introduction générale

Contexte

Aujourd'hui, nous vivons dans une société de consommation orientée technologie, dans laquelle l'échange d'information est primordial. Du point de vue de l'entreprise, cela implique une bonne gestion des activités ainsi qu'une bonne communication au sein de toute l'organisation. Chaque entreprise doit se mettre à jour continuellement afin de suivre l'évolution des technologies de mondialisation et de globalisation qui rendent la concurrence toujours plus féroce. Dans cette logique, les firmes doivent s'adapter à ce contexte en étant toujours plus performantes et flexibles afin de réaliser leurs objectifs et se démarquer de la concurrence.

Actuellement les systèmes informatiques des entreprises sont de plus en plus complexes (en termes de volume de données, nature et complexité des objets stockés, protocoles métiers), de ce fait la communication entre deux systèmes informatique n'est pas toujours évidente et triviale. Pour assurer les échanges de données les chercheurs ont proposé architecture de conception et d'implémentation d'applications nommée architecture orientée service (**SOA**) qui s'avère être une solution adéquate.

Le concept d'architecture orientée service **SOA**, n'a toujours pas de définition exacte et universelle. Cette approche a été développée par diverses entreprises de recherche en informatique telles qu'IBM, BEA, Oracle ou Microsoft qui ont chacun leur propre concept et leur propre définition de cette architecture. Bien que toutes les définitions apparaissant dans la littérature se ressemblent, elles comportent, chacune, des divergences.

L'architecture orientée service est née pour répondre aux inconvénients des technologies orientées composants, telles que CORBA [OMG, 2008] (Common Object Request Broker Architecture), Java RMI [Downing, 1998] (Remote Method Invocation), DCOM (Distributed Component Object Model). [Horstmann et Kirtland, 1997]. Cependant l'interopérabilité qu'offraient ces derniers n'autorisait pas l'incompatibilité entre les langages de programmation.

Selon [Bieberstein et al, 2005] « A service-oriented architecture is a framework for integrating business processes and supporting IT infrastructure as secure, standardized components – services, that can be reused and combined to address changing business priorities ».

Problématique

Aujourd'hui le besoin d'un client est devenu de nature multi objectif (minimiser les coûts, minimiser les délais, augmenter le taux de service…). De plus, comme le nombre de services sur le web est en augmentation constante, il sera nécessaire de développer des outils de recherche qui seront efficace et qui tiennent compte de tous aspects fonctionnels. L'une des principales préoccupations, est de localiser, d'évaluer et de classer tous les services web qui sont adaptés (partiellement / ou globalement) aux besoins fonctionnels du demandeur (requête). Plus précisément, nous avons besoin d'une approche de découverte de services efficace (i.e. temps d'exécution est faible) et effective (i.e. les critères de rappel et de précision sont optimisés).

Contribution

Pour concevoir une approche de découverte performante nous fusionnons cinq (05) algorithmes de matching, qui sont les suivants :cosine, extended jaccard, loss of information, jensen shannon et logique. Nous avons choisi ces cinq (05) mesures de similarités parce qu'elles sont considérées comme des approches les plus performantes dans ce domaine de recherche d'information (RI).

Pour fusionner les classements fournis par les cinq (05) algorithmes, nous avons choisi une approche probabiliste qui se base sur les principes suivants :

- Découper les classements d'entrée en segments, le nombre optimal de segment sera sélectionné empiriquement.
- Le score global de chaque service (celui qui déterminera l'ordre du service dans la liste globale), est calculé avec une combinaison linéaire de probabilités apprise durant l'entrainement.
- Plus ces probabilités de pertinence (entre segment-requête) sont grandes, plus le score global est grand.
- Plus le rang du segment dans lequel un service S appartient est mauvais (grand), plus son score global est petit.

Plan de mémoire

Notre mémoire est composé des chapitres suivants :

Chapitre 1 : Ce chapitre se divisera en deux parties : la première présente l'architecture orienté service (SOA), en exprimant son rôle et ses avantages, en suite, dans la deuxième partie nous parlons du service web et ses standards, après, nous exposons la problématique de découverte de service web en citant quelques approches.

Chapitre 2 : Dans ce chapitre nous allons voir : tout d'abord la description de la collection de test, ensuite nous présentons la conception de nos algorithmes, après nous montrons le prototype ainsi que les résultats expérimentaux, et enfin nous terminons par une conclusion.

Chapitre I

L’architecture SOA

&

Service web

I. Introduction

La discipline des systèmes informatiques intelligents a connu une évolution rapide ces dernières années dans le monde. Ce n'était qu'une question de temps pour que l'intelligence artificielle moderne soit intégrée dans le cursus de graduation. Dans la société d'aujourd'hui, l'intelligence artificielle et l'apprentissage des machines deviennent de plus en plus répandus. Avec l'avènement du web, des millions de personnes sont déjà familières avec des logiciels intégrant l'intelligence artificielle comme la recherche sur le web, le e-commerce, des sites de jeux.

Les dernières décennies ont été marquées par le développement rapide des systèmes d'information distribués, et tout particulièrement par la diffusion de l'accès à Internet. Cette évolution du monde informatique a entraîné le développement de nouveaux paradigmes d'interaction entre applications tels que la SOA (L'Architecture Orienté Service). Cette dernière a été mise en avant afin de permettre des interactions entre applications distantes.

II. L'Architecture Orienté Service (SOA)

II.1 Définition

Le SOA est un paradigme fondé sur la description et l'interaction de services, autrement dit L'AOS est une approche architecturale permettant la création des systèmes basés sur une collection de services développés dans différents langages de programmation, hébergés sur différentes plates-formes avec divers modèles de sécurité et processus métier [Barry, 2003].

Selon [Dodani, 2004], « L'architecture orientée service permet l'intégration d'applications et de ressources de manière flexible en représentant chaque application ou ressource sous la forme d'un service exposant une interface standardisée, permettant à un service d'échanger des informations structurées (messages, documents, objets métier), coordonnant et en organisant les services afin d'assurer qu'ils puissent être invoqués, utilisés et changés efficacement.

L'idée maîtresse de l'architecture orientée service est que tout élément du système d'information doit devenir un service identifiable, documenté, fiable, indépendant des autres services, accessible, et réalisant un ensemble de tâches parfaitement définies [David, 2004].

Nous pouvons décrire la SOA, au niveau de la gestion d'entreprise, comme une approche de conception et de construction de système d'informations (SI) qui utilise des interfaces services pour la création de SI. Cette architecture met à disposition des utilisateurs les fonctionnalités et les services proposés par une entreprise, via une interface standardisée soit à l'interne soit sur Internet (pour le e-commerce par exemple).

II.2 Rôle De l'architecture (SOA)

De nos jours l'SOA est devenu une nécessité, car c'est une architecture basée sur des standards, ainsi elle est caractérisée par son couplage faible entre les services et par l'indépendance par rapport aux aspects technologiques, en plus, elle sécurise l'investissement des applications existantes.

Cette architecture qui s'avère être un modèle d'intégration moderne a comme principal objectif l'augmentation de la flexibilité, la réduction des coûts d'intégration, la mise à disposition de services réutilisables, la modification et la réutilisation des fonctionnalités ainsi que la composition des processus de gestion par l'utilisation de services déjà existants ou à créer.

II.3 Avantages de l'architecture (SOA)

L'SOA encapsule plusieurs avantages bénéfiques pour le domaine de la technologie d'information et de communication. Elle offre :

- La réutilisation et la composition : permettant le partage des modules entre les applications.
- La pérennité : impliquant le support des technologies existantes et à venir.
- L'évolutivité : permettant aux applications d'ajouter de nouveaux modules afin de répondre aux nouveaux besoins fonctionnels.
- Réduction du coût : minimisant le coût de développement des grands projets.
- Une grande tolérance aux pannes avec une souplesse dans la maintenance.

III. Service Web

La SOA est entrée depuis peu de temps dans le domaine du réel, grâce à un ensemble de normes appelées collectivement services Web. Les services Web, réalisation

Concrète des architectures SOA, sont la déclinaison du paradigme des architectures orientées service, sur le Web.

La technologie des services web représente la technologie la plus utilisée pour migrer vers SOA.

III.1 Définition

1) Selon **W3C** (**World Wide Web Consortium**)[1], un Web service(ou service Web) est une application appelable via Internet par une autre application d'un autre site Internet permettant l'échange de données (de manière textuelle) afin que l'application appelante puisse intégrer le résultat de l'échange à ses propres analyses. Les requêtes et les réponses sont soumises à des standards et normalisées à chacun de leurs échanges.

2) **«** *Un service Web est une application accessible à partir du Web. Il utilise les protocoles Internet pour communiquer, et utilise un langage standard pour décrire son interface*»**.** [Melliti 2004].

3) **«** *Les services Web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto-descriptives qui peuvent être publiées, localisées et invoquées depuis le Web. Les services Web effectuent des actions allant de simples requêtes à des processus métiers complexes. Une fois qu'un service Web est déployé, d'autres applications (y compris des services Web) peuvent le découvrir et l'invoquer* **».** [Ponge 2008].

4) « *Les services web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto –contenues et auto –descriptives qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus complexes. Une fois qu'un service Web est déployé, d'autres applications peuvent le découvrir et l'invoquer* ». IBM [Colan, 2003]

III.2Standard des Services Web

Les standards de bases utilisés par le Web Service sont : programmable web, UDDI [Clement et al, 2004], WSDL [Chinnici et al, 2007] et SOAP [Gudin et al, 2003].
UDDI sont proposées par OASIS.
WSDL et **SOAP** font parties des normes W3C.

[1]www.w3c.org

Pour utiliser un Web Services, il faut premièrement savoir qu'il existe.
UDDI (Universal Description, Discovery and Integration Service) est la norme qui définit le mécanisme pour découvrir dynamiquement des services. Un client pointe vers un registraire UDDI, qui lui donnera la définition du service recherché. Le registraire UDDI sert de pages jaunes et liste les services disponibles. Le registraire UDDI est lui-même un Web Service qu'un client peut questionner [hadjila, 2014].

Pour être capable d'utiliser un Web Services et de programmer un client, il est nécessaire d'en connaître la définition. Le langage WSDL (Web Services Définition Langage) décrit l'interface au service. En utilisant XML Schéma, WSDL défini les paramètres d'entrée et de retour d'un appel au service Web.

Les appels comme tel aux Web Services sont effectués avec le protocole SOAP (Simple Object Access Protocol). SOAP offre le transport d'objets sérialisés et autres données en XML et l'appel de procédures distantes. SOAP a pour principe objectif d'assurer la communication entre machines.

La figure 1 présente les facettes générales d'un service web, premièrement un service web est décrit par une interface XML nommée WSDL, il peut échanger des documents XML avec d'autres services à l'aide du protocole SOAP, il peut être recherché dans un annuaire tel que l'UDDI.

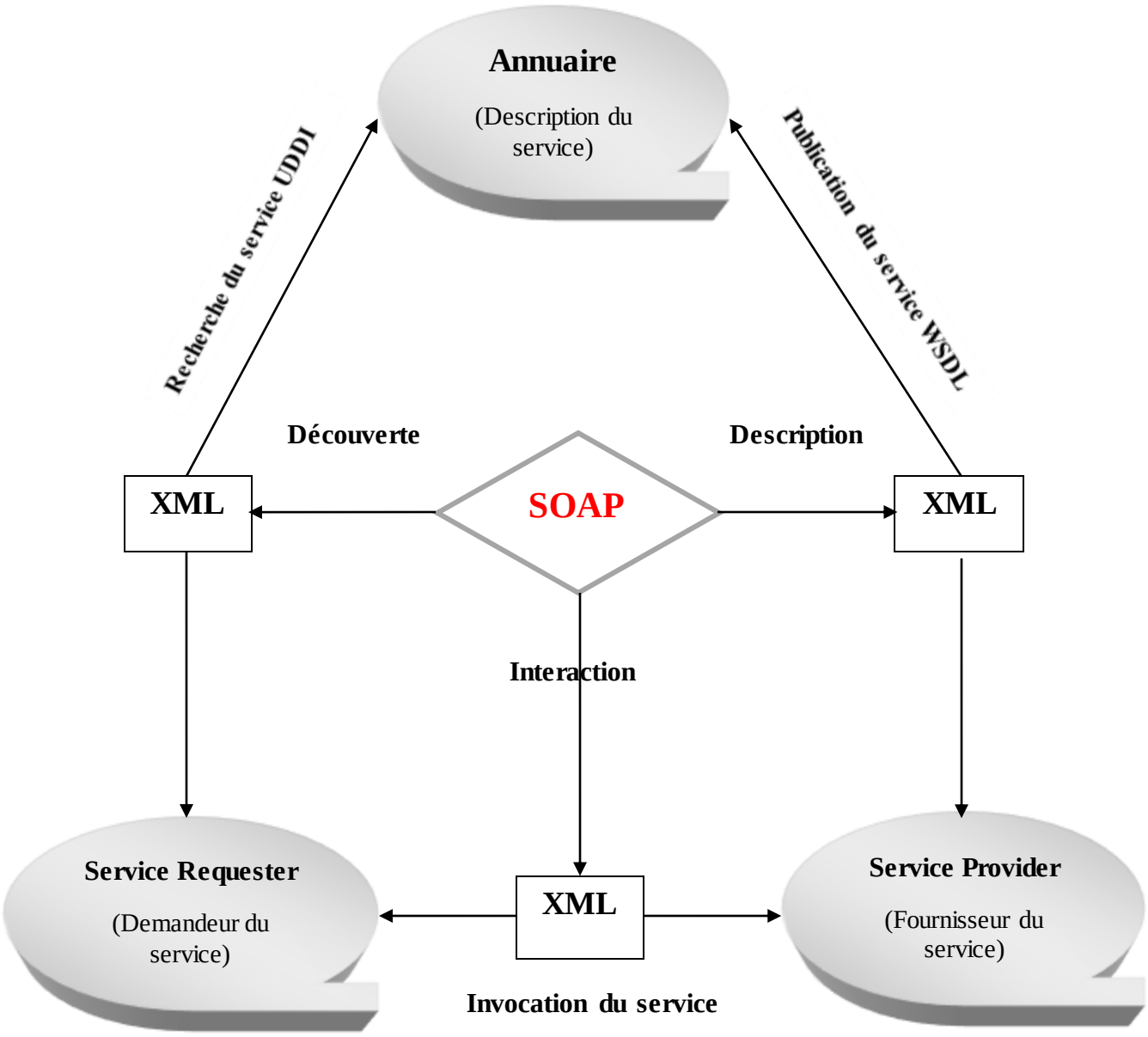

Figure I.1 : Les standards du service Web et les relations entre eux

Décortiquons ce schéma :

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

III.3 Description 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 service	**UDDI**
Description du service	**WSDL**
Communications	**SOAP**
	XML
Transport	**HTTP, SMTP**

Figure I.2 : Couche technologiques des web services.

- Couches technologiques des services Web :
  - Le transport de messages XML-RPC ou SOAP est assuré par le standard HTTP.
  - SOAP ou XML-RPC prévoit la couche de communication basée sur XML pour accéder à des services Web

- La description d'un service Web se fait en utilisant le langage WSDL. WSDL expose l'interface du service.
- 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.

III.4 Caractéristiques d'un service web

La technologie des services Web repose essentiellement sur une représentation standard des données (interfaces, messageries) au moyen du langage XML. Cette technologie est devenue la base de l'informatique distribuée sur Internet et offre beaucoup d'opportunités au développeur Web.

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, comment l'invoquer et où le trouver ?) 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 : SMTP, FTP, BEEP...).
- L'intégration d'application en implémentant des services Web produit des systèmes faiblement couplés, le demandeur du service ne connaît pas forcément le fournisseur.

Ce dernier peut disparaître sans perturber l'application cliente qui trouvera un autre fournisseur en cherchant dans l'annuaire.

III.4 Quelques avantages des services web

L'utilisation de l'architecture des services Web offre les avantages suivants :

- ✓ **Elle favorise l'interopérabilité en réduisant les exigences pour obtenir une compréhension partagée : le** Web Service Description Langage (WSDL), une interface basée sur le langage XML, est la seule exigence pour obtenir une compréhension partagée des informations entre un fournisseur de services et un demandeur de services. En limitant les exigences nécessaires à l'interopérabilité, les services Web qui travaillent

ensemble peuvent être réellement indépendants de la plateforme et du langage. En limitant les exigences requises, les services Web peuvent être intégrés à l'aide de nombreuses infrastructures sous-jacentes différentes.

- ✓ **Elle réduit la complexité par l'isolation :** Tous les composants des services Web sont des services. Ce qui est important, c'est le type de comportement qu'un service a, et non la manière dont il a été mis en œuvre. Un document WSDL est le mécanisme qui permet de décrire le comportement isolé par un service.
- ✓ **Elle permet l'interopérabilité des applications patrimoniales :** En permettant aux applications patrimoniales d'être englobées dans un WSDL, et désignées en tant que services, l'architecture des services Web offre une nouvelle interopérabilité entre elles. De plus, des technologies de sécurité, de logiciels médiateurs et de communication peuvent être englobées pour participer à un service Web en tant que conditions préalables pour l'environnement.

IV. Service Web Sémantique

Le Web sémantique (plus techniquement appelé **« le Web de données »**) permet aux machines de comprendre la sémantique, la signification de l'information sur le Web. Il étend le réseau des hyperliens entre des pages Web classiques par un réseau de lien entre données structurées permettant ainsi aux agents automatisés d'accéder plus intelligemment aux différentes sources de données contenues sur le Web et, de cette manière, d'effectuer des tâches (recherche, apprentissage, etc.) plus précises pour les utilisateurs. Le terme a été inventé par Tim Berners-Lee, co-inventeur du Web et directeur du W3C, qui supervise l'élaboration des propositions de standards du Web sémantique.

Les technologies actuelles basées sur SOAP, WSDL et UDDI sont suffisantes pour mettre en place des composants interopérables et intégrables, mais elles ne permettent qu'une description syntaxique de l'interface des SW. Par conséquent, les technologies du web sémantique telles que les ontologies permettent de décrire sémantiquement les services Web toutes en facilitant leur utilisation automatique. La combinaison des deux dernières technologies, services web et ontologie ont donné naissance aux services Web sémantiques.

L'interdépendance faible (loose coupling) des services est atteinte par les Web Services, mais l'interdépendance faible au niveau des données nécessite des ontologies du Web Sémantiqu

Les approches les plus représentatives des Web services sémantiques sont WSDL-S [Akkiraju et al, 2005], SAWDL, OWLS [Martin et al, 2004], WSMO. Les deux premières sont des annotations sémantiques du standard WSDL, alors que les deux dernières sont des ontologies de services (elles ne sont pas directement liées à WSDL).

V. Problématique de découverte de service

Étant donné un besoin d'un client, qui peut être présenté sous la forme d'un ensemble de concepts d'entrées, de concepts de sorties, et éventuellement des descriptions informelles de la fonctionnalité du service, nous devons créer des mécanismes qui comparent ces besoins avec l'ensemble des services publiés, ces mécanismes doivent gérer la sémantique, en plus ils doivent avoir une bonne performance en termes de rappel, de précision et de temps d'exécution.

V.1 Definitions

Le W3C definite la découverte de services comme suit: "Web service discovery is the act of locating a machine-processable description of a Web service that may have been previously unknown and that meets certain functional criteria. It involves matching a set of criteria with a set of Web service descriptions. The goal is to find an appropriate Web service". [**W3C, 2004 b]**.

[Booth et al, 2004] décrivent le processus de découverte, comme étant la localisation d'une description compréhensible par la machine d'un service éventuellement inconnu au préalable et correspondant à certains critères fonctionnels.

[Toma et al, 2005] Définissent la découverte comme le processus qui prend en entrée une requête utilisateur et retourne une liste de ressources ou services, pouvant combler éventuellement le besoin décrit.

[Keller et al, 2004] Définissent la découverte comme la localisation automatique des services répondant à une requête utilisateur.

Ces définitions mettent l'accent sur le mécanisme de comparaison de la requête avec les services, ainsi que son degré d'automatisation. Selon notre point de vue la découverte de services vise à comparer une requête d'un utilisateur avec les capacités d'un service web, et trie les résultats selon un certain mécanisme.

V.2 Approche de découverte

Il existe plusieurs approches de découverte de service web on va noter quelques-unes :

- **Approches fonctionnelles**

Les capacités fonctionnelles d'un service peuvent inclure, des informations telles que les entrées, les sorties, la catégorie du service, le comportement, les annotations informelles et éventuellement les préférences. Plusieurs approches ont été proposées pour la découverte fonctionnelle de services, on distingue les approches sémantiques, les approches syntaxiques (non sémantiques), et les approches hybrides.

- **Approches syntaxique(ou Basées sur les Interfaces Syntaxiques)**

Utilisent généralement l'interface WSDL, comme description de services, et adoptent les techniques de recherche d'informations et éventuellement le clustering pour le matching.

La recherche dans l'annuaire UDDI est l'exemple typique d'une découverte syntaxique, les mots clés de la requête sont comparés avec les attributs enregistrés, nous pouvons faire une recherche par nom de services ou nom d'entreprise, ou sa catégorie.

- **Approches sémantiques**

Plusieurs interfaces sémantiques ont été créées pour assurer la découverte et la composition de services WSMO [Roman et al, 2005], OWL-S, et SAWSDL [Farrell et Lausen, 2007], toutes les approches de cette catégorie adoptent les ontologies pour le matching de la requête avec les services. Nous notons que ces approches, sont plus complexes et plus fiables que les techniques syntaxiques. Nous distinguons 03 classes d'approches sémantiques, les approches logiques, non logiques et hybrides. Les approches logiques exploitent les inférences pour vérifier la compatibilité entre la requête et le l'annotation de service (subsomption, test de consistance…), alors que les approches non logiques exploitent la sémantique implicite ou informelle des services et la traite avec d'autres techniques, telles que le datamining, le matching de graphes, la recherche d'informations, les mesures de similarité. La troisième classe mélange les deux premiers types.

- **Approches Hybrides**

  Ces dernières implémentent plusieurs filtres, certains d'entre eux sont purement logiques alors que d'autres se basent sur les techniques de recherche d'information, ou le datamining…

Nous notons que certains filtres compensent la défaillance des autres, et de ce fait un service web sera accepté s'il satisfait au moins un filtre. Nous notons aussi que les scores des filtres logiques sont plus élevés que ceux des filtres à base de techniques de recherche d'information (en termes de pertinence).

- **Approches comportementales**

Les services atomiques sont généralement composés pour satisfaire les besoins des clients ou des entreprises. Pour rechercher ces compositions (workflows). Les méthodes de découverte doivent prendre en charge la notion de comportement des processus, (ou les cheminements d'exécution), ce comportement doit être modélisé avec des moyens formels tels que, les réseaux de petri, les automates d'états finis, les processus d'algèbre.

- **Approches Non Fonctionnelles (Prenant en Compte la Qualité de Service(QOS))**
  - **Approches à Base de Réputation et Confiance**

[Ye et al, 2009] se basent sur la réputation comme moyen de découverte de services. De façon générale, les auteurs adoptent le feed-back des communautés d'utilisateurs pour sélectionner les services. Ces approches ont plusieurs inconvénients :

- Pas de modèle précis pour les propriétés de sélection de services
- Ils supposent que les valeurs de propriétés de service, sont facilement calculables, et même ces moyens ne sont pas évalués.
- Les fonctions l'agrégation ne sont pas détaillées.

VI. Conclusion

De plus en plus, avec l'essor d'Internet, le développement tend vers les technologies du Web. Les Services Web sont des composants logiciels représentant une fonction applicative, ils représentent un mécanisme de communication entre applications distantes à travers le Web.

Les services Web permettent le changement de la nature du Web, du Web du document utilisé par les organisations pour la publication des informations, au Web orienté service, qui permet aux serveurs d'applications la communication entre eux. Les services Web sont suffisamment développés pour que les développeurs les utilisent maintenant dans tous les domaines de l'informatique, afin de récolter les divers bénéfices de la technologie.

A travers les différentes sections que nous avons montrées, nous concluons que l'architecture (SOA) et plus particulièrement les services web sont à l'heure actuelle de plus en plus incontournables, ils sont maintenant capables d'échanger des données de façon quasi autonome à travers internet.

Chapitre II

Conception & réalisation

I. Introduction

Dans le cadre de cette partie, notre objectif consiste à résoudre le problème de découverte de service web en adoptant une approche de fusion. Plus particulièrement nous utilisons des probabilités apprises, pour agréger plusieurs fonctions de matching, en suite nous combinons ces probabilités avec une formule linéaire. Le plan de ce chapitre est décrit comme suit : Tout d'abord nous décrivons la collection de test, ensuite nous présentons la conception (les algorithmes RPC, PF, NST), après nous montrons le prototype ainsi que les résultats expérimentaux, et enfin nous terminons par une conclusion.

II. Présentation de la collection de test

Le corpus OWLS-TC version 2.2 est la collection de test utilisée dans notre approche. Elle est développée par le centre allemand pour la recherche en intelligence artificielle (http://www.dfki.de/scallops), La collection est destinée à soutenir l'évaluation de la performance de OWL-S. La majorité de ces services ont été récupérées à partir des registres publics IBM UDDI, et semi- automatiquement transformé de WSDL pour OWL-S.

II.1 OWLS

OWLS[2] est une ontologie de haut niveau qui indique qu'une ressource(entreprise) est liée à un service. Ce dernier est constitué d'un "profile", d'un "service model" et d'un "service grounding". En bref, le "profile" indique ce que fait le service, le "service model" décrit le comment : les étapes qui composent le service et le flot de control, et le "service grounding" décrit la manière d'accéder au service.

- OWLS - TC2 fournit 1007 services web sémantiques segmentés en 07 classes :
    1. Le domaine d'éducation
    2. Le Domaine médical
    3. Le domaine de nourriture
    4. Le domaine militaire
    5. Le domaine de voyages (tourisme

[2] http://projects.semwebcentral.org/projects/owls-tc/.

6. Le domaine de communication
7. Le domaine d'économie.

La base constituée aussi d'un ensemble de requêtes reparties sur les sept (07) classes, ces requêtes sont modélisées sous forme de document OWLS. Chaque document OWLS (service ou requête) comporte dans sa partie « profile » des éléments « profile : hasinput » et « profile : hasoutput », ces derniers sont employés comme entrées pour le module de découverte des services web. Chaque service web (i.e. document OWLS) est étiqueté manuellement par des experts humains comme étant relevant(pertinent) ou non par rapport à une requête donnée. Ceci permet le calcul des rappels et des précisions des approches proposées. Elle offre aussi un ensemble d'ontologies pour décrire les services et les requêtes, chaque classe de services possède une ou plusieurs ontologies. Dans ce qui suit, nous donnons un exemple d'une description de service web et d'une description de requête.

- **Requête** : Cette requête cherche le titre d'un film comédien.
  Concepts d'entrée : books.owl#Title
  Concepts de sortie : my_ontology.owl#ComedyFilm
- **Service :** Ce service cherche les moyens de diagnostique offert par une clinique.
  Nom de service : MEDICALCLINIC_DIAGNOSTICPROCESS_SERVICE
  Concepts d'entrée : Mid-level-ontology.owl#MedicalClinic
  Concepts de sortie : SUMO.owl#DiagnosticProcess

III. Conception

Dans cette partie, nous avons présenté les trois algorithmes utilisés dans notre approche « Relevance Probability Computation **"RPC"** », « Probabilistic fusion **"PF"** », « number segment tuning **"NST"** ».

III.1 les mesures de similarité

1. **Cosinus (cos) :**

D'où : $$\mathbf{Sim_{Cos}(S,R)} = \frac{\sum_{i=1}^{M} Wi,R \times Wi,S}{\sqrt{\sum_{i=1}^{M} W^2 i,R} \times \sqrt{\sum_{i=1}^{M} W^2 i,S}}$$

Sim_x (S, R) : retourne une valeur de similarité selon la méthode 'x' entre un service **'S'**et une requête **'R'**.

$W_{i,R}$: désigne le poids du i-ème index dans la requête **'R'**.

$W_{i,S}$: désigne le poids du i-ème index dans le service **'S'**.

2. **Extended jacquard (EJ) :**

$$Sim_{EJ}(S,R) = \frac{\sum_{i=1}^{M} Wi,R \times Wi,S}{\sum_{i=1}^{M} W^2 i,R + \sum_{i=1}^{M} W^2 i,S - \sum_{i=1}^{M} Wi,R \times Wi,S}$$

3. **Loss of information (LOI) :**

$$Sim_{LOI}(S,R) = \frac{|PC_{R,x} \cup PC_{S,x}| - |PC_{R,x} \cap PC_{S,x}|}{|PC_{R,x}| + |PC_{S,x}|}$$

Avec X ∈ {**In, Out**}, $PC_{R,x}$ et $PC_{S,x}$ ensemble de composants logique Input/output de la requête **'R'** et le service **'S'.**

4. **Jensen-Shannon (JS) :**

$$Sim_{JS}(S,R) = \frac{1}{2log2} \sum_{i=1}^{n} h(Pi,R) + h(Pi,S) - h(Pi,R + Pi,S)$$

5. **Logique (log) :**

Une approche basée sur le raisonnement par subsomption, elle offre cinq (05) score : Exact match, Plug-in match, Subsumer match, Subsumed-by match et Fail.

III.2 Approche RPC

Algorithme "Relevance Probabilty Computation"

Entrés :

dataset : base de services, **SRQ** : sous ensemble de la base des requêtes

m :entier // nombre de fonction de matching, **ns** :entier // nombre de segment

Rel : matrice binaire (req-serv) //indique le service pertinent pour la requête

Sortie : MRelP : matrice de réels de taille ns .m

Début

1. **Pour** i←1 **à** m **faire**
2. ranking$_i$ ← EmptyList;
3. **Pour** j ← 1 **à** SRQ **Faire**
4. **Pour** l ← 1 **à** dataset **Faire**
5. Score ← sim$_i$ (Q_j, S_l);
6. insertInto (score, ranking$_i$);

FinPour

7. decreasing-sort(ranking$_i$);

8. relv-services ← extract (Rel, j);
9. **Pour** r← 1 **à** ns **Faire**
10. segment-members ← extract-seg (ranking$_i$, r);
11. relv[i][j][r] ← |relv-services-segment-members|/ |segment-members|;

FinPour

FinPour

12. **Pour** r← 1 **à** ns **Faire**
13. **Pour** b← 1 **à** |SRQ| **Faire**
14. MRelP[i][r] ← MRelP[i][r]+relv[i][b][r];

FinPour

15. MrelP[i][r] ← MRelP[i][r]/|SRQ|;

FinPour

FinPour

16. return (MRelP);

Fin

Algorithme 1 : Algorithme de Relevance Probabilty computation "RPC"

- ✓ **Les fonctions utilisées**
  - **« Sim » :** elle prend comme paramètre la requête et le service, elle retourne une valeur représentant la similarité entre un service **S** et une requête **Q** et selon la méthode **i.**
  - **« InsertInto »** : elle prend comme paramètre le score et le rang, elle a pour objectif de charger la liste avec les services et les score.
  - **« Decreasing-sort » :** elle a pour rôle de trier la liste construite précédemment
  - **« Extract » :** elle prend en paramètre la matrice (**Rel)** et la requête **j** elle récupère les services pertinents de la requête N°= j
  - **« Extract-seg » :** on récupère les services membres du segment N°=r de la liste i
- ✓ **Déroulement de l'algorithme**

Ligne 1...6 : pour chaque fonction de matching **i** et requête $\mathbf{Q_j}$, on calcule le classement individuel correspondant (rankingi).

Ligne 7…8 : nous trions le classement calculé précédemment et nous obtenons les services pertinents de la requête $\mathbf{Q_j}$**.**

Ligne 9…11 : pour chaque segment **r**, nous calculons la probabilité pour qu'un segment **r** dérivé de la fonction **i** soit pertinent par rapport à la requête $\mathbf{Q_j}$.

Ligne 12…15 : pour chaque segment **r** et chaque fonction de matching **i,** on calcule la moyenne de probabilité de pertinence. Ces probabilités de pertinence sont basées sur les requêtes de l'ensemble d'apprentissage (**SRQ**).

Ligne 16 : Nous retournons les probabilités apprises.

Le second algorithme appelé **PF** (fusion probabiliste) permet de calculer un score fusionné pour chaque service $\mathbf{S_l}$. À cette fin, on se base sur des probabilités de pertinence tirées des cinq (05) classements individuels. PF repose sur deux heuristiques (**H1, H2**) **[Lilllis et al,2006]**, elles sont résumées comme suit :

- ✓ Plus le rang (ou l'identificateur de segment) d'un service $\mathbf{S_l}$ est élevé dans les classements individuels, plus le score fusionné est meilleur (**H1**).
- ✓ Plus la probabilité de pertinence $\mathbf{MRelP_{ri}\ (S_l)}$ est élevée, plus le score fusionné est meilleur(**H2**).

III.3 Approche PF

Algorithme de "Probalistic Fusion"

Entrées :

Q_j : requête courante, **m :** entier // nombre de fonction de matching

K : entier // indique la taille de la liste à retourner, **ns** : entier // nombre de segment

MRelP : matrice de réels de taille ns .m

Sortie : Top-K(CombinedList) : liste fusionnée et ordonnée

Début

1. **Pour** l← 0 **à** |dataset|-1 **Faire**
2. Fscore[l] ← 0;

FinPour

3. CombinedList ←EmptyList;
4. **Pour** i← 1 **à** m **Faire**
5. $ranking_i$ ← EmptyList;
6. **Pour** l← 1 **à** dataset **Faire**
7. score ← sim_i (Q_j, S_l);
8. insertInto (score, S_l,$ranking_i$);

FinPour

9. decreasing-sort($ranking_i$);
10. **Pour** l←1 **à** dataset **Faire**
11. Sid← Get-Seg-ID(S_l,$ranking_i$);
12. Fscore[l] ← Fscore[l]+MRelP[i][sid]/sid;

FinPour

FinPour

13. **Pour** l← 1 **à** dataset **Faire**
14. insertInto(Fscore[l], S_l, CombineList);

FinPour

15. decreasing-sort(CombineList);
16. return Top-k(CombinedList);

Fin.

Algorithme 2 : Algorithme de "Probabilistic Fusion" (PF)

- ✓ **Fonctions utilisé**
  - **« Get-Seg-ID » :** prendre en paramètre le service S_l et la liste i, on récupère le n°= de segment.
- ✓ **Déroulement de l'algorithme**

Ligne 1…3 : nous initialisons les scores fusionnés, par 0 et la liste fusionnée par vide.

Ligne 4…8 : pour chaque fonction de matching **i** et requête Qj, nous appliquons les 02 principes **H1**, **H2**, grâce à la formule suivante :

Fscore[l] ← Fscore[l]+MRelP[i][sid]/sid;

Ligne 9 : nous trions le classement calculé précédemment

Ligne 10…11 : pour chaque service S_l, nous obtenons le n° du segment dans lequel il se trouve (Sid).

Ligne 12 : Nous mettons à jour le score fusionné.

Ligne 13…15 : nous trions la liste combinée, selon l'ordre décroissant du score fusionné.

Ligne 16 : nous retournons les éléments Top K de la liste combinée.

Dans ce qui suit nous présentons le troisième algorithme désigné sous le nom de **« Nmuber of Segment Tuning "NST" ».** Cet algorithme vise à sélectionner le nombre optimal de segments (NS dénoté) ce dernier assure le meilleur MAP (Mean Average Precision) des listes fusionnées.

III.4 Approche NST

Algorithme de "Number of Segment Tuning"

Entrées :

Dataset : base de service, **CRQ :** base de requête

m : entier // nombre de fonction de matching (prendre la valeur 5 par défaut)

Rel : matrice binaire (req-serv) // indique le service pertinent pour chaque requête

Q_j : requête courante

Sortie : ns* : entier //indique le nombre optimal de segments

Début

1. R-prec_moy* ← 0 ; ns* ← 2 ;
2. **Pour** ns ← 2 **à** round(|dataset|/2) **Faire**
3. MRelP ← RPC (dataset, SRQ, m, ns, Rel);
4. **Pour** j ←1 **à** CRQ **Faire**
5. CombinedList [j] ← PF(Q_j, m,k,ns,MRelP);
6. R-prec[j] ← R-precision(CombinedList$_j$,Rel);
7. R-prec_moy[ns] ← R-prec_moy[ns]+R-prec[j];

FinPour

8. R-prec_moy[ns] ← R-precision_moy[ns]/CRQ;
9. **Si** (R-prec_moy[ns]> R-prec_moy[ns]*) **Alors**
10. R-prec_moy* ← R-prec_moy[ns] ;
11. ns* ← ns ;

Finsi

FinPour

12. return (ns*);

Fin.

Algorithme 3: Algorithme de "Number of Segment Tuning" (NST)

- ✓ **Fonctions utilisée**
  - **« RPC »** (déjà expliqué).
  - **« PF »** (déjà expliqué).
  - « **R-précision** » calcule la précision noté (Rprec)
  - « **R-precision_moy** » calcule la moyenne des Rprec

✓ **Déroulement de l'algorithme**

Ligne 1 : nous initialisons ns* et rprec moy*.**Ligne 2...3 : pour chaque valeur possible du ns** nous faisons l'apprentissage des probabilités de pertinence RPC (appel à la méthode RPC).

Ligne 4…8 : nous appliquons la fusion probabiliste (PF) pour toutes les requêtes (0-28) (appel de la méthode PF) jusqu'à l'obtention de classement de chaque requête, pour calculer la R-precision, ainsi la moyenne de R-précision.

Ligne 9…11 : Nous mettons à jour le nombre optimal de segments (ns*), ainsi que sa précision moyenne (R-prec_moy*) correspondant.

Ligne 12 : nous retournons le nombre optimal de segment ns*.

III.5 Exemple

Nous démontrons un scénario qu'illustre le fonctionnement de la fusion probabiliste (à savoir, RPC et PF). Etant donné une base constituée de quatre (04) services {S1, S2, S3, S4} repartie sur deux (02) segments (R1, R2) selon les fonctions de matching (f1, f2) appliqué sur deux (02) requêtes {Q1, Q2}.

Requête Q1 :

Sachant que les services pertinents pour Q1 sont : {S1, S2, S3}, le classement obtenu par la fonction f1 est : <S2, S3, S1, S4>, et par la fonction f2 est :<S1, S2, S3, S4>. Donc, nous appliquons la simulation de l'approche RPC sur les données précédentes, commençons par les résultats obtenus dans le tableau suivant :

$Relv_{111}=1$	$Relv_{112}=0.5$
$Relv_{211}=1$	$Relv_{212}=0.5$

Tableau II.1 :la probabilité des services pertinents pour Q1 selon chaque fonction de matching

D'où :

$Relv_{111}$: la probabilité de relevance pour fonction f1, requête Q1, et de segment R1

$Relv_{212}$: la probabilité de relevance pour fonction f2, requête Q1, et de segment R

Requête Q2 :

Les services pertinents sont : {S2, S3}, le classement obtenu par la fonction f1 : <S3, S2, S4, S1>, et par la fonction f2 : <S4, S3, S1, S2>, nous appliquons les mêmes étapes que la première requête Q1, les résultats présenté dans le tableau suivant :

$Relv_{121}=1$	$Relv_{122}=0.0$
$Relv_{221}=0.5$	$Relv_{222}=0.5$

Tableau II.2 : la probabilité des services pertinents pour Q2 selon chaque fonction de matching

D'où :

$Relv_{121}$: la probabilité de relevance pour fonction f1, requête Q2, et de segment R1

$Relv_{222}$: la probabilité de relevance pour fonction f2, requête Q2, et de segment R2

Ensuite, nous calculons la moyenne de probabilité de relevance($MRelP_{ir}$) de toutes les requêtes, les résultats présenté dans le tableau suivant :

$MRelP_{11}=1$	$MRelP_{12}=0.25$
$MRelP_{21}=0.75$	$MRelP_{22}=0.5$

Tableau II.3 :la moyenne de probabilité de toutes les requête

D'où :

$MRelP_{11}$: la moyenne de probabilité de relevance pour la fonction **f1**, et le segment **R1**

$MRelP_{21}$: la moyenne de probabilité de relevance pour la fonction **f2**, et le segment **R1**

Nous utilisons les résultats obtenus de l'approche RPC et nous appliquons la simulation de l'approche PF, le tableau suivant montre les résultats de calcul de l'approche PF pour les requêtes Q1 et Q2 :

Requête Q1 :

Service	Fscore
S1	0.87
S3	1.25
S2	1.75
S4	0.37

Tableau II.4 : calcul du score des services pour la requête Q1

Le classement final de ces services pour la requête Q1 donné dans le graphe suivant :

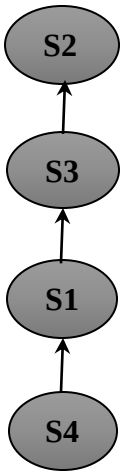

Figure II.3 : graphe du classement selon la probabiliste de fusion pour la requête Q1

Requête Q2 :

Service	Fscore
S1	0.37
S2	1.25
S3	1.75
S4	0.87

Tableau II.5 : calcul du score des services pour la requête Q2

Le classement final de ces services pour la requête Q2 donné dans le graphe suivant :

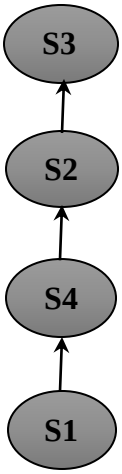

Figure II.4 : graphe du classement selon la fusion probabiliste pour la requête Q2

IV. Présentation de prototype

IV.1 Outils et environnement de développement

Avant de commencer l'implémentation de notre application, nous allons tout d'abord spécifier les outils utilisés qui nous ont semblé être un bon choix vu les avantages qu'ils offrent, Notant que Java est un langage de programmation orienté objet, libre, simple et portable, nous avons utilisé le langage de programmation « Java » (JDK 1.6.0.10), avec l'IDE « NetBeans 6.8 ».

Les API utilisées :

- **JDOM :** est une API open source Java son but est de représenter et manipuler un document XML, elle utilise des classes plutôt que des interfaces. Ainsi pour créer un nouvel élément, il faut simplement instancier une classe, nous avons utilisé cette API aussi pour manipuler les fichiers de description des web services et des requêtes.
- **JFreeChart :** est une API open source Java sous licence LGPL, sa documentation est payante. Elle permet la création des graphiques et des diagrammes de très bonne qualité.

IV.2 présentation du prototype

✧ **Fenêtre d'accueil :**

Figure II.5 : Interface d'accueil

✧ **Fenêtre d'indexation :**

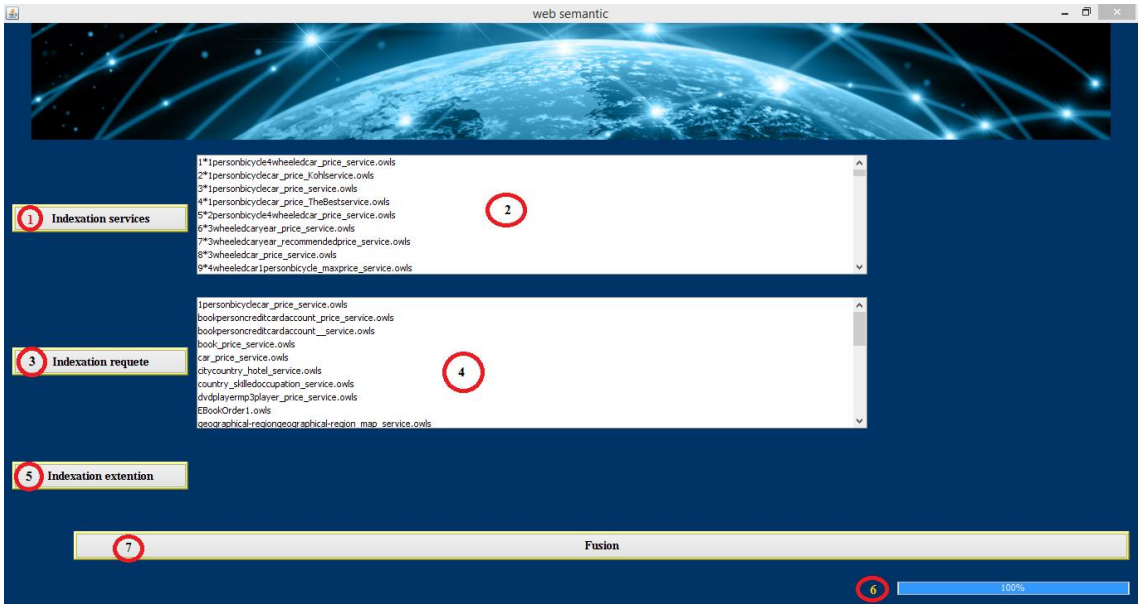

Figure II.6 : Interface d'indexation

Commentaire :

1- Bouton pour lancer l'indexation des services.
2- L'affichage de la liste des services web indexé.
3- Bouton pour lancer l'indexation des requêtes
4- L'affichage de la liste des requêtes indexées.
5- Bouton pour lancer l'indexation des extensions.
6- La barre de progression pour indiquer l'état d'avancement de l'opération qui est en cours d'exécution.
7- Bouton pour lancer l'approche.

✧ **Fenêtre de l'approche RPC**

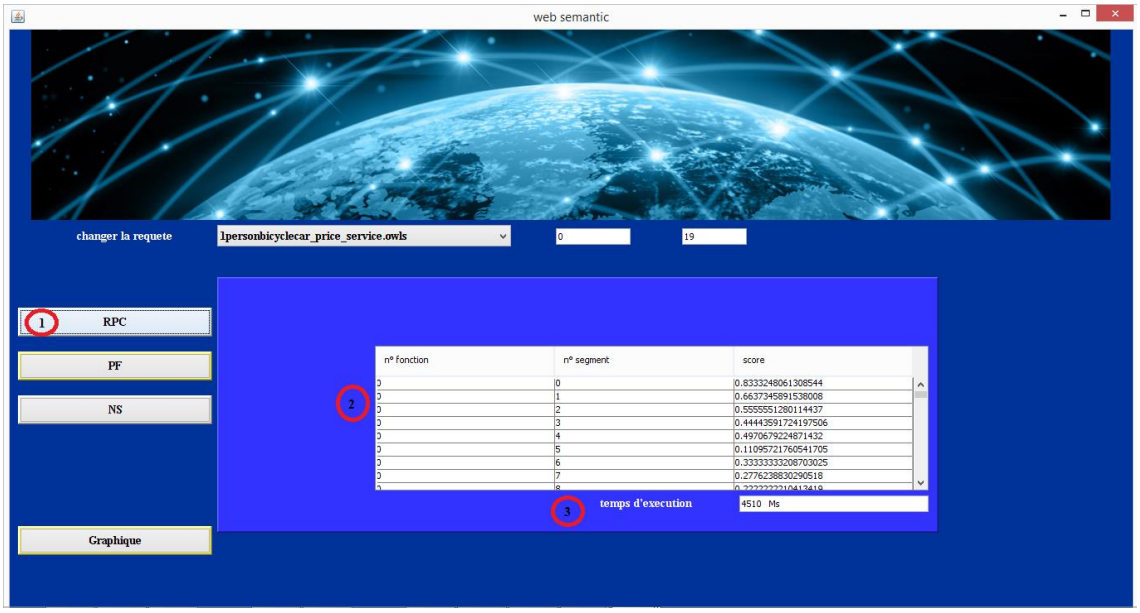

Figure II.7 : Interface de l'approche RPC

Commentaire :

1- Bouton pour lancer l'approche RPC
2- Tableau affiche les scores de RPC
3- Affiche le temps d'exécution de RPC

✧ **Fenêtre de l'approche PF**

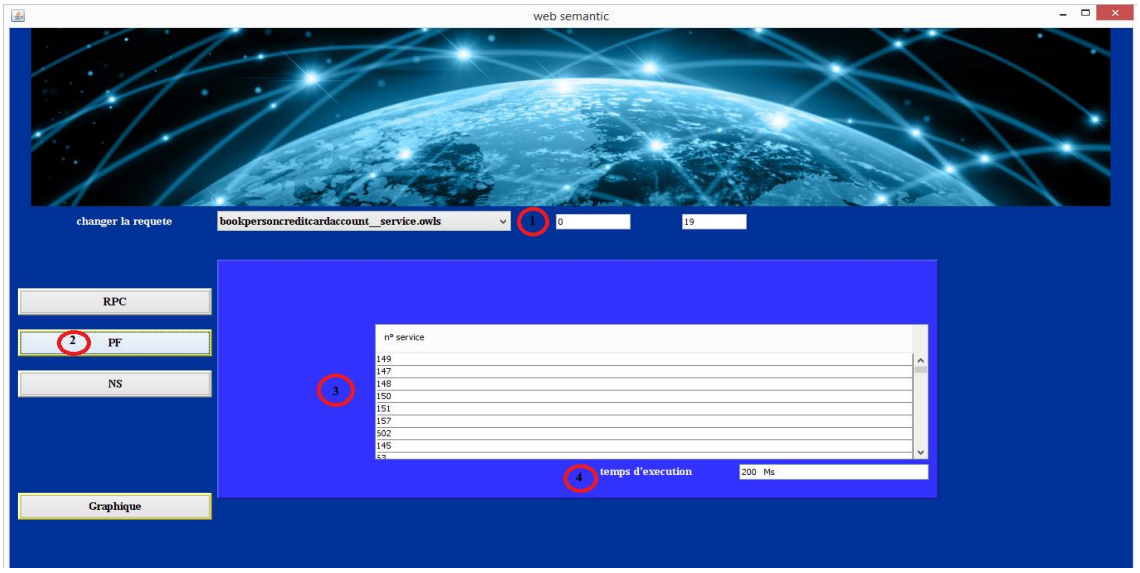

Figure II.8 : Interface de l'approche PF

Commentaire :

1- Nous choisissons une requête
2- Bouton pour lancer l'approche PF
3- Tableau affiche tous les services TOPK
4- Le temps d'exécution

✧ **Fenêtres d'évaluation**

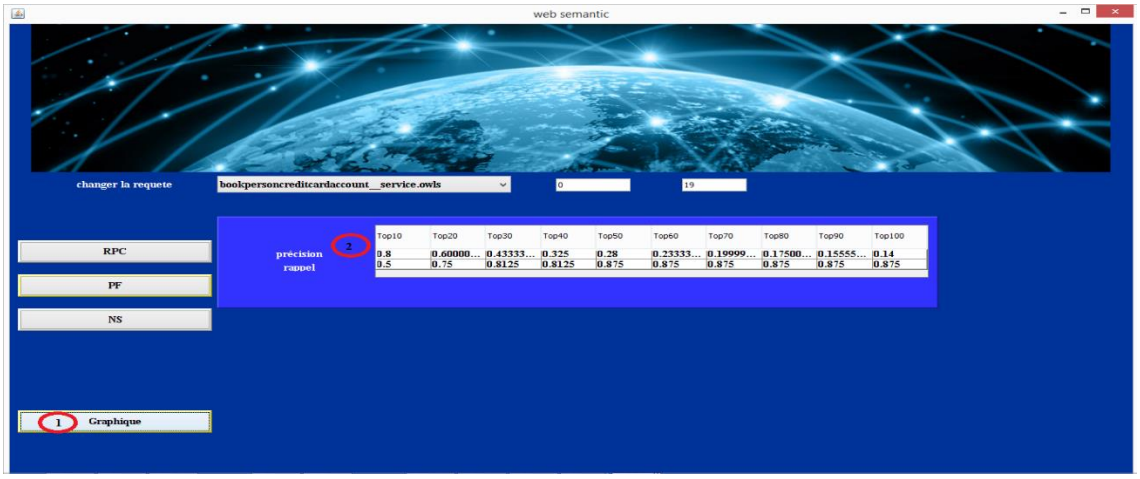

Figure II.9 : fenêtre de TOPK de rappel et précision

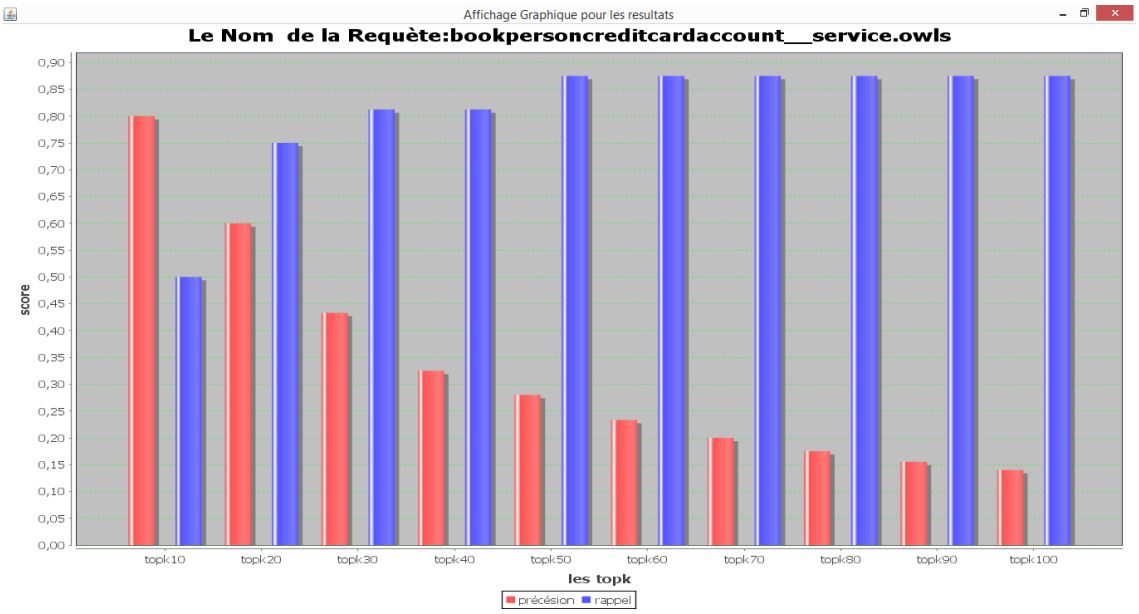

Figure II.10 : fenêtre d'évaluation graphique

Commentaire :

1- Bouton pour lancer l'évaluation numérique
2- Tableau affiche les résultats de l'évaluation
3- Figure affiche les résultats graphiques

V. Expérimentation

Cette section montre nos expériences liées à la fusion probabiliste ainsi que les différentes fonctions correspondantes. Ces expériences ont été menées sur une machine Intel(R) Core(TM)2 Duo CPU T6570 @2.10GHz avec 4.00 Go de la RAM, sous le système d'exploitation Windows 8.1.

Le tableau suivant montre le temps moyen d'exécution de la phase d'apprentissage (fonction RPC), la phase de fusion (fonction PF), ainsi que la durée totale.

- **Le temps d'exécution :** c'est la période de temps qui s'écoule entre la validation de la requête de l'utilisateur et la réception des résultats finaux (pour le cas de découverte, composition ou sélection).

Temps moyen de la fusion (PF)	Temps moyen de l'apprentissage (RPC)	Somme
212 ms	5391 ms	5603 ms

Tableau II.6 : temps d'exécution moyen de la fusion probabiliste

Les deux tableaux suivants montrent la relation entre les performances (en termes de précision et de rappel, et pour toutes les valeurs de K : k = 10 ... k = 60) de la fusion probabiliste et les valeurs possibles de ns.

- **Précision :** mesure la proportion de documents pertinents relativement à l'ensemble des documents restitués par le système. Elle est exprimée par :

$$\textbf{Précision} = \frac{vp}{(vp + fp)}$$

vp : les vrais positifs. (Un résultat correct, et considéré comme étant valide par le système).

fp : les faux positifs. (Un résultat erroné, mais considéré comme étant valide par le système).

- **Rappel :** mesure la proportion de documents pertinents restitués par le système relativement à l'ensemble des documents pertinents contenus dans la base documentaire. Il est exprimé par :

$$Rappel = \frac{vp}{(vp + fn)}$$

fn : les faux négatifs. (Un résultat correct, et considéré comme faux par le systèm

NS	TOP10	TOP20	TOP30	TOP40	TOP50	TOP60
100	**0.419**	**0.661**	**0.777**	**0.829**	**0.867**	**0.908**
150	**0.425**	**0.678**	**0.793**	**0.85**	**0.884**	**0.911**
200	**0.423**	**0.679**	**0.801**	**0.86**	**0.898**	**0.924**
251	**0.421**	**0.679**	**0.815**	**0.871**	**0.904**	**0.931**
300	**0.426**	**0.685**	**0.806**	**0.868**	**0.904**	**0.932**
350	**0.434**	**0.691**	**0.811**	**0.873**	**0.911**	**0.936**
400	**0.433**	**0.691**	**0.811**	**0.872**	**0.911**	**0.936**
500	**0.432**	**0.705**	**0.828**	**0.892**	**0.93**	**0.949**

Tableau II.7 : score du rappel par rapport à ns

NS	TOP10	TOP20	TOP30	TOP40	TOP50	TOP60
100	**0.906**	**0.746**	**0.616**	**0.506**	**0.432**	**0.381**
150	**0.896**	**0.76**	**0.624**	**0.516**	**0.437**	**0.382**
200	**0.893**	**0.762**	**0.633**	**0.524**	**0.446**	**0.389**
251	**0.893**	**0.768**	**0.642**	**0.529**	**0.449**	**0.393**
300	**0.896**	**0.77**	**0.636**	**0.528**	**0.448**	**0.391**
350	**0.917**	**0.775**	**0.637**	**0.531**	**0.453**	**0.393**
400	**0.913**	**0.775**	**0.637**	**0.531**	**0.453**	**0.393**
500	**0.91**	**0.789**	**0.652**	**0.542**	**0.461**	**0.398**

Tableau II.8 : score de la précision par rapport à ns

Nous observons que la performance augmente, comme le nombre de segments ns augmente, et que la meilleure valeur est d'environ 500.

Algorithme	TOP10	TOP20	TOP30	TOP40	TOP50	TOP60
EJ	0.33	0.59	0.73	0.79	0.83	0.85
IL	0.33	0.59	0.74	0.79	0.84	0.86
JS	0.33	0.59	0.72	0.79	0.83	0.86
LOG	0.3	0.46	0.6	0.66	0.72	0.69
COS	0.33	0.59	0.72	0.78	0.82	0.86
PF	**0.5**	**0.95**	**1.0**	**1.0**	**1.0**	**1.0**

Tableau II.9 : Le rappel moyen de toutes les algorithmes

Algorithme	TOP10	TOP20	TOP30	TOP40	TOP50	TOP60
EJ	0.81	0.64	0.58	0.51	0.42	0.36
IL	0.81	0.64	0.58	0.5	0.42	0.36
JS	0.8	0.65	0.57	0.49	0.41	0.36
LOG	0.73	0.53	0.48	0.42	0.37	0.31
COS	0.81	0.64	0.57	0.48	0.4	0.36
PF	**1.0**	**0.95**	**0.66**	**0.5**	**0.4**	**0.39**

Tableau II.10 : La précision moyenne de toutes les approches

Les tableaux 9 et 10 calculent le rappel et la précision moyenne pour toutes les approches. Selon ces tableaux, on remarque que PF est plus efficace que les autres approches. Les résultats PF sont obtenus par la mise en ns à 500. Par contre, les quartes (04) mesures de similarité {Cos, EJ, IL, JS} ont presque la même performance, et le pire des cas est atteint par l'approche logique.

Tableau 11 calcule la précision exacte ou R-précision pour toutes les approches.

❖ **R-précision :** Une mesure communément utilisée est la précision exacte ou R-précision. Si la requête admet n services pertinents, la précision exacte est la précision calculée à partir des n premiers services de la liste ordonnée des services restitués.

Algorithme	R-prec
PF	**0.95**
IL	0.709
LOG	0.594
JS	0.705
EJ	0.707
COS	0.692

Tableau II.11 : R-prec moyenne pour toutes les approches

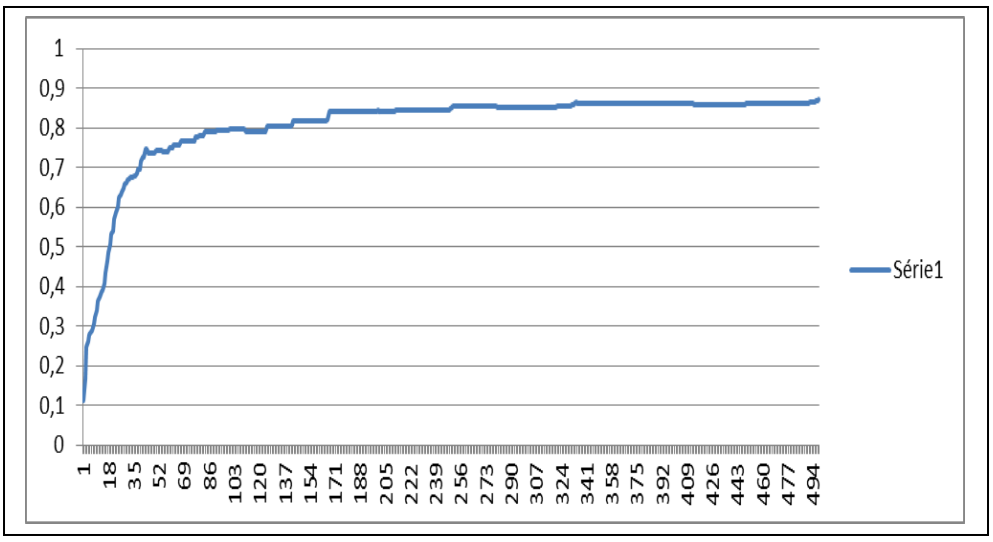

Figure II.11 : Rprec vs ns

L'exécution de NST est représentée dans la figure 10. Elle illustre la relation entre le R-prec_moy et le paramètre ns.

En général, nous discernons trois comportements : tout d'abord, quand ns∈{2…52} nous observons une amélioration rapide de la R-précision estimée, d'autre part, lorsque

ns∈{52…256}, on observe une lente amélioration de la R-précision. Et nous observons une stabilité lorsque ns∈ {257…500}. La valeur optimale est atteinte à environ 500.

VI. Conclusion

Dans ce chapitre, nous avons décrit brièvement la collection de test utilisée dans notre application. Ensuite nous avons présenté la conception en spécifiant l'approche de la fusion probabiliste, qui se base sur l'apprentissage d'un ensemble de probabilité de pertinence.

Nous avons aussi montré dans la partie expérimentation une comparaison entre l'approche proposée et les cinq (05) mesures de similarités les plus fameuses dans l'état de l'art.

Conclusion générale

CONCLUSION GENERALE

Les services Web sont la plus récente technologie adoptée pour l'intégration et l'interopérabilité des systèmes répartis. Ils sont caractérisés par leurs indépendances aux plateformes et aux systèmes d'exploitation, ce qui a impliqué leur adoption par les différentes entreprises et organisations commerciales et industrielles. Ces organisations offrent leurs services à distance via le Web, ce qui augmente le nombre de services offerts (publies). Et par conséquence la tache de découverte de services web devient de plus en plus difficile.

La découverte de services Web constitue un axe de recherche émergeant. Plusieurs approches ont été proposées.

Dans notre travail nous avons présenté une solution basée sur la fusion probabiliste. Cette dernière se base sur l'apprentissage d'un ensemble de probabilité de pertinence. Dans la partie expérimentation, on a fait une comparaison entre l'approche proposée et les cinq(05) mesures de similarité (cosine, extended Jaccard, loss of information, Janson Shannon), les résultats obtenus confirment la supériorité de l'approche proposée.

PERsPECTIVES

- Implémenter la validation croisée pour déterminer la valeur optimale de ns (nombre de segment).
- Implémenter une autre approche de fusion basée sur le concept de dominance probabiliste.
- Etendre le travail pour les versions 3 et 4 de l'owls.tc.

Références bibliographiques

Référence :

[Akkiraju et al, 2005] R. Akkiraju, J. Farrell, J. Miller, M. Nagarajan, M.T. Schmidt, A. Sheth, and K. Verma. Web service semantics - WSDL-S, November 2005.

[Barry, 2003] Barry Douglas K. *The Savvy Manager's Guide to Web Services and ServiceOriented Architectures*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[Bieberstein et al, 2005] N.Bieberstein, S.Bose, M.Fiammante, K.Jones, and R.Shah. Service- Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap, IBM Press, October 19, 2005.

[Booth et al, 2004] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C.Ferris, et D. Orchard. Web services architecture, W3C Working Group Note 11. In http://www.w3.org/TR/ws-arch/ , 2004.

[Chinnici et al, 2007] R.Chinnici, J-J. Moreau, A. Ryman, and S. Weerawarana. Web services description language (wsdl) version 2.0, World Wide Web Consortium, http://www.w3.org/TR/wsdl2.0 . June 2007.

[Clement et al, 2004] A. Clement, C. Hately, V. Riegen, and T. Rogers. Uddi version 3.0.2. http://uddi.org/pubs/uddi_v3.htm , 2004.

[David, 2004] David Booth, Hugo Haas, Francis Mccabe, Eric

Newcomer, Michael Champion, Chris Ferris, and David Orchard. Web services architecture. Technical report, W3C, Web Services Architecture Working Group, February 2004.

[Dodani, 2004] M-H. Dodani. From objects to services. A journey in search of component reuse nirvana. Journal of Object Technology, 2004.

[Downing et, 1998] T. B. Downing and R. M. I. Java. Remote method invocation. Foster City, Calif.: IDG Books Worldwide, 1998.

[Farrell et Lausen, 2007] J. Farrell and H. Lausen. Semantic annotations for wsdl and xml schema. Technical report, W3C Candidate Recommendation, January 2007. http://www.w3.org/TR/2007/ CR-sawsdl-20070126/.

[Gudin et al, 2003] M. Gudgin, M. Hadley, N. Mendelsohn, J.J. Moreau, and H.F. Nielsen. Simple object access protocol (soap) 1.2. World Wide Web Consortium, http://www.w3.org/TR/soap , 2003. July, 23, 1997.

[Hadjila, 2014] Hadjila FethAllah, Composition et interopération des services web sémantiques, thèse de doctorat, UN IVERSITE DE TLEMCEN, 2014.

[Horstmann et Kirtland, 1997] M. Horstmann and M. Kirtland. Dcom architecture. Microsoft Corporation, July, 23, 1997.

[Keller et al, 2004] U. Keller, R. Lara, A.Polleres, I.Toma, M. Kifer, and D. Fensel. Wsmo web service discovery, November 2004.

[Lilllis et al,2006] D. Lillis, F. Toolan, R. Collier, J. Dunnion, ProbFuse: A Probabilistic Approach to Data Fusion, in SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.

[Martin et al, 2004] D. Martin, M. Paolucci, S. Mcilraith, M. Burstein, D. Mcdermott, D. Mcguinness, B. Parsia, T. Payne, M. Sabou, M.Solanki, N. Srinivasan, and K. Sycara. Bringing semantics to web services: The owl-s approach. pages 26-42. Springer, 2004.

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

[OMG, 2008] Object Management Group. Common object request broker architecture (corba/iiop). http://www.omg.org/spec/CORBA/3.1 , January 2008. Une citation à la page 15.

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

[Roman et al, 2005] D. Roman, U.Keller, H.Lausen, J.Bruijn,

R.Lara, Michael Stollberg, A.Polleres, C.Feier, C.Bussler, et D.Fensel. Web service modeling ontology. Appl. Ontol., 1(1) :77-106, 2005.

[Toma et al, 2005] I. Toma, D. Fensel, M. Moran, K. Iqbal, T. Strang, and D. Roman. An Evaluation of Discovery approaches in Grid and Web services Environments. In The 2nd International Conference on Grid Service Engineering and Management, Erfurt, Germany, September 2005.

[VEZAIN, 2005] Arnaud VEZAIN. Les services web - Présentation générale. Technical report, As sociation HERMES, February 2005.

[W3C, 2002 b] W3C. Web services conversation language (WSCL) 1.0, 2002.

[Ye et al, 2009] G.Ye, J.Yue, S.Cheng, and W. Chanle. ―A QoSaware Model for Web Services Discovery. ‖ Proceedings of the 2009 First International Workshop on Education Technology and Computer Science, IEEE. 2009.

Liste des tables

Table des figures

Listes des algorithmes

Résumé

La technologie des services web se répand de plus en plus sur les réseaux à grande échelle. Son ultime objectif est d'assurer l'interopérabilité des applications distribuées, et de créer des services ayant une valeur ajoutée.

Dans notre travail nous proposons une découverte de service web sémantiques à base de fusions probabiliste basée sur cinq (05) algorithmes : Cosinus, Extend Jaccard, Loss of Information, Janson Shannon et Raisonnement logique. Les résultats expérimentaux, montrent que la fusion proposée est très efficace en termes de rappel et précision.

Mots clés : architecture orientée services, découverte de services, fusion probabiliste

Abstract

Web-service technology relies more and more on high speed signal. Its ultimate objective lies in assuring the interoperability of distributed applications, and in creating services according to an added value.

This research work suggests a discovery for semantic web services, based on probabilistic fusions, themselves based on the following five algorithms: Cosinus, Extend Jaccard, Loss of Information, Janson Shannon and Logical Reasoning. Empirical results indicate that the proposed fusion is very efficient regarding recall and precision.

Key words: service-oriented architecture, service discovery, probabilistic fusion

ملخص

تكنولوجيا خدمات الويب ينتشر أكثر وأكثر على الشبكات واسعة النطاق. هدفه النهائي هو لضمان التشغيل البيني للتطبيقات الموزعة، وإنشاء خدمات.

في هدا العمل نقترح الاندماج الاحتمالي الدلالي لاكتشاف خدمة الويب تستند على أساس خمسة خوارزميات، وقد أظهرت النتائج أن عملية الاندماج المقترحة هي فعالة جدا من حيث التذكير والدقة.

الكلمات الرئيسية: ترتيب خدمة الويب، اكتشاف خدمة ويب، دمج البيانات،الانصهار الاحتمالي.