REPUBLIQUE ALGERIENNE DEMOCRATIQUE & POPULAIRE
Ministère de l'Enseignement Supérieur
& de la Recherche Scientifique

Université Djillali Liabès de Sidi Bel Abbes
Faculté de Technologie
Département d'Informatique

Thèse

Présentée Par

Mr. KHATER Maamar

Pour obtenir le diplôme de
Doctorat en Science en Informatique
Option
Informatique

TITRE

La découverte des services web sémantiques

Composition du jury :

Dr. LEHIRECHE Ahmed	**Pr.**	**Université de Sidi Bel Abbes**	**Président**
Dr. MALKI Mimoun	**Pr.**	**Université de Sidi Bel Abbes**	**Encadreur**
Dr. CHIKH Mohamed Amine	**Pr.**	**Université de Tlemcen**	**Examinateur**
Dr. CHOUARFIA Abdallah	**Pr.**	**Université d'Oran USTO**	**Examinateur**
Dr. BENSLIMANE Sidi Mohamed	**Pr.**	**Université de Sidi Bel Abbes**	**Examinateur**
Dr. BOUCHIHA Djelloul	**MCA**	**Centre Universitaire de Naama**	**Examinateur**

2014-2015

Remerciements

الحمد لله الذي بنعمته تتم الصالحات

Tout d'abord, Je remercie Allah qui m'a donné le courage pour accomplir cette tâche, qui au début paraissait une mission difficile.

Je tiens à remercier très chaleureusement Mr Malki Mimoun, mon directeur de thèse, qui m'a guidé pendant plusieurs années. Je le remercie pour toute l'attention qu'il a portée à mon travail, pour sa confiance, son exigence constructive ainsi que son soutien moral.

Je suis très honoré par la présence de Mr Lehireche Ahmed, qui a accepté de présider le jury de ma thèse, je suis également très honoré par la présence de Mr CHIKH Mohamed Amine, Mr CHOUARFIA Abdallah, Mr BENSLIMANE Sidi Mohamed, et Mr BOUCHIHA Djelloul qui ont accepté d'être les examinateurs de cette thèse.

Qu'ils trouvent ici, mes plus vifs remerciements pour l'effort qu'ils ont fait pour lire mon manuscrit et l'intérêt qu'ils ont porté à mon travail.

Je remercie ma famille de m'avoir donné le courage d'accomplir cette thèse.

J'adresse mes sincères remerciements à mes collègues de l'université de Saida ainsi que mes collègues de l'université de SBA pour leurs conseils et soutien continu durant cette thèse.

Dédicaces

الحمد لله الذي هدانا لهذا و ما كنا لنهتدي لو لا أن هدانا الله

A la mémoire de mon père Hadj Mohamed rabi yerehhemou.

A ma chère famille :

Ma mère, mes frères, mes sœurs.

A mon épouse et mes petits enfants : Mohamed Yacine et Oussama Berrezoug.

A tous mes amis.

Qu'ils trouvent ici le témoignage de ma plus profonde reconnaissance.

ملخص

مع عصر الانترنيت و ظهور مفهوم خدمات الويب، تسعى الشركات إلى إدماج هذه الخدمات في أنظمتها المعلوماتية (SI) حيث أصبحت تكنولوجيا خدمات الويب نهجا واعدا في مجال ادماج و تفاعل التطبيقات داخل و بين الشركات.

في هذا السياق أصبحت كفاءة و حسن أداء الأنظمة المعلوماتية المبنية على أساس خدمات الويب عاملا حاسما في نجاح أعمال هذه الشركات.

خدمات الويب الدلالي هي التقاء بين اثنين من مجالات البحث الهامة و التي تخص تقنيات الانترنيت ألا و هي خدمات الويب و الويب الدلالي.

الغرض من مفهوم الويب الدلالي هو انشاء ويب دلالي للخدمات حيث يتم وصف دقيق للخصائص، القدرات، الواجهات و كذا الآثار و ذلك بصفة لا لبس فيها و قابلة للاستخدام من قبل أجهزة الكمبيوتر.

تناقش هذه الأطروحة مشكل علمي معروف في مجال خدمة الويب الدلالي، ألا و هي الاكتشاف الديناميكي لخدمات الويب الدلالي.

تعالج المساهمة الأولى في هذه الأطروحة عملية الاكتشاف الديناميكي لخدمات الويب الدلالي OWL-S الأولية و المشكلة من خدمة واحدة فقط و ذلك باستعمال خوارزمية مطابقة تعتمد على تحويل مصفوفة المطابقة إلى رسم بياني و استعمال كذلك لخوارزمية أقصر طريق.

تعالج المساهمة الثانية عملية الاكتشاف الديناميكي لخدمات الويب الدلالي المركبة و ذلك اعتمادا على مطابقة الأنماط للرسوم البيانية.

تعالج المساهمة الثالثة عملية الاكتشاف الديناميكي لخدمات الويب الدلالي المركبة و ذلك اعتمادا على مطابقة الأنماط للرسوم الآلية (WTIA).

الكلمات المفتاحية : هندسة الخدمات، خدمات الويب الدلالي، تلائم مطابقة، قياس التشابه، أنطولوجيات، أقصر الطرق، الرسوم البيانية، الرسوم الآلية،OWL-S

Résumé

Avec l'air du web et l'apparition de la notion de services Web, les entreprises se sont converties à intégrer les services web dans leurs systèmes d'information (SI). La technologie des services Web est en train de devenir une approche prometteuse d'intégration et d'interaction des applications intra et inter organisations. Dans ce contexte, l'efficacité et la fiabilité des SI à base de services Web deviennent un facteur critique dans la réussite des entreprises. Les services Web sémantiques se situent à la convergence de deux domaines de recherche importants qui concernent les technologies de l'Internet : le Web sémantique et les Web services. L'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines.

Cette thèse est consacrée au problème de la découverte dynamique des services web sémantiques. Notre première contribution traite la découverte des services OWL-S atomiques et consiste à proposer un algorithme de matchmaking basé sur la transformation de la matrice de matching (input/output) en un graphe orienté en tenant compte des règles de mapping, et sur le principe du plus court chemin. Notre seconde contribution traite la découverte des services OWL-S composites et consiste à proposer un algorithme de matchmaking à base de comportement en utilisant les graphes. Notre troisième contribution propose un modèle comportemental à base d'automates (les automates d'interface typés et pondérés –WTIA–). Ce modèle est le résultat d'émergence de deux modèles d'automates, les automates d'interface et les automates pondérés. Notre quatrième contribution consiste à utiliser le formalisme WTIA comme modèle comportemental de services sémantiques afin de réaliser le processus de matchmaking entre la requête de l'utilisateur et les services publiés.

Mots clés : architecture orientée service, services web sémantiques, matchmaking, mesures de similarité, ontologies, OWL-S, plus court chemin, graphes, automates.

Abstract

With the air of the web and the emergence of the Web services concept of, companies are converted to integrate web services into their information systems (IS). The Web services technology is becoming a promising approach of integration and interaction of applications within and across organizations. In this context, the effectiveness and reliability of IS-based Web services become a critical factor in business success. Semantic Web Services are located at the convergence of two major research areas that concern Internet technologies: the Semantic Web and Web services. The purpose of the concept of semantic web services is to create a Semantic Web services whose properties, capabilities, interfaces, and the effects are described unambiguously and used by machines.

This thesis is dedicated to the dynamic discovery problem of semantic web services. Our first contribution deals with the atomic OWL-S services discovery and provides a matchmaking algorithm based on the transformation of the matching matrix (input/output) into a directed graph which takes into account the mapping rules, and the principle of the shortest path. Our second contribution discusses the composite OWL-S services discovery and provides a matchmaking algorithm based on behavior by using the graphs. Our third contribution proposes a behavioral model based on automata (weighted and typed interface automata - WTIA-). This model is the result of the emergence of two models of automata, interface automata and weighted automata. Our fourth contribution is to use the WTIA formalism as behavioral model of semantic services to achieve the matchmaking process between the user request and published services.

Keywords: service-oriented architecture, semantic web services, matchmaking, similarity measures, ontologies, OWL-S, shortest path, graphs, automata.

Table des Matières

Table des figures

Liste des tableaux

Liste des algorithmes

Glossaire des acronymes

SI : Systèmes d'Information

DAML-S : Darpa Agent Markup Language for Services

OWLS : Ontology Web Language for Service

IOPE: Input/Output/Precondition/Effect

WSMO: Web Service Modeling Ontology

WSML: Web Service Modelling Language

SAWSDL: Semantic Annotation for WSDL

REST: REpresentational State Transfer

SA-REST: Semantic Annotation for REST

WSDL-S: WSDL Semantic

USDL: Universal Service Semantics Description Language

SOA: Service Oriented Architecture

WS: Web Service

SWS: Services Web Semantiques

URI: Uniform Resource Identifier

URL: Uniform Resource Locator

HTML: Hypertext Markup Language

FTP: File Transfer Protocol

XML: eXtensible Markup Language

HTTP: HyperText Transfer Protocol

UDDI: Universal Description, Discovery and Integration

WSDL: Web Service Description Language

SOAP: Simple Object Access Protocol

WTIA : Automates d'Interface Typés et Pondérés

RPC: Remote Procedure Call

CORBA: Common Object Request Broker Architecture

J2EE: Java 2 Enterprise Edition

CGI: Common Gateway Interface

PHP : Hypertext Preprocessor

ASP : Active Server Pages

JSP : Java Server Pages

EJB: Enterprise JavaBeans

W3C: World Wide Web Consortium

IBM: International Business Machines

TCP/IP: Transmission Control Protocol/Internet Protocol

QoS: Quality of Service

BPEL4WS: Business Process Execution Language for Web Services

XLANG: XML LANGuage

WSFL: Web Services Flow Language

WSCL: Web Services Conversation Language

BPML: Business Process Modeling Language

SMTP: Simple Mail Transfer Protocol

MIME: Multipurpose Internet Mail Extensions

MOM: Message-oriented middleware

JMS: Java Message Service

CCS : Communicating sequential processes

ACP : Algebra of Communicating Processes

CSP : Communicating Sequential Processes

LOTOS : Language Of Temporal Ordering Specification

PROMELA : PROtocol MEta LAnguage

CSP : Constraint Satisfaction Problem

RDF : *Resource Description Framework*

RDQL: RDF Data Query Language

iRDQL: Imprecise RDQL

TF/IDF: Term Frequency-Inverse Document Frequency

BP-QL: Business Process Query Language

GED : Graph Edit Distance

OWLS-SP: OWL-S – Shortest Path

OWL-S API : OWL-S Java API

OWL API: OWL Java API

OWLS-TC : Service Retrieval Test Collection

Chapitre I :

Introduction Générale

Sommaire

1. Cadre du travail

A. **Cadre général:** un framework de découverte multi-stratégies pour les services web sémantiques.

B. **Cadre précis:** mise en œuvre des approches de matchmaking des services web sémantiques atomiques et composites basées sur OWL-S.

2. Contexte

L'intégration de l'informatique dans les entreprises a permis d'améliorer la circulation et le traitement des informations en utilisant des systèmes d'information -noté SI-.

Avec l'air du web et l'apparition de la notion de services Web, les entreprises sont convertis à intégrer les services web dans leurs SI. La technologie des services Web est en train de devenir une approche prometteuse d'intégration et d'interaction des applications intra et inter organisations. Dans ce contexte, l'efficacité et la fiabilité des SI à base de services Web deviennent un facteur critique dans la réussite des entreprises.

Les services Web sémantiques se situent à la convergence de deux domaines de recherche importants qui concernent les technologies de l'Internet : le Web sémantique et les Web services.

L'expression « web sémantique », due à Tim Berners-lee [berners-Lee et al., 2001] au sein du W3C, fait d'abord référence à la vision du web de demain comme un vaste espace d'échange de ressources entre êtres humains et machines permettant une exploitation, qualitativement supérieure, de grands volumes d'informations et de services variés. Le web sémantique s'intéresse principalement aux informations statiques disponibles sur le web et les moyens de les décrire de manière intelligible pour les machines. Cependant, la tâche principale des services web est d'assurer l'interopérabilité entre les applications via le web en vue de rendre le web plus dynamique.

L'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines.

Les problèmes classiques connus dans le domaine des services web sont imposés implicitement dans la conception et la mise en œuvre d'un tel système, à savoir: la description de service Web, le processus de publication, le processus de découverte, le processus de composition, le processus de sélection, la substitution, et le processus d'adaptation.

Différentes approches ont été proposées dans la littérature pour répondre à ces besoins, ces approches diffèrent dans le langage et le modèle de description utilisé, on trouve des

approches ontologiques qui s'appuient sur la description sémantique du service, et les approches comportementales qui reposent essentiellement sur la description fonctionnelle du service.

Le besoin d'automatisation du processus de conception et de mise en œuvre des services web sémantiques rejoint les préoccupations à l'origine du web sémantique, à savoir comment décrire formellement les connaissances de manière à les rendre exploitables par des machines. Parmi les solutions proposées de la problématique de la description sémantiques des services, OWL-S se présente comme l'ontologie qui détient une large utilisation dans ce domaine.

Nos travaux de recherche s'intéressent aux services web sémantiques à base OWL-S, et visent à proposer des solutions pour améliorer les performances du processus de matchmaking des services atomiques et composites.

3. Problématique :

Le mécanisme de matching fourni par UDDI et WSDL ne peut servir l'objectif de la découverte dynamique des services Web, le principe de ce mécanisme est basé sur la forme syntaxique de services Web et peut produire par conséquent de nombreux faux positifs et faux négatifs, ce qui réduit énormément les performances du système de découverte.

Pour surmonter ces limitations, le concept de la sémantique doit être introduit dans le processus de matching par l'utilisation d'ontologies.

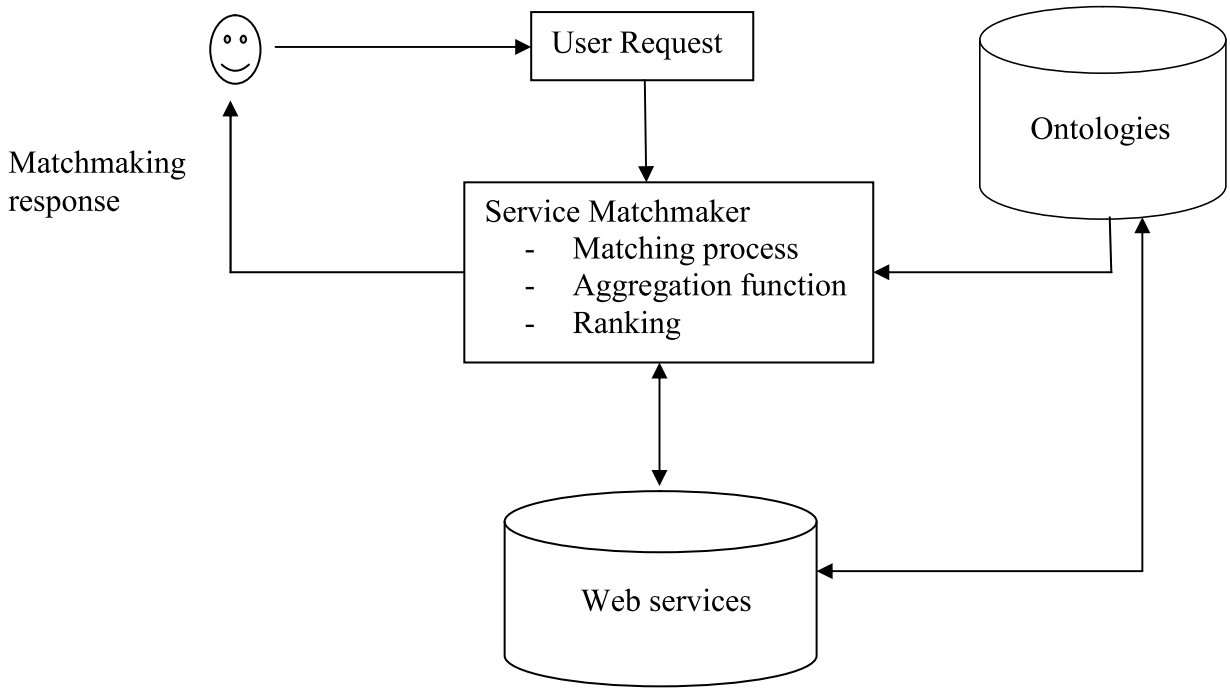


Figure I.1 Matchmaking de services web

La découverte de services sémantiques est un domaine de recherche très actif, et elle est abordée dans un grand nombre de travaux de recherches.

Le matchmaking peut être équivalent au processus de découverte, mais selon d'autres chercheurs la découverte regroupe le matchmaking ainsi que la sélection et l'invocation. Nous considérons que le matchmaking est le processus de trouver des services web pertinents en se basant sur l'évaluation et l'agrégation des résultats des opérations de matching. Les résultats retournés doivent être rangés et classés par ordre de mérite.

Les systèmes de découverte de services web sémantiques peuvent être classés en trois familles selon le mode de raisonnement utilisé par le matchmaker (logique, non-logique, hybride). [Chebab et al., 2011] remarque que les approches de matching les plus citées et les plus élaborées sont celles basées sur la logique pure ou hybride.

4. Objectifs de la thèse:

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 (**black-box** matching), et éventuellement des descriptions sur le fonctionnement du service (**glass-box** matching). Le matchmaker doit créer des mécanismes qui comparent ces besoins avec l'ensemble des services publiés, et de retourner des résultats performants en termes de rappel, de précision et de temps d'exécution.

Deux sous-problématiques ressortent de cette formalisation de la problématique de la découverte :

A. Découverte des services web sémantiques atomiques :

Le matching est basé seulement sur les propriétés (inputs/outputs) entre la requête et les services publiés. L'approche de référence pour cette catégorie de matching est celle proposée par [Paolucci et al., 2002]. L'expérimentation de cette approche a montré que les résultats du matchmaking sont moins performants dans certaines situations (l'ordre de concepts lors de l'opération de matching, retrait (ou non) du service après son matching).

Le premier scenario: sans suppression des concepts du service publié après le matching.

Advertise 'A'

Input	publisher
Output	novel, price

Query 'Q'

Input	publisher
Output	romantic novel, science-fiction novel

liste-candidats={novel, price}

dom(romantic novel, novel)=exact=1.

dom(romantic novel, price)=fail=0.

dom(science-fiction novel, novel)=exact=1.

dom(science-fiction novel, price)=fail=0.

Gdom=exact.

Le matchmaker retourne 'A' comme réponse correcte à la requête 'Q' avec un degré de correspondance « exacte », tandis que 'A' représente un faux positif en réalité (en général, c'est le cas où on trouve deux concepts ou plus de la requête qui sont en correspondance avec un seul concept du service 'A').

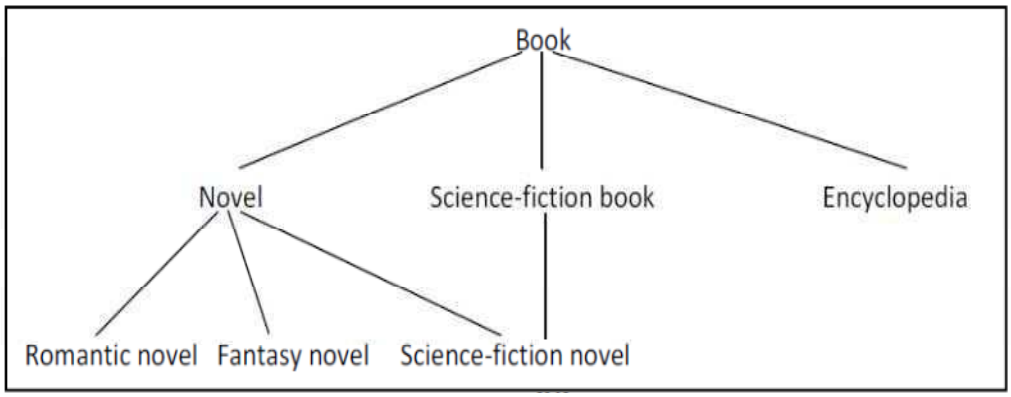


Figure I.2: Partie de book ontology[1]

Le deuxième scenario: avec suppression des concepts du service publié après le matching.

Advertise 'A'

Input	publisher
Output	novel, science-fiction book

Query 'Q'

Input	publisher
Output	science-fiction novel, romantic novel

list-candidat={ novel, science-fiction book }

dom(science-fiction novel, novel)=exact=1.

dom(science-fiction novel, science-fiction book)=1= exact.

Le concept 'Science-fiction novel' correspond au concept 'novel', et 'novel' est retiré de la liste des concepts candidats.

list-candidat={ science-fiction book }

dom(romantic novel, science-fiction book)=fail=0.

[1] OWL-S Service Retrieval Test Collection. http://projects.semwebcentral.org/projects/owls-tc/

Le matchmaker ne retient pas le service ‘A’ comme bonne réponse à la requête ‘Q’ car il a trouvé un échec dans le processus de matching.
On remarque que ‘A’ représente un faux négatif (l’ordre des concepts de la requête influence sur le degré de correspondance et peut changer par conséquence le Gdom).

Nous intéressons à améliorer le processus de matchmaking proposé par l’approche de [Paolucci et al., 2002].

B. Découverte des services web sémantiques composites :

Nous croyons que le matching basé uniquement sur le profil de service (inputs/outputs) ne suffit pas dans le cas des services composites où il est possible que certaines sorties ne sont produites que dans des conditions internes spécifiques.
Le Matching à base I/O ne garantit pas que les services composites retournés présentent le même ordre de messages échangés tel qu’il est décrit par le modèle comportemental de la requête.
Dans cette section, nous présentons les scénarios nécessitant un matchmaking comportemental.

[J.C Corrales et al., 2008] présente dans sa thèse de doctorat deux exemples où il est nécessaire d’effectuer un matchmaking comportemental pour trouver des services qui peuvent satisfaire le besoin de l’utilisateur. Les résultats retournés et en comparant avec un matchmaking input/output sont plus performants et minimisent le coût de développement pour l’utilisateur ainsi que les problèmes d’invocation (sans passer par une étape d’adaptation entre le service trouvé et la requête).

Le premier scénario expose la problématique de découverte des services dans un contexte d'intégration d’applications, il consiste à retrouver les services ayant un comportement compatible. Le second scénario montre qu’une approche de matchmaking comportemental qui peut être utilisée comme un moyen pour quantifier des similarités/dissimilarités entre deux modèles, non seulement dans le contexte de la découverte des services, mais également dans d'autres applications, comme le « delta-analyse ».

Le premier scenario « Intégration des services web »: On considère une entreprise qui utilise un service S pour commander des fournitures. Supposons que l'entreprise veut trouver des partenaires ayant des services web compatibles. Les séquences d'échange de messages acceptés sont appelés protocoles de conversation. La spécification du protocole de conversation est importante, car il arrive rarement que les opérations de service peuvent être invoquées indépendamment les uns des autres. Ainsi, l’entreprise recherche un service ayant

un protocole de conversation compatible. Après trouver de partenaires, le plus compatible doit être choisi parmi eux. Si le service n'est pas compatible, l'entreprise soit adapte son service ou bien développe un adaptateur pour que son application soit conforme au comportement du service trouvé. Dans les deux situations, des modèles de processus doivent être comparés automatiquement afin de déterminer leurs similitudes ou différences en termes de leurs protocoles de conversation.

La recherche des services en tenant compte que des entrées/sorties peuvent être utilisés seulement comme un premier filtre parce qu'elle ne garantit pas que les services trouvés prennent en charge le même ordre de messages échangés.

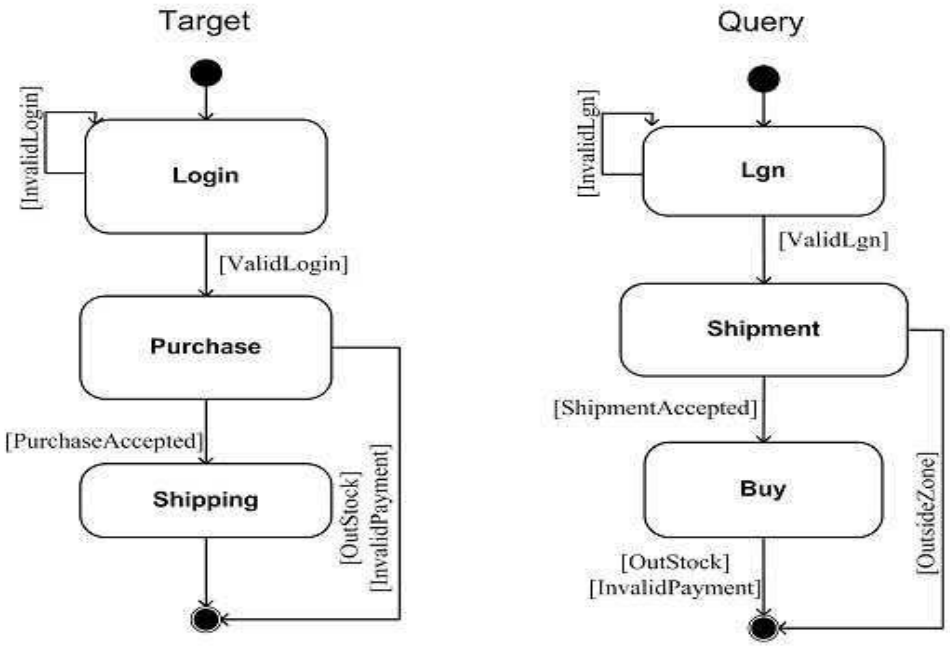


Figure I.3 L'ordonnancement des opérations dans la requête et le service cible [J.C Corrales et al., 2008]

Le deuxième scenario « Delta analyse » *:* le domaine d'application pour ce scénario consiste à déterminer les différences entre les deux modèles. Parmi les défis de l'entreprise est de vérifier si le processus mis en œuvre conforme à une norme. Ainsi, ils ont besoin de comparer le modèle de conversation de leur service existant avec celle prescrite par les normes. Pour les grands modèles et surtout lorsque l'entreprise n'utilise pas le vocabulaire standard, un outil devrait aider les utilisateurs à identifier toutes les différences entre les deux modèles. Sur la base de ces différences, le coût de la réingénierie du service existant pourrait être mieux évalué.

Le matchmaker des services composites doit prendre en charge les propriétés comportementales de ces services lors du processus de matching. Nous intéressons à proposer des solutions qui prennent en charge ces préoccupations.

5. Contributions

Notre première contribution traite la découverte des services OWL-S atomiques et consiste à proposer un algorithme de matchmaking basé sur la transformation de la matrice de matching (input/output) en un graphe orienté en tenant compte des règles de mapping, et sur le principe du plus court chemin. La complexité de notre algorithme proposé est linéaire (d'ordre $O(n^3)$). La validation et l'analyse des performances de notre approche sont basées sur deux indicateurs d'efficacité: la précision et le rappel. La partie expérimentale à montré que notre approche a donné de meilleurs résultats que le matchmaking de Paolucci. Cette approche a permis de développé l'outil «OWLS-SP Discovery » [khater and Malki, 2014].

Notre deuxième contribution traite la découverte des services OWL-S composites et consiste à proposer un algorithme de matchmaking à base de comportement. Le model de processus OWL-S, de la requête et du service publié, est représenté sous forme de graphe orienté où les nœuds sont des services atomiques.

L'objectif est de chercher une copie d'un graphe motif dans un graphe cible (la définition d'isomorphisme de sous-graphe) en se basant sur un raffinage successive d'une fonction de mapping entre la requête et les services publiés.

Notre troisième contribution propose un modèle comportemental à base d'automates (les automates d'interface typés et pondérés –WTIA–). Ce modèle est le résultat d'émergence de deux modèles d'automates, les automates d'interface et les automates pondérés [khater and Malki, à apparaître 2015].

Notre quatrième contribution consiste à utiliser le formalisme WTIA comme modèle comportemental de services sémantiques afin de réaliser le processus de matchmaking entre la requête de l'utilisateur et les services publiés en se basant sur deux relations de similarité: la première qui est de type sémantique entre services, elle est mesurée par l'algorithme de matchmaking de l'approche OWLS-SP, et la seconde est de type comportemental à base de la trace d'exécution du modèle WTIA avec l'utilisation de la distance de Levenshtein [khater and Malki, à apparaître 2015].

6. Organisation du manuscrit

Le manuscrit est composé de manière générale d'une introduction générale, de quatre chapitres et d'une conclusion générale. Il est constitué de deux parties principales :

La première est intitulée « Background et Etat de l'art » qui introduit le contexte et les concepts de base dans lequel se situe cette thèse et présente un état de l'art de la

problématique traitée. La seconde partie est intitulée «Contributions», elle présente les approches proposées pour répondre aux objectifs de cette thèse.

- **Première Partie : « Background et Etat de l'Art»**

Cette partie est organisée en 02 chapitres :

Le chapitre deux « les services web » introduit les principes du paradigme SOA, ainsi que les protocoles, les langages et les modèles qui sont en relation avec la technologie des services web syntaxiques et sémantiques.

Le chapitre trois « la découverte des services web sémantiques » définit formellement la problématique de recherche sémantique de services, présente l'état de l'art sur les systèmes de matchmaking ainsi que les avantages et les inconvénients de chaque catégorie d'approches.

- **Deuxième Partie : « Contributions»**

Cette partie est organisée en 02 chapitres :

Le chapitre quatre « Approche de matchmaking OWLS-SP » présente notre première contribution. Nous commençons par l'évaluation de deux approches logiques ([Paolucci et al., 2002] et [Bellur et al., 2007]). Puis nous présentons notre approche OWLS-Shortest Path avec une partie d'implémentation et d'expérimentation à la fin.

Le chapitre cinq « Matchmaking comportemental » présente les principes du matching comportemental ainsi que les méthodes formelles utilisées en littérature.

La première section est consacrée à présenter notre deuxième contribution « matchmaking comportemental à base d'isomorphisme de sous-graphe ».

La deuxième section propose un formalisme comportemental (WTIA) à base de fusion de deux modèles d'automates qui sont les automates d'interface et les automates pondérés par des entiers.

La dernière section de ce chapitre présente l'application du formalisme WTIA comme modèle comportemental des services web sémantiques.

Enfin, le manuscrit se termine par une conclusion générale qui récapitule les travaux réalisés dans le cadre de cette thèse et propose quelques visions pour nos travaux futurs.

Partie I :

Background et Etat de l'Art

Chapitre II:

Les services Web

Sommaire

1. Introduction

Une approche services web du système d'information de l'entreprise vise à transformer chaque composant, mainframe, bases de données, application métier, application bureautique, en nœud s'exposant sur des standards de l'Internet qui peut consumer ou fournir des services web. Cette approche apporte une réponse aux problèmes que rencontrent actuellement les entreprises au niveau de la réutilisabilité, de l'interopérabilité et de la réduction du couplage entre les composants constituant leurs systèmes d'information.

2. L'architecture SOA :

Dans la littérature, on trouve plusieurs définitions pour les architectures orientés services, parmi les principales celle du W3C : « une architecture orientée service est un ensemble de composants pouvant être invoqués, et dont les descriptions d'interfaces peuvent être publiées et découvertes ». En général, l'architecture des services web s'articule autour de trois acteurs, service provider, service requester et repository (voir la figure II.1).

- Service provider : c'est le fournisseur du service, il correspond au propriétaire du service,
- Service requester : c'est le client, il correspond au demandeur du service, il est constitué par l'application qui va rechercher et invoquer un service (l'application cliente peut être elle-même un service web),
- Repository : c'est l'annuaire des services, il correspond à un registre de descriptions de services offrant des facilités de publication de services à l'intention des fournisseurs ainsi que des facilités de recherche de services à l'intention des clients.

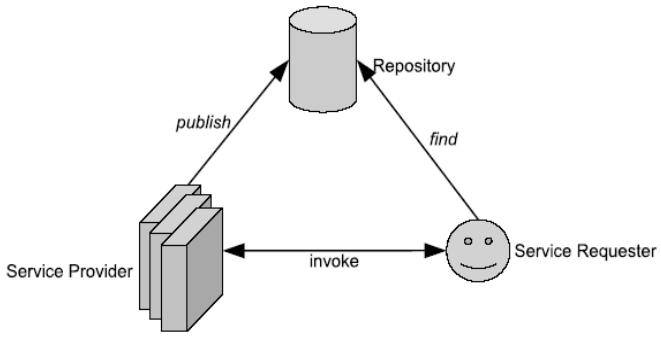


Figure II.1: Web service architecture [Fensel D. et al., 2007].

3. Les services web

Les services web présentent une nouvelle vision de la mise à disposition de composants logiciels pour les systèmes d'information. Contrairement aux technologies classiques des middlewares telles RPC, CORBA, ou J2EE qui emploient des infrastructures logicielles

conçues sur mesure et fortement couplés, les web services prennent le pari d'employer des standards généralistes, fortement répandus et peu coûteux tels XML ou HTTP.

Les services web sont considérés comme étant l'évolution naturelle du web, où le web est devenu un fournisseur de services.

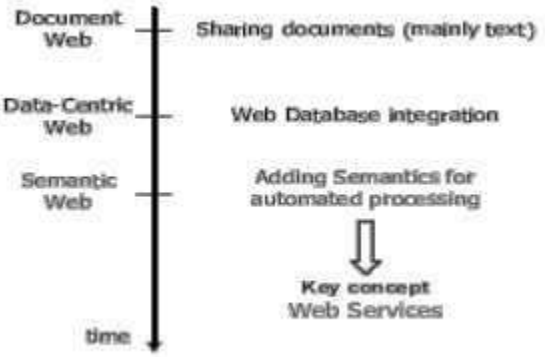


Figure II.2 : Evolution du Web. [Ouazzani M., 2004]

Le web a connu, suivant le perspectif de degré d'automatisation, deux phases de développement :

- Le web statique, le phénomène Internet original, utilisé principalement pour publier des informations, documents, produits, etc. les sites web se basent sur le Html et les liens hypertextes.
- Le web dynamique, c'est le web applicatif où les sites Internet sont devenus plus interactifs, et plus complexes, gérés par des serveurs d'application, dans cette phase, les entreprises ont commencé à utiliser le web à des fines commerciales. Les sites web se basent sur des langages tel que : CGI, PHP, ASP, JSP, EJB, etc.

L'objectif ultime de l'approche services web est d'augmenter le dynamisme et le degré d'automatisation du web où le web devient un fournisseur de services.

On peut donner d'autres classifications en tenant compte d'autres aspects tel que, par exemple, le degré sémantique où on peut distinguer deux générations : le web syntaxique et le web sémantique.

La définition la plus connue pour un service web est : "la notion de service web désigne essentiellement une application (un programme) mise à disposition sur Internet par un fournisseur de service, et accessible par les clients à travers des protocoles Internet standards" [Cassati F., et al., 2001].

Selon IBM[1] "Web services are a new breed of web applications. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the

[1] IBM Web services tutorial. Online: http://www-106.ibm.com/developerworks/webservices/.

Web. Web services perform functions that can be anything from simple requests to complicated business processes".

Cette définition affirme que les services web sont accessibles par d'autres à travers le web, en utilisant des protocoles et des formats standards.

Selon le W3C[2]

"A Web service is a software system identified by a URI and designed to support interoperable machine-to-machine interaction over a network. It has an interface defined and described in a machine-processable format (WSDL). Its definition can be discovered by other software systems. Other systems may then interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards".

Cette définition cite les caractéristiques les plus importantes d'un service web à savoir l'identifiant (URI), l'interface (WSDL), et l'indépendance par rapport aux plateformes.

La figure II.3 schématise les composantes d'un service web syntaxiques et leurs interactions de base, à savoir : la publication, la découverte, et l'invocation.

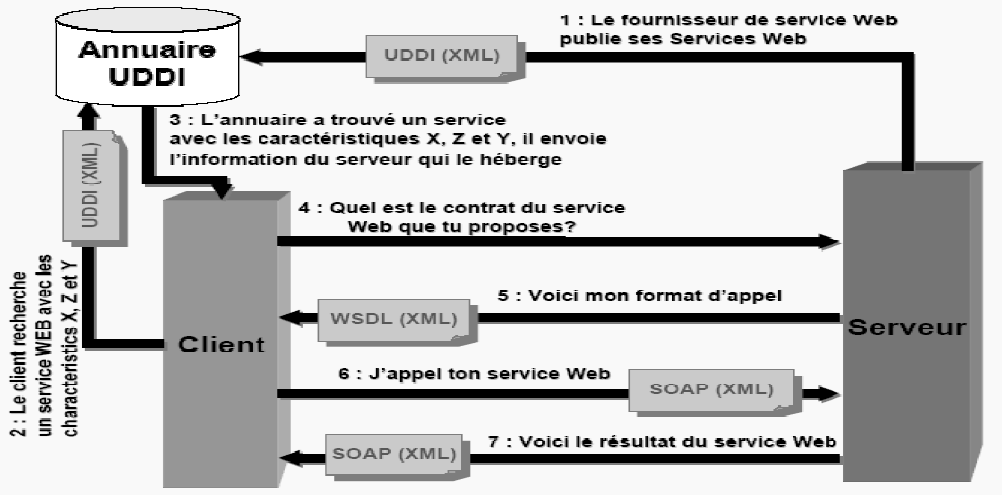


Figure II.3 Architecture de service web syntaxique.

Le consortium W3C propose une architecture standard de service web qui est constituée de plusieurs couches se superposant les unes sur les autres et s'appuyant sur un socle TCP/IP, d'où le nom de pile des services web, voir la figure II.4. La pile est constituée de plusieurs couches où chaque couche s'appui sur un standard particulier. Cette pile présente deux types de couches :

[2] www.w3c.org

- Les couches dites transversales (sécurité, administration, transactions et qualité de services (QoS)) rendent viable l'utilisation effective des web services dans le monde industriel.
- Une couche processus métier - Business process- permet l'utilisation effective des web services dans le domaine du e-business.

On trouve, au-dessus de la couche de transport, les trois couches formant l'infrastructure de base de service web, ces couches s'appuient sur les standards SOAP, WSDL et UDDI.

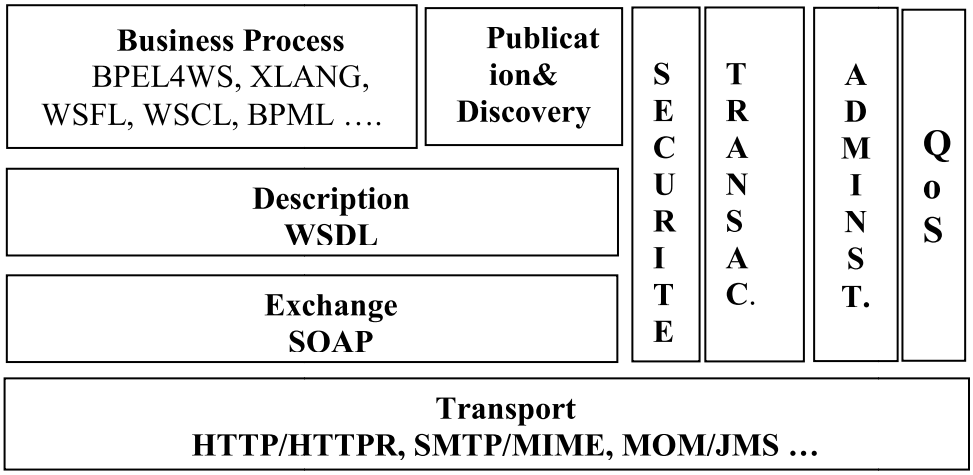


Figure II.4 : Pile des web services

3.1 SOAP :

SOAP (Simple Object Access Protocol) est un protocole d'échange de messages indépendant des plateformes, c'est une recommandation W3C (Word Wide Web Consortium) depuis 2000. Il est constitué de deux parties : une enveloppe XML, et un entête d'un protocole de transport. L'enveloppe XML « Envelope » contient à son tour deux sous éléments : un entête « «header » et un corps appelé body. La figure II.5 présente la structure d'un message SOAP.

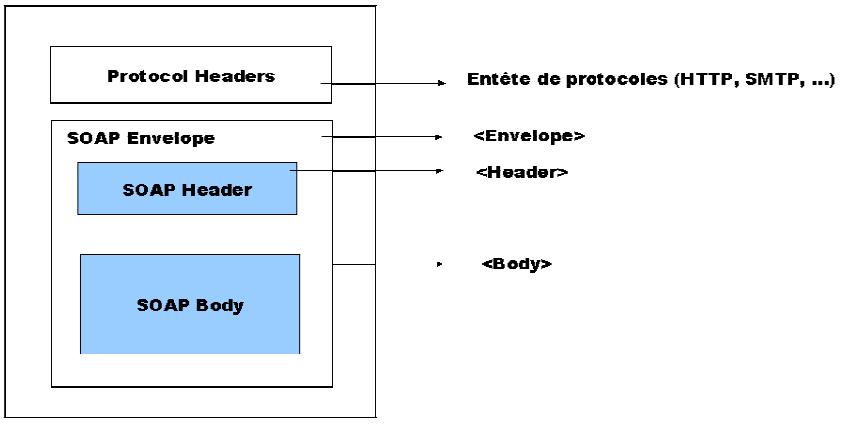


Figure II.5 : Structure d'un message SOAP.[http://www.w3.org/TR/2007/REC-soap12-part0-20070427/]

- ***Un en-tête:*** il débute par la balise <Header> et doit, s'il existe, se situer en première position dans le message SOAP. Il peut contenir des informations utiles mais non nécessaires au service.
- ***Une envelope :*** c'est la racine du document XML; elle débute obligatoirement par la balise <Enveloppe>. Chaque attribut de cette balise doit être associé à un XML namespace afin d'éviter toute ambiguïté possible.
  - ***Un corps:*** Il est indiqué par la balise <BODY> et il contient:

    • soit le nom de méthode, avec les données correspondantes, ou un simple document XML, pour le cas d'une requête.

    • les valeurs de retour ou un message d'erreur pour le cas d'une réponse.

SOAP est indépendant de tout protocole de transport mais il est bien souvent associé à HTTP. Dans ce cas, les requêtes et les réponses qui transitent les messages SOAP sur le réseau auront un en-tête supplémentaire propre à la couche HTTP.

```
POST /StockQuote HTTP/1.1
Hôte : www.stockquoteserver.com
Type de contenu : text/xml; charset="utf-8"
Contenu-Longueur : nnnn
SOAPAction: "urn:stock-quote-services"
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
       <m:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>BORL</symbol>
      </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figure II.6: Message SOAP

3.2 WSDL : langage de description

Le « Web Service Description Language » (WSDL) est un standard du W3C qui permet de définir la structure abstraite et concrète d'un service.

Un document WSDL, écrit en XML, décrit un service Web et la manière de l'appeler. Il expose la signature de méthode du service Web, le protocole à utiliser, l'adresse réseau et le format des données.

Le document WSDL comporte 05 sortes d'éléments XML : <types>, <message>, <portType>, <binding> et <service>.

```
<?xml version="1.0" encoding="utf-8"?>
<definitions>
<types>!--abstract data types</types>
<message>!--message structure</message>
<portType>!--Web Service Interface</portType>
<binding>!--how the service is accessed</binding>
<service>!--who provides the service</service>
</definitions>
```

Figure II.7: Les éléments de l'interface WSDL.

<types> : il définit les types des messages échangés par le service, en utilisant XML schéma ces types peuvent être simples (entiers, réels, chaines de caractères) ou complexes (tableaux, enregistrements,….).

<message> : cet élément définit les messages échangés par le service (entrées ou sorties). Il est composé d'un ou plusieurs éléments nommés <part>. Un <part> est associé à une donnée décrite dans <types>.

<portType> : il représente une collection d'opérations. Une <operation> est une action abstraite accomplie par le service. Elle contient éventuellement un sous élément <input> décrivant le message d'entrée, et un sous élément <output> décrivant le message de sortie.

<binding> : cet élément associe un portType avec des informations concrètes telles que le protocole de transport (HTTP, FTP…), les règles d'encodage de données, et le style du service (RPC ou DOC), le style RPC impose un format précis aux messages SOAP (i.e. le nom de la méthode et les arguments pour la requête, et la valeur de retour pour la réponse).ce style est adapté aux échanges synchrones.

Le style DOC autorise n'importe quel document XML bien formé, comme requête ou réponse, ce style est adapté aux échanges asynchrones.

<service>, cet élément est constitué d'un ensemble de <port>.

Un <port> est une association entre un <binding> et un point d'accès i.e. l'URL qui indique l'adresse du service web. Nous pouvons avoir plusieurs ports par services, (un même binding et des URLs différents, ou des « binding» différents et éventuellement des URLs différents).

La version 1.0 de l'interface WSDL a été créée en 2000 par IBM, Microsoft et Ariba, ensuite le consortium W3C a publié la version WSDL 1.1 en mars 2001.

La spécification WSDL 1.2 est une simplification de la version WSDL 1.1, mais elle n'est pas adoptée comme une norme par le W3C. En juin 2007, la version WSDL 2.0 a été créée comme une recommandation W3C (i.e. norme). WSDL 2.0 a apporté quelques changements à la version WSDL 1.2, ils sont résumés comme suit:

<portType> est devenu <interface>

<port> est remplacé par <endpoints>

<message> est supprimé et l'élément <xs :element> est utilisé pour spécifier les données échangées.

3.3 Un annuaire des services : UDDI

UDDI -Universal Description, Discovery and Integration- normalisé par Oasis offre un mécanisme pour localiser, décrire et recenser les services web dans un registre. Grâce à ce registre, les industries peuvent trouver des partenaires professionnels et se connecter dynamiquement à leurs services Web ainsi que publier des services.

Un registre UDDI ne contient pas les spécifications réelles de la manière dont les entreprises réalisent électroniquement leurs opérations pour partager leurs produits et leurs services, il contient plutôt des pointeurs ou des références menant à ces informations. Certains de ces pointeurs peuvent être des fichiers WSDL décrivant un service Web.

UDDI accepte et organise les trois types d'informations professionnelles suivants :

- Publication - "pages blanches" contenant des informations professionnelles telles que contacts et adresses.
- Localisation - "pages jaunes" contenant des informations de services professionnels, classées par catégories industrielles et localisations géographiques.
- Liaison - "pages vertes" contenant des informations techniques et des spécifications des services (l'accès au service par exemple).

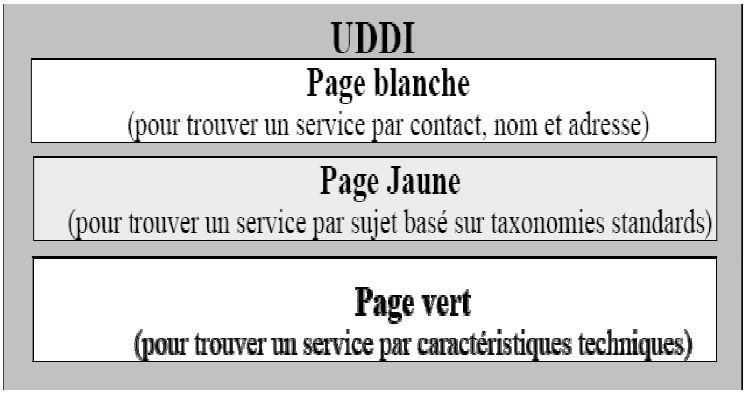


Figure II.8: L'annuaire UDDI

3.4 Lisibilité d'un service :

Il existe plusieurs possibilités pour décrire un service web en tenant compte de sa lisibilité (accessibilité) vers l'extérieur, un service web peut être vu comme une :

- Boite noire (black-box), dont ne sont visibles que les entrées et les sorties.

- Boite semi-transparente (grey-box), une partie seulement de la structure interne est accessible.
- Boite blanche (glass-box), dite transparente, l'ensemble du service web est accessible.

Cette classification est très importante pour effectuer la tâche de la découverte des services web. L'utilité et l'efficacité de l'approche des services web deviennent énormes et importantes dans l'utilisation en groupe de services web reliés entre eux et non pas dans l'utilisation des services web isolés. Pour cela on préfère parler sur la notion de « application à service ».

Une application à services peut être vue comme une collection de services qui interagissent et communiquent entre eux afin de fournir un service de plus haut niveau (business process).

3.5 Cycle de vie des services web :

La figure II.9 présente les états possibles pour un service web, dont les nœuds sont:

- Etat passif « passive »: le service web est crée et peut être publie dans un annuaire de service (UDDI),
- Etat actif « active »: le service web est en train d'être utilisé par un ou plusieurs clients.

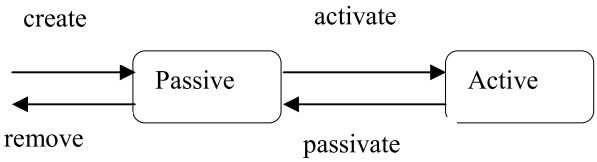


Figure II.9 : Etats d'un service web

Et les transitions sont:

- Create : le service vient d'être crée et la publication dans un annuaire de service le positionne dans l'état « passive ».
- Remove : le service est retiré de l'annuaire de service.
- Activate : le service web est sélectionné par un client.
- Passivate : le service web est sans client.

4. Les services web sémantiques

Les services Web sémantiques se situent à la convergence de deux domaines de recherche importants qui concernent les technologies de l'Internet : le Web sémantique et les Web services.

L'expression « web sémantique », due à Tim Berners-lee [berners-Lee et al., 2001] au sein du W3C, fait d'abord référence à la vision du web de demain comme un vaste espace d'échange de ressources entre êtres humains et machines permettant une exploitation, qualitativement supérieure, de grands volumes d'informations et de services variés. Le web sémantique s'intéresse principalement aux informations statiques disponibles sur le web et les moyens de les décrire de manière intelligible pour les machines. Cependant, la tâche principale des services web est d'assurer l'interopérabilité entre les applications via le web en vue de rendre le web plus dynamique.

L'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines.

Le besoin d'automatisation du processus de conception et de mise en œuvre des services web sémantiques rejoint les préoccupations à l'origine du web sémantique, à savoir comment décrire formellement les connaissances de manière à les rendre exploitables par des machines.

4.1 Description de services web sémantiques :

Les services web sémantiques sont des services avec des descriptions sémantiques, ces dernières sont fournies par des ontologies qui sont l'une des principales technologies du web sémantique. Dans la littérature, il existe deux approches pour décrire les services web sémantiques :

A. Description basée sur les annotations : le service est dans sa forme syntaxique, et il est enrichi par des annotations associées à des ontologies. Dans cette catégorie, la description est indépendante d'un langage sémantique particulier. Plusieurs langages ont été développés dans cette approche, tels que SAWSDL, WSDL-S, USDL.

A.1 SAWSDL, Semantic annotation for WSDL [Farrell et Lausen, 2007] est un langage sémantique de description de Service Web. Il est évolutif et compatible avec les standards des services Web existants, et plus spécifiquement avec WSDL. SAWSDL augmente l'expressivité du langage WSDL avec la sémantique en utilisant des concepts analogues à ceux utilisés dans OWL-S [Martin et al, 2004].

L'annotation des interfaces, opérations, entrées/sorties et les types XML simples s'effectue en leur associant un concept dans une ontologie par le biais des attributs suivants: modelReference, liftingSchemaMapping et loweringSchemaMapping.

L'attribut modelReference permet d'annoter certains éléments WSDL 2.0. Il est utilisé comme attribut dans les éléments <interface>, <operation> et <fault>. Il indique le concept équivalent en précisant son adresse.
Les attributs liftingSchemaMapping et loweringSchemaMapping permettent d'associer à un schema type ou à un élément un concept dans une ontologie partagée.
SAWSDL n'impose pas de langages particuliers pour formaliser les ontologies.

A.2 USDL, a été développé au sein du laboratoire de recherche ALPS (Applied Logic Programming Languages and Systems) de l'Université du Texas. L'approche USDL [Bansal A. et al., 2005] (Universal Service Semantics Description Language) propose une description sémantique et formelle pour les services Web.

USDL décrit un service Web en termes des balises XML "Messages" et "PortType". L'utilisation de WordNet[3] intervient à ces niveaux. Pour décrire les opérations du service (au niveau des PortType) et leurs paramètres (au niveau des Messages), USDL propose de les associer à des concepts basiques, des concepts qualifiés, des concepts inversés, des concepts conjonctifs et des concepts disjonctifs provenant de WordNet.

Il existe d'autres langages dans cette catégorie pour la description des services de type «RESTful»[4] tel que SA-REST.

B. Description basée sur des langages sémantiques : dans cette catégorie, on choisit dès le départ un langage sémantique pour modéliser le service, comme implémentation de cette approche, OWL-S et WSMO.

B.1 OWL-S, Ontology web language for services, est une ontologie pour décrire de façon non ambiguë le service web, elle se base sur le langage OWL. OWL-S modélise le service sur trois axes :

Le service profile "what the service does" : contient le nom du service, la description textuelle, la description des propriétés fonctionnelles (IOPE) et non fonctionnelles (QoS).

Le service model "how to use the service": décrit la logique métier interne du service. Il modélise le service en tant que processus et un ensemble de flux de contrôle, il propose trois types de processus :

[3] WordNet est une ressource lexicale de large couverture, développée par des linguistes du laboratoire des sciences cognitives de l'université de Princeton pour la langue anglaise. Plusieurs autres ressources linguistiques ont été constituées (manuellement ou automatiquement) à partir de, en extension à, ou en complément à WordNet.
[4] Restful : REST (REpresentational State Transfer) est un style d'architecture pour concevoir un service web, utilisant au maximum les possibilités de HTTP.

Processus atomique : correspond à une seule opération (une seule interaction), c'est le seul processus qui est directement exécutable et peut être directement invoqué.

Processus composite : correspond à une combinaison de processus (atomique ou non) liés par des structures de contrôles (Sequence, Split, If-Then-Else etc.).

Processus simple : fournit un mécanisme d'abstraction et multiple vue du même processus.

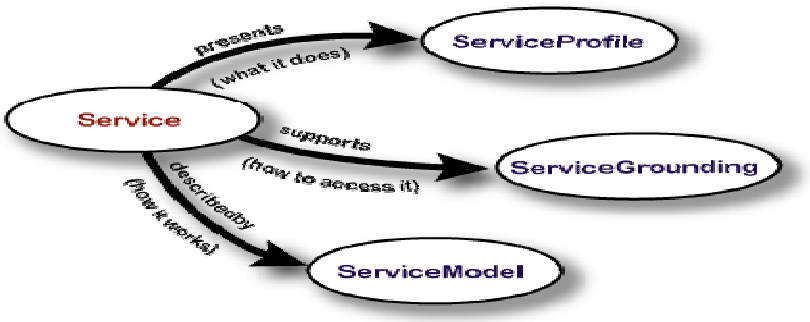


Figure II.10 : Ontologie de service OWL-S

Service grounding spécifie les détails d'accès au service. Il fournit les détails concernant les protocoles, les formats de messages et les adresses physiques. Ces informations sont particulièrement utiles pour l'invocation automatique des services.

B.2 WSMO, est une ontologie qui décrit les différents aspects relatifs à la composition dynamique des services Web, y compris la découverte dynamique, la sélection, la médiation et l'invocation. Elle est basée sur WSMF (Web Service Modelling Framework) [Fensel & Bussler, 2002] qui spécifie les éléments principaux pour décrire les services Web sémantiques. Ces éléments incluent ontologies, objectifs, services Web et médiateurs. Les ontologies définissent la terminologie utilisée par les autres éléments en termes de concepts, relations, fonctions, instances et axiomes. Les buts indiquent ce que l'utilisateur attend du service. La description du service Web définit les fonctionnalités offertes par le service. Les médiateurs lient les différents éléments afin de permettre l'interopérabilité entre les composants hétérogènes.

Le langage WSML (Web Service Modelling Language) [de Bruijn J. et al., 2005] est utilisé pour décrire formellement tous les éléments de WSMO et l'environnement d'exécution WSMX [Bussler C. et al., 2005] permet la découverte, la sélection, la médiation, l'invocation et l'interopérabilité des services Web sémantiques.

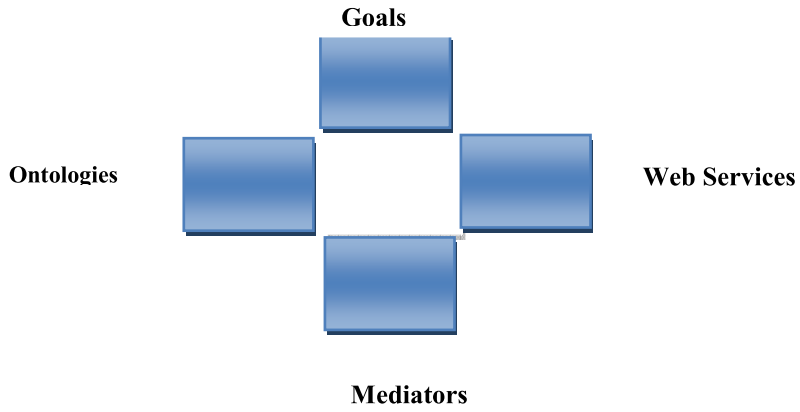


Figure II.11: Les éléments fondamentaux de WSMO

En plus de ces approches, il existe d'autres propositions pour décrire les services web sémantiques, comme easy-L, and pyramid-S [Ngan LD & Kanagasabai R., 2012].

4.2 Evaluation et discussion :

[Chabeb Y., 2010] présente, dans sa thèse de doctorat, une étude comparative entre les langages de description des services sémantiques. Les critères d'évaluation identifiés pour la comparaison sont :

1. Dépendance vis-à-vis d'un langage d'ontologie,
2. Réutilisation et adaptation de descriptions et d'outils,
3. Expressivité,
4. Explicitation de la sémantique.

Symboles	-	+/-	+	++
Interprétations	Ne répond pas au critère	Il y a un manque	Répond mais On peut améliorer	Répond bien au critère

Table II.1 Sémantique des symboles

Critère/Approches	Objectifs	SAWSDL	OWL-S	WSMO	USDL
Dépendance d'un type d'ontologie	++ (indépendant)	++	-	-	-
Adaptation & réutilisation	++(standard)	++	+	+	-
Expressivité	++ (maximale)	+	+/-	+/-	+/-
Sémantique explicite	++ (explicite)	-	+	+	+/-

Table II.2 Comparaison d'approches de descriptions

Selon cette étude comparative, aucune des approches examinées ne réalise complètement les objectifs escomptés.

[Lecue F., 2008] Présente un tableau comparatif des modèles WSMO, OWLS et SAWSDL (voir la table II.3). De façon générale, nous pouvons constater les points suivants :

- WSMO et OWLS sont des ontologies de haut niveau alors que SAWSDL est une extension du format WSDL.
- WSMO et OWLS sont liés à des formalismes particuliers tels que WSML ou OWL, alors que SAWSDL est indépendant de tout langage de définition d'ontologies.

	Ontologie de services				**Extension des standards des services web**	
	OWLS		**WSMO**		**Sa-wsdl**	
fonctionnalité	**Service profile**	**IOPE** **Categorie de services** **Propriétés non fonctionnelles**	**Service capability**	**IOPE** **Propriétés non fonctionnelles**	**Annotations semantiques internes et externes**	**Model refernce, schéma mapping**
processus	**Process model**	**Processus atomiques, simples et composites**	**Chorégraphie, orchestration**	**Transitions avec gardes**	×	
invocation	**Service grounding**	**Liaison avec wsdl, soap**	**Service grounding**	**Liaison avec wsdl ,soap**	**Liaison avec wsdl ,soap**	
Environnement d'exécution	×		**Wsmx**		**Meteor-s**	

Table II.3 Comparaison des principaux standards sémantiques des services web.

Une autre problématique liée à la description de la sémantique des services, c'est comment exploiter cette sémantique dans les différentes tâches qui utilisent le service ?

C'est le problème d'automatisation.

5. Problèmes d'automatisation :

Cette problématique traite le développement des algorithmes et des mécanismes de raisonnement exploitant la sémantique des services pour automatiser les différentes fonctionnalités telle que la découverte, la composition, la sélection, la substitution.

A. découverte, c'est la capacité de localiser automatiquement un service qui répond à des besoins particuliers.

B. composition, c'est la capacité de découvrir, de sélectionner, de composer et de faire interopérer des services existants pour répondre à une requête d'utilisateur.

C. sélection, c'est la capacité d'analyser qualitativement les performances d'un service.

D. substitution, c'est la capacité de remplacer un service par un autre équivalent (propriétés fonctionnelles et non fonctionnelles) et de résoudre les incompatibilités qui résultent.

Ce domaine de recherche reste actif afin de combler les manques et atteindre les motivations du web services sémantiques.

On s'intéresse à la problématique d'automatisation, et en particulier la fonctionnalité de découverte en adoptant OWL-S pour décrire nos services.

6. Conclusion

Les services web proposent une approche flexible pour l'intégration de systèmes hétérogènes en s'appuyant sur un modèle d'intégration basé sur un couplage faible des composants et en exploitant de manière intensive les standards du web (HTTP, XML, WSDL, SOAP, UDDI). Ceci a pour effet de permettre une intégration des applications plus rapide, moins coûteuse et avec des perspectives d'évolution et de réutilisation réelles pour les entreprises.

Ce chapitre présente les principales spécifications qui supportent la technologie des services web syntaxiques, à savoir, le protocole SOAP, l'interface WSDL, et l'annuaire UDDI. Les limites rencontrées lors de l'utilisation des services ont montrés la nécessité d'ajouter une couche sémantique pour améliorer le fonctionnement des applications services web. Nous avons montré aussi les formalismes de description sémantiques tels que SAWSDL, USDL, OWLS, WSMO. Nous notons que ces formalismes sémantiques, sont primordiaux pour l'automatisation du cycle de vie des services web.

Chapitre III:

Découverte de services web sémantiques

Sommaire

1. Introduction

Le mécanisme de matching fourni par UDDI et WSDL ne peut servir l'objectif de la découverte dynamique de services Web. Le principe de ce mécanisme est basé sur la forme syntaxique de services Web et peut produire par conséquent de nombreux faux positifs et faux négatifs, ce qui réduit énormément les performances du système de découverte.

Pour surmonter ces limitations, le concept de la sémantique doit être introduit dans le processus de matching par l'utilisation d'ontologies.

La découverte de services sémantiques est un domaine de recherche très actif, et elle est abordée dans un grand nombre de travaux de recherches.

Certains auteurs considèrent le système de découverte de services comme un matchmaker. Dans d'autres travaux, ce système couvre l'ensemble des tâches de la requête jusqu'au l'invocation des services réponses, ce qui signifie l'inclusion du processus de sélection et d'invocation.

En général, tous les systèmes de découverte de services doivent définir un matchmaker qui évalue la mesure de similarité entre deux services, il est le cœur de toute approche de découverte.

Le système de découverte retourne deux types de résultats à savoir:

Découverte exact: dans le cas où il existe un service qui répond exactement aux besoins des utilisateurs.

Découverte approximative: est considérée comme le cas réaliste, le système de découverte retourne un service (ou un ensemble de services) qui répond à peu près aux besoins des utilisateurs.

Définition « découverte » :

> *La découverte de services est le processus d'identification et de localisation des services les plus similaires à la requête basé sur la description sémantique des propriétés fonctionnelles et/ou non-fonctionnelles.*

Cette définition met l'accent sur trois aspects majeurs qui sont la localisation, la similarité et les propriétés prises en compte par le processus de découverte.

La localisation consiste à trouver l'adresse du service en se basant sur la similarité entre la requête de l'utilisateur et le service du fournisseur, cette similarité est évaluée par le matchmaker qui réalise d'un ensemble d'opérations de matching des propriétés prises en charge par le processus de découverte.

Définition « matching » :

le matching est le processus de mise en correspondance deux à deux entre les propriétés des services selon une mesure.

Définition « matchmaking » :

Le matchmaking peut être équivalent au processus de découverte, mais selon d'autres chercheurs, la découverte regroupe le matchmaking ainsi que la sélection et l'invocation.

Dans le cadre de notre thèse, nous considérons le matchmaking comme étant le processus de trouver des services web pertinents en se basant sur l'évaluation et l'agrégation des résultats des opérations de matching. Les résultats retournés doivent être rangés et classés par ordre de mérite (matchmaking regroupe : mécanisme de matching, fonction d'agrégation, et ranking).

2. La similarité:

Une mesure de similarité est, en général, une fonction qui quantifie le rapport entre deux objets, comparés en fonction de leurs points de ressemblance et de dissemblance. Les deux objets comparés sont, bien entendu, de même type.

2.1 Métrique (distance):

Une métrique est une fonction binaire qui décrit la distance entre deux objets d'un ensemble E. Une distance d est une fonction $E \times E \rightarrow \Re$ telle que, $\forall$ i, j, k $\in E$:

1. d (i, j) = d (j, i) Symétrie
2. d (i, j) ≥ 0 Positivité
3. d (i, j) =0 ⇔ i=j Principe d'identité des indiscernables
4. d (i, j) ≤ d (i, k) + d (k, j) Inégalité triangulaire

Si toutes ces propriétés sont respectées, on se trouve en présence d'une distance métrique. Si l'inégalité du triangle n'est pas respectée, on parle plutôt d'une distance semi-métrique.

2.2 Mesure de similarité:

Une autre méthode de comparer deux objets est de tester leur similarité. Alors que la distance mesure le degré de « différence » entre deux objets, un indice de similarité mesure le degré de « ressemblance » entre deux objets. La vision d'une mesure de similarité est l'inverse d'une distance, plus deux objets sont similaires, moins ils sont distants. Une mesure de similarité S est une fonction $E \times E \rightarrow \Re$ telle que, $\forall$ i, j $\in E$:

1. S (i, j) ≥ 0 Positivité

2. S (i, j) = S (j, i) Symétrie
3. S (i, i) ≥ S (i, j) Maximalité

D'autres propriétés peuvent être requises comme la normalisation qui impose que les valeurs appartiennent à l'intervalle [0,1].

Dans le cas d'une distance, on cherche habituellement les éléments les plus proches, c'est-à-dire qu'on cherche la distance minimale. Dans le cas d'une similarité, on cherche les éléments les plus similaires, c'est-à-dire l'indice de similarité maximal.

2.3 Types de mesures :

Dans la littérature, on peut trouver deux différentes approches pour les mesures de similarités:

2.3.1 Approche syntaxique:

Regroupe les mesures permettant de comparer des objets en se basant sur la comparaison des chaines de caractères (termes ou tokens). C'est l'exemple de la métrique qui mesure la similarité ou la dissimilarité entre deux chaînes de caractères. Par exemple, les chaînes de caractères "voiture" et "voiturier" peuvent être considérées comme très proches, alors que "voiture" et "automobile" pourront être considérées comme très différentes. Une telle mesure sur les chaines de caractères fournit une valeur obtenue algorithmiquement. Parmi de telles mesures de similarité, citons par exemple, la distance de Levenshtein (ou distance d'édition), le coefficient de Dice, l'indice de Jaccard, la distance euclidienne, le cosinus,...(ex. Levenshtein(voiture, voiturier)=0.78).

2.3.2 Approche sémantique:

La similarité sémantique est un concept selon lequel un ensemble d'objets ou de termes se voient attribuer une métrique basée sur la ressemblance de leur signification ou leur contenu sémantique.

Concrètement, cela peut être réalisé en définissant une similitude topologique, par exemple, en utilisant des ontologies pour définir une distance entre les mots, ou en définissant une similitude statistique, par exemple en utilisant un modèle d'espace vectoriel pour corréler les termes et les contextes à partir d'un corpus de texte approprié (co-occurrence).

Parmi de telles mesures de similarité, citons par exemple, Resnik, Wu palmer, Lin,…

3. les méthodes formelles:

Les méthodes formelles traitent les composants d'un système comme des objets mathématiques, et elles fournissent des modèles pour décrire et prédire les propriétés de ces objets ainsi que leur comportement.

Les approches basées sur les modèles formels sont utilisées pour décrire formellement, composer, et vérifier les services Web.

3.1 Graphes

Les graphes sont souvent l'un des modèles de représentation les plus utilisés en science et technologie de l'information qui permettent la description de données structurées. Les graphes sont utilisés dans de nombreuses applications et de domaines tels que la reconnaissance des formes, vision par ordinateur, la chimie (molécules), et les réseaux biologiques.

Un graphe permet de décrire un ensemble d'objets (sont appelés nœuds ou sommets) et leurs relations (sont appelés arête).

Formellement un graphe G est un couple (V,E) où :

V est un ensemble d'objets (sommets).

E est sous-ensemble de VxV. Les éléments de E sont appelés les arêtes du graphe.

3.1.1 Isomorphisme de sous-graphe

Un graphe orienté G = (V, A) est constitué d'un ensemble de nœuds V et un ensemble d'arcs A ⊆ V * V. Deux graphes sont isomorphes s'ils sont structurellement indifférents, plus formellement, G_a et G_b sont isomorphes si et seulement s'il existe une fonction bijective entre les ensembles de nœuds.

Définition (isomorphisme de graphe)*:* soient G=(V,A) et G'=(V',A') deux graphes. Un isomorphisme entre G et G' est la fonction bijective $f: V \rightarrow V'$ où:

$$\forall (a,b) \in V^2, (a,b) \in A \Leftrightarrow (f(a), f(b)) \in A'$$

La complexité du problème de l'isomorphisme de deux graphes est encore indécise [Laura Zager, 2005]. Certains auteurs définissent une nouvelle classe de complexité dans ce cas, notée isomorphe-complet.

Lorsque le nombre de nœuds dans les deux graphes est différent, le problème d'isomorphisme de graphe devient plus difficile. Ce problème est connu sous le nom isomorphisme de sous-graphe où l'objectif est de chercher des copies isomorphes d'un graphe motif dans un graphe cible.

Définition (isomorphisme de sous-graphe) : un isomorphisme de sous-graphe entre un graphe motif Gm = (Vm, Am) et un graphe cible Gc = (Vc, Ac) est la ***fonction totale:*** $f: Vm \rightarrow Vc$, où:

1. La fonction f est ***injective***,
2. f est un ***isomorphisme***: $(a,b) \in Am \Leftrightarrow (f(a), f(b)) \in Ac$

Dans la littérature, il existe deux types d'isomorphisme de sous-graphe, le premier est généré par les sommets (sous-graphe induit) et le deuxième est généré par les arcs (sous-graphe partiel).

Dans le sous-graphe induit G'(V', A') du graphe G (V, A), on a V'⊆V et tous les arcs des nœuds V' sont préservés. Dans le sous-graphe partiel, des arcs de G peuvent ne pas être préservés dans G '.

Définition (sous-graphe induit): soient G=(V, A) et G'=(V', A') deux graphes. G' est un sous-graphe induit de G si:

1. V'⊆V
2. A'=A∩V'*V'

Définition (sous-graphe partiel): soient G=(V, A) et G'=(V', A') deux graphes. G' est un sous-graphe partiel de G si:

1. V'⊆V
2. A'⊆A∩V'*V'

3.1.2 Voisinage de nœud:

Le voisinage d'un nœud est l'ensemble de ses nœuds successeurs et prédécesseurs.

Le voisinage du nœud u, notée adj(u), est définie par : adj(u)={u' | (u, u') ∈ A}, où A est l'ensemble des arcs.

L'ensemble des successeurs d'un nœud u, dénoté par succ(u), est définie par : succ (u)= {u' | ∃ a∈ A, u → u'}.

L'ensemble des prédécesseurs d'un nœud v, dénoté par pred(v), est définie par : pred (v)= {v' | ∃ a∈ A, v' → v}.

Degré de nœud :

Chaque nœud dans les deux graphes (motifs et cibles) possède deux propriétés, degré-in et degré-out.

Degré-in d'un nœud « v » est le nombre d'arcs entrants où « v » représente le nœud terminus pour ces arcs.

Degré-out d'un nœud « v » est le nombre d'arcs sortants où « v » représente le nœud source pour ces arcs.

3.2 Automates

Les automates finis offrent un formalisme de description peu puissant mais avec beaucoup d'algorithmes efficaces. Ils sont très utilisés dans plusieurs domaines et notamment la description de comportement dynamique de systèmes.

Formellement un automate fini est un quintuplet A=(Σ, Q, δ, i, F) où :

Σ est un ensemble fini de symboles appelé alphabet.

Q est un ensemble fini dont les éléments sont appelés états.

δ est une relation de transition : $Q^*\Sigma \rightarrow Q$

i est un état de Q appelé état initial.

F est un sous-ensemble de Q appelé ensemble des états finaux de A.

Un automate peut etre représenté comme un graphe orienté dont les sommets sont les états et les arcs sont les transitions.

La façon intuitive dans lequel un automate peut modéliser le comportement d'un système a conduit à une variété de modèles de spécification sur la base des automates, tels que, par exemple, les automates d'entrés/sorties (I/O automata), les automates d'interface, les automates pondérés par des entiers.

3.3 Algèbre de processus

Algèbres de processus sont des langages formels qui permettent de décrire les protocoles de communication entre les processus concurrents et de vérifier leurs propriétés. Plusieurs de modèles d'algèbre de processus ont été proposés: CCS [Milner, 1989], ACP [Baeten & Weijand, 1990], CSP [Hoare, 1984], et leur extension π-calcul [Parrow, 2001], LOTOS [Bolognesi et Brinksma, 1989], Promela [Holzmann, 2003] et Timed CSP [Schneider et al., 1992]. L'interprétation sémantique sous-jacente de ces modèles est basée sur les systèmes de transitions étiquetées.

Algèbres de processus sont des formalismes précis, bien étudiées dans la littérature et qui permettent la vérification automatique de certaines propriétés de l'aspect comportemental des systèmes. Ils fournissent une théorie riche sur l'analyse de bisimulation (déterminer si deux processus ont des comportements équivalents).

3.4 Réseaux de pétri

PetriNets [Petri, 1962] sont des outils d'analyse et de modélisation de processus, ils sont très utilisés dans les systèmes concurrents et parallèles. Un réseau de Pétri est un graphe biparti orienté, dans lequel chaque nœud est soit une place ou une transition. Les places sont utilisées pour représenter les états du processus, tandis que les transitions représentent les actions élémentaires du processus. Arcs lient entre les places et les transitions, et jamais entre les places ou entre les transitions.

L'arc est étiqueté par une valeur (ou un poids), qui est un nombre entier positif. Un marquage assigne un nombre entier à chaque place, si un marquage assigne un entier k à une place p, on

dit que p est marqué par k jetons. Les jetons sont utilisés pour simuler les activités concurrentes et dynamiques des systèmes.

Formellement un PN est un triplet N=(P, T, F) où :

P et T sont deux ensembles ($P \cap T=\phi$), P les places et T les transitions.

$F \subseteq (P \times T) \cup (T \times P)$ représente les arcs.

Graphiquement, les places sont présentées par des cercles, et les transitions par des rectangles noirs.

3.5 Discussion :

Une étude comparative de ces modèles formels a été réalisée par Juan Carlos Corrales [J.C Corrales, 2008] dans sa thèse de doctorat. La table III.1 présente cette évaluation dont les critères d'évaluation sont :

Complétude (completeness): un ensemble complet de propriétés syntaxiques et sémantiques suffisantes pour la modélisation du système.

Composabilité (composability): une propriété qui permet de raisonner sur un système composé en se basant sur ses éléments composants sans le besoin de savoir d'autres informations sur la mise en œuvre de ces parties.

Parallélisme (parallelism): un aspect clé du formalisme qui doit être remplie pour modéliser avec précision la composition des services web.

Complexité de matching (polynomial matching): la complexité temporelle de l'algorithme du matching dans le domaine des services web.

Table III.1 résume les représentations formelles des processus expliqués dans cette section. Le signe (+) signifie la présence de cette propriété (en colonne) par le modèle formel. Le symbole (-) signifie l'absence de la propriété.

Le signe (+/-) signifie que la propriété est légèrement soutenu.

Enfin, le symbole N/A (not applicable) signifie que la propriété ne s'applique pas au modèle avec lequel il a été associé.

Formal representation	Critères d'évaluation			
	Completeness	Composability	Parallelism	Polynomial matching
Process algebra models	+	+	+	N/A
Petri Nets models	+	-	+	+
Finite state machine	+	+	+/-	+
Graph representation	+	+	+	+/-

Table III.1 Comparaison entre les modèles formels

4. Les critères de découverte des services web sémantiques :

Selon [Ngan LD & Kanagasabai R., 2012], les systèmes de découverte des services web ont plusieurs critères importants qui doivent être remplies, et qui peuvent varier selon les contextes d'utilisation et les domaines d'application :

Conformité aux normes: le framework de découverte doit être conforme aux normes pour assurer l'interopérabilité, et la réutilisation.

Expressivité: le formalisme de description sémantique doit être assez riche pour encoder les descriptions de service, les requêtes et les objectifs précisément.

Autonomie: le processus de découverte doit minimiser au maximum la participation de l'utilisateur. Le processus de découverte idéal implique un seul utilisateur qui est le client qui recherche de services ayant des besoins spécifiques.

Qualité de service (QoS): QoS se réfère principalement à la qualité, à la fois fonctionnel et non fonctionnel, d'un service web. Les propriétés QoS principales sont : la performance, la fiabilité, l'intégrité, l'accessibilité, la disponibilité, l'interopérabilité et la sécurité. Ces propriétés doivent être prises en compte dans le mécanisme de découverte pour le ranking et la sélection.

Scalabilité: la scalabilité est une qualité importante qui intervient lorsque le nombre des utilisateurs (clients et fournisseurs) du système de découverte devient de plus en plus grand.

Robustesse: le mécanisme de découverte doit être capable de faire face aux pannes et aux changements de l'environnement, et réagir de manière appropriée à la présence de conditions anormales.

Dynamisme: la description sémantique des services web peut être insuffisante pour offrir un processus de découverte dynamique qui englobe toutes les possibilités offertes.

Hétérogénéité: le mécanisme de découverte de services devrait être en mesure de faire face à l'hétérogénéité, qui pourrait résulter en raison de différentes plates-formes, différents formats de données, ainsi que due à des hétérogénéités entre ontologies de services ou ontologies de domaine.

L'hétérogénéité importante est celle des différents langages de description de services. Le framework WSMO a abordé ce problème en proposant une solution à base de médiation (à travers des médiateurs).

La table III.2 présente une comparaison des approches de découverte, où les lignes représentent les approches de découverte et les colonnes représentent les critères d'évaluation.

Le symbole "√" désigne la prise en charge de la propriété, et dans le cas contraire c'est le symbole "-". Tandis que le symbole "+" désigne un support partiel de la propriété [Ngan Ngan LD & Kanagasabai R., 2012].

	Autonomy	Scalability	Dynamism	Heterogeneity	Context-awareness
OWLS-MX [Klusch et al, 2006]	-	-	-	-	-
iMatcher [Bernstein A. et Kiefer C., 2006]	-	-	-	-	-
Rajesh [Rajesh et al., 2009]	-	-	+	-	-
MODiCo [Ngan LD, et al., 2009)	-	-	-	+	-
Keller [Keller U, et al., 2005]	√	-	√	√	-
WSMO-MX [Kaufer et Klusch, 2006]	-	-	-	√	-
SWE-ST [Brambilla M, et al., 2009]	+	-	√	√	√
DERI [Vitvar T, et al., 2009]	√	-	√	√	√
Oundhankar [Oundhakar S, et al(2005]	-	√	-	-	+
Kourtesis [Kourtesis et al., 2008]	-	-	-	-	-
SAWSDL-MX [Klusch et Kapahnke, 2008]	-	-	-	-	-

Easy-L [Mokhtar SB, et al., 2008]	-	√	-	-	-
Pyramid-S [Pilioura T, T salgatidou A, 2009]	-	√	-	-	-
Verma [Verma K, et al., 2003]	-	√	-	-	-
DIANE [KuSter U et al., 2007]	√	+	+	-	+
Paolucci [Paolucci et al., 2002]	-	-	-	-	-
Li [Li L. & Horrocks I., 2003]	-	-	-	-	-

Table III.2 Évaluation d'approches de découverte [Ngan LD & Kanagasabai R., 2012]

5. Classification des approches

5.1 Principes :

Les systèmes de découverte de services web sémantiques peuvent être classés en trois familles selon le mode de raisonnement utilisé par le matchmaker (logique, non-logique, hybride). Ainsi que dans chaque famille de découverte, les approches proposées peuvent être aussi classées selon les propriétés exploitées dans le processus de matching (matching au niveau d'interface, matching au niveau du processus, matching hybride).

Matching logique: les matchmakers utilisent les relations sémantiques et l'inférence logique pour mesurer la similarité entre deux services. Le degré de mise en correspondance basé sur la logique peut être déterminée, soit (a) exclusivement dans la théorie de la logique considérée au moyen d'un raisonnement logique, ou (b) par une combinaison d'inférences logiques à l'intérieur de la théorie et de traitement algorithmique en dehors de la théorie [Klusch M., 2008].

Matching non-logique: le matchmaker effectue hors de toute logique raisonnement pour déterminer la similitude entre les services. Ils utilisent d'autres techniques de d'autres domaines de recherche comme : isomorphisme de graphes, les techniques de la recherche de l'information ... etc.

La découverte basée sur le matching non logique, trouve ses origines dans les mécanismes de recherche d'information. L'inconvénient majeur des approches non logiques est qu'elles relèvent toujours des problèmes linguistiques, terminologiques ou statistiques. Des résultats

de découverte non pertinents peuvent être causés par plusieurs problématiques bien connues dans ce domaine, telles que les mauvaises et les fausses interprétations de la syntaxe.

***Matching hybride**:* le matchmaker utilise une combinaison de mécanismes logiques et non logiques permettant d'effectuer le processus de matching.

L'idée est de remédier à certaines limites de chacun de ces deux mécanismes grâce à différentes combinaisons hybrides qui réussissent là où chacun de ces deux mécanismes échoue.

Matching au niveau du profile (black-box matching): le matchmaker exploite les propriétés du profile dans le processus de matching. Un nombre important d'approches est basé sur un matching des entrées/sorties entre le service et la requête, on peut citer comme exemple de cette catégorie, l'algorithme de Paolucci. Autres matchmakers exploitent d'autres éléments du profile de service comme IOPE, PE, E, QoS dans le processus de matching.

Matching au niveau du processus (glass-box matching): le matchmaker est basé sur le matching du comportement entre le service et la requête. Les approches de cette catégorie se basent sur la modélisation des actions (observables et/ou internes) et les états d'évolution du modèle comportemental du service. Elles utilisent différents formalismes tels que, la théorie des graphes, l'algèbre de processus, les automates, les réseaux de pétri,…etc.

***Matching hybride**:* le matchmaker utilise une combinaison des mécanismes de matching du profile de service et des mécanismes de matching du processus. L'idée est de remédier à certaines limites de chacun de ces deux mécanismes grâce à différentes combinaisons hybrides qui réussissent là où chacun de ces deux mécanismes échoue.

Un aperçu d'approches existantes dans la découverte des services web sémantiques est illustré par la table III.4. La description donnée par [Klusch M., 2008] est utilisée avec l'intégration d'autres catégories et d'autres approches. Dans la section suivante nous présentons en détails quelques approches de chaque catégorie.

5.2 Les approches logiques :

5.2.1 Approches basées sur le profile

[Paolucci et al., 2002]

L'approche de paolucci [paolucci et al., 2002] s'inscrit dans la catégorie des approches logiques qui manipulent les concepts inputs et outputs du service profile de l'ontologie OWL-S dans le processus de découverte dynamique.

L'algorithme de matching de cette approche est un algorithme glouton -Greedy algorithm- qui propose de trouver un max-match entre chaque concept de la requête (entrées/sorties) et les

concepts du service publié (entrées/sorties). Il définit quatre classes sémantiques pour représenter le degré de correspondance entre les concepts en se basant sur la relation de subsomption de l'ontologie de domaine. La mesure de similarité entre concepts est une fonction discrète qui retourne quatre valeurs (exact, plugin, subsumes, fail) selon la véracité de la relation de subsomption.

Notons que le matching des concepts outputs est différent de celui des concepts inputs, les algorithmes III.1 et III.2 présentent les procédures utilisées dans les deux cas :

degreeOfMatch(outR, outA) :

if outA=outR then return exact

if outR subclassOf outA then return exact

if outA subsumes outR then return plugIn

if outR subsumes outA then return subsumes

otherwise fail

Algorithme III.1: Règles d'affectation de degré de correspondance des concepts de sorties

degreeOfMatch(inA, inR) :

if inR=inA then return exact

if inA subclassOf inR then return exact

if inR subsumes inA then return plugIn

if inA subsumes inR then return subsumes

otherwise fail

Algorithme III.2: Règles d'affectation de degré de correspondance des concepts d'entrées

Avec :

outR est le concept output de la requête

outA est le concept output du service publié

inR est le concept input de la requête

inA est le concept input du service publié

Les classes sémantiques de matching :

- ✓ Exact : dans le cas où les deux concepts sont équivalents ou un concept est un père directe de l'autre. Dans la figure III.1, outR=car et outA=vehicle, alors Dom(car,vehicle)=exact

  Dans le cas des concepts inputs, inA=car et inR=vehicle, alors Dom(car,vehicle)=exact.

- ✓ Plugin : dans le cas où outA englobe outR dans la procédure de correspondance des concepts d'outputs, et inR englobe inA dans la procédure de correspondance des concepts d'inputs. outA=vehicle et outR=sedan, alors Dom(sedan, vehicle)=plugin
- ✓ subsumes: dans le cas où le concept du service ne répond pas complètement au concept de la requête. Le concept output de la requête englobe le concept output du service. outR=vehicle et outA=sedan, alors Dom(vehicle, sedan)=subsumes.
- ✓ Fail : dans le cas où n'existe aucune relation de subsumption entre les concepts de la requête et du service.

Dans la littérature, généralement les auteurs affectent des valeurs numériques aux classes de degré de correspondance entre les concepts.

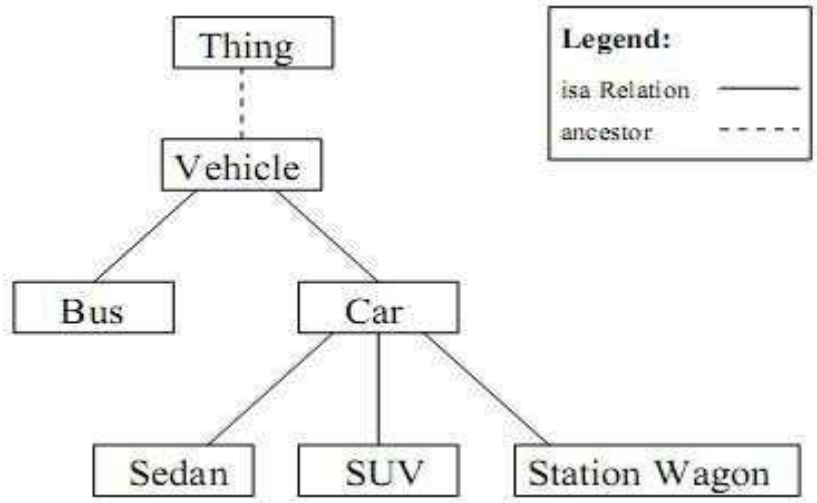


Figure III.1 : Fragment de l'ontologie « vehicle »

Procédure de matching :

Le matching entre les services et la requête se fait sur deux étapes :

- ➢ Matching des concepts outputs : consiste à faire le matching entre tous les concepts output de la requête et les concepts outputs du service, le résultat est représenté par la variable match.output (nommée aussi Gdom_out) qui contient le plus petit degré trouvé.
- ➢ Matching des concepts inputs : consiste à faire le matching entre tous les concepts input du service et les concepts outputs de la requête, le résultat est représenté par la variable match.input (nommée aussi Gdom_in) qui contient le plus petit degré trouvé.

L'algorithme III.3 présente la procédure de matching des outputs.

```
outputMatch(outputsRequest, outputsAdvertisement) {
        globalDegreeMatch=Exact
        forall outR in outputsRequest do {
```

```
            find outA in outputsAdvertisement such that
                    degreeMatch=maxDegreeMatch(outR, outA)
                    if (degreeMatch==fail) return fail
            if (degreeMatch < globalDegreeMatch)
                    globalDegreeMatch=degreeMatch
        return sort(recordMatch);}
}
```

Algorithme III.3 : Procédure de matching des concepts de sorties

Ranking des résultats :

L'algorithme trie les résultats du matching stockés dans la structure match[i] où chaque résultat possède deux attributs (match.output ou Gdom_out et match.input ou Gdom_in).

```
sortRule (match1,match2) {
    if match1.output > match2.output then match1 > match2
        if match1.output = match2.output
            & match1.input > match2.input then match1 > match2
        If match1.output = match2.output
            & match1.input = match2.input then match1 = match2
}
```

Algorithme III.4 : Règles de tri des résultats de matching

Le meilleur score de matching est le résultat qui possède l'attribut match.output plus élevé, et dans le cas d'égalité, on fait le recourt au deuxième attribut match.input comme un critère secondaire pour différencier les résultats de matching.

La procédure du matchmaker :

La fonction principale du matchmaker de Paolucci est présentée par l'algorithme III.5, et qui a comme argument la requête de l'utilisateur. Cette fonction consiste à balayer la base des services en enregistrant que les services qui peuvent répondre à cette requête (services avec score de matching différent de fail).

```
        match (request) {
                recordMatch = empty list
                forall adv in advertisements do {
                        if match (request, adv) then
                        recordMatch.append(request, adv) }
                return sort(recordMatch);}
```

Algorithme III.5 : La fonction principale de l'algorithme du matchmaker

Complexité de l'algorithme :

Soit n_1 et n_2 le nombre de concepts input de la requête et du service.

Soit m_1 et m_2 le nombre de concepts output de la requête et du service.

La complexité de l'algorithme de matching est donnée par la formule suivante : $((n_1*n_2)+(m_1*m_2))$.

Le Matchmaker répète « t » fois l'exécution de l'algorithme de matching selon la cardinalité des services dans la base. La complexité de l'algorithme de matchmaker de Paolucci est donnée alors par la formule suivante : $(t*((n_1*n_2)+(m_1*m_2)))$.

On voit clairement que la complexité est de classe polynomiale et en particulier d'ordre $O(N^3)$ dans le pire cas, c'est-à-dire quand $t=n_1=n_2=m_1=m_2=N$, alors le matchmaker de Paolucci est un algorithme efficace rapide de complexité cubique.

[Benatallah et al, 2005]

Les auteurs formalisent la découverte comme un problème de réécriture de concepts en logique des descriptions (DL).

L'algorithme proposé cherche de découvrir une meilleure couverture sémantique d'une requête par les concepts d'une ontologie exprimée en logiques de description (DL). La requête est constituée d'un ensemble de concepts d'entrées et de sorties. Les auteurs démontrent que le problème de découverte de la meilleure couverture sémantique, est similaire au problème de calcul, à coût minimal, des transversales minimes d'un hypergraphe pondéré (computing the minimal transversals with minimum cost of a weighted hypergraph).

Les résultats expérimentaux de cette approche [Rey et al, 2003] sont satisfaisants, mais nécessitent des améliorations pour prendre en compte un grand nombre d'ontologies, éventuellement hétérogènes.

[Bellur U., et al 2008]

Pour résoudre les problèmes de l'algorithme glouton de Paolucci, [Bellur U., et al., 2008] propose une amélioration dans les performances de matching en se basant sur un graphe biparti et l'algorithme hongrois[1] pour atteindre une solution optimale de correspondance de

[1] L'algorithme hongrois : L'algorithme Hongrois est un algorithme spécialisé permettant de résoudre le problème d'affectation. Il s'agit d'une procédure itérative qui transforme la matrice des coûts en une suite de matrices équivalentes, jusqu'à ce que l'obtention d'une solution optimale soit évidente. La matrice finale est telle que toutes les entrées sont soit positives ou nulles et qu'une affectation faisant appel seulement aux entrées nulles est possible. Cette affectation de coût 0 est alors nécessairement optimale.

concepts. Comme le montre le tableau III.3, l'algorithme affecte des poids numériques aux différents degrés de correspondance.

Degree of match	Weight
Exact	$W_1=1$
Plugin	$W_2=(w_1*\|v_0\|)+1$
Subsume	$W_3=(w_2*\|v_0\|)+1$

Table III.3: Pondération des degrés de matching

Où $|V_0|$ est la cardinalité des nœuds du graphe biparti.

La recherche d'une affectation optimale (maximale ou minimale) par la méthode hongroise est basée sur la notion des zéros indépendants dans une matrice carrée. Ces zéros n'appartiennent ni à la même ligne ni à la même colonne.

L'approche proposée calcule un matching complet des concepts inputs et outputs entre la requête et le service, ensuite elle fait appel à l'algorithme hongrois qui retourne la solution optimale de correspondance entre les nœuds du graphe biparti. Le principe est de minimiser le degré maximum du matching (la fonction objectif : min (max(w_i))).

Le résultat du matching des concepts outputs est match.out qui contient le degré maximum de matching qui correspond à une classe sémantique de matching.

Le résultat du matching des concepts inputs est match.in qui contient le degré maximum de matching qui correspond à une classe sémantique de matching.

Pour le tri des résultats, les auteurs ne spécifient aucun critère de comparaison des résultats de matching. Dans notre expérimentation de cette approche, on a adopté la même procédure de trie utilisée par l'algorithme de Paolucci.

Complexité :

Nous utilisons les mêmes variables pour calculer la complexité de cet algorithme. La complexité temporelle de l'algorithme hongrois est d'ordre $O(n^3)$ où "n" est la cardinalité de concepts (entrée / sortie). La complexité de la mise en correspondance est donnée par $((n_1*n_2)+n_2^3+(m_1*m_2)+m_1^3)$. La cardinalité du répertoire des services est « t » alors la complexité du matchmaker de Bellur est donnée par : $(t*((n_1*n_2)+n_2^3+(m_1*m_2)+m_1^3))$.

La complexité est polynomiale et en particulier d'ordre $O(N^4)$ dans le pire cas, c'est-à-dire quand $t=n_1=n_2=m_1=m_2=N$, alors le matchmaker de Bellur est un algorithme efficace rapide de complexité $O(N^4)$.

Performance :

L'expérimentation de cette approche, menée par Bellur et al., montre une forte amélioration dans les performances de l'algorithme du matchmaker par rapport à l'algorithme

de Paolucci. Ce constat est motivé par la minimisation des faux positifs et des faux négatifs.

[Kourtesis et al., 2008]

Les auteurs proposent une approche de découverte à base des annotations SAWSDL avec une ontologie OWL et un raisonnement en DL –logiques de descriptions- pour le matchmaking. L'approche présente une implémentation d'un registre sémantique nommé FUSION (FUSION project). Ce dernier est un projet de recherche financé par l'UE qui vise à promouvoir l'intégration des processus métiers -business process- et l'interopérabilité intra et inter entreprises, à travers une approche sémantique pour l'intégration d'applications d'entreprise (EAI) dans les entreprises. Le registre sémantique de fusion repose sur une combinaison de trois normes du domaine de services Web et les technologies du web sémantique pour atteindre ses objectifs: UDDI pour stocker et récupérer les informations syntaxiques et sémantiques sur les services et leurs fournisseurs (recherche par mots-clés). SAWSDL pour la création de descriptions sémantiques qui annotent les interfaces des services, et OWL pour réaliser un matchmaking logique à base les logiques de descriptions.

[Rajesh et al., 2009]

Les auteurs proposent une approche de découverte à base OWL-S où le matchmaker ne retourne aucun faux positif. Le matching se fait sur deux étapes : matching à base du profile et vérification de la consistance des résultats retournés.

Il propose cinq degrés de matching à base de la relation de subsomption :

Exact ($S \equiv Q$), plugin ($S \subseteq Q$), subsumed ($Q \subseteq S$), intersection ($\neg(S \cap Q \subseteq \bot)$), mismatch ($S \cap Q \subseteq \bot$).

Chaque service est associé avec deux profiles:

Profile conceptuel : c'est l'équivalent de TBOX[2], utilisé pour déterminer l'ensemble des correspondances possibles.

Profile instance : c'est l'équivalent de ABOX[3] utilisé pour déterminer les capacités concrètes du service (l'instanciation de profiles conceptuels).

L'algorithme de matching est divisé en deux étapes :

[2] TBOX=terminological box (connaissance conceptuelle, intentionnelle ou Terminologique). Mère $\sqsubseteq \exists$mèreDe.$\top$ (relation de sumbsomption)

[3] ABOX = assertion box (connaissances des instances, extensionnelle ou Assertionnelle) mèreDe(fatima, karim),
Une BC en logiques de description= Abox+Tbox

Etape 1 : consiste à identifier les correspondances possibles en utilisant les profiles conceptuels dont l'objectif est de réduire l'espace de recherche. Le résultat est une liste de services candidats.

Etape 2 : consiste à filtrer les résultats de l'étape 1. Créer un profile d'instances pour chaque service candidat, et évaluer l'exactitude du matchmaker (si ce service peut satisfaire la demande), dans le cas défavorable le service est retiré de la liste.

L'algorithme de matchmaking est modélisé comme un système de satisfaction de contraintes -Constraint Satisfaction Problem (CSP)- où les profiles de la requête et du service sont représentés comme des contraintes. Cependant, cette approche ne présente pas les mécanismes de tri des résultats retournés.

La complexité temporelle de la première étape du processus de découverte est d'ordre $O(N^2)$.

En supposant qu'il n'y n profiles conceptuels de services, chacun avec un maximum de m instances, le nombre de vérifications de contraintes qui doivent être effectuées pour trouver tous les services correspondants est en O (n * m) dans le pire des cas.

Pour la deuxième étape du processus de découverte qui traite le problème de satisfaction de contraintes (constraint satisfaction problem) dont la complexité est NP-difficile. Les auteurs supposent qu'en général, il y'a que peu de variables et de contraintes dans un profil et que la vérification de consistance devient un processus facile et rapide.

[Guilherme et al., 2012]

Les auteurs proposent une approche de découverte à base SAWSDL, le matchmaker cherche un service qui répond à la requête utilisateur, et dans le cas défavorable, le système lance une procédure de composition d'éventuels services qui peuvent être combinés pour répondre aux besoins de l'utilisateur.

5.2.2 Approches basées sur le modèle du processus

[Ehrig et al., 2007]

Les auteurs proposent une approche de découverte à base de modèle de processus du service OWL-S. Ce dernier est modélisé par les réseaux de pétri. L'idée de cette approche est de transformer les éléments du réseau de pétri qui modélisent le service à une ontologie OWL-DL, le résultat de ce passage est dénoté « semantic business process model SBPM».

Le matching entre deux SBPM se fait à la base de trois mesures de similarité (linguistique, syntaxique, et structurelle). La décision du matching est basée sur l'agrégation de ces trois mesures.

5.2.3 Approches hybrides :

[Günay A. et al., 2010]

Les auteurs présentent une approche de matchmaking de services basée sur la vérification de modèle (model checking). L'idée de base de cette approche est de représenter les services comme des modèles de systèmes, et les requêtes comme un ensemble de formules LTL (Linear Temporal Logic : logiques de temps linéaire) où chaque formule correspond à une propriété fonctionnelle spécifique de la requête. Le matchmaker utilise des techniques de vérification de modèles pour identifier des matchings exactes (le modèle vérifie toutes les propriétés spécifiées en LTL) ou partiaux. Un matching partiel est réalisé par deux méthodes: la première utilise les ontologies pour capturer des similarités sémantiques entre les concepts du service et la requête, et la deuxième utilise la relation de subsomption entre les opérateurs LTL à travers une structure hiérarchique.

5.3 Approches non-logiques :

5.3.1 Approches basées sur le profile

iMatcher1

[Bernstein A. et Kiefer C., 2006] offre un système de découverte non logique grâce à un matchmaker syntaxique basé sur les profiles de services. Il prend en entrée un ensemble de profiles de service spécifiés en OWL-S qui sont stockés sous forme de graphes RDF sérialisés dans une base de données RDF avec une extension de RDQL, appelée iRDQL [Bernstein A. & Kiefer C., 2006]. Le degré de matching est calculé à partir de quatre métriques de similarité syntaxique de recherche d'information: TF/IDF (Term Frequency-Inverse Document Frequency) [Salton G. 1983], la distance de similarité de Levenshtein [Levenshtein V. 1966], la mesure du vecteur Cosinus [Zhu S. & Xia G. 2010], la mesure de divergence de Jensen-Shannon [Fuglede B. & Topsoe F. 2004]. Les résultats sont classés en fonction des scores numériques de ces mesures de similarités syntaxiques et d'un seuil défini par l'utilisateur.

5.3.2 Approches basées sur le modèle du processus

[Minor et al., 2007]

Les auteurs présentent une approche qui permet d'évaluer la distance entre deux Workflows représentés sous forme de graphes. La mesure de similarité proposée repose sur la similarité de structure du workflow, et la similarité de modèles de contexte. La première similarité prend en charge que les tâches séquentielles (le contrôle de flux « séquence »), et la distance entre deux structures de workflow (tâches séquentielles) est la somme des opérations

d'éditions sur les nœuds et les arcs des deux structures, cette distance est donnée par la formule suivante : $\delta(G1,G2)=|N1|+|E1|+|N2|+|E2|$ où les opérations d'éditions sont: N1 nœuds dans G1 , mais pas dans G2 ; N2 nœuds dans G2, mais pas dans G1 ; E1 arcs dans G1 , mais pas dans G2 ; E2 arcs dans G2 , mais pas dans G1. Deux nœuds sont équivalents ou égaux si les noms de leurs prédécesseurs et successeurs sont égaux, cette équivalence est basée seulement sur la comparaison syntaxique entre chaines de caractères.

Hybrid (Profile and Process)	[Corrales et al., 2008]	[Günay et al., 2010]	[Gater et al., 2010], [BenMokhtar et al., 2005], [Majithia et al., 2004]
Process Model	[Nejati et al., 2007], [Vander aalst et al 2006], [Minor et al., 2007], [Beeri et al., 2008], [Dijkman et al., 2009]	[Vaculin R. & Sycara K., 2007], [Ehrig M. Et al., 2007]	[Bansal et al., 2003]
combined (QoS and functional)	[Bernstein A. & Kiefer C., 2006]	[Chakraborty D. et al., 2001]	[Kiefer C. & Bernstein A., 2008], [Bianchini D. et al., 2006], [Abhijit et al., 2004]
IOPE	[Klein M. & Konig-Ries B., 2004], [Li Jing 2013]	[Stollberg M. et al., 2007], [Della Valle E. et al., 2005], [Rajesh et al., 2009], [Benatallah et al, 2005], [Kourtesis et al., 2008]	[Kaufer F. & Klusch M., 2006], [Sycara K. et al., 2002], [Klusch M. & Kapahnke P, 2008]
PE		[Botelho L. et al., 2008]	
E		[Di Noia et al., 2004], [Li L. & Horrocks I., 2003]	
IO	[Verma K. et al., 2005], [Constantinescu & Faltings, 2003]	[Jaeger M.C. et al., 2005], [Guilherme et al., 2012], [bellur et al., 2008], [Paolucci et al., 2002]	[Klusch M. et al., 2005]
QoS	[Vu L.H. et al., 2006]		[Fernandez A. et al., 2006]
	Non-logic	**Logic**	**Hybrid**

Table III.4 Synthèse de quelques approches existantes.

[Beeri et al., 2008]

Les auteurs proposent le langage BP-QL qui est à base de graphes pour l'interrogation des processus BPEL. BP-QL utilise le système Active XML comme une plate-forme de mise en œuvre. Une partie de données XML sont données explicitement. Quand une requête est évaluée sur un ensemble de documents, les appels relatifs à l'exécution de la requête sont dynamiquement invoquées, et que les données nécessaires sont matérialisées. En BP-QL, l'utilisation des appels de services AXML a pour objectif de localiser, en cas de besoin, les spécifications de processus distants, qui supportent le traitement distribué.

[Dijkman et al., 2009]

Les auteurs présentent quatre algorithmes qui évaluent la similarité entre processus de services. Les algorithmes proposés utilisent la technique de matching de graphes, et tentent d'établir une correspondance un-à-un entre les nœuds dans les modèles de processus comparés (trouver une relation bijective entre deux graphes). En outre, les auteurs proposent une similarité structurelle basée sur l'algorithme GED -graph edit distance- qui utilise les opérations d'édition pour calculer la similarité comportementale, et déterminent des mesures de similarité lexicale et syntaxique pour comparer entre les vocabulaires de différents modèles.

5.3.3 Approches hybrides :

[Corrales et al., 2008]

Les auteurs considèrent le problème d'équivalence de comportement entre deux processus de services web comme étant un problème d'appariement de graphes. Ils présentent un prototype qui prend en entrée deux modèles BPEL et évalue la distance sémantique entre eux. L'approche proposée présente une solution de découverte de services à base de leurs spécifications comportementales en BPEL. Elle se base en premier lieu sur la transformation de processus BPEL en graphe, ensuite de calculer la distance entre graphes pour mesurer leur similarité. La mesure de similarité proposée repose sur les opérations d'édition entre graphes qui peuvent être utilisées pour rendre un graphe G1 identique à un autre graphe G2, et à chaque opération d'édition est affecté un coût. La meilleure solution c'est celle qui présente un coût minimal.

5.4 Approches hybrides :

5.4.1 Approches basées sur le profile

[Klusch et Kapahnke, 2008]

SAWSDL-MX (première version) accepte en entrée des services spécifiés en SAWSDL [Farrell et Lausen, 2007]. Ce matchmaker est inspiré par les matchmakers OWLS-MX [Klusch et al, 2006] et WSMO-MX [Kaufer et Klusch, 2006]. La découverte utilise à la fois:

- une approche de matching logique basée sur le raisonnement par subsomption,
- une approche de matching syntaxique basée sur les techniques de recherche d'information.

Partant du fait qu'une description SAWSDL est une extension d'une description WSDL, le matching recouvre les éléments de description suivants :

- l'interface si on découvre un service spécifié en WSDL 2.0 et portType si on découvre un service spécifié en WSDL 1.1,
- operation, input, output : pour WSDL 1.1 et WSDL 2.0.

Pour réaliser le matching des interfaces, le matchmaker effectue des matching sur des graphes bipartis avec :

- des nœuds représentant des opérations d'un service,
- des arcs valués dont les valeurs sont calculées à partir des degrés de matching des opérations.

Pour réaliser le matching des opérations, le matchmaker utilise différentes techniques :

- basées sur des approches syntaxiques (ou textuelle) : Loss-of-Information, Extended Jaccard, Cosine et Jensen-Shannon. Ces scores appartiennent à l'intervalle [0,1]. En effet, chaque signature sémantique d'une opération de service est transformée en paire de vecteurs de mots-clés pondérés. nous aurons deux vecteurs : un vecteur pour les entrées et un autre pour les sorties, ces vecteurs sont comparés avec les vecteurs de la requête en utilisant les mesures de similarité citées ci-dessus.
- basées sur une approche logique en utilisant les annotations sémantiques spécifiées par les attributs modelReference dans les éléments annotés (les entrées et les sorties d'opérations).

Le Matchmaker SAWSDL-MX combine les deux types d'approches (similarité syntaxique ou textuelle et les filtres logiques), en effet les services ayant le même score logique seront classés avec la similarité textuelle.

Au moment de l'agrégation des degrés de matching des différentes opérations (i.e. le calcul de matching global de deux services), l'approche opte pour la plus petite valeur. Ceci est appelé le principe d'agrégation "Valeur minimale" (notée Min).

Par exemple, dans la Figure III.2 l'agrégation des deux degrés Exact et Plug-in, donne lieu au degré de matching final Plug-in.

Il est important de noter que cette version SAWSDLMX1 se concentre sur les annotations sémantiques de la signature d'opérations et ne prend pas en compte la structure d'arbre du document WSDL.

[Klucsh et al, 2009 b] propose un nouveau filtre nommé similarité structurelle dans la version SAWSDLMX2, pour comparer les arbres XML, ce filtre est basé sur WSDL-Analyzer présenté dans [Zinnikus et al, 2006].

La version SAWSDLMX2 applique trois algorithmes de matching pour une paire (requête-service), ces algorithmes sont les filtres logiques, la similarité textuelle, la similarité structurelle, les trois scores servent comme entrées pour entrainer un classificateur de type SVM (support vector machines).

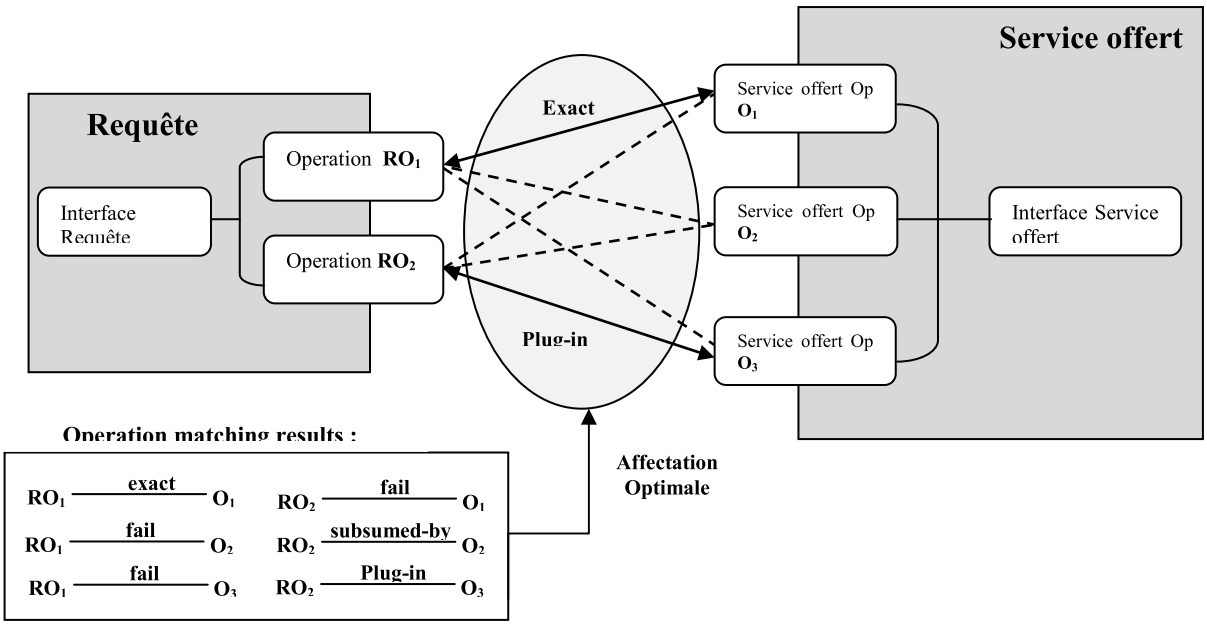


Figure III.2 Exemple d'application du principe d'agrégation de Valeur minimale [Klusch et Kapahnke, 2008]

Les expérimentations montrent que SAWSDLMX2 est plus performant en termes de précision et rappel, que le matchmaker logique, ou les mesures structurelles ou textuelles prises individuellement.

En plus, il est au moins aussi performant que SAWSDLMX1. Sa précision moyenne est 0.68, et son temps de réponse moyen est 7.9 sec (selon la compétition de sélection de services S3 édition 2009).

5.4.2 Approches basées sur le modèle du processus

[Bansal et al., 2003]

Les auteurs considèrent que le modèle de processus de service fournit une description beaucoup plus riche que son profile. Les algorithmes proposés dans cette approche exploitent l'information du modèle comportemental du service pour fournir des capacités de matching dans des situations où les algorithmes de matching classiques à base des propriétés IOPE sont incapables de déterminer des correspondances appropriées pour une requête utilisateur. Les modèles de processus DAML-S sont présentés comme des arbres, et le processus de matching implémente une stratégie de recherche par profondeur.

5.4.3 Approches hybrides :

[Gater et al., 2010]

Les auteurs utilisent la même architecture et les mêmes principes de l'approche proposée par [Corrales et al., 2008]. Ils proposent une approche de découverte de services OWL-S à base d'appariement de graphes. L'idée repose sur la transformation du modèle de processus OWL-S à un graphe orienté selon une spécification proposée qui prend en charge l'ensemble d'opérateurs de contrôles de flux. Le prototype proposé prend en entrée deux OWL-S process model et évalue leur distance sémantique en se basant sur les opérations d'édition de graphe où chaque opération a un coût. L'algorithme qui calcule GED « graph edit distance » est NP-complet [Bunke et al., 1994], et peut être implémenté par l'algorithme de recherche A*[4]. La mesure de similarité entre nœuds regroupe deux mesures, la première évalue la similarité linguistique entre les noms des nœuds du graphe, et la deuxième évalue la similarité sémantique des I/O à base d'ontologies.

6. Conclusion

Nous avons présenté dans ce chapitre les travaux existants qui traitent le problème de découverte des services web, nous avons mis en évidence les avantages et les inconvénients de chaque type d'approches.

[4] A* est un algorithme glouton de recherche de chemin dans un graphe

[Chebab Y., 2011] estime que les approches de matching les plus citées et les plus élaborées sont celles basées sur la logique (pure ou hybride). Le plus important est que ces mêmes travaux utilisent les langages de description de services les plus connus et répondus, à savoir SAWSDL, OWL-S et WSMO.

Nos contributions se basent essentiellement sur l'utilisation de l'ontologie de service OWL-S pour décrire les services web.

Notre première approche OWLS-SP s'inscrit dans la catégorie des approches logiques, et qui exploite les propriétés IO du profile OWL-S dans les opérations de matching.

Notre deuxième approche SGI_OWLS-SP s'inscrit dans la catégorie des approches hybrides, et qui formalise le « process model » de service sous forme de graphes pour réaliser un matching comportemental.

Notre troisième approche WTIA_OWLS-SP s'inscrit dans la catégorie des approches hybrides, et qui formalise le « process model » de service sous forme d'automates pour réaliser un matching comportemental.

Ces deux dernières approches réutilisent la première approche pour réaliser le matching entre les services atomiques.

Partie II :

Contributions

Notre contribution consiste principalement à offrir aux utilisateurs un framework de découverte en adoptant deux stratégies de matchmaking :

A. Matchmaking atomique réalisé par l'approche OWLS-SP
B. Matchmaking comportemental réalisé par l'approche SGI_OWLS-SP
C. Matchmaking comportemental réalisé par l'approche WTIA_OWLS-SP

Cette partie comporte deux chapitres qui exposent nos contributions dans la découverte des services web sémantiques.

Chapitre IV:

Approche de matchmaking OWLS-SP

Sommaire

1. Introduction

Dans cette section nous présentons les motivations de notre première contribution qui porte sur la découverte des services web atomiques styles OWL-S.

1.1 Performance de l'algorithme de Paolucci :

L'approche de Paolucci [Paolucci et al., 2002] s'inscrit dans la catégorie des approches logiques qui manipulent les concepts inputs et outputs du service profile de l'ontologie OWL-S dans le processus de découverte dynamique.

L'algorithme de matching de cette approche est un algorithme glouton -Greedy algorithm- qui propose de trouver un max-match entre chaque concept de la requête (entrées/sorties) et les concepts du service publié (entrées/sorties).

Les performances d'un algorithme sont mesurées en utilisant deux indicateurs à savoir la précision et le rappel, et qui manipulent trois variables :

Les vrais positifs sont les réponses correctes qui sont retournées par l'algorithme,

Les faux positifs sont les réponses incorrectes qui sont retournées par l'algorithme,

Les faux négatifs sont des réponses correctes qui ne sont pas retournées par l'algorithme.

Dans cette section, nous évaluons les performances de l'algorithme de Paolucci en présentant des exemples où certains résultats de matchmaking sont incorrects.

De plus, cet algorithme présente une ambiguïté où il ne décrit pas si un concept est retiré une fois qu'il a été sélectionné par le processus de matching ou non, nous présentons des exemples dans les deux scénarios.

1.2 Scénarios de motivation

Les concepts du service publié «A» et la requête « Q » sont définis par l'ontologie « book ontology» illustré dans la figure IV.1. On note le degré de correspondance par «dom», et le degré global de correspondance par « Gdom ».

a. le premier scenario: sans suppression des concepts du service publié.

Advertise 'A'

Input	publisher
Output	novel, price

Query 'Q'

Input	publisher
Output	romantic novel, science-fiction novel

liste-candidat={novel, price}

dom(romantic novel, novel)=exact=1.

dom(romantic novel, price)=fail=0.

dom(science-fiction novel, novel)=exact=1.

dom(science-fiction novel, price)=fail=0.

Gdom=exact.

Le matchmaker retourne 'A' comme réponse correcte à la requête 'Q' avec un degré de correspondance « exacte », tandis que 'A' représente un faux positif en réalité (en général, c'est le cas où on trouve deux concepts ou plus de la requête qui sont en correspondance avec un seul concept du service 'A').

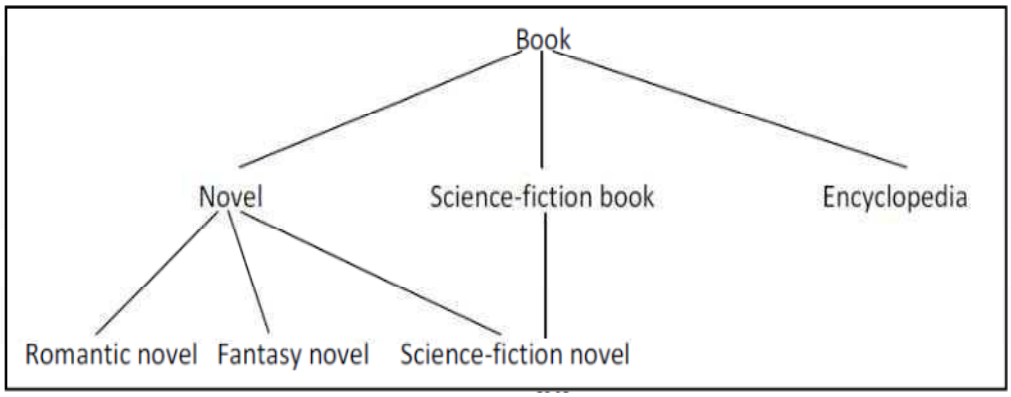


Figure IV.1: Partie de book ontology[1]

b. le deuxième scenario: avec suppression des concepts du service publié.

Advertise 'A'

Input	publisher
Output	novel, science-fiction book

Query 'Q'

Input	publisher
Output	science-fiction novel, romantic novel

list-candidat={ novel, science-fiction book }

dom(science-fiction novel, novel)=exact=1.

dom(science-fiction novel, science-fiction book)=1= exact.

Le concept 'Science-fiction novel' correspond au concept 'novel', et 'novel' est retiré de la liste des concepts candidats.

list-candidat={ science-fiction book }

dom(romantic novel, science-fiction book)=fail=0.

Le matchmaker ne retient pas le service 'A' comme bonne réponse à la requête 'Q' car il a trouvé un échec dans le processus de matching.

[1] OWL-S Service Retrieval Test Collection. http://projects.semwebcentral.org/projects/owls-tc/

On remarque que 'A' représente un faux négatif (l'ordre des concepts de la requête influence sur le degré de correspondance et peut changer par conséquence le Gdom).

Dans la littérature, certains travaux de recherche proposent une amélioration de cet algorithme. [Bellur U., et al., 2008] utilise l'algorithme hongrois pour déterminer le meilleur matching entre le service et la requête par des graphes bipartites. [J. Phatak et al., 2005] ajoute des mappings à base d'ontologie et des contraintes de qualité de service (QoS). [Michael C. et al. , 2004] propose un matching basé sur l'hiérarchie du profile de service.

2. L'approche OWLS-Shortest Path :

Notre proposition consiste à améliorer les performances du matchmaking des services atomiques OWL-S.

L'algorithme proposé s'inscrit dans la catégorie des approches logiques qui exploitent les propriétés input/output du service profile de l'ontologie OWL-S.

On propose de représenter le processus de matching par une matrice $M_{n,m}$, et de retenir les mêmes règles d'attribution de degré de correspondance (exact, plugin, subsumes, fail) et de les faire associer à des poids, qui sont des valeurs numériques définis de la façon suivante :

Degré de matching (dom)	Poids
Exact	W1=1
Plugin	W2=(w1*|M|)+1
Subsumes	W3=(w2*|M|)+1
Fail	W4=(w3*|M|)+1

Table IV.1: Pondération des degrés de matching de OWLS-SP

Où |M| est la dimension de la matrice M.

Ce sont les mêmes poids utilisés par Bellur et al., l'idée est d'écarter entre les poids pour servir notre objectif final qui utilise la notion du plus court chemin.

Définition « matrice de matching » : le matching est représenté par une matrice $M_{n,m}$, avec :

m_{ij}= dom(C_i,C_j) tels que C_i et C_j sont des concepts de l'ontologie de domaine.

par exemple, soit $M_{3,3}$ une matrice de matching des concepts "output" entre un service et une requête:

$$M = \begin{pmatrix} exact & plugin & fail \\ plugin & fail & exact \\ fail & subsumes & plugin \end{pmatrix} \; se\ traduit\ en, \; M = \begin{pmatrix} 1 & 4 & 40 \\ 4 & 40 & 1 \\ 40 & 13 & 4 \end{pmatrix}$$

2.1 Les étapes de l'algorithme de matching:

1. calculer la matrice de matching, dans le cas d'une matrice non carrée n>m alors n représente les lignes et m représente les colonnes.
2. Transformer la matrice de matching $M_{n,m}$ en un graphe G en tenant compte les règles suivantes:

a. Les sommets sont les m_{ij} de la matrice M, les arcs sont organisés par colonne (arc (c_i, c_{i+1})), et il n'est pas autorisé de créer un arc entre la première et la 3ème colonne.

b. Il n'est pas permis de connecter deux sommets de la même ligne ou de la même colonne.

c. Chaque sommet (élément) dans une colonne est connecté à tous les autres sommets (éléments) de la colonne suivante, si elle existe.

d. Le sommet « source » est connecté à tous les sommets de la première colonne.

e. Tous les sommets de la dernière colonne sont connectés au sommet « fin »

f. Les arcs sortants du sommet « source » sont pondérés par zéro « 0 »

g. Chaque arc Ai qui relie deux sommets n_i et n_j est pondéré par la valeur n_i où n_i est le nœud source de l'arc Ai. P_{ij} est le poids de la transition entre n_i et n_j, alors P_{ij}=poids(Ai(n_i,n_j))=n_i.

3. on cherche le plus court chemin dans G; l'algorithme de Dijkstra [Dijkstra, E.W., 1971] est appliqué avec la spécification de notre graphe G.
4. La solution optimale de la matrice de matching M est donnée par les sommets du plus court chemin, noté π.
5. Le plus court chemin est valide lorsque tous leurs sommets ne partagent pas entre eux ni la même ligne ni la même colonne dans $M_{n,m}$, il est constitué de nœuds indépendants.
6. La valeur retournée est le Gdom (Gdom_out équivalent à match.out, et Gdom_in équivalent à match.in dans l'algorithme de Paolucci) qui représente la longueur du plus court chemin π, Gdom=|π|.

Pour l'exemple précédent, la solution optimale est donnée par π tel que: $\pi=m_{21}m_{12}m_{33}$

π est un chemin valide, et Gdom=|π|=12.

Exemples: soit M1, M2, M3, M4 quatre matrices de matching, l'objectif est de les triées selon notre approche:

$$M1 = \begin{pmatrix} 1 & 4 & 40 \\ 40 & 1 & 13 \\ 4 & 1 & 40 \end{pmatrix}$$ $\pi=m_{11}m_{32}m_{23}$, Gdom_out=15

$$M2 = \begin{pmatrix} 4 & 1 & 13 \\ 1 & 13 & 4 \\ 13 & 4 & 1 \end{pmatrix}$$ $\pi=m_{21}m_{12}m_{33}$, Gdom_out=3

$$M3 = \begin{pmatrix} 1 & 13 & 40 \\ 4 & 1 & 4 \\ 40 & 1 & 13 \end{pmatrix}$$ $\pi=m_{11}m_{32}m_{23}$, Gdom_out=6

$$M4 = \begin{pmatrix} 1 & 4 & 40 \\ 40 & 4 & 13 \\ 4 & 13 & 40 \end{pmatrix}$$ $\pi=m_{31}m_{12}m_{23}$, Gdom_out=21

Les résultats retournés dans un ordre ascendant de Gdom_out selon l'ordre de mérite: M2, M3, M1, M4.

2.2 Algorithme de matchmaking:

Dans cette section, on présente la fonction de base de l'algorithme de matchmaking de notre approche ainsi que ses fonctions et ses procédures.

L'algorithme de matchmaking prend comme entrée la requête utilisateur Q constituée d'un ensemble de concepts d'entrées dont le nombre est card(Q_{in}) et un ensemble de concepts de sorties dont le nombre est card(Q_{out}). Il commence par balayer la base des services disponibles en vérifiant la compatibilité de cardinalité avant de procéder aux deux formes de matching (des sorties et des entrées) entre la requête et un service de la base.

Il retourne comme résultat un ensemble de services réponses où chaque service a un score de matching évalué à partir de la fonction d'agrégation proposée : Score[A, Q]=(5*Gdom_out + Gdom_in)/6. Cette fonction représente le résultat de matching entre la requête et un service donné dans la base des services publiés par une valeur numérique (un nombre), elle donne beaucoup d'importance au matching des sorties par rapport au matching des entrées ce qui explique le choix des coefficients de la pondération utilisés, à savoir 5/6 pour Gdom_out et 1/6 pour Gdom_in.

La quantification des deux formes de matching (sorties/entrées) par un nombre permet de classer les résultats trouvés et qui sont les réponses aux besoins de la requête selon notre approche. Les services sont rangés en ordre croissant de leur score de matching.

Matchmaking Algorithm

Input: Query Q

Output: set of services ranked in ascending order, called Result

Result=empty

For each service A_i in repository do

If card(Q_{out})>card(A_{out}) then Gdom_out=0

Else Gdom_out=Matching_output_concepts (Q_{out}, A_{out})

If Gdom_out ≠ 0 then If card(A_{in})>card(Q_{in}) then Gdom_in=0

Else Gdom_in= Matching_input_concepts (A_{in}, Q_{in})

If (Gdom_out≠0 and Gdom_in≠0 and not_exist_fail_in (Gdom_out)) then

*Score[A, Q]=(5*Gdom_out + Gdom_in)/6*

Append.Result(A_i, score[A_i])

End_for

Ranked Result in ascending order of score, the best score is which have the lowest value.

Return Result.

Algorithme IV.1 Algorithme de Matchmaking

La procédure « degree_of_match_out() » prend en entrée deux concepts qui sont Q_{out} (un concept de sortie de la requête), A_{out} (un concept de sortie du service publié), et retourne le degré de maching (dom) correspondant en utilisant une ontologie de domaine.

Degree_of_match_out()

Input: two concepts: Q_{out}, A_{out},

Output: dom, where dom={w1, w2 , w3 , w4}

If Q_{out} = A_{out} then return w1 //exact

If Q_{out} subclassOf A_{out} then return w1 //exact

If Q_{out} subsumed by A_{out} then return w2 //plugin

If Q_{out} subsumes A_{out} then return w3 //subsumes

Otherwise return w4 //fail

Algorithme IV.2 Assignation des degrés des sorties.

La procédure « degree_of_match_in() » prend en entrée deux concepts qui sont A_{in}(un concept d'entrée du service publié), Q_{in} (un concept d'entrée de la requête), et retourne le degré de maching (dom) correspondant en utilisant une ontologie de domaine.

Degree_of_match_in()

Input: two concepts: A_{in}, Q_{in}, Max(n,m)

Output: dom, where dom={ w1, w2 , w3 , w4}

If A_{in}=Q_{in} then return w1 //exact

If A_{in} subclassOf Q_{in} then return w1 //exact

If A_{in} subsumed by Q_{in} then return w2 //plugin

If A_{in} subsumes Q_{in} then return w3 //subsumes

Otherwise return w4 //fail

Algorithme IV.3 Assignation des degrés des entrées.

La procédure « Output_Matching_Matrix() » construit la matrice de matching des concepts de sorties entre la requête et le service publié. Cette procédure fait appel à la procédure match_out().

Output_Matching_Matrix()

Input: two set of output concepts: Q_{out}, A_{out} *//vectors*

Output: $M\_out_{n,m}$

For i= 1 to n do

For j= 1 to m do M[i,j]=degree_of_match_out(Q_{out}*[i],* A_{out}*[j])*

Return $M\_out_{n,m}$

Algorithme IV.4 matrice de matching des sorties.

La procédure « Input_Matching_Matrix() » construit la matrice de matching des concepts d'entrées entre le service publié et la requête. Cette procédure fait appel à la procédure match_in().

Input_Matching_Matrix()

Input: two set of input concepts: A_{in}, Q_{in} *// two vectors*

Output: $M\_in_{n,m}$

For i= 1 to n do

For j= 1 to m do M[i,j]=degree_of_match_in(A_{in}*[i],* Q_{in}*[j])*

Return $M\_in_{n,m}$

Algorithme IV.5 matrice de matching des entrées.

La Matching_output_concepts () est le cœur du matchmaker, elle permet de calculer la matrice de matching des concepts de sorties $M\_out_{n,m}$ (appel de la procédure Output_Matching_Matrix()) dont les lignes « n » représentent les concepts du service publié et les colonnes « m » représentent les concepts de la requête (avec n≥m car la solution est constituée à partir des affectations de colonnes). Elle permet aussi de transformer la matrice de matching $M\_out_{n,m}$ en graphe orienté et pondéré G. L'étape suivante consiste à déterminer le plus court chemin π dans G dont les nœuds représentent les affectations de la solution optimale du matching. Le résultat retourné par cette procédure est le Gdom_out qui accepte comme valeur soit zéro (0) soit π la longueur du plus court chemin dans G.

Matching_output_concepts () // each Q_{out} *needs to be matched with* A_{out}

Input: m concepts of Q_{out}, *n concepts of* A_{out} *(n≥m).*

Output: Gdom_out // the global degree of matching for output concepts

1. *Call the procedure: Output_Matching_Matrix(), this step returns* $M\_out_{n,m}$
2. *If all elements of a line in the matrix* $M\_out_{n,m}$ *equal to W4 then Gdom_out=0,*

goto (5).

3. *Call the procedure: matrix_to_weighted_di-graph($M\_out_{n,m}$) which transforms the $M\_out_{n,m}$ to weighted di-graph G.*
4. *Call the procedure: Dijkstra algorithm for shortest path(graph G) //search an optimal matching (π) which represents the shortest path in G.*
5. *Gdom_out=$|\pi|$*
6. *Return Gdom_out, END.*

Algorithme IV.6 Evaluation de Gdom des sorties.

Le même travail effectué avec le matching des concepts de sorties est reproduit avec le matching des concepts d'entrées en tenant compte que les colonnes représentent les concepts du service publié. Le résultat retourné par cette procédure est le Gdom_in qui accepte comme valeur soit zéro (0) soit π la longueur du plus court chemin dans G.

Matching_input_concepts () // each A_{in} needs to be matched with Q_{in}
Input: m concepts of A_{in}, n concepts of Q_{in} ($n \geq m$).
Output: Gdom_in // the global degree of matching for input concepts

1. *Call the procedure: Input_Matching_Matrix(), this step returns $M\_in_{n,m}$*
2. *If all elements of a line in the matrix $M\_in_{n,m}$ equal to W4 then Gdom_in=0,*
   goto (5).
3. *Call the procedure: matrix_to_weighted_di-graph($M\_in_{n,m}$) which transforms the $M\_in_{n,m}$ to weighted di-graph G.*
4. *Call the procedure: Dijkstra algorithm for shortest path(graph G) //search an optimal matching (π) which represents the shortest path in G.*
5. *Gdom_in=$|\pi|$*
6. *Return Gdom_in, END.*

Algorithme IV.7 Evaluation de Gdom des entrées.

La procédure Matrix_To_Weighted_Di-Graph($M_{n,m}$) permet de transformer une matrice de matching $M_{n,m}$ (des sorties ou des entrées) à un graphe orienté et pondéré G en respectant un ensemble de règles.

Matrix_To_Weighted_Di-Graph($M_{n,m}$)
Input: $M_{n,m}$
Output: weighted di-graph G.

- *The vertices are mij, the arcs are organized by column (arc (ci, ci +1)), and it is not allowed to create an arc between the first and the 3rd column.*

- *It is not allowed to connect two vertices of the same line or column.*
- *The source vertex of G is connected to all the vertices of the first column.*
- *Each vertex (element) in a column is connected to all other vertices (elements) of the next column, if exist.*
- *The terminus vertex of G is connected to all the vertices of the last column.*
- *Each arc Ai that connects two vertices ni and nj is weighted by the ni value where ni presents the source of this arc. Pij is the weight of the transition between nodes ni and nj, where Pij = ni (P_{ij}= weight ($Ai(n_i,n_j)$)=n_i).*
- *Outgoing arcs of the source vertex are weighted by zero "0".*

Algorithme IV.7 Règles de mapping entre la matrice de matching et le graphe.

On cherche le plus court chemin dans G en utilisant l'algorithme de Dijkstra [Dijkstra, E.W., 1971] qui est appliqué en tenant compte la spécification de notre graphe G.

La solution optimale de la matrice de matching M est donnée par les sommets du plus court chemin, noté π. Le plus court chemin est valide lorsque tous leurs sommets ne partagent pas entre eux ni la même ligne ni la même colonne dans $M_{n,m}$, il est constitué de nœuds indépendants.

Dijkstra algorithm for shortest path:

Input: weighted di-graph G.

Output: shortest path denoted π,

1. $T \leftarrow \emptyset$
2. *For* $v \in V$ *do* $d(v) \leftarrow \infty$
3. $d(s) \leftarrow 0$
4. *while* $(T \neq V)$

   $v \leftarrow rgmin\{d(u): u \notin T\}$

   $T \leftarrow T \cup \{v\}$

   For $u \in voisins(v)$ *do* $d(u) \leftarrow min\{d(u), d(v) + w_{vu}\}$
5. *Return optimal matching expressed by π, END.*

Algorithme IV.8 Algorithme de Dijkstra

La complexité temporelle de notre algorithme de matchmaking est calculée par les formules suivantes:

On dénote :

Card(Adv), le nombre de services publiés dans la base des services.

Card(A_{out}), le nombre de concepts « output » dans le service.

Card(A_{in}), le nombre de concepts « input » dans le service.

Card(Q_{out}), le nombre de concept « output » dans la requête.

Card(Q_{in}), le nombre de concept « input » dans la requête.

La complexité temporelle de « *Matchmaking Algorithm* » est d'ordre $O(n^3)$.

La complexité temporelle de « *Dijkstra algorithm for shortest path* ($M_{n,m}$) » est d'ordre $O(n^2)$.

La complexité temporelle de « *Matching output concepts(vector* Q_{out}, *vector* A_{out}*)* » est d'ordre $O(n^2)$.

La complexité temporelle de « *Output Matching Matrix(vector* Q_{out}, *vector* A_{out}*)* » est d'ordre $O(n^2)$.

La complexité temporelle de « Degree of match_out(Q_{out}, A_{out}) » est d'ordre O(1).

La complexité est de classe polynomiale et en particulier d'ordre $O(N^3)$, alors le matchmaker de OWLS-SP est un algorithme efficace rapide de complexité cubique.

2.3 Expérimentation:

Dans cette section, nous implémentons notre algorithme de matchmaking, et nous utilisons les outils suivants:

OWL-S API (pour lire et invoquer les requêtes, les services, et les ontologies). OWLS-TC comme base de tests qui contient un ensemble de services OWL-S, un ensemble d'ontologies OWL, un ensemble de requêtes OWL-S et ensemble de fichiers WSDL.

L'architecture de notre application est illustrée dans la figure IV.2.

Pour mesurer la précision de notre algorithme, nous utilisons une collection de services Web (OWLS-TC V3). Cette collection compte plus de 500 services couvrant plusieurs domaines d'application (à savoir : communication, économie, éducation, médical, transport,…).

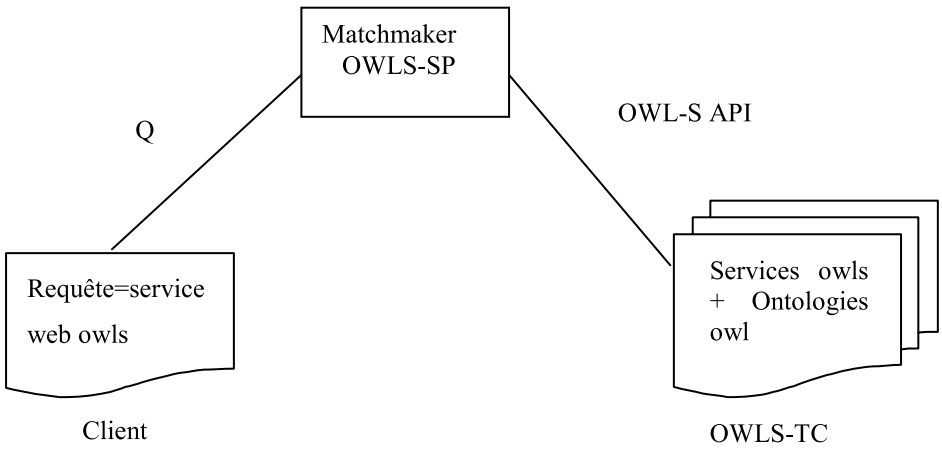


Figure IV.2: L'architecture du matchmaker OWLS-SP

Dans cette section, l'objectif est de mesurer l'efficacité de notre algorithme et de comparer nos résultats avec les deux algorithmes: Paolucci et Bellur. À cet effet, nous avons fait une série de tests (requêtes et services) pour l'évaluation de la performance des trois matchmakers.

Requêtes:

Q1: Car_price_service

Input: Car

Output: Price

Q2: GROCERYSTORE_BUTTERSELLING_SERVICE

Input: GROCERYSTORE

Output: BUTTER, SELLING

Q3: book_taxedprice_service_modified

Input: Book

Output: TaxedPrice, RECOMMENDEDPRICE

Q4: bicyclecar_priceyear_service

Input : CAR, BICYCLE

Output : PRICE, YEAR

Q5: bookprice service0

Input: BOOK, NOVEL, PUBLICATION

Output: RECOMMENDEDPRICE, PRICE, TAXEDPRICE

Q6: bookprice service4

Input: ROMANTICNOVEL, MONOGRAPH, FANTANSYNOVEL

Output: GENRE, EUROPETAXEDPRICE, RECOMMENDEDTAXED-PRICE

Q7: bookprice service5

Input: PRINTEDMATERIAL, FANTANSYNOVEL, GRADE

Output: PRICE, RECOMMENDEDPRICE, MAXPRICE

Q8: bookprice service3

Input: ENCYCLOPEDIA, MONOGRAPH, SHORTSTORY

Output: MAXPRICE, TAXEDPRICE, EUROPETAXED-PRICE

Q9: bookprice service1

Input: SERIALPUBLICATIONS, BOOK, SCIENCEFICTION-SHORTSTORY

Output: PRICE, TAXEDPRICE, RECOMMENDEDPRICE-INDOLLAR

Q10: bookprice service2

Input: ENCYCLOPEDIA, MONOGRAPH, ROMANTICNOVEL

Output: PRICE, TAXEDPRICE, MAXPRICE

Ontology:

C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\ontology\books.owl

C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\ontology\concept.owl

Dans l'objectif de bien montrer l'apport de notre approche, on a enrichie la base de test avec des services qui présentent plusieurs entrées/sorties (07 requêtes ajoutées, car Q1, Q2, Q4, Q5 existent dans la base OWLS-TC3 et 27 services OWL-S avec plusieurs entrées/sorties). Cet ajout est justifié par le fait que la majorité des services OWLS-TC V3 comportent une seule entrée/sortie (un seul concept input/output).

Figure IV.3: L'interface OWLS-SP

Les résultats détaillés sont présentés dans la table IV.2.

	S1	S2	S3	S4	S5
inputs	SERIALPUBLICA--TIONS BOOK SCIENCEFICTION--SHORTSTORY	ENCYCLOPEDIA MONOGRAPH ROMANTICNOVEL	ENCYCLOPEDIA MONOGRAPH SHORTSTORY	ROMANTICNOVEL MONOGRAPH FANTANSYNOVEL	PRINTEDMATERIAL FANTANSYNOVEL GRADE
outputs	PRICE TAXEDPRICE RECOMMENDEDPRICE-INDOLLAR	PRICE TAXEDPRICE MAXPRICE	MAXPRICE TAXEDPRICE EUROPETAXED-PRICE	GENRE EUROPETAXEDPRICE RECOMMENDEDTAXED-PRICE	PRICE RECOMMENDEDPRICE MAXPRICE
Matrice in	0.0 1.0 0.7 0.0 0.3 0.0 1.0 0.7 0.7	1.0 0.3 0.7 0.0 0.3 1.0 0.7 1.0 0.7	1.0 0.3 1.0 0.0 0.3 0.0 0.7 1.0 0.7	0.7 0.3 0.7 1.0 0.3 1.0 0.7 1.0 0.7	0.3 0.7 0.0 0.3 1.0 0.0 0.3 0.7 0.0

Matrice out		1.0 1.0 1.0 0.0 0.3 1.0 0.3 0.3 0.0	1.0 1.0 1.0 0.0 0.3 1.0 0.0 0.3 0.0	0.0 0.3 0.0 0.0 0.3 1.0 0.0 0.3 0.3	0.0 0.0 0.0 0.0 0.3 0.3 0.0 0.3 0.3	1.0 1.0 1.0 1.0 0.3 0.0 0.0 0.3 0.0
Paulucci with remove	IN	0.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	1.0 0.0 0.0 0.0 0.3 0.0 0.0 0.0 0.7	0.7 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	0.0 0.7 0.0 0.3 0.0 0.0 0.0 0.0 0.0
	OUT	1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.3 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.3 0.0	0.0 0.3 0.0 0.0 0.0 1.0 0.0 0.0 0.0	0.0 0.0 0.0 0.0 0.3 0.0 0.0 0.0 0.3	1.0 0.0 0.0 0.0 0.3 0.0 0.0 0.0 0.0
Paulucci without remove	IN	0.0 1.0 0.0 0.0 0.3 0.0 1.0 0.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	1.0 0.0 0.0 0.0 0.3 0.0 0.0 1.0 0.0	0.7 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0	0.0 0.7 0.0 0.0 1.0 0.0 0.0 0.7 0.0
	OUT	1.0 0.0 0.0 0.0 0.0 1.0 0.3 0.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.3 0.0	0.0 0.3 0.0 0.0 0.0 1.0 0.0 0.3 0.0	0.0 0.0 0.0 0.0 0.3 0.0 0.0 0.3 0.0	1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.3 0.0
Hungarian	IN	0.0 0.0 10.0 0.0 91.0 0.0 1.0 0.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	1.0 0.0 0.0 0.0 91.0 0.0 0.0 0.0 10.0	10.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	91.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 820.0
	OUT	1.0 0.0 0.0 0.0 0.0 1.0 0.0 91.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 91.0 0.0	820.0 0.0 0.0 0.0 0.0 1.0 0.0 91.0 0.0	820.0 0.0 0.0 0.0 91.0 0.0 0.0 0.0 91.0	0.0 0.0 1.0 1.0 0.0 0.0 0.0 91.0 0.0
OWLS-SP	IN	0.0 0.0 4.0 0.0 13.0 0.0 1.0 0.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	1.0 0.0 0.0 0.0 13.0 0.0 0.0 0.0 4.0	4.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0	13.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 40.0
	OUT	1.0 0.0 0.0 0.0 0.0 1.0 0.0 13.0 0.0	1.0 0.0 0.0 0.0 0.0 1.0 0.0 13.0 0.0	40.0 0.0 0.0 0.0 0.0 1.0 0.0 13.0 0.0	40.0 0.0 0.0 0.0 13.0 0.0 0.0 0.0 13.0	0.0 0.0 1.0 1.0 0.0 0.0 0.0 13.0 0.0

Table IV.2 Résultats expérimentaux.

		S1	S2	S3	S4	S5
Paulucci With Remove	Match.in	0≡fail	1≡exact	0..3≡ subsumes	0.7≡plugin	0≡fail
	Match.out	0.3≡ subsumes	0..3≡ subsumes	0≡fail	0≡fail	0≡fail
	Ranking	2	1	-	-	-
Paulucci without Remove	Match.in	0≡fail	1≡exact	0≡fail	0≡fail	0≡fail
	Match.out	0≡fail	0..3 ≡subsumes	0≡fail	0≡fail	0≡fail
	Ranking	-	1	-	-	-
hungarian	Match.in	91≡ subsumes	1≡exact	91 ≡subsumes	10≡plugin	820≡fail
	Match.out	91 ≡subsumes	91≡ subsumes	820≡fail	820≡fail	91 ≡subsumes
	Ranking	2	1	-	-	3
OWLS-SP	Match.in	18	3	18	6	54
	Match.out	15	15	54	66	15
	Score	15.5	13	48	56	21.5
	Ranking	2	1	-	-	3

Table IV.3: Partie des résultats expérimentaux.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
<!ENTITY books.owl "http://127.0.0.1/ontology/books.owl">
<!ENTITY owl "http://www.w3.org/2002/07/owl#">
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
<!ENTITY simplified_sumo.owl "http://127.0.0.1/ontology/simplified_sumo.owl">
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<rdf:RDF xml:base="&books.owl;"
xmlns:owl="&owl;"
xmlns:rdf="&rdf;"
xmlns:rdfs="&rdfs;">

<!-- Ontology Information -->
<owl:Ontology rdf:about=""
rdfs:label="Book Ontology"
owl:versionInfo="1.0">
<rdfs:comment>An ontology containing information about books</rdfs:comment>
<owl:imports>
<owl:Ontology rdf:about="&simplified_sumo.owl;"/>
</owl:imports>
</owl:Ontology>

<!-- Classes -->
<owl:Class rdf:about="#A">
<rdfs:subClassOf rdf:resource="#Grade"/>
</owl:Class>

<owl:Class rdf:about="#Article">
<rdfs:subClassOf rdf:resource="#Text"/>
</owl:Class>

<owl:Class rdf:about="#Author">
<rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>

<owl:Class rdf:about="#B">
<rdfs:subClassOf rdf:resource="#Grade"/>
</owl:Class>

<owl:Class rdf:about="#Book">
<rdfs:subClassOf rdf:resource="#Monograph"/>
<rdfs:subClassOf>
<owl:Restriction rdf:nodeID="b10">
<owl:allValuesFrom rdf:resource="#Author"/>
<owl:onProperty rdf:resource="#writtenBy"/>
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:allValuesFrom rdf:resource="#Book-Type"/>
<owl:onProperty rdf:resource="#hasType"/>
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:allValuesFrom rdf:resource="#Title"/>
<owl:onProperty rdf:resource="#isTitled"/>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:about="#Book-Type"/>
<owl:Class rdf:about="#C">
<rdfs:subClassOf rdf:resource="#Grade"/>
</owl:Class>
```

Algorithme IV.9: Fragment de l'ontologie "book"

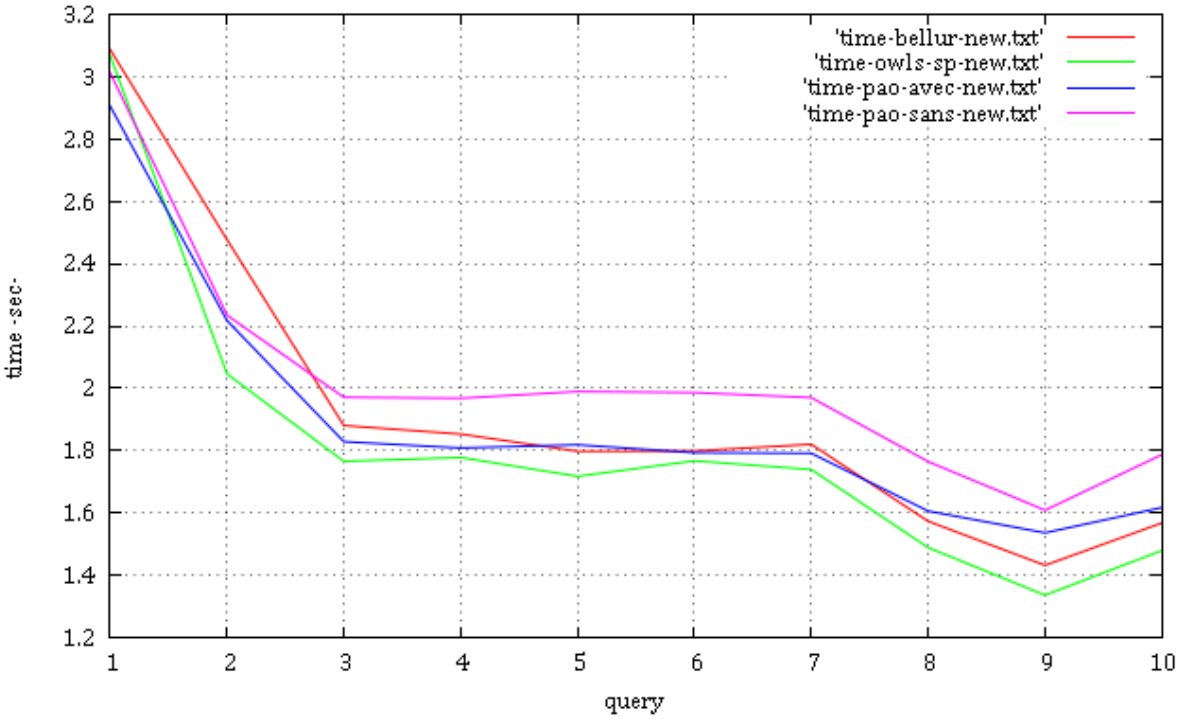


Figure IV.4: Evaluation du temps d'exécution des quatre approches.

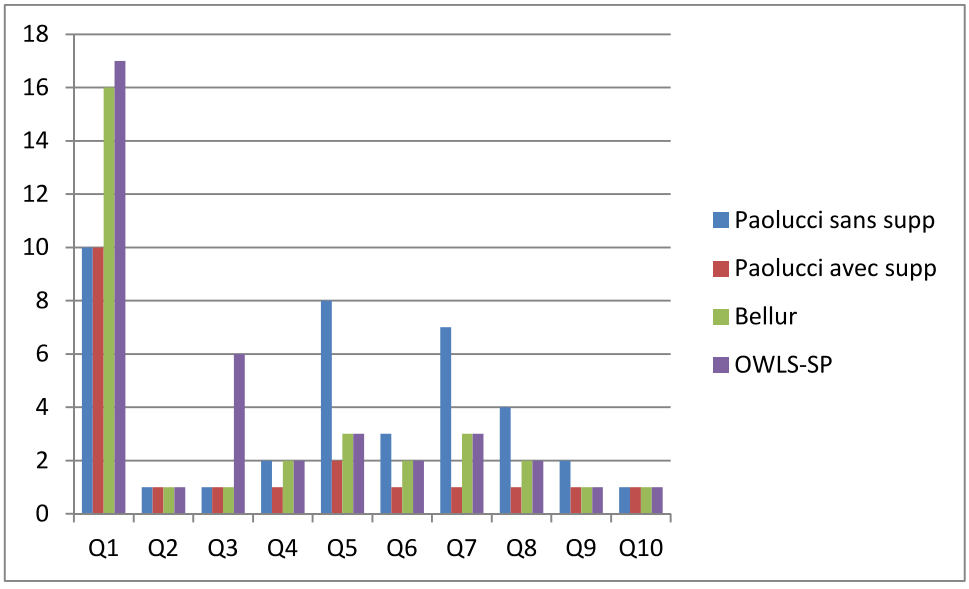


Figure IV.5: Nombre de réponses des quatre approches.

Histogramme de la figure IV.5 présente le nombre de réponses des quatre approches pour chaque requête. Nous concluons que notre approche OWLS-SP et l'approche de Bellur présentent de meilleurs performances que le matchmaker de Paolucci. Les résultats de notre algorithme qui utilise le principe de plus court chemin sont similaires à ceux retournés par l'algorithme de Bellur qui utilise la méthode hongroise pour améliorer la qualité du matchmaking. La différence réside dans la complexité des deux algorithmes.

	L'approche de Paolucci	L'approche de Bellur	Notre approche: OWLS-SP
Dom	exact, plugin, subsume, fail	exact=w1=1 plugin=w2=(w1*|v|)+1 subsumes=w3=(w2*|v|)+1	exact=w1=1 plugin=w2=(w1*|M|)+1 subsumes=w3=(w2*|M|)+1 fail=w4=(w3*|M|)+1
Algorithme de Matching	Greedy algorithm	Hungarian algorithm	Shortest path algorithm
Ranking	Descending order of Gdom_out	Gdom=max-weight edge in Graph	Gdom=Length of shortest path in Graph
Complexité	$O(N^3)$	$O(N^4)$	$O(N^3)$

Table IV.4: Comparaison des trois approches

3. Conclusion

La qualité des résultats du processus de découverte est basée principalement sur l'exactitude « correctness » du matchmaker utilisé. Nous avons proposé un algorithme de matchmaking qui résout les problèmes de l'algorithme Paolucci en utilisant l'algorithme du plus court chemin qui détermine la correspondance optimale entre la requête de l'utilisateur et le service de fournisseur. La complexité de l'algorithme en question est polynômiale de l'ordre $O(N^3)$, ce résultat est meilleur que la complexité de l'algorithme de Bellur qui utilise la méthode hongroise et elle est délimitée par $O(N^4)$. Nous avons effectué des expériences pour valider notre approche et d'analyser l'amélioration de la précision obtenue par notre algorithme basé sur la comparaison des résultats avec les autres approches (Paolucci et Bellur). Enfin, nous avons développé l'outil « OWLS-SP Discovery » comme implémentation de notre approche.

Chapitre V:

Matchmaking comportemental

Sommaire

1. Introduction

La découverte de services est le processus d'identification et de localisation de services les plus similaires de la requête en se basant sur la description sémantique des propriétés fonctionnelles et/ou non-fonctionnelles. Le matchmaking à base du profile de service (inputs/output) n'est pas suffisant quand il s'agit de services composites, et ceci peut être facilement prouvé quand le contexte d'utilisation du service est en particulier une composition ou une substitution. Pour surmonter cette limite, nous proposons deux approches de découverte basées sur le comportement, la première à base d'isomorphisme de sous-graphe et la deuxième à base d'automate d'interface pondéré. Les deux approches exploitent l'approche OWLS-SP [khater et al., 2014] pour calculer le degré de similarité sémantiques à base de concepts I/O de services OWL-S. L'objectif est de vérifier l'adéquation et la compatibilité de la structure comportementale entre deux modèles de services sémantiques.

Nous croyons que le matching basé uniquement sur le profil de service (inputs/output) ne suffit pas dans le cas des services composites où il est possible que des sorties ne sont produites que dans des conditions internes spécifiques.

Le Matching à base I/O ne garantit pas que les services composites retournés présentent le même ordre de messages échangés tel qu'il est décrit par le modèle comportemental de la requête. Par conséquent, les services les plus similaires retournés à l'utilisateur échouent de réaliser ses besoins et il doit développer un adaptateur pour réconcilier la communication selon ses préférences.

[J.C Corrales et al., 2008] présente dans sa thèse de doctorat deux exemples où il est nécessaire d'effectuer un matchmaking comportemental pour trouver des services qui peuvent satisfaire le besoin de l'utilisateur. Les résultats retournés et en comparant avec un matchmaking input/output sont plus performants et minimisent le coût de développement pour l'utilisateur ainsi que les problèmes d'invocation (sans passer par une étape d'adaptation entre le service trouvé et la requête).

Le premier scénario expose la problématique de découverte des services dans un contexte d'intégration d'applications, il consiste à retrouver les services ayant un comportement compatible. Le second scénario montre qu'une approche de matchmaking comportemental qui peut être utilisée comme un moyen pour quantifier des similarités/dissimilarités entre deux modèles, non seulement dans le contexte de la découverte des services, mais également dans d'autres applications, comme le « delta-analyse ».

Le premier scenario « Intégration des services web »: On considère une entreprise qui utilise un service S pour commander des fournitures. Supposons que l'entreprise veut trouver

des partenaires ayant des services web compatibles. Les séquences d'échange de messages acceptés sont appelés protocoles de conversation. La spécification du protocole de conversation est importante, car il arrive rarement que les opérations de service peuvent être invoquées indépendamment les uns des autres. Ainsi, l'entreprise recherche un service ayant un protocole de conversation compatible. Après avoir trouvé des partenaires, le plus compatible doit être choisi parmi eux. Si le service n'est pas compatible, l'entreprise doit soit adapter son service ou développer un adaptateur pour que son application soit conforme avec le comportement du service trouvé. Dans les deux situations, des modèles de processus doivent être comparés automatiquement afin de déterminer leurs similitudes ou différences en termes de leurs protocoles de conversation.

La recherche des services en tenant compte que des entrées/sorties peuvent être utilisés seulement comme un premier filtre parce qu'elle ne garantit pas que les services trouvés prennent en charge le même ordre de messages échangés.

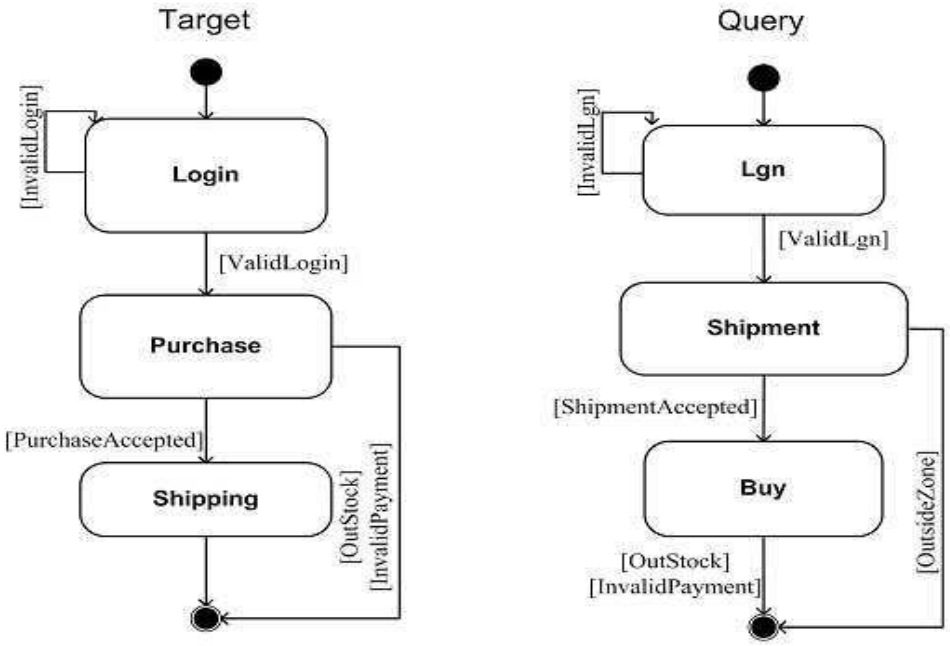


Figure V.1 L'ordonnancement des opérations dans la requête et le service cible [J.C Corrales et al., 2008]

Le deuxième scenario « Delta analyse » *:* le domaine d'application pour ce scénario consiste à déterminer les différences entre les deux modèles. Parmi les défis de l'entreprise est de vérifier si le processus mis en œuvre conforme à une norme. Ainsi, ils ont besoin de comparer le modèle de conversation de leur service existant avec celle prescrite par les normes. Pour les grands modèles et surtout lorsque l'entreprise n'utilise pas le vocabulaire standard, un outil devrait aider les utilisateurs à identifier toutes les différences entre les deux

modèles. Sur la base de ces différences, le coût de la réingénierie du service existant pourrait être mieux évalué.

Le matchmaker des services composites doit prendre en charge les propriétés comportementales de ces services lors du processus de matching.

2. Approche à base de graphe matching:

Dans cette partie, nous concentrons notre étude sur l'utilisation d'isomorphisme de sous-graphe afin de vérifier l'adéquation de la structure comportemental entre deux modèles de processus OWL-S de services composites.

2.1 Travaux connexes (graph based matching) :

Dans la littérature, plusieurs approches ont été proposées pour mesurer la similarité entre graphes, elles sont fondées sur un ensemble de techniques telles que « l'isomorphisme de graphe », « isomorphisme de sous-graphe», «le plus grand sous-graphe commun», et «distance d'édition de graphe ».

Nous avons présenté dans la section 5 du chapitre 3 une vue générale de différentes approches qui traitent le problème de découverte à savoir les approches comportementales à base de correspondance de graphes ([Bansal et al., 2003], [Minor et al., 2007], [Corrales et al., 2008], [Gater et al., 2010], [Beeri et al., 2008], [Dijkman et al., 2009]).

L'idée de base est de trouver une meilleure correspondance entre deux graphes, ceci se réfère à trouver une correspondance entre les nœuds et les flux de contrôle d'un graphe G1 avec les nœuds et les flux de contrôle d'un autre graphe donné G2 en fonction d'un ensemble de contraintes d'optimalité.

Notre but dans cette première approche est de détecter un isomorphisme de sous-graphe partiel entre deux modèles de processus OWL-S présentés comme des graphes orientés.

2.2 Algorithme de matching comportemental:

Dans cette section, nous présentons l'algorithme qui détecte l'existence d'isomorphisme de sous-graphe entre deux services composés. Cet algorithme se base sur les résultats d'une fonction de mapping entre les nœuds des deux graphes (motif et cible).

1 On calcule les degrés (degré-in et degré-out) de chaque nœud dans les deux graphes, cible et motif, (pour le nœud v, degré(v)=(degré-in(v), degré-out(v))).

2 On définit une fonction de mapping «f» entre les nœuds des deux graphes, f : V'→V. soit $u'_i \in V'$, tel que degré(u'_i)=(x'_i, y'_i) où x'_i=degré-in(u'_i) et y'_i=degré-out(u'_i), et $u_j \in V$, tel que degré(u_j)= (x_j, y_j) où x_j=degré-in(u_j) et y_j=degré-out(u_j). la construction de la

fonction f se fait suivant la règle suivante : M = $f(u'_i, u_j) = \{u_j \mid x'_i \leq x_j \; et \; y'_i \leq y_j\}$, f retourne l'ensemble des nœuds candidats dans le graphe cible qui satisfont la règle précédente et ceci pour chaque nœud dans le graphe motif.

3 Pour chaque ensemble de mapping mk=f(Vi) où Vi ∈ Gm graphe motif, faire:
Calculer l'ensemble de voisinage N pour chaque nœud Vi, et associer les degrés à chaque nœud dans N.
Calculer pour chaque nœud Ui ∈mk (mk=f(Vi)) l'ensemble de voisinage M, et associer les degrés à chaque nœud dans M.

4 Filtrage à base de la relation de voisinage: cette étape consiste à raffiner les résultats retournés par la fonction de mapping f. pour chaque élément "ev" dans l'ensemble de voisinage du nœud Vi avec Vi∈Gm, vérifier l'existence de correspondance de degré « injective » avec un élément "eu" appartenant à l'ensemble de voisinage de Ui avec Ui ∈ f(Vi) (on élimine un nœud U où U∈mk=f(Vi) s'il n'existe pas de correspondance injective de degré entre l'ensemble de voisinage de Vi et l'ensemble de voisinage de U).

5 On développe la relation « succ » entre les instances de la fonction de mapping m_i. la règle est de préserver le même séquencement d'origine entre les nœuds du graphe motif. $f(u'_i) = m_i$, succ (m_i)= {m_k | m_k= $f(s'_i)$ et ∃a∈A, $u'_i \rightarrow s'_i$ }.

6 Filtrage à base des relations successeur "succ" et prédécesseur "pred": après la construction de la relation « succ » entre les instances de mapping mi, cette étape consiste à raffiner les résultats de la fonction f. On supprime les nœuds qui ne vérifient pas la relation « succ » et la relation « pred » entre les instances de mapping en se basant sur la défintion de la relation « succ » entre les nœuds du graphe cible.

(a) Succ(f(A))= $\bigcup_{vi \in f(A)}\{ui \mid ui \in mk \; et \; vi \rightarrow ui\} \cap mk$, avec f(A)→mk

(b) Pred(mk)= $\bigcup_{ui \in mk}\{vi \mid vi \in f(A) \; et \; vi \rightarrow ui\} \cap f(A)$, avec f(A)→mk
L'intuition derrière ces deux règles est de supprimer les nœuds orphelins dans les ensembles de mapping (nœud sans successeur ou prédécesseur).

Cet algorithme raffine au maximum les résultats de la fonction de mapping f.

2.3 Matching des nœuds atomiques:

Les nœuds d'un graphe qui représente un service web composé sont des services atomiques, et l'appariement entre deux nœuds de deux graphes se réfère à l'appariement entre deux services atomiques. Nous réalisons cette tâche en utilisant notre approche OWLS-SP déjà développée dans un précédent travail [khater et al., 2014]. Dans le chapitre 3, nous avons

présenté un algorithme de matchmaking qui résout les problèmes de découverte de services atomiques en utilisant l'algorithme du plus court chemin qui détermine la correspondance optimale entre la requête d'utilisateur et le service de fournisseur. La complexité de l'algorithme proposé est polynômiale et elle est bornée par $O(N^3)$.

Algorithme de Matching des nœuds atomiques:

L'algorithme de matching entre les concepts Input/Output d'un service A et d'une requête Q utilise les fonctions suivantes: Matching_output_concepts (Q_{out}, A_{out}), Matching_input_concepts (A_{in}, Q_{in}), et l'algorithme de plus court chemin après la transformation des deux matrices en graphes orientés et pondérés.

Matching Atomic Services Algorithm: OWLS-SP

Input: two atomic services Ni and Nj // Ni=Q and Nj=A

Output: Score of matching

Score= $W_{max}=|A|^{|Q|}$ *// |A| and |Q| are the number of nodes in A and Q respectively*

If card(Q_{out}*)>card(*A_{out}*) then Gdom_out=0*

Else Gdom_out=Matching_output_concepts (Q_{out}*,* A_{out}*)*

If Gdom_out ≠ 0 then If card(A_{in}*)>card(*Q_{in}*) then Gdom_in=0*

Else Gdom_in= Matching_input_concepts (A_{in}*,* Q_{in}*)*

If (Gdom_out≠0 and gdom_in≠0 and not_exist_fail_in (Gdom_out) and

not_exist_fail_in (Gdom_in)) then Score=(α**Gdom_out +* β**Gdom_in)/(*α*+*β*)*

//α*=5,* β*=1 in our experiment*

Return Score.

Algorithme V.1: Matching des services atomiques.

Matching_output_concepts (Q_{out}, A_{out}) est la fonction qui calcule la solution optimale de matching entre les concepts de output du service fournisseur A et le service de la requête Q, et retourne la valeur de matching Gdom_out ($|\pi_{output}|$) qui est affecté à la variable Gdom_out.

Matching_input_concepts (A_{in}, Q_{in}) est la fonction qui calcule la solution optimale de concepts d'entrée correspondant entre la publicité et le service de la requête et retourne la valeur de correspondance ($|\pi_{intput}|$) qui est affecté à la variable Gdom_in.

Les deux fonctions utilisent l'algorithme de plus court chemin pour déterminer la meilleure affectation des concepts entre la requête et le service.

2.4 Algorithme de Matchmaking comportemental

L'algorithme prend comme argument la requête d'utilisateur Q qui est un service composite, et scanne la base de services composites en enregistrant que les services pertinents qui peuvent répondre à la demande Q (GSOM de matching différent de zéro -le cas d'échec-).

L'algorithme de Matchmaking repose sur deux fonctions :

1. Matching(Q,A): prend comme argument deux services composites A et Q, et retourne leur score globale de matching -Gsom-, cette fonction se base sur quatre fonctions:

a. Set_mapping_f Subgraph_Isomorphism(Q, A):

Cette fonction implémente l'algorithme de matching comportemental de la section 2.3 de ce chapitre, elle vérifie l'adéquation de structure entre deux modèles de processus OWL-S de services composites Q et A, et retourne en réponse la fonction de mapping f: Q→A. Cette étape est importante pour garantir que les services composites retrouvés supportent le même ordre de messages échangés tel qu'il est défini par la requête.

b. Score OWLS-SP(NQi, NAj): cette fonction calcule le score de matching entre les nœuds des deux graphes qui représentent des services atomiques. Cette fonction implémente l'algorithme de matching proposé dans le chapitre 4 (OWLS-Shortest Path).

c. Directed_Graph Matrix_to_graph($M_{n,m}$): cette fonction transforme la matrice M, dont les colonnes sont les nœuds de la requête et les lignes sont les nœuds de service fournisseur, à un graphe orienté et pondéré suivant les règles suivantes :

- Lignes "n" sont les nœuds du graphe cible,
- Colonnes "m" sont les nœuds du graphe motif,
- Les sommets sont m_{ij}, les arcs sont organisés par colonne (arc (C_i, C_{i+1})), il est autorisé seulement de créer des arcs entre une colonne C_i et la colonne suivante C_{i+1}. Il n'est pas autorisé de créer des arcs entre les nœuds de la même ligne ou de la même colonne. Le sommet « source » est connecté avec tous les sommets de la première colonne. Chaque sommet dans une colonne est relié avec tous les sommets de la colonne suivante en respectant les règles précédente. Le sommet « terminus » est relié avec tous les sommets de la dernière colonne.
- Chaque arc Ai qui relie deux sommets ni et nj est pondéré par la valeur de ni où ni est le sommet départ ou source de l'arc Ai. P_{ij} est le poids de la transition entre ni et nj, avec P_{ij}=ni
- Les arcs sortants du sommet source sont pondérés par la valeur zéro "0".

Cette fonction retourne un graphe G orienté et pondéré à partir de la matrice des scores issue de l'algorithme OWLS-SP.

d. Shortest_Path Dijkstra_algorithm_for_shortest_path (G):

Cette fonction explore le graphe orienté étiqueté en identifiant les nœuds qui constituent le plus court chemin à partir du nœud "source" au nœud "terminus". Il s'agit d'appliquer l'algorithme de Dijkstra en tenant compte la spécification du graphe G. un chemin est valide s'il est constitué de nœuds indépendants qui ne partagent pas les mêmes lignes et les mêmes colonnes. La solution optimale du matching est donnée par les nœuds du plus court chemin qui présentent la meilleure affectation entre les concepts de la requête et les concepts du service fournisseur. Le plus court chemin est dénotée par π.

2. Ranking(result): cette fonction classe les résultats dans l'ordre croissant du GSOM, le meilleur service est celui qui présente la plus faible valeur de GSOM.

Score Global de matching -Gsom-: Gsom est utilisé pour classer les résultats retournés par l'algorithme du matchmaking, il s'agit de la longueur du plus court chemin π : $\mathrm{Gsom} = |\pi|$

Matching Algorithm

Input: two composite services Q and A

Output: Gsom(Q,A), the global score of matching between two composites service Q and A

Gsom=0;

Set_mapping_f Subgraph_Isomorphism (Q, A);

If not_exist NQi where f(NQi) is empty then begin

For each node NQi in Q do // each node in Q represents a column m

For each node NAj in A do // each node in A represents a line n

If (NAj ∈ f(NQi)) then M[i,j]=OWLS-SP(NQi, NAj);

Else M[i,j]=W_{max}=$|A|^{|Q|}$ // |A| and |Q| are the number of nodes in A //and Q respectively

End_for

End_for

Directed_Graph G=Matrix_to_graph($M_{n,m}$) // transform the matrix M to a graph

Shortest_Path Dijkstra_algorithm_for_shortest_path (G);

If (not_exist_W_{max}_in(π)) then Gsom=|π|; //π is the shortest path in G and |π| its length

End_if

Return Gsom,

END.

Algorithme V.2: Matching des services composés.

Matchmaker Algorithm

Input: composite service Q

Output: Result Ranked in ascending order of Gsom, the best Gsom is which have the lowest value

Result=empty

For each composite service A_k in repository do

Gsom=Matching(Q, A_k);

If (Gsom ≠ 0) then Append.Result(A_k, Gsom[A_k]);

End_For

Ranking(Result); // this function ranks Result in ascending order of Gsom

End.

Algorithme V.3: Algorithme de Matchmaking.

2.5 Expérimentation

Pour mesurer la précision de notre algorithme, nous utilisons un ensemble de services Web (OWLS-TC), qui comprend plus de 500 services couvrant plusieurs domaines d'application, pour créer notre petite base de services composites OWL-S.

On a fait le recours à cette solution pour surmonter l'inexistence de base de test pour les services OWL-S composites. Nous avons mis en œuvre l'algorithme de matchmaking décrit précédemment en utilisant le langage Java, et des bibliothèques existantes comme OWL-S API.

Exemple de requête:

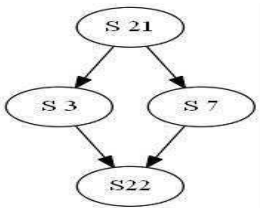


Figure V.2: Service requête.

S21: PERSON_ADDRESS_SERVICE

Input : PERSON, Affiliated-Person, Legal-Agent

Output : Town, ADDRESS, Postal-Address

S3: CAR_PRICE_SERVICE

Input : CAR, SportsCar, RacyCarDriver

Output : PRICE, TaxedPriceInEuro, MaxPrice

S7: TOWNCOUNTRY_HOTEL_SERVICE

Input : COUNTRY, TOWN, Web-Site

Output : LuxuryHotel, HOTEL, BudgetHotelDestination

S22: COFFEESANDWICH_PRICE_SERVICE

Input : SANDWICH, COFFEE, PreparedFood

Output : PRICE, TaxedPrice, MAXPRICE

***Ontologies**:*

books.owl, concept.owl, Mid-level-ontology.owl, portal.owl, travel.owl,…

Le degré « exact » de découverte a pour Gsom:

Gsom_exact=((card(output)*α+card(input)*β)/6)*card(Query)

α=5, β=1, card(output)=card(input)=3, W_{exact}=1, alors:

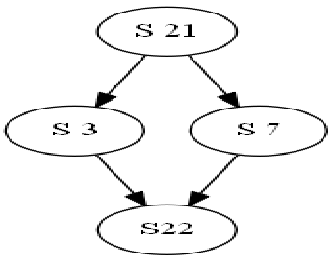
Gsom_exact=12,

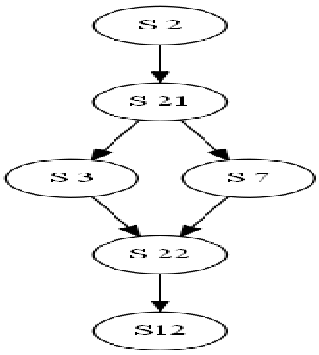
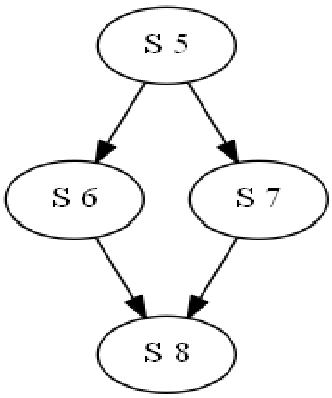
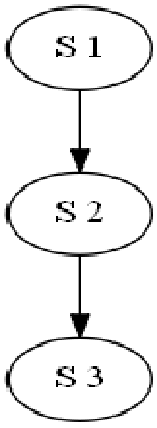
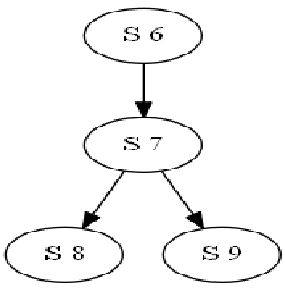
Score_owls-sp_fail= $W_{max}=|A|^{|Q|}=|A|^4$,

Gsom_fail si le plus court chemin (π) contient un poids W_{max}.

Nous avons créé un petit ensemble de services composites (une petite base de test) qui contient plus de 28 services OWL-S où chaque service composite comprend un nombre aléatoire de services atomiques ordonnancés de manière aléatoire.

Les résultats retournés, par notre matchmaker après le traitement de la requête Q, montrent clairement l'efficacité de notre méthode de découverte des services composites qui se base en premier lieu sur la vérification de la compatibilité comportementale, et en second lieu sur un matching logique des concepts input/output entre les services atomiques.

Si	service Composite	∃(iso)	Mapping function f	Matching Matrix	Gsom
S1		True	map(S21)= {S21} map(S3)= {S3,S7} map(S7)= {S3,S7} map(S22)= {S22}	3.0 256 256 256 256 3.0 20.0 256 256 120 3.0 256 256 256 256 3.0	12.0 Exact

S2		True	map(21)= {21} map(3)= {3,7} map(7)= {3,7} map(22)= {22}	1296 1296 1296 1296 3.0 1296 1296 1296 1296 3.0 120 1296 1296 120 3.0 1296 1296 1296 1296 3.0 1296 1296 1296 1296	12.0 Exact
S3		True	map(21)= {5} map(3)= {6,7} map(7)= {6,7} map(22)= {8}	256 256 256 256 256 256 120 256 256 120 3.0 256 256 256 256 256 ∃ W_{max} in the matching of atomic services and owls-sp() returns the value W_{max} =256 for M_{11}, M_{22}, and M_{44}, then Gsom=0, not returned,	0 Fail
S4		False			Fail
S5		False			Fail

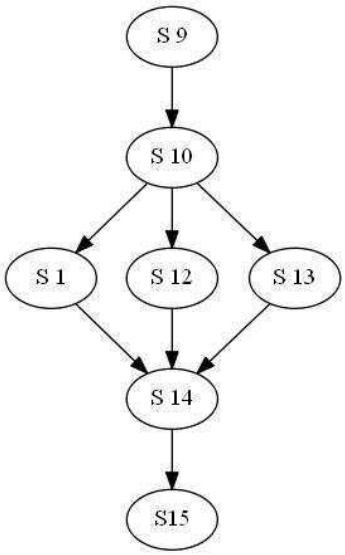
S6		True	map(21)= {10} map(3)= {1,13,12} map(7)= {1,13,12} map(22)= {14}	2401 2401 2401 2401 2401 2401 2401 2401 2401 22.5 120.0 2401 2401 120.0 2401 2401 2401 120.0 120.0 2401 2401 2401 2401 32.5 2401 2401 2401 2401 ∃ W_{max} in the matching of atomic service and owls-sp() returns the value W_{max} =2401 for M_{21} and M_{33}, then Gsom=0, not returned,	0 Fail

Table V.1: Partie des résultats de notre approche.

La structure result={S1 avec Gsom=12, S2 avec Gsom=12}, notons que Gsom=12 représente le degré de matching « exact » pour cette requête.

2.6 Conclusion

Nous croyons que le matchmaking basé uniquement sur les propriétés du profile de service (entrées/sorties) ne suffit pas dans le cas des services composites où existe des sorties qui sont produites que dans des conditions internes spécifiques.

I/O Matching ne garantit pas que les services composites retournés prennent en charge le même ordre de messages échangés décrites par le modèle comportemental de la requête. L'objectif principal de notre approche est de développer un matchmaker sémantique qui exploite aussi les propriétés comportementales de services. On utilise le modèle de processus de services composites dans la première étape de l'algorithme afin de détecter la correspondance comportementale en se basant sur une technique qui s'inspire de l'isomorphisme de sous-graphe. Dans la seconde étape de l'algorithme de découverte proposé, nous effectuons l'appariement des nœuds du modèle de processus (services atomiques) basé à OWLS-SP matchmaker (OWLS-Shortest Path, voir chapitre 4).

Notre matchmaker traite la découverte des services composites en utilisant un matching un-à-un entre les nœuds des graphes qui représentent des services OWL-S atomiques. On propose

comme perspective de cette approche, d'étudier d'autres types de matching entre les nœuds de deux graphes, comme par exemple 1-n et n-m.

La difficulté de l'expérimentation de cette approche réside dans l'absence d'une base standard de test. Cette limitation est bien reconnue dans ce domaine [Cuzzocrea A. et al., 2011, Gater et al., 2010].

3. Approche basée sur les automates d'interface typés et pondérés (WTIA)

3.1 Introduction

L'utilisation des automates dans cette deuxième approche permet d'améliorer les deux propriétés suivantes :

A. Le niveau de détail de l'échange : la représentation du comportement par des graphes où les arcs ne sont pas étiquetés ne permet pas de raffiner le niveau de détail dans l'échange entre les services. Dans les graphes, les nœuds sont des services atomiques, par contre avec les automates, on peut exprimer facilement le protocole d'échange entre les services (opération d'envoie avec des paramètres, opération de réception,…).
B. La complexité temporelle : l'isomorphisme de sous-graphe est connu comme un problème NP. Notre approche précédente utilise les graphes comme modèle comportemental, et pour remédier au souci de la complexité d'isomorphisme, nous avons utilisé une fonction de correspondance entre les deux graphes (cible et motif). Cette solution devient couteuse en temps d'exécution si le nombre de nœuds (services atomiques) évolue de manière exponentielle.

Dans cette section, nous nous concentrons sur la description du comportement du service Web via le formalisme d'automate d'interface pondéré et typé qui fusionne deux modèles d'automates, les automates d'interface et les automates pondérés. Le formalisme des automates d'interface a été introduit par [Henzinger et Alfaro, 2001], il permet de décrire l'ordre temporel des interactions fonctionnelles de service. Dans notre approche, l'interface d'une opération est donnée par les types de ses entrées et sorties, et chaque transition est étiquetée par les propriétés non-fonctionnelles (temps de latence, la disponibilité, la réputation, le coût ...).

La pondération de l'automate représentant le comportement du service est importante dans l'étape de sélection qui consiste à déterminer les meilleurs services en fonction des préférences de l'utilisateur. Elle est également impliquée dans le choix d'un remplacement d'un service défaillant.

Notre objectif est de modéliser les services web sémantiques via le formalisme des automates pondérés et typés qui offre une représentation comportementale et sémantique des services. En se basant sur le modèle WTIA, deux relations de similarité sont mises en place afin de développer des algorithmes de découverte et d'adaptation de services web sémantiques.

3.2 Automate d'interface pondéré et typé (WTIA):

L'automate est un modèle bien connu dans l'informatique théorique, il peut présenter les interactions d'un système en se basant sur des états et des transitions étiquetées. L'automate d'interface pondéré et typé provient de l'émergence de deux types d'automates qui sont les automates d'interface et les automates pondérés par des entiers. Automates d'interface ont été introduits par [Henzinger et Alfaro, 2001] et [chouali et al., 2010] pour la description des composants. Dans notre approche, l'automate d'interface typée est un automate d'interface où chaque transition est dénotée par la signature de l'opération de service Web.

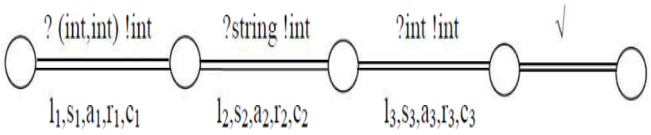


Figure V.3: Automate d'interface pondéré et typé -WTIA-

Un automate pondéré par des entiers est un automate dont chaque transition est munie d'un poids dont la valeur appartient à Z. Dans notre formalisme chaque transition est munie d'un ensemble de poids qui expriment des mesures des propriétés non fonctionnelles (PNF ou QoS).

3.3 Modélisation du comportement de service

Nous proposons un modèle comportemental des services web sémantiques qui combine entre le formalisme d'automate d'interface pondéré et typé pour représenter les interactions d'échange de service et le profile OWL-S par ses propriétés Input/Output pour décrire la sémantique de service.

Définition: Formellement, un service web est représenté par un automate d'interface pondéré et typé (WTIA), service web=(S, P, F, S_0, T), avec:

S est un ensemble fini d'états.

P est le service profile qui contient les concepts I/O du service et sa trace d'exécution.

$S_0 \in$ S est l'état initial.

T $\subseteq$ S est un ensemble fini d'états finaux.

F est la fonction de transition: $(S \times action \times Z^n) \rightarrow S$.

Action décrit l'alphabet de l'automate WTIA et définie par:

$?m_1!m_2$: décrit la réception de message m_1 et l'émission de message m_2

?m: décrit la réception de message m

!m: décrit l'émission de message m

τ: décrit une action interne non-observable de l'extérieur

e: décrit le lancement d'une exception e

√: décrit la terminaison de l'exécution

Z^n : décrit les poids QoS pour chaque opération comme (temps d'exécution, fiabilité, disponibilité, réputation, coût).

Fig.V.3 présente le formalisme d'automate d'interface pondéré et typé **-WTIA-**

Définition: les propriétés non-fonctionnelles (PNF) incluent les attributs de qualité de service comme latence, fiabilité, disponibilité, réputation, et coût.

Les fonctions d'agrégation [Alrifai et al, 2012] des attributs QoS dans le cadre d'une composition de services sont données dans la table V.2. Les propriétés de QoS adoptées sont:

Temps de réponse : c'est la période qui s'écoule entre l'envoi de la requête et la réception des résultats.

Coût : c'est le prix fixé par le fournisseur du service.

Disponibilité : c'est la probabilité d'accessibilité du service. (Le pourcentage de temps au cours duquel le service est accessible).

Fiabilité : le nombre d'invocations réussies sur le nombre total d'invocations.

Réputation : elle est mesurée avec les feedbacks des utilisateurs.

Attribut de QOS	Fonction d'agrégation (n est la taille de la composition)			
	Séquence	Boucle	Parallèle	Choix conditionnel
Temps d'exécution	Q1' (C) =$\sum^{n}_{j=1}$ Q1(sj)	Q1' (C) =$\sum^{n}_{j=1}$ Q1(s)	Q1' (C) = $MAX^{n}_{j=1}$ Q1(sj)	Q1' (C) = $MAX^{n}_{j=1}$ Q1(sj)
Réputation	Q2' (C) = 1/n * $\sum^{n}_{j=1}$ Q2(sj)	Q2(s)	Q2' (C) = 1/n * $\sum^{n}_{j=1}$ Q2(sj)	Q2' (C) = $MIN^{n}_{j=1}$ Q2(sj)
Cout	Q3' (C) = =$\sum^{n}_{j=1}$ Q3(sj)	Q3' (C) = =$\sum^{n}_{j=1}$ Q3(s)	Q3' (C) = =$\sum^{n}_{j=1}$ Q3(s_j)	Q3' (C) = $MAX^{n}_{j=1}$ Q3(sj)
Fiabilité	Q4' (C) = $\Pi^{n}_{j=1}$ Q4(sj)	Q4' (C) = $\Pi^{n}_{j=1}$ Q4(s)	Q4' (C) = $\Pi^{n}_{j=1}$ Q4(sj)	Q4' (C) = $MIN^{n}_{j=1}$ Q4(sj)
Disponibilité	Q5' (C) = $\Pi^{n}_{j=1}$ Q5(sj)	Q5' (C) = $\Pi^{n}_{j=1}$ Q5(s)	Q5' (C) = $\Pi^{n}_{j=1}$ Q5(sj)	Q5' (C) = $MIN^{n}_{j=1}$ Q5(sj)

Table V.2: Les fonctions d'agrégation de valeurs de QOS [Alrifai et al, 2012].

Pour une exécution séquentielle « ; » des opérations op_i :

Latence=$\sum_{i=1}^{n} lat(op_i)$,

Fiabilité =$\prod_{i=1}^{n} saf(op_i)$,

Disponibilité =$\prod_{i=1}^{n} aval(op_i)$,

Réputation =$\frac{1}{n} * \sum_{i=1}^{n} rep(op_i)$,

Coût =$\sum_{i=1}^{n} cost(op_i)$.

Le profile de service est défini par: profile = (entrées, sorties, O), où: Entrées et sorties décrivent un ensemble de concepts de l'ontologie de domaine désigné par O. le profile présente l'interface de service qui est attachée à une ontologie de domaine.

Le formalisme d'automate d'interface pondéré et typé décrit formellement l'activité d'un service (la séquence d'opérations) avec la possibilité de mesurer certaines propriétés non-fonctionnelles (mémoire, temps de latence, fiabilité, disponibilité, réputation, coût).

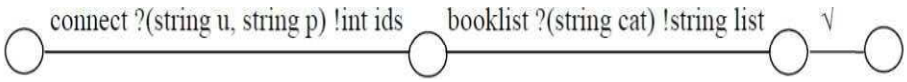


Figure V.4: Automate de service « librairie »

Définition -Trace- *:* on appelle trace de service, la séquence d'opérations exécutées d'un chemin réussi de l'état « début » à l'état « fin » (l'ordre d'exécution des opérations).

La trace du service de la figure V.4 est donnée par :

Trace=connect ?(string u , string p) !int ids ; booklist ?(string cat) !string list; √

On prend en charge deux opérateurs d'ordonnancement d'opérations:

L'opérateur « ; » qui désigne la relation successeur, dans la figure V.4 booklist() est le successeur de connect ().

L'opérateur « + » qui désigne la relation parallèle, $op_1()$ + $op_2()$ signifie que $op_1()$ s'exécute en parallèle que $op_2()$.

Raffiner la trace d'exécution revient à supprimer les noms des opérations et les noms de paramètres, tels que:

Trace (WTIA service) =?(string , string) !int ; ?(string) !string; √

On utilise cette forme de trace dans le processus de découverte.

3.4 Sémantique formelle du modèle WTIA:

Afin de formaliser le processus WTIA, nous présentons les règles opérationnelles associées à certains constructeurs du langage BPEL [Hadad et al., 2006].

Le processus vide (représente le processus qui ne fait rien) est donné par la règle:

$$empty \longrightarrow \surd \qquad [1]$$

le processus ?m (resp !m) correspond à l'opération de réception du message m (resp. l'envoie du message m), il exécute l'action ?m (resp. !m) et le processus devient « empty ».

$$*m \xrightarrow{*m} \text{empty avec } * \in \{?, !\} \quad [2]$$

Le processus $?m_1!m_2$ correspond à l'exécution d'une opération qui fait la réception du message m_1 et l'envoie du message m_2, il exécute l'action $?m_1\ !m_2$ et le processus devient « empty ».

$$?m_1\ !m_2 \xrightarrow{?m_1\ !m_2} \text{empty} \quad [3]$$

Le processus P;Q, qui correspond à l'exécution du processus p suivi par l'exécution du processus Q, il devient le processus P';Q si P devient P' après l'exécution d'une action act.

$$\forall\ act \neq \surd \qquad \frac{P \xrightarrow{act} P'}{P;Q \longrightarrow P';Q} \quad [4]$$

Dans le cas où P a terminé son exécution (P exécute l'action « $\surd$ ») alors le processus séquentiel P;Q devient le processus Q après l'exécution d'une action interne « τ ».

$$\frac{P \xrightarrow{\surd}}{P;Q \xrightarrow{\tau} Q} \quad [5]$$

En tenant compte de ces règles, l'exemple précédent de la figure V.3 dont la trace est :

Trace = ?(int,int) !int ; ?string !int ; ?int !int ; $\surd$

La trace peut être représentée par: P;Q;R;S

La sémantique opérationnelle de cet exemple est:

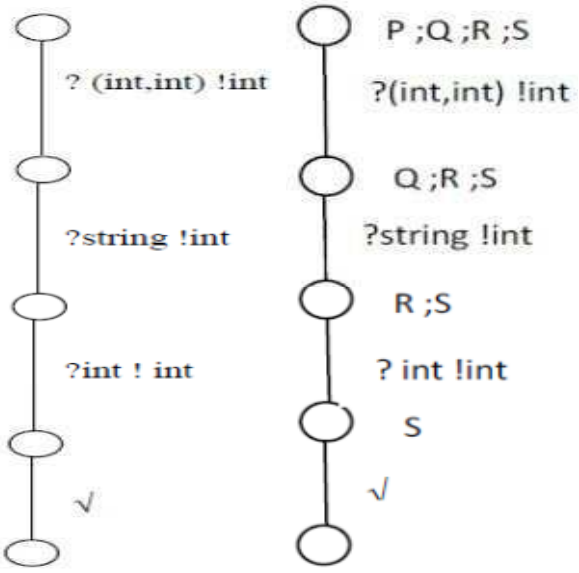


Figure V.5: (a) Trace WTIA (b) Sémantique opérationnelle

3.5 Proximité non-fonctionnelle:

Qualité de service (QoS) pour les composants et services à travers un ensemble de propriétés non fonctionnelles (PNF) sont largement étudiées dans la littérature [Héam P-C, 2009; Jérome

V. et al., 2011].

Notre objectif est d'expliciter l'intégration des attributs QoS dans le modèle de représentation des services web sémantiques.

Définition: deux services sont non-fonctionnellement similaires (QoS-similaires) s'ils présentent les mêmes propriétés non-fonctionnelles (identiques ou approximatives).

Les attributs QoS dans le modèle WTIA sont définis par un vecteur, noté PNF, où PNF = (latence, fiabilité, disponibilité, réputation, coût)^t.

Comparer les propriétés QoS de deux services revient à comparer leurs vecteurs PNF.

Définition: soit la relation R, dénotée par ≤, on dit que $PNF_1 \leq PNF_2$ si et seulement si :

$$\begin{cases} 0 \leq latence1 \geq latence2 \\ 0 \leq fiabilité1 \leq fiabilité2 \\ 0 \leq disponibilité1 \leq disponibilté2 \\ 0 \leq réputation1 \leq réputation2 \\ 0 \leq coût1 \geq coût2 \end{cases}$$

Notons que $PNF_1 \leq PNF_2$ signifie que le service qui présente les QoS par le vecteur PNF_2 est meilleur que le service qui présente les QoS par le vecteur PNF_1.

Le théorème1 fournit un résultat important de décidabilité pour les automates finis pondérés par des entiers.

Un automate pondéré par des entiers est dit k-ambigu si pour tout mot il existe au plus k chemins réussis d'étiquette ce mot. Un automate 1-ambigu est dit non-ambigu. Un automate est dit finiment ambigu s'il existe un k tel qu'il soit k-ambigu [Héam P-C, 2009].

Notons que dans notre modèle WTIA l'automate est déterministe et non-ambigu.

Théorème 1: soient A1 et A2 deux automates pondérés par des entiers. On ne peut pas décider si pour tout chemin réussi π_1 de A1 il existe un chemin réussi π_2 de A2 tel que coût(π_1) ≤ coût(π_2) [Krob D. 1994]. Ce problème est décidable en tems polynomial si A1 et A2 sont finiment ambigus [Hashiguchi K. et al., 2002 ; Weber A. et al, 1994].

Lemme: décidable en temps polynomial pour tester si pour chaque chemin réussi $\pi1 \in L(A)$, il existe un chemin réussi $\pi2$ de A2 tel que : $PNF_{A2}(\pi2) \geq PNF_{A1}(\pi1)$ si A1 et A2 sont finiment ambigus.

Preuve:

Soient A_1 et A_2 sont deux WTIA automates finiment ambigus (ou non-ambigus), π_1 décrit le chemin réussi de A_1 avec le vecteur QoS PNF_1, π_2 décrit le chemin réussi de A_2 avec le vecteur QoS PNF_2. On procède la démonstration par composante du vecteur PNF en appliquant le théorème 1.

Selon le théorème 1:

Pour la première composante du vecteur QoS concernant le temps de latence: décidable en temps polynomial pour tester l'existence de π_2 tel que: latence($A_1(\pi_1)$) ≥ latence($A_2(\pi_2)$).

Pour la deuxième composante du vecteur QoS concernant la fiabilité: décidable en temps polynomial pour tester l'existence de π_2 tel que: fiabilité($A_1(\pi_1)$) ≤ fiabilité($A_2(\pi_2)$)

Pour la troisième composante du vecteur QoS concernant la disponibilité: décidable en temps polynomial pour tester l'existence de π_2 tel que: disponibilité($A_1(\pi_1)$) ≤ disponibilité($A_2(\pi_2)$)

Pour la quatrième composante du vecteur QoS concernant la réputation: décidable en temps polynomial pour tester l'existence de π_2 tel que: réputation($A_1(\pi_1)$) ≤ réputation($A_2(\pi_2)$)

Pour la cinquième composante du vecteur QoS concernant le coût: décidable en temps polynomial pour tester l'existence de π_2 tel que: Coût($A_1(\pi_1)$) ≥ Coût($A_2(\pi_2)$)

Alors selon la relation R «≤» on a: $PNF_{A2}(\pi_2) \geq PNF_{A1}(\pi_1)$.

3.6 Processus de découverte:

La tâche de découverte est la capacité de localiser automatiquement des services qui répondent aux besoins spécifiques d'une requête utilisateur, cette tâche nécessite la mise en correspondance de la requête utilisateur et les services publiés dans l'annuaire (disponible).

Notre idée est de développer une nouvelle approche pour la découverte de services basée sur deux mesures de similarité, la première est basée sur l'approche OWLS-SP, et la deuxième assure la compatibilité comportementale en se basant sur la trace du modèle WTIA.

3.6.1 Similarité sémantique:

Dans cette section, nous présentons les principaux principes de notre précédente approche OWLS-SP qui constitue la première partie de l'algorithme de découverte. La Similarité sémantique entre deux services est mesurée par l'algorithme de matchmaking de l'approche OWLS-SP du chapitre 4.

I/O Matching Algorithm

Input: two TIWA services Ni and Nj // Ni=Q and Nj=A

Output: Score of matching

Q_{out} *= Extract_of_profile(output) //* Q_{out} *output concepts of the query*

A_{out} *= Extract_of_profile (output) //* A_{out} *output concepts of the Advertise*

Q_{in} *= Extract_of_profile (input) //* Q_{in} *input concepts of the query*

A_{in} = Extract_of_profile (input) // A_{in} input concepts of the Advertise

Score= $W_{max}=|A|^{|Q|}$ // $|A|=max(|A_{out}|, |A_{in}|)$ and $|Q|= max(|Q_{out}|, |Q_{in}|)$

If card(Q_{out})>card(A_{out}) then Gdom_out=0

Else Gdom_out=Matching_output_concepts (Q_{out}, A_{out})

If Gdom_out ≠ 0 then If card(A_{in})>card(Q_{in}) then Gdom_in=0

Else Gdom_in= Matching_input_concepts (A_{in}, Q_{in})

*If (Gdom_out≠0 and gdom_in≠0 and not_exist_fail_in (Gdom_out) and not_exist_fail_in (Gdom_in)) then Score=(α*Gdom_out + β*Gdom_in)/(α+β) //we use α=5, β=1*

Return Score.

End.

Algorithme V.4: Matching des services atomiques.

Les détails de cet algorithme sont déjà présentés dans le chapitre 4.

3.6.2 Similarité comportementale:

Cette mesure consiste à utiliser la trace d'exécution de WTIA et la distance de Levenshtein pour vérifier l'équivalence de comportement entre les deux automates de services.

behaviour Matching:

Input: two WTIA automata Q and A

Output: Score of matching

Score=-1;

String Tr_Q=Extract_trace(Q);

String Tr_A=Extract_trace(A);

Double distance = ((double) leveishtein(Tr_Q, Tr_A))/(max(Tr_Q.length(), Tr_A.length()));

If (distance≤threshold) then score=distance // we use: threshold=0.3

Return score;

End.

Algorithme V.5: Algorithme de Matching comportemental.

3.6.3 Algorithme de Matchmaking: il est composé de trois étapes:

- L'étape de matching sémantique des concepts input/output: cette première étape consiste à sélectionner les services pertinents selon un matching sémantique sur les concepts I/O en appliquant l'approche OWLS-SP. Le résultat de cette étape est donné par score 1.

- L'étape de matching de comportement: cette deuxième étape consiste à filtrer les services retournés de la première étape. Elle consiste de vérifier la compatibilité de comportement entre la requête utilisateur et les résultats de l'étape précédente, cette vérification est basée sur la notion de trace WTIA. Le résultat de cette étape est donné par score 2.
- L'étape de classement: cette troisième étape consiste à déterminer l'ordre de classement des résultats de l'algorithme de matchmaking. Le trie de résultats est basé sur la valeur du score globale de matching.

Score globale de matching-GSOM-: GSOM est utilisé pour classer les résultats par ordre croissant selon la règle suivante: GSOM = (α * Score1 + β * Score2) / (α + β), où α et β sont deux poids positifs fixés par la valeur 1.

Matchmaking Algorithm:

Input: TIWA automaton of service requester (Q)

Output: Result Ranked in ascending order of Gsom, the best Gsom is which have the lowest value

Result=empty

For each service A_k in repository do // service A_k in WTIA format

Score1 =I/O Matching(Q, A_k);

If Score1 ≠ W_{max}(where $W_{max}=|A|^{|Q|}$) then

Score2= behaviour Matching(Q, A_k)

If Score2 ≠ -1 then

*Gsom =(α*Score1 + β*Score2)/ (α+β) //both α and β are fixed to 1*

Append.Result(A_k, Gsom[A_k])

End_if

End_if

End_For

Ranking(Result); // this function ranks Result in ascending order of Gsom

Return Result;

End.

Algorithme V.6: Algorithme de Matchmaking.

3.7 Conclusion et perspectives

Dans la section 3, nous avons proposé une approche pour la découverte des services web à base du formalisme WTIA (automate d'interface typé et pondéré). Ce dernier permet de décrire une opération par le biais de leur type de ses entrées et de ses sorties, et chaque transition est étiquetée par les propriétés non-fonctionnelles (temps de latence, la disponibilité, la réputation, le coût ...).

La modélisation des services web sémantiques via le formalisme des automates pondérés et typés permet d'offrir une représentation comportementale et sémantique des services. En se basant sur le modèle WTIA, deux relations de similarité sont mises en place afin de développer des algorithmes de découverte et d'adaptation de services web sémantiques.

L'expérimentation de cette approche est réalisée seulement sur quelques exemples et ceci à cause du problème de la base de test dans ce cas. Pour bien tester notre proposition, nous devons créer un benchmark de services composés avec OWL-S, et de créer pour chaque service son automate d'exécution. Cette limitation est bien reconnue dans ce domaine [Cuzzocrea A. et al., 2011, Gater et al., 2010].

Plusieurs perspectives de l'utilisation du formalisme WTIA peuvent être envisagées pour entamer d'autres problématiques :

3.7.1 Problème d'adaptation:

Adapter signifie changer pour supporter de nouveaux critères ou besoins. Le concept d'adaptation devient une nécessité dans les environnements qui présentent une forte hétérogénéité, une variabilité élevée, et de nombreuses possibilités d'évolution. L'objectif principal de l'adaptation est de maintenir les conditions de vie d'une application ou d'un système. Dans la littérature, de nombreuses techniques d'adaptation sont étudiés et en particulier [Ledoux et al., 2001] qui fournit une synthèse de différentes approches dont le but est de supporter la capacité d'adaptation. [Aksit et al., 2003] présente un état de l'art sur l'adaptabilité, et donne un aperçu sur la reconfiguration dynamique et les techniques d'adaptation (dix techniques pour l'adaptation dynamique sont étudiées). [Desertot et al., 2007] présente un ensemble de techniques qui adresse les problèmes d'adaptabilité. Dans cette section, l'adaptation d'un service web consiste dans une première étape à rechercher un équivalent en termes de propriétés fonctionnelles et non-fonctionnelles, et qui répond aux nouveaux besoins et exigences. Dans l'échec de la première étape, nous proposition est de générer l'automate d'interface qui interagit correctement avec le nouveau service après évolution.

3.7.2 Problème de substitution:

L'objectif est de s'adapter à la nouvelle situation sans provoquer un dysfonctionnement du client ou de re-planifier le schéma de composition de services. Nous sommes intéressés par le processus de substitution parce que le coût de ce processus est moins couteux que le processus de re-planification, et il est généralement possible de trouver un service équivalent qui peut le remplacer.

Le problème de substitution dans le modèle WTIA revient à décider si A2 (automate WTIA) peut avoir le même comportement que «A1» (l'automate du service à remplacer) avec une qualité similaire ou supérieure, le problème de substitution affecte à la fois les propriétés fonctionnelles (PF) et les propriétés non-fonctionnelles (PNF).

Problème de Substitution dans le modèle WTIA:

Input: two WTIA automata A1 and A2,

Output: true if and only if (a) and (b) are both satisfied

False otherwise,

(a): A2 et A1 ont un comportement équivalent: $Gsom(A2) \cong Gsom(A1)$ //$trace(A1) \subseteq trace(A2)$

(b): A2 a des propriétés QoS supérieures que A1: $PNF(A2) \geq PNF(A1)$

Produit séquentiel:

Soient A1 et A2 deux automates WTIA. Le produit séquentiel de A1 et A2, dénoté A1.A2, est l'automate A12 avec:

$Q12=Q1 \cup Q2$

$A12=A1 \cup A2$

$E12=E1 \cup E2 \cup E12$ où $E12=\{(p,\tau,PNF,q) \mid p \in F1,\ q \in I2, NFP=(0,0,0,0,0)^t\}$

$I_{12}=I_1$

$F_{12}=F_2$

Chemin du produit séquentiel:

Soient A_1 et A_2 deux WTIA automates, π_1 représente la trace séquentielle de A_1, et π_2 représente la trace séquentielle de A_2. La trace de $A_{12}=A_1.A_2$ est π_{12} où $\pi_{12}=\pi_1;\pi_2$ avec un vecteur QoS $PNF_{12}=PNF_1+PNF_2$.

***Proposition*:** soient A_1, A_2, A_3, A_4 des automates WTIA, si A_1 substitut A_3 et A_2 substitut A_4, alors $A_1.A_2$ substitut $A_3.A_4$ ($\pi_{34} \subseteq \pi_{12}$ and $PNF_{12} \geq PNF_{34}$)

***Preuve*:**

A_1 substitut A_3 alors $\pi_3 \subseteq \pi_1$ avec $PNF_1 \geq PNF_3$ et A_2 substitut A_4 alors $\pi_4 \subseteq \pi_2$ with $PNF_2 \geq PNF_4$.

A_1. A_2 par construction on a: $\pi_{12}=\pi_1;\pi_2$ et $PNF_{12}=PNF_1+PNF_2$

A_3 .A_4 par construction on a: $\pi_{34}=\pi_3;\pi_4$ et $PNF_{34}=PNF_3+PNF_4$

A_1 substitut A_3 alors: (1) $\pi_3 \subseteq \pi_1$ donc: $\pi_3;\pi_4 \subseteq \pi_1;\pi_4$

(2) $PNF_1 \geq PNF_3$ donc: $PNF_1 + PNF_4 \geq PNF_3 + PNF_4$

A_2 substitut A_4 alors (1) $\pi_4 \subseteq \pi_2$ donc: $\pi_1;\pi_4 \subseteq \pi_1;\pi_2$

(2) $PNF_2 \geq PNF_4$ donc: $PNF_2 + PNF_1 \geq PNF_4 + PNF_1$

La relation d'inclusion est transitive, et par conséquent on a: $\pi_3;\pi_4 \subseteq \pi_1;\pi_2$ d'où: $\pi_{34} \subseteq \pi_{12}$

Et de même: $PNF_2 + PNF_1 \geq PNF_3 + PNF_4$ (car $PNF_2 + PNF_1 \geq PNF_1 + PNF_4 \geq PNF_3 + PNF_4$)

d'où : $PNF_{12} \leq PNF_{34}$

L'algorithme de substitution proposé dans notre approche est basé sur des mesures non-fonctionnelles du modèle WTIA après l'exécution de la découverte dynamique.

Le premier critère de classement des résultats est la compatibilité fonctionnelle en utilisant le score Gsom. Le second critère est les attributs QoS intégrés dans le modèle WTIA.

Algorithme de Substitution:

Input : sws_substituted

Output: Result Ranked in ascending order of Gsom, the best Gsom is

which have the lowest value of Gsom and high value of NFP attributes.

Result=empty;

extract_NFP(sws_substituted)

Result= Matchmaking Algorithm(sws_substituted)

If (Result not empty) then

For each service A_k in Result do

Extract_NFP(A_k)

Append.Result(A_k, Gsom[A_k], NFP[A_k])

End_for

Ranking(Result)

End_if

Else Substitution=false

Return Result

End.

Algorithme V.7: Algotithme de substitution.

3.7.3 Génération dynamique de l'automate du client:

Dans cette section, nous présentons la sémantique opérationnelle d'un ensemble d'opérateurs utilisés dans la génération dynamique de l'automate du client qui correspond à la spécification de l'automate du service après son évolution (ce service n'a pas de substituant dans la base des services).

L'objectif est de générer automatiquement l'automate du client qui communique correctement avec l'automate de service (après son évolution) basé sur un ensemble de règles opérationnelles liées aux actions observables de l'automate WTIA du service.

Nous présentons la sémantique opérationnelle associée aux nouveaux opérateurs introduits (comme complément, composer, et décomposer) afin de générer dynamiquement la spécification du client qui correspond à la description du service.

Opérateur « Complement »:

Complement (act): spécifie la négation logique de l'action observable act (act ∈ {?, !, ?!, !?, √}). Les règles sémantiques associées à cet opérateur sont:

complement (?m) $\xrightarrow{?m}$!m

complement (!m) $\xrightarrow{!m}$?m

complement ($?m_1!m_2$) $\xrightarrow{?m_1!m_2}$! $m_1?m_2$

complement (√) $\xrightarrow{\surd}$ √

Opérateur « Decompose »:

Decompose (act): spécifie la décomposition d'une action composée de plusieurs actions primitives, les règles sémantiques de cet opérateur sont :

Decompose (?m1!m2) → ?m1;!m2

Decompose ($!m_1?m_2$) $\xrightarrow{!m_1?m_2}$ $!m_1;?m_2$

Opérateur « Compose »:

Compose (act): spécifie l'assemblage de deux actions successives qui forment un processus séquentiel de la forme (?m1;!m2) dans une seule action de la forme ?!.

La règle sémantique associée à cet opérateur est:

compose ($?m_1;!m_2$) $\longrightarrow$ $?m_1!m_2$

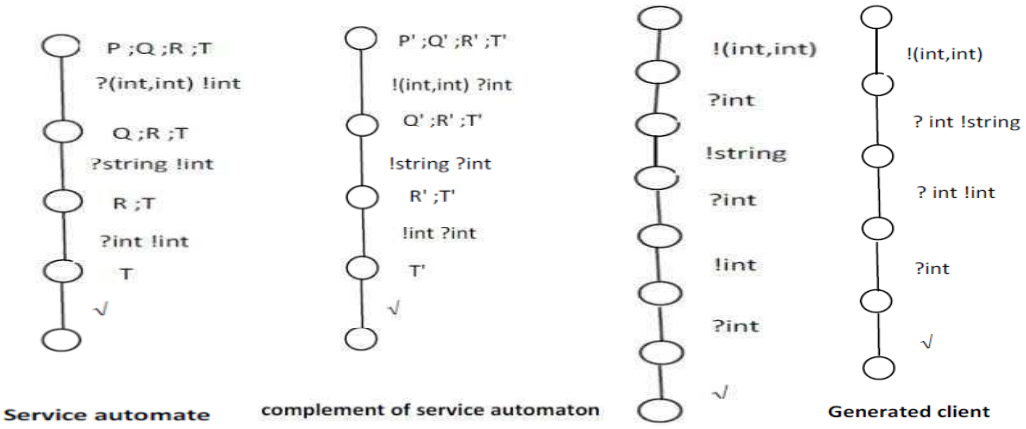


Figure V.6: Etapes de génération de l'automate client du service après évolution.

4. Conclusion

Nous avons présenté dans ce chapitre deux approches de découverte de services web sémantiques en tenant compte leur comportement. Elles partagent l'étape de similarité sémantique en utilisant l'algorithme de matchmaking de l'approche OWLS-SP vue dans le chapitre 4.

La première approche utilise la technique d'isomorphisme de sous-graphe pour vérifier la compatibilité comportementale entre la requête et le service. L'expérimentation de cette approche a permis de montrer son efficacité.

La deuxième approche utilise la technique d'équivalence de trace d'automates d'interface typé et pondéré pour déterminer la similarité comportementale entre deux services web sémantiques.

Chapitre VI:

Conclusion générale

Sommaire

Plusieurs types de paradigmes de développement d'applications distribuées ont été proposés au cours de ces dernières années. Le paradigme SOA est l'un des modèles les plus convenables au développement d'applications distribuées sur le web où les services web sémantiques sont considérés comme une implémentation du paradigme SOA.
L'objectif visé par la notion de services Web sémantiques est de créer un Web sémantique de services dont les propriétés, les capacités, les interfaces et les effets sont décrits de manière non ambiguë et exploitable par des machines.

1. Synthèse

La problématique traitée dans le cadre de cette thèse est le processus de découverte des services web sémantiques. Ces derniers représentent un domaine de recherche très actif où la problématique de découverte est abordée dans un grand nombre de travaux de recherches.

Notre vision pour répondre à ce besoin consiste à le décomposer en deux sous-problèmes :

La découverte des services atomiques où le matching se base principalement sur les propriétés du profile (IOPE) de l'ontologie de service OWL-S.

La découverte des services composés où il est nécessaire de prendre en charge l'information comportementale offerte par le « process model » de l'ontologie de service OWL-S.

La première partie comporte les trois premiers chapitres, elle présente les bases théorique et un état de l'art du domaine des services web sémantiques où nous avons mis en évidence les principaux standards liés à la technologie des services web sémantiques tels que : WSDL, SOAP, UDDI, OWL-S, WSDL-S, SAWSDL, USDL,..

La deuxième partie regroupe les deux derniers chapitres, elle présente nos contributions proposées pour répondre à la problématique de découverte. Les différentes approches proposées font l'objet de plusieurs publications internationales dans des revues indexées (SCOPUS, DBLP, INSPEC) ainsi que dans des conférences internationales avec comité de lecture (IEEE).

2. Travaux réalisés dans le cadre de cette thèse :

- **Maamar Khater, Mimoun Malki (2008):** An approach for managing web services changes. Int. J. of Information Systems and Change Management

2008 (IJISCM). Inderscience publishers- Vol. 3, No.4 pp. 314 – 326. (SCOPUS, DBLP,…).

- **Maamar Khater, Mimoun Malki (2009) :** une approche pour l'adaptation des services web. International Conference on Multimedia Computing and Systems. ICMCS'09 – April 02-04, 2009 -Ouarzazate, IEEE Morocco. ISBN: 978-1-4244-3756-6. DOI: 10.1109/MMCS.2009.5256732. On page(s): 56-61.
- **Maamar Khater, Mimoun Malki (2009):** Un framework pour les services web sémantique. Second International Conference on Web and Information Technologies "ICWIT'09". 12-14 june 2009, Kerkennah Islands, Sfax, Tunisia. ISBN: 978-9973-868-23-7. Edition IHE.
- **Maamar Khater, Salah-eddine Habibeche, Shwan Khaled, Mimoun Laouni, Mimoun Malki (2014):** Shortest-path based approach for improving the performance of semantic web services discovery. ICA2IT International Conference on Artificial Intelligence and Information. ouargla 2014. http://www.icaiit2014.com/
- **Maamar Khater and Mimoun Malki (2014)**: Improving the Performance of Semantic Web Services Discovery: Shortest Path based Approach. (2014). MECS IJ Information Technology and Computer Science. DOI: 10.5815/ijitcs.2014.07.05. (INSPEC,….).
- **Maamar Khater and Mimoun Malki (to appear 2015)**: Formal method based behaviour for semantic web services discovery. International Journal of Web Science. Inderscience Publishers Ltd. (DBLP, INSPEC,….).

3. Perspectives

Plusieurs améliorations et extensions peuvent être envisagées pour enrichir les approches de découverte proposées.

L'utilisation d'une mesure de similarité dans le calcul du degré de matching entre les concepts de la requête et du service publié (tel que : Wu Palmer, Resnik,….) au lieu de la fonction discrète de matching proposée par Paolucci.

L'utilisation d'un autre modèle formel que les automates pour représenter le comportement du service tel que (les réseaux de pétri, algèbre de processus,…).

Etendre l'approche qui utilise la fonction de mapping entre le graphe motif et le graphe cible pour prendre en charge aussi le matching de type 1-plusieurs entre les nœuds des deux graphes.

Nos approches peuvent être aussi utiles pour entamer d'autres problématiques abordées dans le domaine des services web sémantiques telles que : la substitution et l'adaptation des services.

Références bibliographiques

[Abhijit et al., 2004]: Abhijit A. Patil, Swapna A. Oundhakar, Amit P. Sheth, Kunal Verma. Meteor-s web service annotation framework. World Wide Web Conference Series-WWW, pp. 553-562, 2004.

[Alrifai et al, 2012] : M. Alrifai, T. Risse, and W. Nejdl, A Hybrid Approach for Efficient Web Service Composition with End-to-End QoS Constraints. ACM Transactions on the Web, Vol. 6, No. 2, Article 7, 2012.

[Aksit et al., 2003]: Aksit M., Chokair Z., « Dynamic, Adaptive and Reconfigurable Systems Overview and Prospective Vision », Actes des ICDCSW'03, providence, Rhode Island,, USA, May 19-22, 2003., p. 84-92.

[Baeten & Weijand, 1990]: Baeten, J. C. M., and Weijand, W. P. 1990. Process Algebra, volume 18 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

[Bansal et al., 2003]: Sharad Bansal, José M.Vidal, Matchmaking of Web Services Based on the DAML-S Service Model.AAMAS'03, July 14–18,2003,Melbourne, Australia.ACM 1-58113-683- 8/03/0007

[Bansal A. et al., 2005]: A. Bansal, S. Kona, L. Simon, and T.D. Hite. A universal service-semantics description language. Web Services, European Conference on:214–225, 2005.

[Beeri et al., 2008]: Catriel Beeri, Anat Eyal, Simon Kamenkovich, Tova Milo. Querying Business Processes with BP-QL. Proceedings of the 31st VLDB Conference, Trondheim,Norway,2005.

[Bellur et al., 2008]: Bellur Umesh, Kulkarni Roshan, Improved Matchmaking Algorithm for Semantic Web Services Based on Bipartite Graph Matching. Conference: International Conference on Web Services-ICWS , pp. 86-93, 2007

[Benatallah et al, 2005]: B. Benatallah, M-S Hacid, A.Leger, C.Rey, and F.Toumani. On automating web services discovery. The VLDB Journal,14(1) :84-96, 2005.

[BenMokhtar S et al., 2005]: BenMokhtar S, Nikolaos Georgantas, Valérie Issarny: Ad Hoc Composition of User Tasks in Pervasive Computing Environments. Software Composition 2005: 31-46.

[BenMokhtar S et al., 2008]: efficient semantic service discovery in pervasive computing environments with QoS and context support. J Syst Softw 81(5):785–808.

[Berners-Lee et al., 2001]: Tim Berners-Lee, James Hendler and Ora Lassila, "The Semantic Web", Scientific American, May 2001, p. 29-37.

[Bernstein A. & Kiefer C., 2006]: Bernstein A., Kiefer C.: Imprecise RDQL: Towards Generic Retrieval in Ontologies Using Similarity Joins. Proceedings ACM Symposium on Applied Computing, Dijon, France, ACM Press, 2006.

[Bianchini D. et al., 2006]: Bianchini D., De Antonellis V., Melchiori M. Salvi D.: Semantic-enriched Service Discovery. Proceedings of IEEE ICDE 2nd International Workshop on Challenges in Web Information Retrieval and Integration (WIRI06), Atlanta, Georgia, USA, 2006.

[Bolognesi et Brinksma, 1989]: Bolognesi, T., and Brinksma, E. 1989. Introduction to the ISO Specification Language LOTOS. In van Eijk, P. H. J.; Vissers, C. A.; and Diaz, M., eds., The Formal Description Technique LOTOS. Elsevier Science Publishers North-Holland.23–73.

[Botelho L. et al., 2008]: Botelho L., Fernandez A., Klusch M., Pereira L., Santos T., Pais P., Vasirani M.: Service Discovery. Chapter 10. 2008.

[Bunke et al., 1994]: Bunke, H., Messmer, B.T.: Similarity Measures for Structured Representations. In Wess, S., Althoff, K. D., Richter,M.M., eds.: Topics in Case-Based Reasoning, First European Workshop, EWCBR-93, Kaiserslautern, Germany, November1-5, 1993, Selected Papers. Volume837 of Lecture Notes in Computer Science., Springer (1994)106–118

[Bussler C. et al., 2005]: Bussler C : Web Service Execution Environment (WSMX) http://www.w3.org/Submission/WSMX/

[Brambilla M, et al., 2009]: Brambilla M, et al. Comparison: mediation on WebML/WebRatio and jABC/jETI. In: Petrie C, Margaria T, Lausen H, Zaremba M(eds)emantic web services challenge. Springer,US,pp153–166

[Cassati F., et al., 2001]: Cassati F., Shan M-C., Models and languages for Describing and Discovering E-services. In ACM SIGMOD, Santa Barbara, USA. 2001.

[Chakraborty D. et al., 2001]: Chakraborty D., Perich F., Avancha S., Joshi A. Reggie D.: Semantic Service Discovery for M-Commerce Applications. Proceedings of the International Workshop on Reliable and Secure Applications in Mobile Environment, 2001.

[Chouali et al., 2010]: Chouali samir, sebti mouelhi, hassan mountassir. adaptation des protocoles des composants par les automates d'interface. congrés approches formelles dans l'assistance au développement de logiciels (AFADL'10). france 2010.

[Constantinescu & Faltings, 2003]: Constantinescu I. Faltings B.: Efficient matchmaking and directory services Proceedings of IEEE Conference on Web Intelligence WI, 2003.

[Corrales et al., 2008]: Juan Carlos Corrales,Daniela Grigori, and Mokrane Bouzeghoub. BPEL Processes Matchmaking for Service Discovery. LNCS 4275,pp.237–254, 2006. Springer-Verlag. Berlin Heidelberg 2006.

[Cuzzocrea A., 2011] : Alfredo Cuzzocrea1, Juri Luca De Coi, MarcoFisichella, and Dimitrios Skoutas : Graph-based Matching of Composite OWL-S Services. GDB 2011, Hong Kong, China.

[de Bruijn J. et al., 2005]: de Bruijn Jose: Web Service Modeling Language (WSML) http://www.w3.org/Submission/WSML/

[Della Valle E. et al., 2005]: Della Valle E., Cerizza D., Celino I. : The Mediators Centric Approach to Automatic Web Service Discovery of Glue. Proceedings of 1st International Workshop on Mediation in Semantic Web Services (MEDIATE), CEUR Workshop proceedings, 168, 2005.

[Desertot et al., 2007]: Desertot M., « Une architecture flexible et adaptable pour les serveurs d'applications », thèse de PhD l'Université Joseph Fourier de Grenoble, 27 Avril 2007.

[Dijkman et al., 2009]: Dijkman, R.M., Dumas, M., García-Bañuelos, L.: Graph Matching Algorithms for Business Process Model Similarity Search. In: Dayal, U., Eder, J., Koehler, J.,

[Dijkstra, E.W., 1971]: Dijkstra, E.W., A Short Introduction to the Art of Computer Programming, Technische Hogeschool, Eindhoven, 1971.

[Di Noia et al., 2004]: Di Noia T., Di Sciascio, Donini FM : Semantic Matchmaking as Non-Monotonic Reasonning: A Description Logic Approach. Artificial Intelligence Research (JAIR), 29:269-307, 2007.

[Ehrig M. Et al., 2007]: Ehrig M, A. Koschmider, and A. Oberweis. Measuring similarity

between semantic business process models. In Proc. of APCCM 2007, pages 71-80, 2007.

[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/.

[Fensel & Bussler, 2002]: Fensel, D., & Bussler, C. (2002). The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications, 1(2).

[Fensel D. et al., 2007]: Fensel. D., Lausen. H, Polleres. A, Bruiji. J, Stollberg. M, Roman. D, and Domingue, J., Enabling Semantic Web Services : The Web Service Modeling Ontology. Springer Berlin Heidelberg New York. 2007.

[Fernandez A. et al., 2006]: A. Fernandez, M. Vasirani, C. Caceres, S. Ossowski: A role-based support mechanism for service description and discovery. In: huang et al. (eds.), Service Oriented Computing: Agents, Semantics, and Engineering. LNCS 4504, Springer, 2006.

[Fuglede B. & Topsoe F. 2004]: Fuglede B. and Topsoe F. Jensen-shannon divergence and Hilbert space embedding. In Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on, page 31, June 2004.

[Gater et al., 2010]: Gater A, Daniela Grigori, Mokrane Bouzeghoub: A Graph-Based Approach for Semantic Process Model Discovery. Graph Data Management 2011: 438-462

[Guilherme et al., 2012]: Hobold G C, Siqueira F (2012) Discovery of semantic web services compositions based on SAWSDL annotations. In: IEEE 19th international conference on web services. Hawaii,USA

[Günay et al., 2010]: Günay A, Pinar Yolum: Service matchmaking revisited: An approach based on model checking. J. Web Sem. 8(4): 292-309 (2010)

[Hadad et al., 2006]: HADDAD S., Moreaux P., Rampacek S. client synthesis for web services by way of a timed semantics. ICEIS (4), p. 19-26, 2006.

[Hashiguchi K. et al., 2002]: K. Hashiguchi, K. Ishiguro, and S. Jimbo. Decidability of the Equivalence Problem for Finitely Ambiguous Finance Automata. IJAC, 12(3), 2002.

[Henzinger et Alfaro, 2001]: Henzinger T. and Alfaro L. Inerface automata. ACM Press, 9th Annual Aymposium of FSE (Foundations of Software Engineering), pages 109-120, 2001

[Héam P-C, 2009]: Pierre-Cyrille Héam. automates finis pour la fiabilité logicielle et l'analyse d'accessibilité. Mémoire d'habilitation à diriger des recherches. université de franche-comté. 2009.

[Hoare, 1984]: Hoare, C. A. R. 1984. Communicating Sequential Processes. Prentice-Hall.

[Holzmann, 2003]: Holzmann, G. 2003. The SPIN Model-Checker: Primer and Reference Manual. Addison-Wesley.

[Jaeger M.C. et al., 2005]: Jaeger M.C., Rojec-Goldmann G., Liebetruth C., M"uhl G., Geihs: Ranked Matching for Service Descriptions Using OWL-S. Proceedings of 14. GI/VDE Fachtagung Kommunikation in Verteilten Systemen KiVS, Kaiserslautern, 2005.

[Kaufer F. & Klusch M., 2006]: Kaufer F. & Klusch M.: WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. Proceedings 4th IEEE European Conference on Web Services (ECOWS), Zurich,

Switzerland, IEEE CS Press, 2006.

[Keller U, et al., 2005]: Keller U, et al(2005) Automatic location of services. In: ESWC, pp1–16

[Khater et Malki, 2008]: Maamar Khater, Mimoun Malki: An approach for managing web services changes. Int. J. of Information Systems and Change Management 2008 (IJISCM). Inderscience publishers- Vol. 3, No.4 pp. 314 – 326.

[Khater et Malki, 2009]: Maamar Khater, Mimoun Malki: une approche pour l'adaptation des services web. International Conference on Multimedia Computing and Systems. ICMCS'09 – April 02-04, 2009 - Ouarzazate, IEEE Morocco. ISBN: 978-1-4244-3756-6. DOI: 10.1109/MMCS.2009.5256732. On page(s): 56-61.

[Khater et Malki, 2009]: Maamar Khater, Mimoun Malki (2009): Un framework pour les services web sémantique. Second International Conference on Web and Information Technologies "ICWIT'09". 12-14 june 2009, Kerkennah Islands, Sfax, Tunisia. http://icwit09.aigtunisie.org/accepted.php.

[Khater et al., 2014]: Maamar Khater, Mimoun Malki, "Improving the Performance of Semantic Web Services Discovery: Shortest Path based Approach", IJITCS, vol.6, no.7, pp.32-39, 2014. DOI: 10.5815/ijitcs.2014.07.05

[Khater et al., 2015]: Maamar Khater, Mimoun Malki (to appear 2015): Formal method based behaviour for semantic web services discovery. International Journal of Web Science. Inderscience Publishers Ltd. DBLP.

[Kiefer C. & Bernstein A., 2008]: Kiefer C, Bernstein A (2008) The creation and evaluation of iSPARQL strategies for matchmaking. In 5th European semantic web conference (ESWC). Springer, Spain, pp 463–477.

[Klein M. & Konig-Ries B., 2004]: Klein M. & Konig-Ries B. : Coupled Signature and Specification Matching for Automatic Service Binding. European Conference on Web Services (ECOWS 2004), Erfurt, 2004.

[Klusch M. et al., 2005]: M. Klusch, B. Fries, M. Khalid, and K. Sycara. Owls-mx: Hybrid owl-s service matchmaking. In Proceedings of 1 st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, USA, November 2005.

[Klusch M., 2008]: Matthias Klusch: "Semantic Web Service Coordination", Semantic Web Service Coordination CASCOM: Intelligent Service Coordination in the Semantic Web Whitestein Series in Software Agent Technologies and Autonomic Computing 2008, pp 59-104. Print ISBN 978-3-7643-8574-3

[Klusch M. & Kapahnke P, 2008]: M. Klusch, & P.Kapahnke, Semantic Web Service Selection with SAWSDL-MX. CEUR Proceedings of 2nd International Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web (SMR2), Karlsruhe, Germany, CEUR 416. 2008.

[Kourtesis et Paraskakis, 2008]: D.Kourtesis and I.Paraskakis Combining SAWSDL, OWL-DL and UDDI for Semantically Enhanced Web Service Discovery. Proceedings of 5th European Semantic Web Conference ESWC 2008, LNCS 5021, pp. 614 – 628, 2008.

[Krob D. 1994]: D. Krob. The Equality Problem for Rational Series with Multiplicities in the Tropical Semiring is Undecidable. IJAC, 4(3),1994.

[KuSter U et al., 2007]: KuSter U et al(2007) DIANE: a matchmaking-centered frame-work for automated service discovery, composition, binding, and invocation on the web. Int J Electron Commer 12 (2):41–68.

[Lecue F., 2008]: F.Lecue. Composition de Services Web: Une Approche basée liens Sémantiques. Thèse de doctorat en Informatique. L'Ecole Nationale Supérieure des Mines de Saint-Etienne. octobre 2008.

[Ledoux et al., 2001]: Ledoux T., Projet RNTL ARCAD D.1.1, « état de l'art sur l'adaptabilité », école de mines de Nantes, 4, rue Alfred Kastler, 44307 Nantes Cedex, 2001.

[Levenshtein V. 1966]: Levenshtein V. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady, 10:707, 1966.

[Li Jing 2013]: Li Jing. A Fast Semantic Web Services Matchmaker for OWL-S Services. JOURNAL OF NETWORKS, VOL. 8, NO. 5, MAY 2013

[Li L. & Horrocks I., 2003]: Li L. & Horrocks I.: A software framework for matchmaking based on semantic web technology. In Proceedings of the Twelfth International Conference on World Wide Web, pages 331-339. ACM Press, 2003.

[Majithia et al., 2004]: Majithia S., David W. Walker and W. A. Gray. A framework for automated service composition in service-oriented architecture. In 1st European Semantic Web Symposium, 2004.

[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.

[Michael C. et al. , 2004]: Michael C. Jaeger, Stefan Tang, Ranked Matching for Service Descriptions using DAML-S Conference on Advanced Information Systems Engineering - CAiSE , pp. 217-228, 2004

[Milner, 1989]: Milner, R. 1989. Communication and Concurrency. International Series in Computer Science. Prentice Hall.

[Minor et al., 2007]: Minor M, A. Tartakovski, and R. Bergmann. Representation and structure-based similarity assessment for agile workows. In Proc. of the Intl. Conf. on Case-Based Reasoning, volume 4626 of LNAI, pages 224-238. Springer, 2007.

[Nejati et al., 2007]: Nejati S, M. Sabetzadeh, M. Chechik, S. Easterbrook, and P. Zave. Matching and merging of statecharts specifications. In Proc. of ICSE 2007, pages 54-63, 2007.

[Ngan LD, et al., 2009]: Ngan LD, et al (2009) MODiCo: a multi-ontology web service discovery and composition system. In: ICWE2009, Spain, pp531–534

[Ngan LD & Kanagasabai R., 2012]: Ngan L.D., Kanagasabai R. Semantic Web service discovery: state-of-the-art and research challenges. Personal and Ubiquitous Computing journal, september 2012. Springer-Verlag. Online ISSN 1617-4917

[Ouazzani M., 2004]: Mourad Ouzzani: Efficient Delivery of Web Services. Thèse de doctorat en informatique. 2004. Virginia Polytechnic Institute and State University.

[Oundhakar S, et al., 2005]: Oundhakar S, et al(2005) Discovery of web services in amulti-ontology and federated registry environment. Int J Web Serv Res 1(3):1–32.

OWL-S. http://www.w3.org/Submission/OWL-S/ visited 15/09/2013

[Paolucci et al., 2002]: Paolucci, M., Kawamura, T., Payne, T. R., & Sycara, K. (2002).

Semantic matching of Web services capabilities. In Proc. of 1st International Semantic Web Conference (ISWC). LNCS Volume 2342, 2002, pp 333-347

[Parrow, 2001]: Parrow, J. 2001. An Introduction to the π-Calculus. In Bergstra, J.; Ponse, A.; and Smolka, S., eds., Handbook of Process Algebra. Elsevier. Chapter 8, 479–543.

[Petri, 1962]: Petri, C. A. 1962. Kommunikation mit Automaten. Schriften des IIM Nr.2, Bonn: Institut fur Instrumentelle Mathematik. Second Edition:, NewYork: Griffiss Air Force Base,Technical Report RADC-TR-65377,Vol.1, 1966, Pages: Suppl.1,English translation.2.2.1.

[J. Phatak et al., 2005]: Phatak J. et al. A Framework for Semantic Web Services Discovery. WIDM, 2005.

[J. Phatak et al.,, 2007]: J. Pathak. Interactive and verifiable web services composition, specification reformulation and substitution. Phd Thesis in Computer Sciences. Iowa State University 2007.

[Pilioura T, T salgatidou A., 2009]: Pilioura T, T salgatidou A(2009) Unified publication and discovery of semantic web services. ACM Trans Web 3(3):1–44.

[Rajesh et al., 2009]: Rajesh T, Mayer W, Stumptner M (2009): Semantic service discovery by consistency-based matchmaking. In: Advances in data and web management. Springer, Berlin, pp 492–505.

[Rijsbergen C. J. V., 1976]: Rijsbergen C. J. V., “Information Retrieval,” Buttersworth, London, 1976.

[Salton G. 1983]: Salton G. Introduction to Modern Information Retrieval. Mc Graw-Hill, New York, 1983.

[Stollberg M. et al., 2007]: M. Stollberg, U. Keller, H. Lausen, S. Heymans: Two-phase Web Service discovery based on rich functional descriptions. Proceedings of European Semantic Web Conference, Buda, Montenegro, LNCS, Springer, 2007.

[Schneider et al., 1992]: Schneider, S.; Davies, J.; Jackson, D. M.; Reed , G. M.; Reed, J. N.; and Roscoe, A.W. 1992. Timed CSP: Theory and Practice. In Proc. of REX Workshop on Real-Time: Theory in Practice, 640–675.

[Sycara K. et al., 2002]: Sycara K. Klusch M., Widoff S., Lu J.: LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. Autonomous Agents and Multi-Agent Systems, 5(2): 173-204, Kluwer Academic, 2002.

[Vaculin R. & Sycara K., 2007]: R. Vaculin, K. Sycara: Towards automatic mediation of OWL-S process models. IEEE International Conference on Web Services (ICWS 2007), 2007.

[Vander aalst et al 2006]: van der Aalst W, A.K. Alves de Medeiros, and A. Weijters. Process Equivalence: Comparing two process models based on observed behavior. In Proc. of BPM 2006, volume 4102 of LNCS, pages 129-144. Springer, 2006.

[Verma K. et al., 2003]: Verma K, et al(2003): METEOR-S WSDI:a scalable infrastructure of registries for semantic publication and discovery of web services. J Inf Technol Manag6:17–39.

[Verma K. et al., 2005]: Verma K., Sivashanmugam K., Sheth A., Patil A., Oundhakar S., Miller J.: METEORS WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. Information Technology and Management, Special Issue on

Universal Global Integration, vol. 6, No. 1, 2005.
[Vitvar T, et al., 2009]: Vitvar T, et al (2009) Mediation using WSMO, WSML and WSMX. In: Semantic web services challenge. Springer,US, pp31–49.
[Voinot J. et al., 2011]: Voinot Jérôme, Intégration de la qualité de service dans la vérification de la substitutivité des services Web. ARA COPS. 29/30 mars 2007 – Nancy.
[Vu L.H. et al., 2006]: Vu L.H., Hauswirth M., Porto F., Aberer K.: A Search Engine for Qos-enabled Discovery of Semantic Web Services. Ecole Politechnique Federal de Lausanne, LSIR-REPORT-2006-002, Switzerland, 2006.
[Weber A. et al, 1994]: A. Weber. Finite-valued Distance Automata. Theoretical Computer Science, 134, 1994.
[Zhu S. & Xia G. 2010]: S. Zhu, J. Wu, and G. Xia. Top-k cosine similarity interesting pairs search. In Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on, volume 3, pages 1479–1483, auguste 2010.