

composition des services web

TAOURIRIT Salah Eddine

1^{er} décembre 2014

version 0.2.3

Table des matières

1	Introduction	2
I	Étude bibliographique	3
2	Les services web : Vue d'ensemble	4
2.1	Notions de base et technologies associées	5
2.1.1	Définition et caractéristiques	5
2.1.2	L'évolution des styles des services web	6
2.1.3	L'architecture de référence et technologies associées	6
2.1.3.1	Communication : SOAP	8
2.1.3.2	Description : WSDL	8
2.1.3.3	Découverte : UDDI	10
2.2	Description des services web	11
2.2.1	Description syntaxique de services	11
2.2.2	Ajout de la sémantique	13
2.2.2.1	Définition des services Web sémantiques	13
2.2.2.2	WSDL-S	14
2.2.2.3	SAWSDL	15
2.2.2.4	OWL-S	16
2.2.2.5	WSMO	16
2.3	Découverte des services web	17
2.4	Conclusion	18
3	La Composition des services web	19
3.1	Définition et stratégies de composition	20

3.1.1	Définitions	20
3.1.2	Cycle de vie d'une composition	21
3.1.3	Procédés de coordination	23
3.1.3.1	Orchestration	24
3.1.3.2	Chorégraphie	24
3.1.4	Stratégies de composition	25
3.1.4.1	Composition statique/dynamique	26
3.1.4.2	Composition manuel/automatique	26
3.2	Langages pour la composition	27
3.2.1	BPEL	27
3.2.2	WS-CDL	28
3.2.3	WSMF	28
3.2.4	OWL-S	29
3.2.5	Comparaison	32
3.3	Composition dynamique des services web	33
3.3.1	Les approches basées sur les workflow	33
3.3.2	Les approches guidées par les modèles	33
3.3.3	Les approches mathématiques	33
3.3.4	Techniques de planification	33
3.4	Conclusion	33

4 Les approches de composition dynamique des services Web sémantiques basées sur le modèle graphe 34

4.1	Préliminaires	35
4.1.1	Matching des services Web	35
4.1.2	Mésures de similarité	36
4.1.3	Graphe de dépendance	36
4.2	Travaux connexes	36
4.2.1	Génération Online du graphe de dépendance	36
4.2.2	Génération Offline du graphe de dépendance	36
4.2.3	Autres travaux basés sur le graphe matching	36
4.3	Vers les bases de données graphe	36
4.4	Conclusion	36

II	L'approche proposée	37
5	Une méthode de composition dynamique des services Web sémantiques utilisant neo4j	38
5.1	Reformulation mathématique du problème	38
5.1.1	Définitions de base	38
5.1.2	Hypothèses du travail	38
5.1.3	Objectifs	38
5.2	Exemple d'illustration	38
5.3	Annotation sémantiques des services web	38
5.4	Architecture du système	38
5.5	Conclusion	38
6	L'Implémentation d'un prototype	39
6.1	Exemple d'illustration	39
6.2	Outil d'annotaion sémantiques des services	39
6.3	Publication des services web atomiques	39
6.4	Découvert d'un service web composites	39
6.5	Conclusion	39

Acronyms

BPEL	Business Process Execution Language.	28
BPEL4WS	Business Process Execution Language for Web Services.	28
DAML	the DARPA Agent Markup Language.	14, 16
DARBA	Defence Advanced Research Projects Agency.	16
HTTP	Hypertext Transfer Protocol.	5
QoS	Quality of service.	22
SAWSDL	Semantic Annotations for WSDL and XML Schema.	15
SOAP	Simple Object Access Protocol.	7
UDDI	Universal Description Discovery and Integration.	6, 7, 10
W3C	The World Wide Web Consortium.	5, 8, 11
WS-BPEL	Web Service Business Process Execution Language.	28
WS-CDL	Web Services Choreography Description Language.	28
WSCI	Web Service Choreography Interface.	29
WSDL	Web Services Description Language.	7, 8, 11, 29
WSFL	Web Services Flow Language.	28
WSMF	The Web Service Modeling Framewor.	29
XLANG	XML Business Process Language.	28
XML	Extensible Markup Language.	28

Table des figures

2.1	La structure d'un message SOAP	9
2.2	Web evolution to Semantic Web services [1].	14
3.1	Cycle de vie d'une composition des services [2].	21
3.2	Orchestration vs Chorégraphie.	23
3.3	Classification des stratégies de composition [3].	25
3.4	Les éléments d'une ontologie OWL-S	29

Liste des tableaux

3.1	Comparaison des standards et langages de composition	32
-----	--	----

Chapitre 1

Introduction

Première partie

Étude bibliographique

Chapitre 2

Les services web : Vue d'ensemble

Ce chapitre établit une étude du fondement théorique de notre travail à savoir les concepts de base du paradigme service Web. Nous commençons d'abord par présenter un tour d'horizon définissant l'architecture de référence de ce paradigme ainsi que quelques définitions proposées dans la littérature. Ensuite nous nous intéressons à montrer les limitations de l'approche syntaxique de la description des services web et l'apport de l'enrichissement sémantique de cette dernière aux processus de la découverte et la composition des services Web.

2.1 Notions de base et technologies associées

Les services Web constituent une approche pour mettre en œuvre le paradigme de service, et peut être vue comme une instance de l'architecture orienté service.

Dans cette section va parler aussi d'un socle technologique très sollicité, On va aussi Détailler l'architecture de base d'un service web, ensuite nous introduisons l'architecture étendus.

2.1.1 Définition et caractéristiques

Les services Web sont la technologie la plus connue et la plus populaire dans le monde industriel et académique pour la mise en place d'architectures à services.

Les Web services ont été proposés initialement par IBM [4] et Microsoft, puis en standardisés par le W3C¹ et définis [5] par :

“Un service web est un système conçu pour permettre d'interopérabilité des applications à travers un réseau. Il est caractérisé par un format de description interprétable/compréhensible automatiquement par la machine, D'autres systèmes peuvent interagir avec le Service Web selon la manière prescrite dans sa description et en utilisant des messages SOAP, généralement transmis via le protocole HTTP et sérialisés en XML et en d'autres standards du Web ”.

Cette définition surligne les caractéristiques clés de services Web [6] :

- **Basés sur des protocoles Internet** : L'utilisation de HTTP pour le transport des informations permet de traverser les contrôles d'accès dans un environnement hétérogène.
- **Interopérables** : Le standard SOAP [7] définit comme étant un protocole destiné à l'échange de messages structurés véhiculé généralement sur HTTP et sérialisé en XML, permettant le support pour l'interopérabilité.
- **Basés sur XML** : Le méta-langage de balisage XML *eXtensible Markup Language* est un standard Web ouvert par W3C [8] offre un cadre standard pour la définition de documents Interprétable par des machines.

1. <http://www.w3.org/>

M. P. Papazoglou [9] apporte une autre définition de services web :

“Les services Web sont des éléments auto-descriptifs et indépendants des plateformes permettent la composition faible coût d’applications distribuées. Les services Web effectuent des fonctions allant de simples requêtes des processus métiers complexes. Les services Web permettent aux organisations d’exposer leurs programmes résultats sur Internet (ou sur un intranet) en utilisant des langages (basés sur XML) et des protocoles standardisés et de les mettre en œuvre via une interface auto-descriptive basée sur des formats standardisés et ouverts”

Curbera et al. [10] de ça part proposent la définition suivante :

“Un service Web est une application réseau capable d’interagir par le moyen des standards et des protocoles via des interfaces bien spécifiés, dans lequel est décrits utilisant un langage de description fonctionnel standardisé”.

2.1.2 L’évolution des styles des services web

2.1.3 L’architecture de référence et technologies associées

[11] [12] [4] [5] Cette architecture a été proposée afin de promouvoir l’interopérabilité et l’extensibilité des services Web Dans l’ensemble, une architecture complète de services Web est constitué d’un fournisseur de service², un annuaire de services³, et un client⁴ de service. La figure x montre comment ces trois rôles interagissent.

- **Le fournisseur** Un prestataire de services fournit l’interface pour le service Web et l’implémentation de l’application. Le fournisseur de service est également responsable de la création de la définition du service et de publier cette définition pour répondre à la spécification UDDI
- **L’annuaire** Un registre de service est une façon dont les services Web sont officiellement publiés. Le registre de service est basée sur la spéci-

2. Providers

3. Service Registry

4. Service Requester

fication UDDI et reflète des informations sur les services fournis par le fournisseur de services. Le registre de service fournit un demandeur de services avec une description de service WSDL et un URL qui pointe vers le service lui-même.

- **Le client** Un client de services est le consommateur d'un service Web, il utilise le registre de service pour obtenir des informations et pour pouvoir accéder à un service Web.

Pour qu'une application profite des services Web, trois comportements doivent avoir lieu : la publication des descriptions du service, correspondances des descriptions du service et la liaison ou l'invocation des services basés sur la description du service. Ces comportements peuvent se produire d'une manière singulière ou d'une manière itérative. En détail, ces opérations sont les suivantes :

- **Publish** Pour être accessible, une description de service doit être publiée afin que le demandeur de service puisse la trouver. l'emplacement où elle est publiée peut varier selon les exigences de l'application.

- **Découvrir** Dans l'opération de recherche, le demandeur de service récupère une description du service directement ou interroge le registre de service pour le type de service requis. Cette opération peut être impliquée dans deux différentes phases du cycle de vie pour le demandeur de service : au moment du design pour récupérer la description d'interface de service pour le développement du programme et à l'exécution pour récupérer la liaison du service et l'emplacement de la description pour l'invocation.

- **lier (bind)** Finalement, un service doit être invoqué. Dans l'opération de liaison, le client de service invoque ou initie une interaction avec le service à l'exécution en utilisant les détails de liaison dans la description du service pour localiser, contacter et appeler le service.

Les services Web sont construits autour de standards qui sont SOAP, WSDL et UDDI assurant respectivement leur communication, leur description et leur découverte.

2.1.3.1 Communication : SOAP

Développé par IBM⁵ et Microsoft⁶ [7], L'approche SOAP est une recommandation W3C qui le définit comme étant un protocole destiné à l'échange de messages structurés, permettant d'invoquer des applications sur des réseaux distribués [13].

Ce protocole SOAP est basé sur XML pour mettre en place un mécanisme valable d'échange des données indépendant du modèle de programmation de l'application et du système d'exploitation.

Un message SOAP est un document XML constitué d'une enveloppe SOAP obligatoire, d'un en-tête SOAP facultatif et d'un corps SOAP obligatoire :

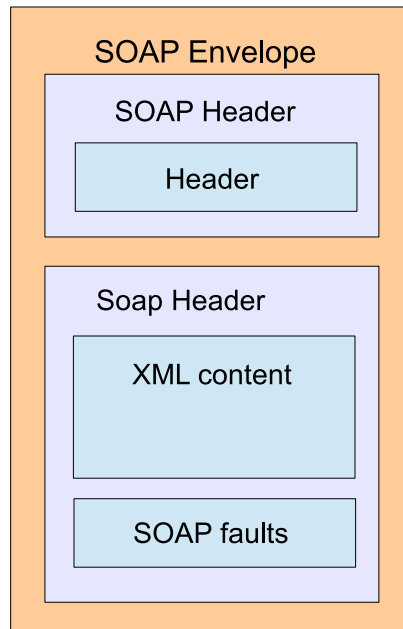
- **Enveloppe** : L'élément racine du message SOAP , définissant le contexte du message, son destinataire et son contenu, il englobe l'en-tête et le corps.
- **En-tête <Header>** : Un mécanisme générique permettant d'ajouter des fonctions à un message SOAP d'une manière modulaire sans accord préalable entre les parties en communication. Des exemples d'extension qui peuvent être implémentées comme des en-têtes sont des authentifications, des transactions, des paiements
- **Corps <Body>** : Contient les informations obligatoires destinées à l'ultimate destinataire du message, il sert comme un container pour les informations mandataires à l'intention du récepteur du message. SOAP définit un élément pour le corps, qui est l'élément <Fault> (Erreur) utilisé pour rapporter les erreurs.

2.1.3.2 Description : WSDL

Le langage de description des services Web WSDL [14] est une recommandation du W3C, maintenant dans sa deuxième version. WSDL est basé sur XML pour décrire les fonctions opérationnelles de services Web. La description des WSDL sont composées d'une interface et des implémentations. L'interface est une définition abstraite et réutilisable service qui peut être référencée par plusieurs implémentations.

5. <http://www.ibm.com>

6. <http://www.microsoft.com>



(a) Les éléments d'un message SOAP

```

<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<soapenv:Body>
<symbol xmlns="http://stock.samples">XXX</symbol>
</soapenv:Body>
</soapenv:Envelope>

```

(b) Exemple de message SOAP

FIGURE 2.1 – La structure d'un message SOAP

Le WSDL sert à décrire :

- le protocole de communication (SOAP RPC ou SOAP orienté message)
- le format de messages requis pour communiquer avec ce service
- les méthodes que le client peut invoquer
- la localisation du service.

2.1.3.3 Découverte : UDDI

UDDI [15] est une standardisation pour la publication et la découverte des services Web initialement conçue et spécifiée par le Consortium de standards OASIS⁷, et il est le résultat d'un accord d'un ensemble d'industriels Ariba⁸, IBM, Microsoft, etc en vue de devenir le registre standard de la technologie des services Web.

UDDI complète les technologies basiques de services Web en permettant de créer un **annuaire** permettant de localiser sur le réseau les services web recherchés, les services référencés dans UDDI sont accessibles par l'intermédiaire du protocole de communication SOAP, et la publication des informations concernant les fournisseurs et les services doit être spécifiée en XML afin que la recherche et l'utilisation soient faites de manière **dynamique** et **automatique**.

Un UDDI peut appartenir à un domaine public comme internet ou tout autre réseau accessible à un nombre non limité d'utilisateurs, comme il peut appartenir à un domaine restreint comme l'intranet d'une entreprise ou d'un groupe d'entreprise.

Les données stockés dans l'UDDI sont structurées (en XML) et organisées en trois parties connues :

Pages blanches : fournissent des descriptions générales sur les fournisseurs de services à savoir le nom de l'entreprise qui fournit le service, son identificateur commercial, ses adresses, etc.

Pages jaunes : comportent des descriptions détaillées sur les fournisseurs de services catalogués dans les pages blanches d'une de façon taxonomique (selon secteurs d'activités par exemple).

7. <https://www.oasis-open.org>

8. <http://www.ariba.com/>

Pages vertes : fournissent des informations techniques sur les services Web catalogués. Ces informations incluent la description du service, les adresses URL, du processus de son utilisation et des protocoles utilisés pour son invocation.

2.2 Description des services web

Une description du service Web est un document par lequel le fournisseur de services communique au client les spécifications pour invoquer le service Web. Dans cette section nous présentons les modèles de description des services web. Nous détaillons dans la première sous-section le modèle de description syntaxique WSDL [14] développé et standardisé par le W3C qui est devenu un élément essentiel dans des technologies services web. Ensuite en mettant l'accent sur les limitations majeurs de cette approche dans un environnement hétérogène qui nécessite un certain degré de dynamisme et d'automatisation. Finalement, Nous présentons les divers approches sémantiques visant à préciser la description d'un service en insistant sur les approches d'annotation sémantique et sur les ontologies de services.

2.2.1 Description syntaxique de services

Le langage de description de services Web WSDL [14] fournit un modèle ainsi qu'un langage basé sur XML de description de services Web. Un fichier WSDL comprend une description des fonctionnalités d'un service, mais il ne se préoccupe pas de l'implantation de celles-ci. Il contient aussi des informations concernant la localisation du service, ainsi que les données et les protocoles à utiliser pour l'invoquer. En pratique, le document WSDL⁹ est un document XML qui se divise en deux parties [16] :

- La définition **abstraite** de l'interface du service avec les opérations supportées par le service Web, ainsi que leurs paramètres et les types des données.
- La définition **concrète** de l'accès au service avec la localisation, par une

9. <http://www.w3.org/TR/wsdl20/>

adresse réseau du fournisseur de service¹⁰, et les protocoles spécifiques d'accès.

Un document WSDL constitué de quatre éléments principaux [14] : `<Types>`, `<Interface>`, `<Binding>`, `<Service>`.

[17] [16] [18]

- `<Types>` : L'élément `Types` sert à un conteneur définissant les données figurant dans les messages échangés par le service. WSDL supporte des types élémentaires prédéfinis (tels que les entiers, les chaînes de caractères et les dates). Si les données échangées possèdent une structure particulière, il est possible de les décrire à travers un schéma XML [19].

- `<Interface>` : Les interfaces WDSL offrent une manière abstraite de décrire la fonctionnalité du service, Contrairement à la représentation concrète offerte par les éléments de `<Bindings>` et de `<Services>` qui sera décrit plus tard. Une interface WSDL est constitué d'un ensemble d'opérations, chacun d'entre eux décrivant d'une simple interaction entre le service et le client. Une opération décrit un séquence des messages d'entrées/sorties ou un modèle d'échange de message¹¹ suivie lorsque l'opération est invoqué. Pour chaque message contenu dans le motif¹², un type de message est spécifié à l'aide des types qui ont été définis précédemment dans le document. WSDL contient huit modèles de messages prédéfinis, mais on peut facilement définir de nouveaux.

- `<Binding>` : [16] [17] L'élément `Binding` reprend les opérations de l'élément `<Interface>` et leurs associe un protocole de transfert et des spécifications des formats de données de message. La définition des protocoles de communication utilisés pour l'invocation du service Web permet d'établir le lien, d'une part, entre le document et les messages SOAP et d'autre part, entre les messages SOAP et les opérations invoquées.

- `<Service>` : Cet élément définit la localisation du service Web décrit. Pour chaque interface décrite, un élément service lui est associé. Le sous-élément `<endpoint>` définit un port d'accès en référant l'élément

10. Service Endpoint

11. message exchange pattern

12. pattern

<binding> associé et en déclarant l'URL localisant le service (avec l'attribut <address>).

2.2.2 Ajout de la sémantique

Malgré les améliorations apportées au standard WSDL dans son deuxième version [14], la description du service reste uniquement au niveau fonctionnel, c'est-à-dire qu'elle contient la manière dont on peut utiliser le service et non ce que fait le service, le standard WSDL est limité à l'énumération des opérations et à la description des types des paramètres d'entrée et de sortie associés, elle ne caractérise pas la sémantique de la fonctionnalité accomplie par le service. Par conséquent, la description WSDL reste insuffisante lors du processus de sélection. Pour pallier cette difficulté, plusieurs approches proposent de rajouter une couche au dessus sémantique de WSDL complétant la description syntaxique par des précisions sémantiques.

Dans un premier temps, on va essayer de clarifier la notion d'un services Web sémantique, puis étudie les langages émergents qui permettent de décrire ce type de services Web.

2.2.2.1 Définition des services Web sémantiques

L'objectif premier du Web sémantique est de définir et lier les ressources du Web afin de simplifier leur utilisation, leur découverte, leur intégration et leur réutilisation dans le plus grand nombre d'applications [20]. Le Web sémantique doit fournir l'accès à ces ressources par l'intermédiaire de descriptions sémantiques exploitables et compréhensibles par des machines. En effet, Les technologies du Web sémantique complètent le Web actuel avec des outils sémantiques. Il ne s'agit donc pas de créer un nouveau Web ou un Web séparé de l'existant : ce Web de données repose entièrement sur les technologies et concepts qui ont fait le succès du Web tel que nous le connaissons aujourd'hui [21].

La réalisation du Web sémantique trouve ces racines dans le développement des langages de balisage inspiré par des travaux issue de la communié AI [22], tels que OIL [23], DAML+OIL [24] et DAML+OTN [25] (ces deux derniers

langages sont parties de la famille DAML).

Ces langages ont une sémantique bien définies et permettent le balisage et la manipulation des taxonomie complexe et Des relations logiques entre les entités sur le Web. [26]

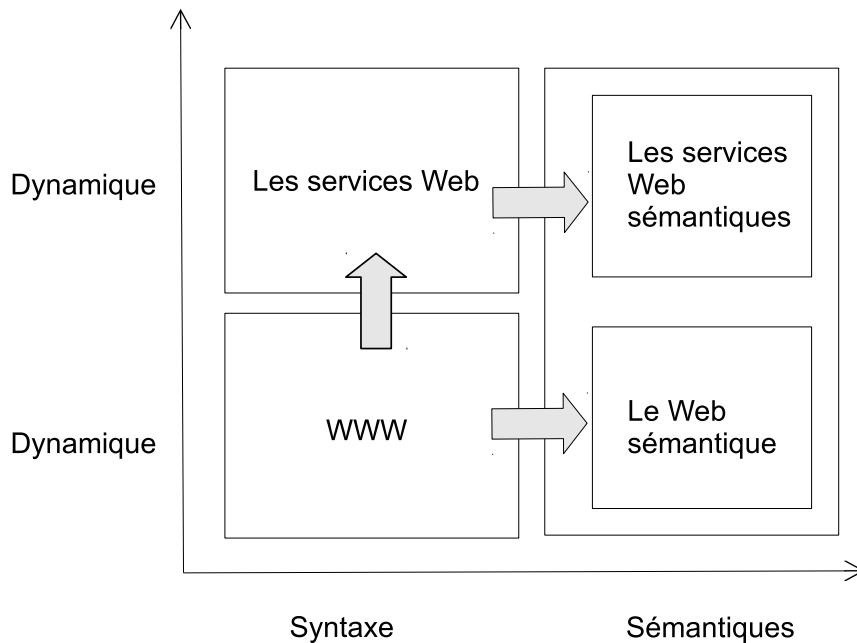


FIGURE 2.2 – Web evolution to Semantic Web services [1].

Cette description repose sur des ontologies. Selon Gruber [27], une ontologie est une spécification explicite d'une conceptualisation. Une conceptualisation est un modèle abstrait qui représente la manière dont les personnes conçoivent les choses réelles dans le monde et une spécification explicite signifie que les concepts et les relations d'un modèle abstrait reçoivent des noms et des définitions explicites. Le Web sémantique est devenu un domaine à part entière, preuve en est la création en 2001 du groupe de travail sur ce sujet par le W3C.

2.2.2.2 WSDL-S

WSDL-S [28] est le résultat d'un travail collaboratif entre IBM, laboratoire LSDSI et l'université de Geogia¹³. La spécification a devenue une recommandation

13. <http://www.uga.edu/>

W3C depuis 2005. Son objectif principal est de fournir un processus d'annotation sémantique compatible avec les technologies existantes. Pratiquement, Le méta-modèle WSDL-S repose sur les capacités du modèle WSDL en rajoutant trois éléments majeurs `<category>`, `<effect>` et deux attributs `modelReference` et `schemaMapping`. Les éléments introduits permettent de rajouter des informations qui n'étaient pas prises en compte dans WSDL comme *les préconditions* et *les effets* d'une opération. Tandis que les attributs permettent de référencer des concepts dans des ontologies de référence, ces préconditions et effets ensemble avec les annotations sémantiques des éléments `<inputs>` et `<outputs>` permet de l'automatisation du processus de découverte de services.

- L'élément `<category>`
- `<precondition>`
- `<effect>`
- L'attribut `modelReference`
- `schemaMapping`

2.2.2.3 SAWSDL

La spécification SAWSDL [29] est la suite de WSDL-S et il partage les mêmes principes de ce dernier. issue d'initiative du groupe de travail d'annotations sémantiques pour WSDL¹⁴ et soumise au W3C en 2007, SAWSDL définit un mécanisme d'annoter sémantiquement les interfaces et les opérations WSDL, ainsi que les types XML SCHEMA en les reliant à des concepts dans une ontologie. Cette annotation repose sur la définition d'attributs étendant le standard de description. Les annotations sémantiques référencent des ontologies pré-existantes. Le mécanisme d'annotation de SAWSDL est indépendant de tout langage de représentation [18] d'ontologies.

SAWSDL propose deux sortes d'annotations sémantiques : une pour identifier le concept sémantique (représentée par l'attribut `modelReference`) et une autre pour faire le lien entre le concept et le document WSDL (représentée par les attributs `liftingSchemaMapping` et `loweringSchemaMapping`).

14. Semantic Annotations for WSDL and XML Schema

2.2.2.4 OWL-S

2.2.2.5 WSMO

[17]

2.3 Découverte des services web

WS discovery is related to getting appropriate service for a request. It is one of the critical steps in the process of developing applications based on SOA. It can be done using syntactic matching or semantic matching[30].

2.4 Conclusion

Chapitre 3

La Composition des services web

3.1 Définition et stratégies de composition

Cette section a pour but d'exposer, d'une part, quelques définitions et objectifs de la composition des services Web proposées par la communauté, et d'autre part, les différents types et mécanismes de composition selon différents points de vue rencontrés dans la littérature.

3.1.1 Définitions

Martin *et al.* [31] définissent la composition comme étant *“le processus de sélection, de combinaison et d'exécution de services en vue d'accomplir un objectif donné”*.

Selon S. Dustdar et W. Schreiner [32] : *“ L'infrastructure de base des services Web suffit pour la mise en œuvre d'interactions simples entre un client et un service Web. Si la mise en œuvre d'une application métier implique l'invocation d'autres services web, il est nécessaire donc de combiner les fonctionnalités de plusieurs services web. Dans ce cas, nous parlons d'une composition de services Web”*.

En d'autre terme, La composition de services Web désigne une opération qui consiste à construire de nouvelles applications ou services appelés **services composites** ou agrégats par l'assemblage ou l'agrégation de services existants nommés **services atomiques** ou élémentaires.

Medjahed [33] de ça part a défini un service Web composite comme un *“conglomérat de sous-traitance services Web (services appelés participants) travaillant en tandem pour offrir un service à valeur ajoutée.”*

La composition de services Web vise essentiellement quatre objectifs [34] :

1. Créer de nouvelles fonctionnalités en combinant des services déjà existants.
2. Résoudre des problèmes complexes auxquels aucune solution n'a été trouvée.
3. collaborer plusieurs entreprises ensemble.
4. Optimiser et améliorer une fonctionnalité existante.

3.1.2 Cycle de vie d'une composition

Comme l'illustre la figure, le cycle de vie de la composition de services Web comprend quatre phases [2] : la phase de *définition*, La phase de *sélection*, la phase de *déploiement* et la phase d'*exécution* .

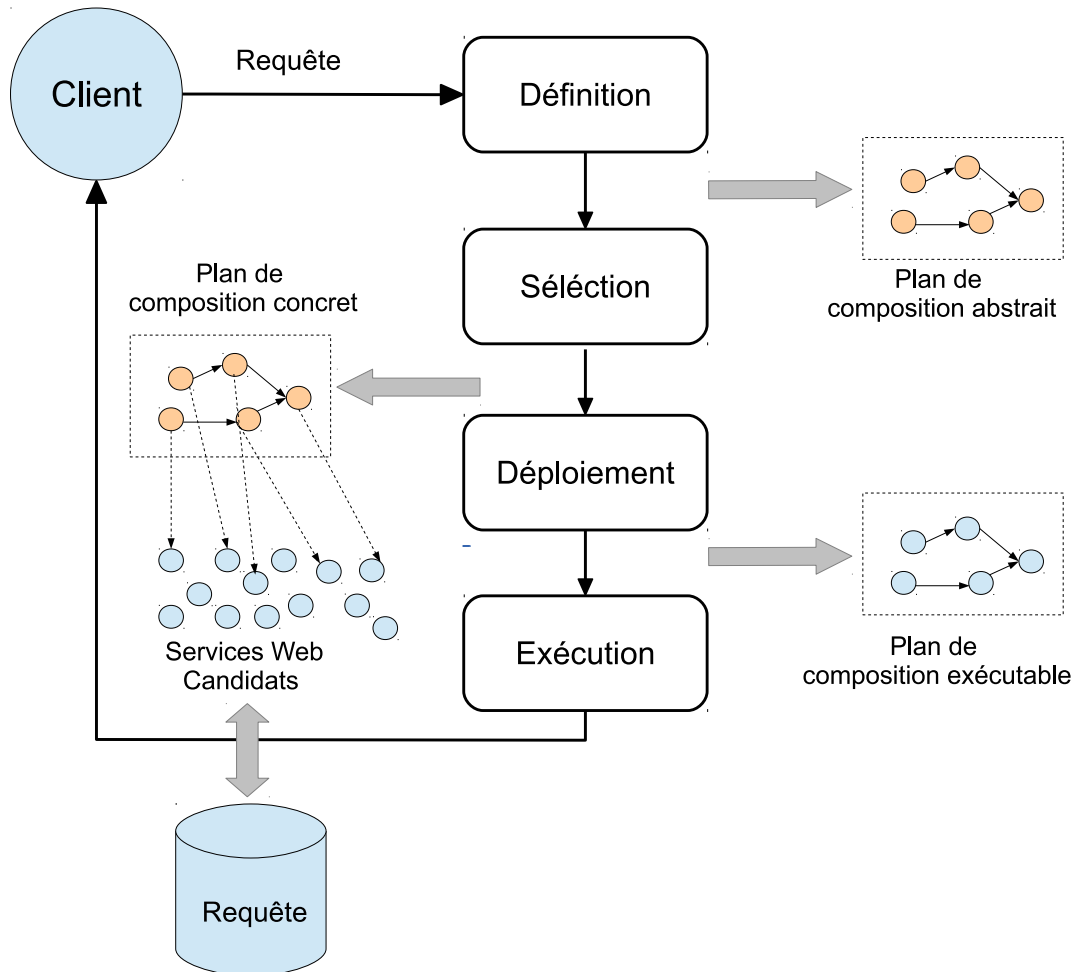


FIGURE 3.1 – Cycle de vie d'une composition des services [2].

- **La phase de définition** Pendant cette phase, le client de service spécifie les exigences de composition des services en termes des besoins et des préférences pour le service composite. L'exigence est ensuite *décomposée*, soit semi-automatique ou automatique, dans un modèle de **processus abstrait** (ce est à dire, le service composite abstraite), qui spécifie un ensemble d'activités, le contrôle et le flux de données entre eux, la qualité de service QoS et la gestion des exceptions.

- **La phase de sélection.** Dans cette phase, pour chaque activité dans le service composite, les services Web appropriés qui répondent aux exigences de l'activité sont situés en cherchant sur le registre de service, sur la base des informations contenues dans les documents de description de service publiées. Il est probable que plus d'un service de candidat de répondre aux exigences. Par conséquent, le meilleur le service identifié doit être sélectionné. Après tous les services Web requis sont identifiés et liés aux activités correspondantes, le service composite construit est produite.

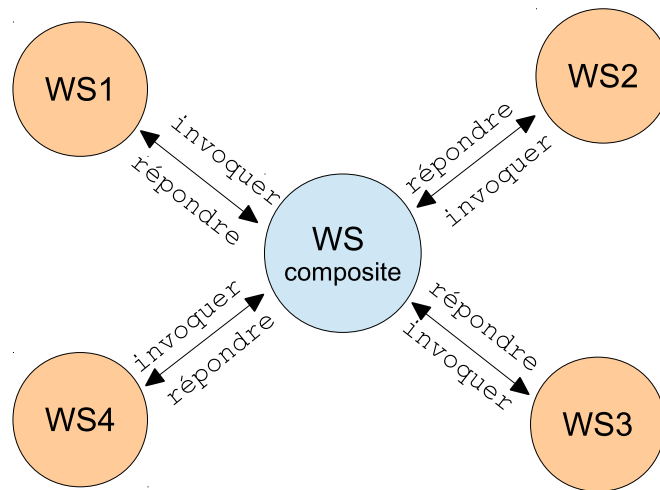
- **La phase de déploiement.** Dans cette phase, le service composite construit est déployé pour permettre son instanciation et l'invocation par les utilisateurs finaux. Le résultat de cette phase est le service composite exécutable.

- **La phase d'exécution.** Dans cette phase, l'instance de service composite est créé et exécuté par le moteur d'exécution, qui est aussi responsable de l'invocation des composants de service atomiques. Pendant l'exécution de l'instance de service composite, les tâches de surveillance, y compris le suivi d'exécution, mesure de la performance et la gestion des exceptions, doivent être effectuées.

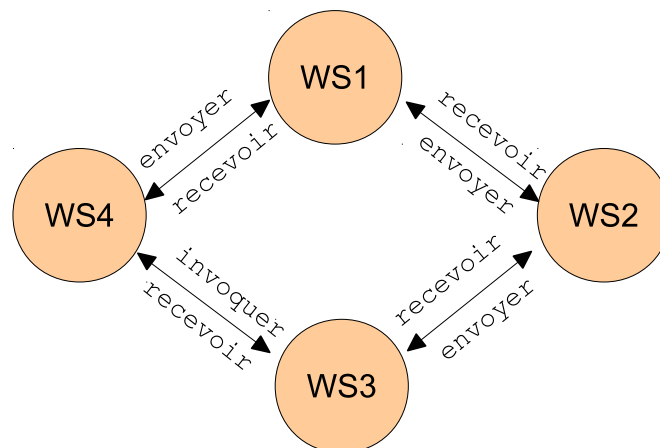
3.1.3 Procédés de coordination

Nous distinguons deux méthodes utilisées pour décrire la composition de services dans un flot de processus métier : l'*orchestration* de services et la *chorégraphie* des services. Ces deux procédés de coordination décrivent deux aspects de création des processus métiers à partir des services Web composites [35]. Un **procédé** est représenté par un graphe orienté d'activités ou un flot de contrôle qui donne l'ordre d'exécution des activités et la logique de coordination des services. Chaque activité représente une fonctionnalité réalisée concrètement par un service [36].

La figure 3.2 illustre ces deux approches en conjonction.



(a) l'orchestration des services.



(b) la chorégraphie des services.

FIGURE 3.2 – Orchestration vs Chorégraphie.

3.1.3.1 Orchestration

Selon Sonia *et al.* [37] : “*L’orchestration des services Web permet de définir l’arrangement et l’enchaînement de ces services selon un canevas bien défini. Elle décrit la manière par laquelle les services peuvent interagir ensemble tout en incluant l’ordre d’exécution des différentes interactions*”.

Barros *et al.* [38] définissent l’orchestration comme un ensemble de processus exécutés dans un ordre prédéfini afin de répondre à un but [18]. Ce type de composition se base sur un procédé métier exécutable permettant de décrire d’enchaînement et les interactions des différents services basiques collaborant dans une composition.

L’orchestration offre **une vision centralisée** de contrôle, le procédé est toujours contrôlé par l’un des partenaires métiers. Ce dernier joue le rôle d’un chef d’orchestre qui se charge d’appeler les services de la composition suivant l’ordre d’exécution déjà défini par le processus métier. Le principe de l’orchestration est illustré par La figure 3.2a.

3.1.3.2 Chorégraphie

Selon Sonia *et al.* [37] : “*La chorégraphie permet de tracer la séquence de messages échangés dans un contexte de composition de services Web. Elle est typiquement liée à la description de conversations existantes entre les services tout en impliquant plusieurs parties, incluant les clients, les fournisseurs et les partenaires*”.

D’après Barros *et al.* [38], la chorégraphie permet de décrire la composition comme un moyen d’atteindre un but commun en utilisant un ensemble de services Web. La collaboration entre chaque service Web de la collection (faisant partie de la composition) est décrite par des flots de contrôle [18].

La chorégraphie offre **une vision décentralisée** et **globale** du système et exprime une vue d’ensemble des services interagissant dans le cadre d’une composition de services. Selon Peltz [35], la chorégraphie illustre les différents échanges de messages entre les participants. Le principe de la chorégraphie est illustré par la figure 3.2b.

3.1.4 Stratégies de composition

Un modèle de composition de service peut être relativement complexe. Il requiert la description et l'organisation de l'interaction entre les services et nécessite la gestion de plusieurs aspects comme les échanges de données entre les services, les pannes ou erreurs éventuelles, le contexte d'interaction, le degré d'automatisation des tâches, etc...

Il existe une variété de spécifications, de langages et d'approches formelles développées par la littérature concernant la composition. Ces techniques sont également classées en fonction de différentes dimensions, et selon les travaux effectués dans le champ des services web, les définitions des types de composition diffèrent d'une communauté à l'autre.

Barros *et al.* [38] classent la composition des services Web en trois catégories : la composition comportementale, l'orchestration et la chorégraphie, à l'instar de Barros *et al.*, Peltz [35] considère que les deux dernières (*orchestration*, *chorégraphie*).

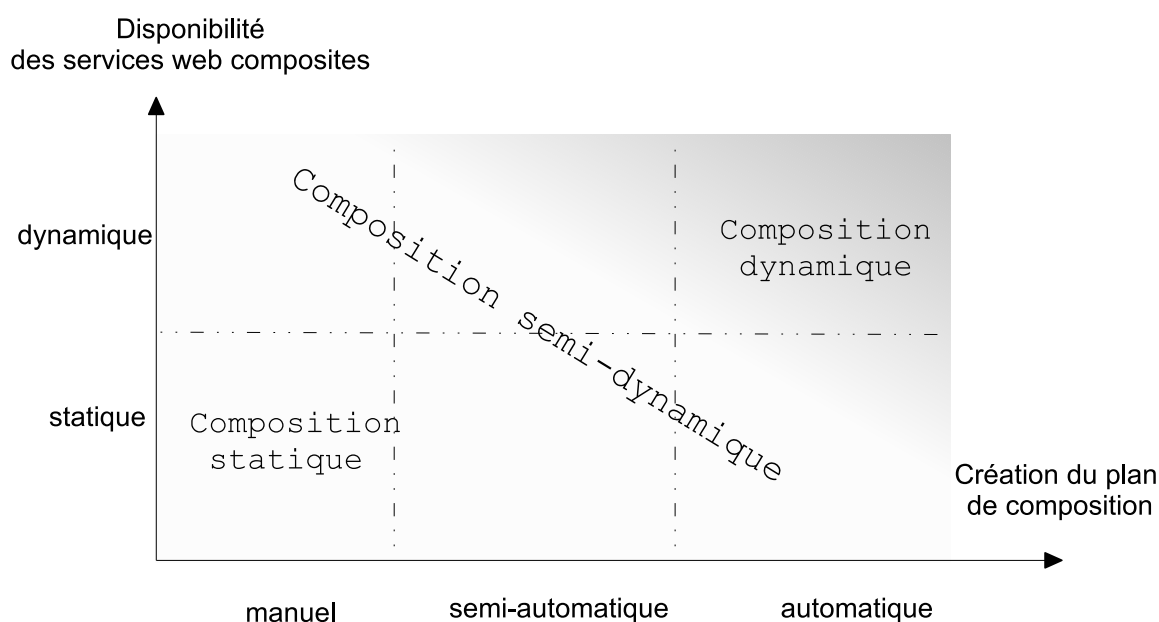


FIGURE 3.3 – Classification des stratégies de composition [3].

D'une autre façon, Fluegge *et al.* [3] dans une analyse de l'état de l'art considèrent l'orchestration et la chorégraphie comme des modèles d'exécution appli-

qués dans le contexte d'une composition. Il distingue trois stratégies de composition selon la disponibilité des services Web composites lors de composition et de le degré d'automatisation : composition **statique**, **semi-dynamique** et **dynamique** (voir la figure 3.3).

3.1.4.1 Composition statique/dynamique

Selon la disponibilité des services composites, La composition des services Web peut être soit une composition statique soit une composition dynamique [34] :

- **Composition statique** : est appelé aussi composition *off-line*, précompilé ou encore proactive. c'est une composition qui utilise des services basiques qui sont au préalable définis d'une façon figée et qui ne peuvent pas changer en fonction du contexte du client. Ce type de composition engendre des applications peu flexibles, parfois inappropriées avec les exigences des clients.
- **Composition dynamique** : appelée aussi composition *on-line*, post-compilée ou encore réactive. Elle se réfère à la sélection des services basiques à la volée. Autrement dit, la sélection des services basiques ne peut pas être définie à l'avance mais elle sera faite au moment de l'exécution en fonction des contraintes imposées par le client. Ceci permet d'élaborer différents scénarios de composition qui offrent les mêmes fonctionnalités et qui tiennent compte de la dynamique de la situation du client.

3.1.4.2 Composition manuel/automatique

Classification basée sur le degré d'automatisation.

- **Composition manuel** : Suppose que l'utilisateur gère la composition avec sa main, via un éditeur de texte et sans l'aide d'outils dédiés.
- **Composition semi-automatique** : C'est un pas en avant en comparaison avec la composition manuelle, dans le sens qu'elle fait des suggestions sémantiques pour aider à la sélection des services Web dans le processus de composition.

- **Composition automatique** : La composition automatique (ou encore dynamique selon [3]) permet un développement plus rapide des applications à base de services. Elle consiste à préciser la requête d'un utilisateur sous forme d'objectifs à satisfaire. Un moteur de composition "*intelligent*" choisit la combinaison de services répondant à l'objectif décrit. Il génère la composition de service adéquate de manière transparente à l'utilisateur. Ce principe a interpellé plusieurs communautés de recherche travaillant dans le domaine de l'Intelligence Artificielle. [16]

3.2 Langages pour la composition

Afin de supporter la composition de services Web, plusieurs langages de composition de services ont été proposés pour décrire et mettre en oeuvre une composition. Dans cette section on va faire un tour d'horizon de quelques standards et langages principaux rencontrés dans la littérature.

3.2.1 BPEL

BPEL est une spécification du consortium OASIS¹ issue de la fusion des spécifications XLANG Microsoft² et WSFL d'IBM³, il hérite les caractéristiques d'un langage structuré en blocs de XLANG, ainsi que les caractéristiques d'un graphe direct de WSFL [34].

BPEL (*appelé aussi BPEL4WS ou WS-BPEL*) est le langage d'**orchestration** le plus utilisé dans l'industrie permettant la coordination des interactions entre l'instance du service composite et ses partenaires sous forme d'un schéma XML (*le script d'orchestration*), il définit le processus, l'enchaînement et l'ordonnement des actions qui seront exécutées par le moteur d'orchestration, agissant comme une machine virtuelle capable d'exécuter **le procédé métier** intéreptable de **coordination** [36].

BPEL repose sur un modèle constitué d'activités de coordination qui peuvent être de deux types, les activités de base ou élémentaires comme l'invocation (in-

1. <https://www.oasis-open.org>

2. <http://www.microsoft.com>

3. <http://www.ibm.com>

voke) d'un service, l'attente d'une réponse et la génération d'une réponse (**reply**), et les activités composites permettant du contrôle du flot de données comme les séquences (**sequence**), les exécutions en parallèle (**flow**) et les branchements (**switch, if**).

3.2.2 WS-CDL

WS-CDL est un langage de composition de services de type **chorégraphie** qui permet de décrire une vision **globale** des collaborations entre les services Web [16], à l'instar des standards de services Web, WS-CDL est basé sur XML, il complète la description WSDL des services Web afin de décrire les interactions entre les participants (les services Web) de la composition.

WS-CDL reprend et développe la spécification WSCI décrivant les séquences ordonnées de messages impliquant plusieurs entités (services Web) engagés dans une composition visant à accomplir un objectif commun.

WS-CDL consiste à définir un fichier XML décrivant une chorégraphie, il permet de [16] :

- désigner les variables et les types de données échangées.
- décrire les activités impliquées.
- décrire les structures illustrant les interactions entre les activités.

3.2.3 WSMF

WSMF [39] est une initiative européenne pour fournir une plate-forme riche de modélisation décrivant plusieurs aspects de services web. Son objectif principale est de permettre le commerce électronique (*E-commerce*) par l'application de web sémantique aux services Web.

Le standard WSMF est centré autour de deux principes complémentaire :

3.2.4 OWL-S

OWL-S [31] désigné par DAML-S [40] dans les versions antérieures, est un langage issue des travaux de DARBA⁴ et son programme DAML⁵ en collaboration avec des chercheurs de plusieurs universités et organisations. Il a été intégré au consortium W3C en 2004, au sein du groupe d'intérêt sur les services Web sémantiques, lors de la recommandation du langage OWL [24] [41]. Ankolekar *et al.* [40] présentent une ontologie pour les services web dans le but d'automatiser la *découverte*, *l'invocation*, la *composition* et la *surveillance* de l'exécution des services [42], les auteurs reprennent la notion de classes d'OWL et proposent l'ontologie OWL-S.

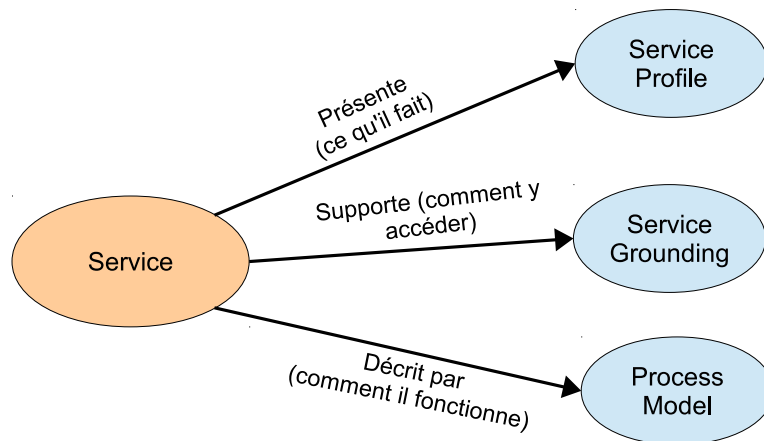


FIGURE 3.4 – Les éléments d'une ontologie OWL-S

L'objectif principal de ces recherches est d'établir une plateforme dans laquelle les descriptions des services Web sont partagés en utilisant une ontologie standard, constituée d'un ensemble de classes de base et des propriétés pour résoudre les ambiguïtés et de rendre la description d'un service compréhensible par une machine.

la figure 3.4 décrit la structure tripartite d'une ontologie OWL-S. Elle est composée de trois sous-ontologie : un *service profile*, d'un *service model* et d'un *service grounding*.

4. <http://www.darpa.mil/>

5. <http://www.daml.org/services/>

- **ServiceProfile**. il offre une description informelle des fonctionnalités rendues par le service (**serviceName**, **textDescription**) et des informations concernant son fournisseur (**contactInformation**). D'un autre côté, il spécifie des fonctionnalités offertes par le service (comportement fonctionnel) en terme de transformation d'information dénotée par les *Entrées/Sorties* (I/O) (**hasInput**, **hasOutput**) et de changement d'état après l'exécution du service dénoté par les *pré-conditions/effets* (P/E) (**hasPrecondition**, **hasResult**).

Du point de vue de découverte et de la composition, Le **ServiceProfile** est la partie la plus importante de la définition du service OWL-S.

- **ServiceModel**. il décrit le fonctionnement du service du en indiquant comment les résultats sont produits étape par étape précisant la façon un client peut interagir avec le service afin d'atteindre sa fonctionnalité. Ceci est fait en exprimant la transformation de données avec *Entrées/Sorties* (I/O) et la transformation de l'Etat avec *pré-conditions/effets* (P/E).

le **ServiceProfile** est généralement considéré comme un sous-ensemble du **ServiceModel**, contenant uniquement l'information nécessaire pour annoncer le service Web pour une découverte ultérieure.

- **ServiceGrounding**. Permet de spécifier les détails d'accès au service en précisant le protocole, le format des messages, la sérialisation et l'adressage. Il représente une correspondance (*mapping*) entre la définition abstraite d'un processus OWL-S décrivant le service et la définition WSDL concrète des éléments nécessaires pour interagir avec le service.

Le rôle de mise en correspondance est principalement à combler l'écart entre la description sémantique des services Web (détallée dans les deux premières sous-ontologies) et les modèles de description de service existants qui sont principalement syntaxique (WSDL).

Le **ServiceProfile** fournissent éventuellement d'autres informations supplémentaires sur le service comme la qualité qu'il assure en terme du temps de réponse et du coût, une classification possible d'un service (**serviceCategory**), et un paramètre générique **serviceParameter**.

- Le processus atomique directement invoqué par l'intermédiaire d'un Grounding
- Le processus composé décomposable en d'autres processus plus simples en utilisant les commandes de contrôle (par exemple : Si-Alors-Sinon, ...)
- Le processus simple non-invoquable, il fournit simplement une vue d'un processus atomique ou une représentation simplifiée d'un processus composé.

Les composants principaux d'un modèle de processus sont l'ontologie de processus (Process Ontology) et l'ontologie de contrôle du processus (Process Control Ontology). L'ontologie de processus, décrit un service en termes d'IOPE. Cette ontologie peut être utilisée afin de supporter l'invocation et la composition automatique de services web. L'ontologie de contrôle de processus décrit chaque processus comme étant un état en prenant en compte son activation, son exécution et sa terminaison.

Nous pouvons constater que le langage OWL-S s'adapte bien au besoin des méthodes de planification qui exigent une description de l'état du monde avant et après l'exécution d'un service donné.

Any language is barely practical if its usage is not supported by tools. Several tools are proposed to create, or manipulate the OWL-S descriptions of services. The most usable are OWL-S API, WSRF2OWLS, JAX-SA (Bab et al., 2006, ik 2007; Habala et al., 2006). JAX-SA is an enhancement of WSRF2OWLS. The tool provides an interface on the top of the OWL-S API to create OWL-S service descriptions. The OWL-S generation process is semi-automatic. It requires a manually created configuration file depicting the mapping between the I/O parameters defined in WSDL and the ontological classes defined in OWL. Based on this, it automatically generates the semantic annotation. The disadvantage of all the tools is a lack of support to describe the pre-/post-conditions.[43]

nous montrons dans la section suivante qu'elle présente des limitations pour en décrire les aspects non fonctionnels.[44]

- les limitations de OWLS [44]
- d'augmentation de OWLS par QoS [17], [45]

3.2.5 Comparaison

Une composition des services Web nécessite la satisfaction de plusieurs exigences techniques [2], [46], le tableau 3.1 montre une comparaison entre les langages et standards étudiés dans cette section selon les critères suivants :

Langages	BPLE	WS-CDL	WSFM	OWLS-S
Composabilité	+	+	+	+
Representation du rôle	+	+	+	+
Support des structures complexes	+	+	+	+
Gestion des exceptions	+	+	+	+
Dupport du sémantique	+	+	+	+
Support industriel	+	+	+	+

(+) support, (-) pas du support.

TABLE 3.1 – Comparaison des standards et langages de composition

- **La composabilité** indique la capacité d’assembler des services participants dans un processus de composition et de modéliser les interactions entre eux.
- **La representation du rôle** indique La représentation de rôle indique la capacité de refléter la le comportement que le participant doit présenter afin d’interagir dans le processus de composition.
- **Le support des structures complexes** la capacité de modéliser les structures complexes qui reflètent les règles des actions réalisées dans le processus de composition logique d’exécution et de commande.
- **La gestion des exceptions** est la capacité de gérer les exceptions de processus lors de l’exécution du processus de composition
- **le support du sémantique** est la capacité de représenter la sémantique de services participants pour faciliter la découverte des services et la composition dynamique.
- **le support industriel** est la capacité de représenter la sémantique de services participants pour faciliter la découverte des services et la composition dynamique.

3.3 Composition dynamique des services web

3.3.1 Les approches basées sur les workflow

3.3.2 Les approches guidées par les modèles

3.3.3 Les approches mathématiques

3.3.4 Techniques de planification

3.4 Conclusion

Introduire la composition dynamique basé sur le modèle graphe qui se sera détaillé dans le prochain chapitre.

Chapitre 4

Les approches de composition dynamique des services Web sémantiques basés sur le modèle graphe

4.1 Préliminaires

4.1.1 Matching des services Web

([47], [48], [49])

Le matching, l'alignement, le mapping...ect. Sont tous des termes utilisés pour référencer le concept de recherche de similarité. Voici quelques définitions de ces termes utilisés.

Definition 1 (Matching). *c'est le processus de découverte des liaisons et des correspondances entre les entités de différentes représentations à travers un algorithme de matching.*

Dans le contexte d'une architecture à services, le *Matchmaking* (ou le *Matching*) est définie comme un processus qui nécessite un annuaire des services de prendre une requête en entrée, et de revenir tous les services qui peuvent satisfaire les exigences spécifiées dans la requête d'entrée [49]. Formellement, Le processus de *Matchmaking* peut se spécifié comme de suit :

Soit Φ est l'ensemble de toutes les services référencés dans un annuaire des services donné. Pour une requête R .

Definition 2 (Alignement). *on parle souvent de l'alignement des ontologies. C'est un ensemble de correspondances entre deux ou plusieurs ontologies. L'alignement est la sortie d'un processus de matching*

Definition 3 (Mapping). *chercher des correspondances pour établir des transformations entre deux objets de même nature mais pas de même forme. Par conséquent, le mapping utilise les résultats du matching pour effectuer les transformations des objets.*

Comme on le voit, tous ces termes prennent bien en compte la notion de recherche de correspondance entre des concepts, dans certains contextes ils sont utilisés indifféremment. Pour les services web, on parlera du matchnig ou match-making pour exprimer la mise en correspondance entre deux entités, et du mapping pour exprimer la transformation ou conversion des types de données.

4.1.2 Mésures de similarité

4.1.3 Graphe de dépendance

[30]

4.2 Travaux connexes

4.2.1 Génération Online du graphe de dépendance

4.2.2 Génération Offline du graphe de dépendance

4.2.3 Autres travaux basés sur le graphe matching

4.3 Vers les bases de données graphe

4.4 Conclusion

Deuxième partie

L'approche proposée

Chapitre 5

Une méthode de composition dynamique des services Web sémantiques utilisant neo4j

5.1 Reformulation mathématique du problème

5.1.1 Définitions de base

5.1.2 Hypothèses du travail

5.1.3 Objectifs

5.2 Exemple d'illustration

5.3 Annotation sémantiques des services web

5.4 Architecture du système

5.5 Conclusion

Chapitre 6

L'Implémentation d'un prototype

6.1 Exemple d'illustration

6.2 Outil d'annotation sémantiques des services

6.3 Publication des services web atomiques

6.4 Découvert d'un service web composites

6.5 Conclusion

Bibliographie

- [1] Dieter FENSEL et Christoph BUSSLER : Semantic web enabled web services. *Advances in Artificial Intelligence*, page 316, 2002.
- [2] Quan Z SHENG, Xiaoqiang QIAO, Athanasios V VASILAKOS, Claudia SZABO, Scott BOURNE et Xiaofei XU : Web services composition : A decade's overview. *Information Sciences*, 280:218–238, 2014.
- [3] Matthias FLUEGGE, Ivo JG SANTOS, Neil Paiva TIZZO et Edmundo RM MADEIRA : Challenges and techniques on the road to dynamically compose web services. *In Proceedings of the 6th international conference on Web engineering*, pages 40–47. ACM, 2006.
- [4] Heather KREGER *et al.* : Web services conceptual architecture (wsca 1.0). *IBM Software Group*, 5:6–7, 2001.
- [5] W3C Working GROUP : Web services architecture. <http://www.w3.org/TR/ws-arch/>, 2004.
- [6] Paul FREMANTLE, Sanjiva WEERAWARANA et Rania KHALAF : Enterprise services. *Communications of the ACM*, 45(10):77–82, 2002.
- [7] Don BOX, David EHNEBUSKE, Gopal KAKIVAYA, Andrew LAYMAN, Noah MENDELSON, Henrik Frystyk NIELSEN, Satish THATTE et Dave WINER : Simple object access protocol (soap) 1.1, 2000.
- [8] Tim BRAY, Jean PAOLI, C Michael SPERBERG-MCQUEEN, Eve MALER et François YERGEAU : Extensible markup language (xml). *World Wide Web Consortium Recommendation REC-xml-19980210*. <http://www.w3.org/TR/1998/REC-xml-19980210>, 1998.
- [9] Mike P PAPAIOGLOU : Service-oriented computing : Concepts, characteristics and directions. *In Web Information Systems Engineering, 2003. WISE*

2003. *Proceedings of the Fourth International Conference on*, pages 3–12. IEEE, 2003.
- [10] Francisco CURBERA, William NAGY et Sanjiva WEERAWARANA : Web services : Why and how. *In OOPSLA 2001 Workshop on Object-Oriented Web Services*, 2001.
 - [11] Francisco CURBERA, Matthew DUFTLER, Rania KHALAF, William NAGY, Nirmal MUKHI, Sanjiva WEERAWARANA *et al.* : Unraveling the web services web. *IEEE Internet computing*, 6(2):86–93, 2002.
 - [12] Karl GOTTSCHALK, Stephen GRAHAM, Heather KREGER et James SNELL : Introduction to web services architecture. *IBM Systems journal*, 41(2):170–177, 2002.
 - [13] Nilo MITRA, Yves LAFON *et al.* : Soap version 1.2 part 0 : Primer. *W3C recommendation*, 24:12, 2003.
 - [14] Roberto CHINNICI, Jean-Jacques MOREAU, Arthur RYMAN et Sanjiva WEERAWARANA : Web services description language (wsdl) version 2.0 part 1 : Core language. *W3C recommendation*, 26:19, 2007.
 - [15] Luc CLEMENT, Andrew HATELY, Claus von RIEGEN, Tony ROGERS *et al.* : Uddi version 3.0. 2, uddi spec technical committee draft. *OASIS UDDI Spec TC*, 2004.
 - [16] ABI LAHOUD ELIE : *Composition dynamique de services : application à la conception et au développement de systèmes d’information dans un environnement distribué*. Thèse de doctorat, Université de Bourgogne, 2010.
 - [17] George BARYANNIS et Dimitris PLEXOUSAKIS : Automated web service composition : State of the art and research challenges. *ICS-FORTH, Tech. Rep*, 409, october 2010.
 - [18] Céline LOPEZ-VELASCO : *Sélection et composition de services Web pour la génération d’applications adaptées au contexte d’utilisation*. Thèse de doctorat, Université Joseph-Fourier-Grenoble I, 2008.
 - [19] XML Schema PART : 2 : Datatypes. *W3C Recommendation*, 2, 2001.
 - [20] Tim BERNERS-LEE, James HENDLER, Ora LASSILA *et al.* : The semantic web. *Scientific american*, 284(5):28–37, 2001.

- [21] Alexandre BERTAILS, Ivan HERMAN et Sandro HAWKE : Le web sémantique. *Réalités industrielles*, (4):84–89, 2010.
- [22] Sheila A McILRAITH, Tran Cao SON et Honglei ZENG : Semantic web services. *IEEE intelligent systems*, 16(2):46–53, 2001.
- [23] Dieter FENSEL, Frank VAN HARMELEN, Ian HORROCKS, Deborah L MCGUINNESS et Peter F PATEL-SCHNEIDER : Oil : An ontology infrastructure for the semantic web. *Intelligent Systems, IEEE*, 16(2):38–45, 2001.
- [24] Ian HORROCKS *et al.* : Daml+oil : a description logic for the semantic web. *IEEE Data Eng. Bull.*, 25(1):4–9, 2002.
- [25] Deborah L MCGUINNESS, Richard FIKES, Lynn Andrea STEIN et James A HENDLER : Daml-ont : An ontology language for the semantic web. *In Spinning the Semantic Web*, pages 65–93, 2003.
- [26] Dieter FENSEL, Jim HENDLER, Henry LIEBERMAN et Wolfgang WAHLSTER : Creating of semantic web. *Im Internet verfügbar unter <http://citeseer.nj.nec.com/481673.html>*, 2000.
- [27] Thomas R GRUBER : A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.
- [28] Rama AKKIRAJU, Joel FARRELL, John A MILLER, Meenakshi NAGARAJAN, Amit SHETH et Kunal VERMA : Web service semantics-wsdl-s. 2005.
- [29] Jacek KOPECKY, Tomas VITVAR, Carine BOURNEZ et Joel FARRELL : Sawsdl : Semantic annotations for wsdl and xml schema. *Internet Computing, IEEE*, 11(6):60–67, 2007.
- [30] Abrehet Mohammed OMER : *A framework for Automatic Web Service Composition based on service dependency analysis*. Thèse de doctorat, Technical University of Dresden, 2011.
- [31] David MARTIN, Mark BURSTEIN, Jerry HOBBS, Ora LASSILA, Drew McDERMOTT, Sheila McILRAITH, Sridhar NARAYANAN, Massimo PAOLUCCI, Bijan PARSIA, Terry PAYNE *et al.* : Owl-s : Semantic markup for web services. *W3C member submission*, 22:2007–04, 2004.
- [32] Schahram DUSTDAR et Wolfgang SCHREINER : A survey on web services composition. *International journal of web and grid services*, 1(1):1–30, 2005.

- [33] Brahim MEDJAHED : *Semantic Web Enabled Composition of Web Services*. Thèse de doctorat, Virginia Polytechnic Institute and State University, 2004.
- [34] Maha DRISS : *Approche multi-perspective centrée exigences de composition de services Web*. Thèse de doctorat, Université Rennes 1, 2011.
- [35] Chris PELTZ : Web services orchestration and choreography. *Computer*, 36(10):46–52, 2003.
- [36] Stéphanie CHOLLET : *Orchestration de services hétérogènes et sécurisés*. Thèse de doctorat, Université Joseph-Fourier-Grenoble I, 2009.
- [37] Sonia JAMAL *et al.* : *Environnement de procédé extensible pour l’orchestration-Application aux services web*. Thèse de doctorat, Université Joseph-Fourier-Grenoble I, 2005.
- [38] Alistair BARROS, Marlon DUMAS et Phillipa OAKS : Standards for web service choreography and orchestration : Status and perspectives. *In Business process management workshops*, pages 61–74. Springer, 2006.
- [39] Dieter FENSEL et Christoph BUSSLER : The web service modeling framework wsmf. *Electronic Commerce Research and Applications*, 1(2):113–137, 2002.
- [40] Anupriya ANKOLEKAR, Mark BURSTEIN, Jerry R HOBBS, Ora LASSILA, David MARTIN, Drew McDERMOTT, Sheila A McILRAITH, Srinu NARAYANAN, Massimo PAOLUCCI, Terry PAYNE *et al.* : Daml-s : Web service description for the semantic web. *In The Semantic Web ISWC 2002*, pages 348–363. Springer, 2002.
- [41] Deborah L MCGUINNESS, Frank VAN HARMELEN *et al.* : Owl web ontology language overview. *W3C recommendation*, 10(2004-03):10, 2004.
- [42] Sheila A McILRAITH et David L MARTIN : Bringing semantics to web services. *Intelligent Systems, IEEE*, 18(1):90–93, 2003.
- [43] Peter BARTALOS : Effective automatic dynamic semantic web service composition. *Inf. Sci. and Technol. Bulletin ACM Slovakia*, 3(1):12–14, 2011.
- [44] Stéphane JEAN, Francisca LOSAVIO, Alfredo MATTEO et Nicole LEVY : Prise en compte des standards de qualité et des préférences utilisateurs pour la modélisation des propriétés non fonctionnelles dans owl-s. *Technique et Science Informatiques*, 31(1):39–69, 2012.

- [45] Kyriakos KRITIKOS et Dimitris PLEXOUSAKIS : Requirements for qos-based web service description and discovery. *Services Computing, IEEE Transactions on*, 2(4):320–337, 2009.
- [46] Antonio BUCCHIARONE et Stefania GNESI : A survey on services composition languages and models. *In International Workshop on Web Services–Modeling and Testing (WS-MaTe 2006)*, page 51, 2006.
- [47] Freddy LÉCUÉ et Alain LÉGER : A formal model for semantic web service composition. *In The Semantic Web-ISWC 2006*, pages 385–398. Springer, 2006.
- [48] Massimo PAOLUCCI, Takahiro KAWAMURA, Terry R PAYNE et Katia SYCARA : Semantic matching of web services capabilities. *In The Semantic Web ISWC 2002*, pages 333–347. Springer, 2002.
- [49] Lei LI et Ian HORROCKS : A software framework for matchmaking based on semantic web technology. *International Journal of Electronic Commerce*, 8(4), 2004.