Modélisation et utilisation de ressources et services Web et indexation de données dans un contexte d'incertitude.

Asma Omri

THÈSE

DOCTORAT EN COTUTELLE

Délivré par : *Université Claude Bernard Lyon 1, France*
Université de Sousse, Tunisie Institut Supérieur d'Informatique et Techniques de Communication.

En vue de l'obtention du diplôme de
DOCTEUR EN SCIENCES de l'INFORMATIQUE

Présentée et soutenue le *30/08/2018* **par :**

Asma Omri

Modélisation et utilisation de ressources et services Web et indexation de données dans un contexte d'incertitude.

JURY

Ouajdi KORBAA	Université de Sousse	Examinateur
Hajer BAAZAOUI	Université de la Mannouba	Examinatrice
Nadjet KAMEL	Université de Sétif	Examinatrice
Allel HADJALI	ENSMA Poitier	Rapporteur
Ikram AAMOUS	Université de Sfax	Rapporteur
Djamal BENSLIMANE	Université Claude Bernard Lyon 1	Directeur de thèse
Mohamed Nazih OMRI	Université de Sousse	Directeur de thèse
Karim BENOUARET	Université Claude Bernard Lyon 1	Co-directeur de thèse

---

Unités de Recherche :
LIRIS
MARS

Résumé — Il est communément admis que la production de données connait, depuis plusieurs années, un développement spectaculaire en raison de la multiplication des nouvelles technologies telles que les réseaux sociaux, les nouveaux appareils mobiles, les compteurs intelligents, les capteurs et le cloud computing. De fait, cette explosion de données devrait se poursuivre et même accélérer. S'interroger sur la façon dont on devrait traiter cette masse de qui devient de plus en plus variée, complexe et moins structurée, est alors essentiel. DaaS (Data As A Service) peut être définie comme l'approvisionnement, la gestion et la fourniture de données présentées dans un format immédiatement consommable aux utilisateurs professionnels des organisations en tant que service. Les données retournées par ces services se caractérisent généralement par l'incertitude et l'hétérogénéité.

Nombreux sont les approches qui traitent les données selon le cycle de vie du service Web qui repose sur 6 phases à savoir la création, la sélection, la découverte, la modélisation, l'invocation et la composition des services, dans le but de résoudre le problème de volume de données, de son hétérogénéité ou de sa vitesse d'évolution. En revanche, il y a très peu d'approches qui s'intéressent à la qualité de données et au traitement de son incertitude dans le Web.

Nous nous sommes naturellement intéressés, dans cette thèse, à la question des services Web dans un contexte de systèmes distribués et hétérogènes. La principale contribution à apporter dans le cadre de ce travail de recherche est d'étudier la composition de services et/ou de ressources Web et l'indexation de données dans un contexte incertain.

Dans un premier temps, au travers des apports de la littérature, le cadre théorique relatif aux spécificités du concept de service DaaS incertain, est présenté en adoptant la théorie possibiliste. Le problème de la composition de services Web et l'impact de l'incertitude, qui peut être associée à la sortie d'un service, sur les processus de sélection et de composition des services sont explicités. Pour ce faire, nous avons proposé une approche possibiliste afin de modéliser l'incertitude des données renvoyées par des services incertains. Plus précisément, nous avons étendu les normes de description de service Web (par exemple, WSDL) pour représenter les degrés d'incertitude des sorties. Nous avons également étendu le processus d'invocation de service pour prendre en compte l'incertitude des données d'entrée. Cette extension est basée sur la théorie des mondes possibles utilisée dans les bases de données possibilistes. Nous avons également mis en avant un ensemble d'opérateurs de composition, sensibles aux valeurs d'incertitude, dans le but d'orchestrer

des services de données incertains.

Dans un deuxième temps, nous avons étudié l'impact de l'incertitude sur la représentation et la manipulation des ressources Web. Nous avons défini le concept de ressource Web incertaine et proposé des mécanismes de composition de ressources. Pour ce faire, un modèle de description de l'incertitude à travers le concept de ressource Web incertaine a été présenté. Celui-ci est basé sur un modèle probabiliste où chaque ressource peut avoir plusieurs représentations possibles, avec une certaine probabilité.

Enfin, et dans un dernier temps, nous avons proposé des méthodes d'indexation documentaire des données de type Big Data. Au commencement, nous avons adopté une approche d'indexation syntaxique de données incertaines, ensuite, nous avons suivi une méthode d'indexation sémantique incertaine. Enfin, et pour booster cette démarche, nous avons proposé une méthode hybride d'indexation dans un contexte incertain.

Mots clés : Services Web, Ressources Web, Incertitude, Indexation, Recherche d'informations, possibiliste, probabiliste.

Remerciements

Cette thèse est l'aboutissement de trois années d'apprentissage nourries de multiples échanges sans lesquels elle n'aurait pu voir le jour. Je tiens à remercier toutes les personnes m'ayant accompagné tout au long de cette thèse.

Tout d'abord, je tiens à adresser mes chaleureux remerciements à mes directeurs de thèse Djamal Benslimane et Mohamed Nazih omri pour leurs aides et leurs précieux conseils ainsi que leurs qualités humaines. Leurs compétences et leurs rigueurs scientifiques m'ont beaucoup appris. Je remercie également mon co-encadrant Karim Benouaret, pour toutes les réunions de travail, pour son aide, ses conseils et l'attention portée sur mes travaux. J'ai pris beaucoup de plaisir à travailler avec eux et je leur adresse ma très profonde gratitude.

Je souhaite adresser mes sincères remerciements aux personnes qui ont accepté la tâche délicate de rapporter ce mémoire et qui ont eu la patience de juger ce travail Allel Hadjali, Professeur des Universités en Informatique à l'Université Chasseneuil-du-Poitou, et Ikram Aamous, Maître de Conférences en Informatique à l'université de Sfax (Tunisie).

Je souhaite remercier également les examinateurs et membres du jury, Nadjet kamel, Professeur des Universités à l'université de Sétif (Algérie), Hajer Baazaoui, Maître de Conférences, Université de la Mannouba Tunis (Tunisie), et, Ouajdi Korbaa Professeur à Université de Sousse (Tunisie) d'avoir accepté de participer au jury.

Je remercie spécialement les secrétaires de laboratoire LIRIS Madame Brigitte Gudayer et Isabelle pour sa spontanéité et son sourire qui a égayé nos journées.

Mes remerciements vont également à tous les membres du laboratoire LIRIS pour leurs compétences et leur esprit de dialogue. Mes remerciements s'adressent aussi au professeur Mohand Said.Hacid, directeur du LIRIS, pour les très bonnes conditions de travail qu'il m'a offertes les premières années de ma thèse et pour le financement de nombreuses conférences où j'ai pu présenter mes travaux. Je suis également très reconnaissant aux membres de laboratoire MARS. Je souhaite également remercie ceux que j'ai eu l'occasion de rencontrer lors de cette thèse et qui, par leurs commentaires sur mes différents travaux, m'ont permis d'améliorer et d'enrichir mon. Enfin, tous remerciements à ma famille, mes proches et mes amis pour tous les encouragements et la confiance qu'ils m'ont témoignés.

Les années de la thèse n'auraient pas été possibles sans le soutien de certaines personnes formidables : Je remercie en premier lieu ma très chère amie de Tunisie : Rim pour tout ce qu'elle représente pour moi.

J'adresse toute mon affection à ma famille et en particulier à mes parents, mes sœurs, et mes frères. Malgré mon éloignement, votre confiance, encouragement et amour me guident tous les jours. C'est grâce à vous que je suis arrivée jusqu'ici. Je vous aime beaucoup. Enfin, le dernier et pas des moindres, je remercier tout particulièrement mon fiancé Waleed pour son soutien quotidien, sa patience et son amour, je remercie également sa famille.

Liste des publications

[1] Asma Omri, Karim Benouaret, Djamal Benslimane, Mohamed Nazih Omri : Towards an understanding of cloud services under uncertainty : A possibilistic approach. Int. J. Approx. Reasoning 98 : 146-162 (2018)

[2] Asma Omri, Karim Benouaret, Mohamed Nazih Omri, Djamal Benslimane : Toward a New Model of Indexing Big Uncertain Data. MEDES 2017 : 93-98

[3] Asma Omri, Karim Benouaret, Mohamed Nazih Omri, Djamal Benslimane : Querying Data Services in an Uncertain Environment : A Possibilistic-Based Approach. SITIS 2016 : 246-251

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction Générale

Sommaire

1.1 Contexte général

Au cours des dernières années, il y a eu une énorme explosion dans la production de données en raison de la multiplication et l'adoption des nouvelles technologies telles que les réseaux sociaux, les plus puissants appareils mobiles, les compteurs intelligents, les capteurs, etc. Cette explosion des données devrait se poursuivre et même s'accélérer. Comme les données sont de plus en plus variées, plus complexes et moins structurées, il est devenu impératif de pouvoir les traiter efficacement.

Au cours de la dernière décennie, le Web a subi une transformation majeure, passant d'un environnement de simple partage de données entre individus à un environnement qui permet également aux organisations d'offrir leurs services et de mener à bien leurs activités. Les entreprises modernes ont déplacé leurs opérations sur Internet en adoptant la technologie de service Web [Alo+04] pour fournir une interface interopérable et programmatique avec leurs systèmes internes. La technologie des services Web incarne le paradigme de l'informatique orientée service (SOC) [Pap03], dans lequel les applications logicielles à l'intérieur et à l'extérieur des murs d'entreprises sont encapsulées comme des services pouvant être exécutés, composés et coordonnés de façon flexible. En simples termes, un service Web est une application logicielle dont l'interface et la liaison peuvent être définies, décrites et découvertes sous forme d'artefacts XML [Cur+02] et accessibles via des protocoles Web omniprésents, des langages et des formats de données standard tels que HTTP, XML, SOAP, WSDL et UDDI.

Alors que les services Web répondent individuellement à des besoins précis, dans certains cas, les utilisateurs doivent composer différents services Web pour réaliser une tâche plus complexe qui ne peut pas être satisfaite par un service Web individuel. La composition des services Web est une solution puissante pour la création de services à valeur ajoutée [Sin01], [MBE03]. Ainsi, la composition de services Web est un aspect crucial de la technologie des services Web, ce qui nous donne l'opportunité de sélectionner de nouveaux services Web et de mieux répondre à nos besoins.

La prolifération des données sous une forme ou une autre est une aubaine pour les utilisateurs et il devient important de pouvoir les utiliser avec aisance. Ces sources de données peuvent présenter différentes formes d'hétérogénéité structurelle, syntaxique et sémantique. Ce qui nécessite un haut niveau d'adaptation et différentes solutions ont été proposées dans la littérature scientifique. Les sources de données sont parfois sujettes à des incertitudes, ce qui signifie que les données qu'elles contiennent peuvent être contradictoires, non pleinement attestées, ou qu'elles peuvent ne pas être fiables en raison d'une défaillance des

mécanismes de collecte. Malheureusement le traitement de l'incertitude dans un contexte d'intégration de données n'a pas été profondément traité, encore moins dans les approches orientées services.

Le Web regorge de données dont une partie peut être incertaine. Cette incertitude n'est malheureusement pas traitée dans les technologies du Web. Elle n'est pas traitée au niveau des modèles de représentation de l'information sur le Web (HTML, JSON, ...). Elle n'est pas aussi traitée au niveau du traitement de requêtes sur le Web. En effet, les moteurs de recherche actuels ne tiennent pas compte de la part d'incertitude de l'information présente sur le Web.

Dans ce contexte, cette thèse se veut comme une ambition de faire évoluer le Web que nous connaissons vers un Web incertain. Un Web incertain qui serait capable de représenter et de traiter l'information incertaine sur le Web et d'exploiter cette incertitude dans les mécanismes de recherche d'information.

Dans notre thèse, nous nous focalisons sur l'impact de l'incertitude des données sur les mécanismes de représentation et de composition de services/ressources Web et sur les techniques de recherche d'information. L'incertitude et l'incomplétude sont deux caractéristiques communes de l'information.

1.2 Challenges

L'accès aux données et la recherche d'information sur le Web représente une passerelle entre l'utilisateur et cet univers de données et d'informations qui ne cesse de croître. Les systèmes de recherche d'information se veulent ainsi comme des messagers dont l'objectif principal est de répondre aux interrogations des utilisateurs. Cependant, comme tout système bâti sur un autre, ils doivent leur efficacité au processus d'indexation qui s'efforce de donner une représentation fidèle, compacte et accessible de l'information recherchée. Pour améliorer ce processus d'accès aux données et de recherche d'informations, l'un des axes auxquels on s'est intéressé en premier est l'incertitude dans le Web. En effet, le Web présente un ensemble de risques et des défis que les chercheurs de ce domaine veulent résoudre. L'un de ces risques réside dans l'incertitude des données et de sources de données. Ainsi, si le Web offre l'opportunité de partager les données de tout types et toute valeur, peut-on garantir l'opportunité de partager des données totalement ou partiellement incertaines ? Autrement dit, peut-on garantir une protection contre l'incertitude des données sans perdre l'utilité de ces données lors de leur partage ?

Les challenges auxquels nous nous sommes intéressés dans le contexte du Web incertain sont formulés de la manière suivante :

— ***Modélisation de l'incertitude au niveau des services Web :*** Les services Web sont conçus pour être invoqués avec des données certaines et retourner des résultats certains. La présence de l'incertitude dans les données d'entrée d'un service Web impactera nécessairement les modèles de représentation des services. Il est alors question d'étudier l'impact de l'incertitude des données sur la modélisation de tels services que nous appellerons services Web incertains. Très peu de travaux se sont intéressés à cette question. Une approche probabiliste a été proposée dans [Mal+15], [MB13]. Dans cette thèse, il est question de considérer la modélisation des services Web incertains en adoptant une approche possibiliste. Cette approche doit être capable de représenter un service Web incertain en précisant les degrés de nécessité et de possibilité des données des services, et de définir la sémantique d'un service Web incertain (possibiliste).

— ***Invocation des services incertains :*** Les services Web sont définis et fournis pour être invoqués avec des données certaines. En présence d'incertitude sur les données d'entrée des services, une invocation classique de services est obsolète. Il est donc question d'adapter l'invocation classique des services pour tenir compte des données d'entrée incertaines représentées dans un modèle possibiliste.

— ***Composition des services Web incertains :*** Le traitement de requêtes dans un contexte orienté services consiste à composer des services qui couvrent la requête. En présence de services Web incertains, la composition de services ne peut être qu'incertaine, c'est-à-dire que la même composition répondra en partie aux besoins de la requête et fournira aussi une ou plusieurs autres sémantiques qui ne correspondent pas à la requête initiale. Il est donc important de revisiter la composition de services Web incertains représentés dans un modèle possibiliste et en définir la sémantique.

— ***Représentation des ressources Web REST incertaines :*** Les ressources Web sont représentées avec des données certaines. A notre connaissance, il n'y a pas de travaux qui s'intéressent à la représentation des ressources Web dans le cas où une partie des informations associées à la ressource est incertaine. Lorsque les ressources Web ont un contenu incertain, la représentation standard (JSON, ...) nécessite d'être modifiée ou adaptée pour la prise en compte de cette incertitude. Ainsi, il est question d'étudier comment modéliser les ressources Web incertaines en adoptant la théorie des données probabilistes et comment représenter les degrés

d'incertitude en prenant en considération les dépendances (c-à-d, corrélations) qui peuvent exister entre les différentes interprétations possibles.

— ***Évaluation des requêtes utilisateur à travers des ressources Web incertaines :*** Les techniques d'évaluation de requêtes sur un ensemble de ressources Web certaines sont insuffisantes dans le cas de ressources incertaines. Ces techniques classiques nécessitent donc une adaptation pour la prise en compte de la part d'incertitude dans la description des ressources. La composition de ressources certaines et incertaines devient elle-même une composition incertaine, et devra retournera non plus un résultat mais différents résultats accompagnés de leurs degré d'incertitude. Ainsi, il est question d'étudier et d'adapter les techniques d'évaluation de requêtes dans le cas de ressources incertaines dont la modélisation est basée sur la théorie des probabilités.

— ***Calcul de l'incertitude niveau documents textuels :*** Pour trouver l'ensemble des documents répondant à une requête utilisateur, tout système de recherche d'information développe une méthodologie pour affirmer si les termes de chaque document correspondent à ceux de la requête. La plupart des systèmes s'appuient sur l'hypothèse que les termes extraits des documents ont été parfaitement reconnus ou certains. Dans certains cas les termes extraits ont une sémantique incertaine, Il est donc important de revisiter le principe d'extraction des termes des documents et de proposer une méthode qui est capable de déterminer la nature des termes contenu dans un document en se basant sur une approche probabiliste.

— ***Indexation syntaxique et sémantique incertaine :*** La recherche d'information se base sur l'indexation des documents et des requêtes pour pouvoir répondre correctement à l'utilisateur. La présence d'incertitude dans les données indexées nécessite la modification et l'adaptation des méthodes d'indexation pour faire face à cette incertitude. Il est donc question d'étudier comment indexer sémantiquement et syntaxiquement les documents textuels dans un modèle probabiliste.

1.3 Contributions

L'objectif de cette thèse est donc d'apporter des réponses aux challenges précédents. Ainsi, nos contributions se déclinent comme suit :

— ***Contribution 1. Modélisation et composition des services Web incertains :*** nous proposons une approche pour la modélisation des services Web incertains. Ce modèle est basé sur la théorie des possibilités et étend la description

standard d'un service Web pour y associer des valeurs de possibilité, nécessité et provenance. Ces valeurs qualifient leurs degrés de certitude. Nous proposons également un modèle d'invocation possibiliste qui permet l'invocation des services Web incertains lorsque les données d'entrées sont incertaines. Ce modèle permet l'exécution de services incertains sans que les fournisseurs de services n'aient à le supporter. Enfin, nous proposons d'utiliser une algèbre de composition de services sensible aux valeurs d'incertitude des sorties de services. Cette algèbre de composition doit être étendue pour permettre de calculer les valeurs de possibilité, nécessité et provenance pour déterminer la qualité des sorties de la composition de services.

— ***Contribution 2. Modélisation et composition de ressources Web incertaines :*** nous proposons un modèle pour la représentation des ressources Web à sémantique incertaine. Ce modèle est basé sur la théorie des probabilités. Pour analyser le modèle incertain que nous proposons, nous avons développé un Parser probabiliste qui permet de valider syntaxiquement la description des ressources certaines et des ressources incertaines. Pour déterminer l'ensemble des interprétations d'une ressource incertaine, nous proposons d'utiliser un ensemble d'opérations algébriques. Cette algèbre nous permet d'évaluer cette incertitude dans un contexte d'une navigation hypertexte classique sur un ensemble de ressources certaines et incertaines. Nous proposons un ensemble d'algorithmes qui permet d'évaluer efficacement les requêtes utilisateurs sur un ensemble de ressources.

— ***Contribution 3. Indexation de documents en présence d'incertitude :*** nous proposons une approche pour représenter l'incertitude présente dans les termes extraits d'un document textuel pour les besoins de la recherche d'information. Nous proposons trois méthodes d'indexation de documents textuels dans un environnement incertain : l'indexation syntaxique incertaine, l'indexation sémantique incertaine et l'indexation hybride incertaine. Ces méthodes doivent prendre en considération les degrés de probabilités des termes contenus dans les documents.

— ***Implémentation et évaluation :*** Nous implémentons nos différentes techniques proposées et évaluons leur efficacité.

1.4 Organisation de la thèse

Le reste de cette thèse est organisé comme suit.

Dans le chapitre 2, nous fournissons le contexte, que nous estimons nécessaire pour

comprendre le contenu de cette thèse. Nous présentons d'abord les concepts clés de la technologie des service Web. Nous nous concentrons ensuite spécifiquement sur les données incertaines et les principaux domaines qui l'ont traité. Enfin nous présentons les notions de base de la recherche d'informations.

Dans le chapitre 3, nous présentons un cadre qui identifie les compositions de services Web en présence de données incertaines. Un nouveau modèle possibiliste de représentation des services Web est proposé pour mieux présenter l'incertitude au niveau des services Web. Nous proposons également une méthode pour invoquer les services Web incertains. Un algorithme efficace est proposé pour la composition des services Web incertains.

Dans le chapitre 4, nous présentons un nouveau concept appelé ressources Web incertaines basé sur la théorie des probabilités pour permettre aux utilisateurs de définir ses ressources Web certaines et incertaines. Nous proposons ensuite un nouveau modèle pour la représentation de ce type de ressources ainsi qu'un analyseur (Parser) que nous avons développé. Enfin, nous développons un algorithme efficace pour évaluer les requêtes utilisateurs sur ces ressources Web incertaines.

Dans le chapitre 5, nous présentons une nouvelle méthode d'indexation de données dans un environnement incertain. Nous proposons un algorithme d'indexation syntaxique des documents textuels en traitant le problème d'incertitude de données. Ensuite, nous proposons un algorithme d'indexation sémantique en présence d'incertitude de données. Enfin, nous proposons un algorithme hybride pour l'indexation de données incertaines.

Dans le chapitre 6, nous fournissons une implémentation de nos différentes techniques et leur application dans certains domaines. Nous mettons en place les services Web incertains et fournissons une étude de performance de notre structure de composition. Nous présentons le parser probabiliste développé pour analyser les modèles de représentations proposés pour les ressources Web incertaines. Nous développons ensuite les algorithmes proposés pour l'évaluation des requêtes au sein des ressources incertaines pour calculer efficacement leurs performances. Enfin, Nous évaluons également notre approche d'indexation en calculant la complexité de chaque algorithme.

Dans le chapitre 7, nous concluons ce travail par un rappel de nos principales contributions et une discussion sur certaines orientations possibles pour de futures recherches.

Chapitre 2

Services Web, gestion d'incertitude et indexation de données : État de l'art

Sommaire

2.1 Introduction

Avec la quantité de données transférées et échangées à travers le Web, les mails envoyés, les fichiers partagés, le besoin d'une telle famille de techniques et d'approches permettant de gérer et de guider ce croisement énorme présente une nécessité importante dans le cadre de recherche des sciences de l'information. De nos jours, en raison de la croissance de la technologie, il existe une production de masse de données, disponible sous forme numérique (de grand volume). Ces données actuellement disponibles ne sont pas unifiées. Elles apparaissent dans différents formats et dans différents types. La diversité des données est basée sur le type d'informations qu'il contient, tels que les documents textuels, les images, les vidéos et les audio ou même sur sa source telles que des données provenant de capteurs (haute variété).

Dans ce chapitre, nous présentons les concepts clés dans les domaines où nos contributions ont lieu. Nous définissons tout d'abord la diversité architecturale des applications Web, depuis les principes HTTP et URI jusqu'aux architectures de services de haut niveau. Dans un deuxième temps, nous présentons le principe d'incertitude de données et d'intégration de données. Dans cette section, nous présentons aussi l'effet d'incertitude dans d'autres domaines tels que le Web, la base de données, les ressources Web,.... Dans la troisième section, nous effectuons une revue rapide des technologies de recherche d'information et des principes d'indexation, qui sont utiles pour améliorer le résultat de la requête d'un utilisateur. La dernière section de ce chapitre sera consacrée à une conclusion .

2.2 Services Web : généralité

Différentes architectures logicielles et technologies ont été proposées ces dernières années pour faciliter le développement et le déploiement de systèmes distribués ; par exemple, middleware pour des objets distribués [Emm00]. Cependant, la généralisation d'Internet et la diversification des dispositifs en réseau ont conduit à la définition d'un nouveau paradigme informatique : l'architecture SOA (Service-Oriented Architecture) qui permet de développer un logiciel comme service livré et consommé à la demande [VN02]. L'utilisation de la technologie de service Web permet d'interconnecter et d'intégrer les applications à divers endroits du World Wide Web de manière faiblement couplée, comme si elles faisaient partie d'un seul grand système de technologie de l'information.

2.2.1 Services Web : définition

Une variété de définitions sur les services Web est données dans la littérature. Cependant, celle proposée par le Consortium Word Wide Web (W3C) est considérée comme une référence : « Un service Web est un système logiciel conçu pour prendre en charge l'interopérabilité d'une machine à l'autre sur un réseau. Il a une interface décrite dans un format pouvant être traité par machine (en particulier WSDL). Les autres systèmes interagissent avec le service Web d'une manière prescrite par sa description à l'aide de messages SOAP, généralement transmis via HTTP avec une sérialisation XML en conjonction avec d'autres normes Web. » Donc, le W3C définit généralement un service Web comme un système logiciel conçu pour prendre en charge l'interaction inter-machines sur un réseau. Les services Web sont définis comme des unités logiques, applicatives, autonomes et modulaires qui fournissent des fonctionnalités métier à d'autres applications via une connexion internet. Fondamentalement, les services Web définissent une façon standardisée d'intégrer des applications ou des fonctionnalités Web en utilisant différents types de technologies, de langages et de formats. Un service Web est considéré comme une application accessible à d'autres applications sur le Web, [VN02]. La définition du W3C est assez précise et suggère également la façon dont les services Web devraient fonctionner. Cette définition met en évidence les principaux avantages technologiques et commerciaux des services Web, c'est à dire :

— **L'interopérabilité** : C'est l'avantage le plus important des services Web. Ils fonctionnent généralement en dehors des réseaux privés, offrant aux développeurs un itinéraire non exclusif vers leurs solutions. Les services Web développés sont donc susceptibles d'avoir une durée de vie plus longue, offrant un meilleur retour sur investissement du service Web développé. Les services Web permettent également aux développeurs d'utiliser leurs langages de programmation préférés.

— ***L'utilisabilité :*** Les services Web permettent d'exposer la logique métier de nombreux systèmes différents sur le Web. Cela donne à vos applications la liberté de choisir les services Web dont elles ont besoin. Au lieu de réinventer la roue pour chaque client, vous devez uniquement inclure une logique métier spécifique à l'application côté client. Cela vous permet de développer des services et/ou du code côté client en utilisant les langues et les outils que vous voulez. De plus, grâce à l'utilisation de méthodes de communication normalisées, les services Web sont virtuellement indépendants des plateformes.

— ***La réutilisation :*** Les services Web ne fournissent pas un modèle de développement d'applications basé sur des composants, mais le plus proche possible du codage à zéro du déploiement de tels services Web. Cela facilite la réutilisation des compo-

sants de service Web dans d'autres services Web. Il facilite également le déploiement de code hérité en tant que service Web.

— ***La déployabilité :*** Les services Web sont déployés sur des technologies d'internet standard. Ce dernier permet de déployer des services Web, même sur le pare-feu, vers des serveurs fonctionnant sur internet à l'autre bout du monde. En outre, grâce à l'utilisation de standards communautaires éprouvés, la sécurité sous-jacente est déjà intégrée.

En fait, il existe deux types de services Web : les services SOAP et les services REST. Ces deux types de services sont caractérisés par différents langages et par différents standards pour mettre en place les principes de la SOA.

2.2.2 Service Web SOAP

2.2.2.1 Définition

Les services Web SOAP représentent la première technologie proposée pour implémenter l'architecture SOA. Cette technologie de la SOA est basée sur une panoplie de standards qui sont tous basés sur le modèle XML. Ce modèle est un format universel compréhensible par n'importe quelle plateforme de programmation. De ce fait, il respecte les principes de la SOA tels que : le couplage faible et l'interopérabilité. Le W3C propose la définition suivante pour les services Web SOAP [BHB04] : "Un service Web est une entité logicielle. Cette entité doit être capable d'être interopérable avec d'autres systèmes distribués. Il a une interface décrite avec une description facilement exploitable, appelée "WSDL". Les autres systèmes réagissent avec le service Web en utilisant des messages SOAP qui sont généralement transmis à travers le protocole HTTP en utilisant une sérialisation XML".

2.2.2.2 Modèle

Le modèle des services Web repose sur une architecture orientée vers les services. La figure 1 montre une représentation graphique du modèle de service Web traditionnel. Celle-ci fait intervenir trois catégories d'acteurs : le fournisseur des services qui offre simplement le service Web (c.-à-d. les entités responsables du service Web). Le fournisseur des services doit décrire le service Web dans un format standard, qui est caractérisé par la traduction en langage XML, et le publier dans un registre de service central. Le registre de service contient des informations supplémentaires sur le fournisseur telles que l'adresse, le contact de la société fournissant le service, des détails techniques concernant celui-ci,... Les clients

servent d'intermédiaires aux utilisateurs de services. Et les annuaires offrent aux fournisseurs des services la capacité de publier leurs services services pour effectuer la promotion de ceux-ci auprès de leurs clients. Les clients ont ainsi la capacité de spécifier leurs besoins en termes de services. Le consommateur de service récupère les informations caractérisant le registre et utilise la description du service qu'il a trouvée pour appeler et lier le service Web. La dynamique entre ces trois acteurs contient donc les opérations de publication, de recherche et de liens ("binding").

Ensemble, ces rôles et ces opérations agissent sur les artefacts des services Web : le module logiciel de service Web et sa description. Les méthodes appropriées sont illustrées dans la figure 2.1 par les mots-clés « publish», «bind» et «find». Afin d'assurer la communication entre les applications fonctionnantes sur différentes plates-formes et écrites dans des langages de programmation différents, des normes sont nécessaires pour chacune de ces opérations. Cette dynamique est normalisée à travers 3 standards : un protocole abstrait de structuration et de description des messages, SOAP [Box+00], une spécification XML qui permet la localisation et la publication des services dans les annuaires, UDDI [Cle+04] et un format de description des services Web : WSDL [7]. Un service WSDL contient un ensemble d'opérations élémentaires, chacune décrite par un flux de messages échangés entre le client et le service [JO02] [Bar13].

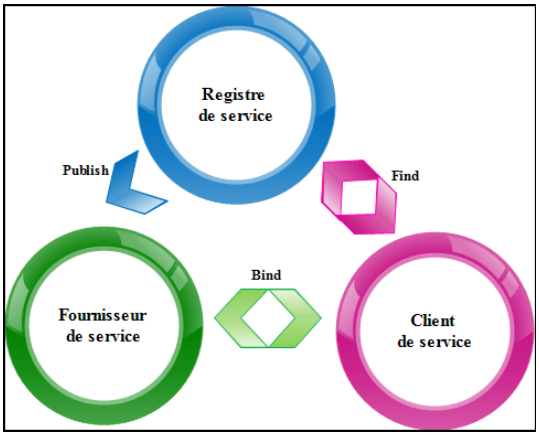


FIGURE 2.1 – Le modèle des services Web.

2.2.2.3 Standard

En se basant sur la définition précédente, les services Web SOAP présentent un ensemble de standards décrits comme suit :

— ***SOAP :*** est l'acronyme de "Simple Object Access Protocol". Il est un protocole caractérisé par une syntaxe basée sur XML. Il assure des échanges standardisés de données. Ces échanges permettent de résoudre les conflits syntaxiques et techniques. SOAP assure l'indépendance de l'invocation des services Web par rapport à la plate-forme d'exécution. En outre, il s'appuie sur d'autres protocoles de la couche d'application, notamment HTTP, et cela pour la négociation et la transmission des messages.

— ***WSDL :*** est l'acronyme de "Web Services Description Language". C'est un langage de description à base d'XML. Cet acronyme est utilisé pour décrire les fonctionnalités présentées par un service Web (l'interface du service). Le fichier WSDL d'un service représente la façon dont le service peut-être appelé, spécifie suivant quels paramètres et caractérise la structure des données qu'il retourne.

— ***UDDI :*** est l'acronyme de "Universal Description, Discovery and Integration". C'est une plateforme indépendante, à base d'XML. Cette plateforme assure le stockage, le regroupement et la diffusion des descriptions de services Web.

— ***WS-BPEL :*** est l'acronyme de "Web Services Business Process Execution Language". C'est une spécification du consortium OASIS. Cet acronyme est un langage d'orchestration de services basé sur la langage XML,tout comme les autres standards. WS-BPEL assure la construction des compositions de services. Ces compositions sont généralement interprétables et exécutables par un moteur d'orchestration (ActiveBpel, OAEapache, etc).

— ***WS-* :*** [Bon04] Il existe plusieurs spécifications associées aux services Web WS-*. Ces spécifications sont à des niveaux de maturité parfois différents, et sont maintenus par diverses organisations de standardisation : WS-Security, WS-Trust , WS-Inspection, etc.

2.2.3 Service REST

Avec le développement du Web2.0/3.0, SOAP n'est pas le seul protocole d'échange des messages dans le monde du Web. Cependant, il existe une forme de SOA encore moins restrictive qu'un transfert d'état de représentation de service Web. C'est le type REST qui est plus léger et assez facile à utiliser. De nombreux fournisseurs de services tels que Google, Yahoo ou Amazon proposent une variété de services REST. L'auteur de [FT02] a décrit ce principe pour la première fois dans sa thèse. REST acronyme de "Representational State Transfer" correspond à un style d'architecture orienté vers les services (ou

"ressource") et non un protocole. C'est une architecture de services Web, élaborée en 2000 par Roy Fiedling qui est un des créateurs du serveur Apache http, du protocole HTTP, et de nombreux autres travaux fondamentaux. REST est la manière de construction d'une application pour les systèmes distribués comme le World Wide Web. REST est une collection de principes qui sont indépendants de la technologie, excepté pour l'exigence afin qu'ils soient basés sur HTTP. Nous parlons de ce concept pour les systèmes qui suivent l'architecture REST. Un service RESTful est donc un autre moyen pour construire les services Web. Ce type de services Web suscite un intérêt de plus en plus croissant dans l'industrie et il a été largement adopté par certaines entreprises en raison de sa simplicité et de sa légèreté (light-weight). Le modèle RESTful a été conçu pour un accès utilisateur simple via un navigateur. Restfull est un système qui se conforme à l'ensemble des principes suivants :

— Tous les composants du système communiquent via des interfaces avec des méthodes clairement définies et un code dynamique. Chaque composant est identifié de manière unique via un lien hypermédia (URL).

— Une architecture client/serveur est suivie

— Toute communication est sans état c'est à dire qu'elles ne dépendent pas d'un contexte conservé sur le serveur. Un appel pour utiliser un service avec des paramètres différents peut retourner un JSON, un fichier, un flux, une image,etc.

— Les requêtes sont simples et formulées sous la forme URI avec des verboses HTTP du type GET/POST/PUT/DELETE,etc. avec des entêtes de requête pour décrire les informations envoyées. Les requêtes sont conservées et mises en cache pour une réutilisation future.

— L'architecture est hiérarchisée et les données peuvent être mises en cache à n'importe quelle couche. Cette architecture se base sur un ensemble de contraintes pour permettre l'optimisation de certains critères propres au cahier des charges du système à concevoir.

L'information de base d'une architecture REST est appelée "ressource". La ressource est définie comme toute information qui peut-être nommée telle qu'un article dans un journal, une photo, un service ou n'importe quel concept. Chaque ressource est un composant distribué. Elle est gérée par une norme et par une interface commune rendant possible la manipulation des ressources. En plus de la ressource, l'architecture REST est basée sur les éléments suivants :

— *Identifiant de la ressource :* Une ressource est caractérisée par un identifiant de ressource. Cet identificateur permet aux composants de l'architecture d'identifier les ressources qu'ils manipulent. Sur le Web, ces identificateurs sont généralement représentés par les URI (Uniform Resource Identifier).

— *Représentation de la ressource :* Les composants de l'architecture traitent les ressources Web en se basant sur le transfert de leurs représentations. Sur le Web, nous trouvons aujourd'hui la plupart des représentations au format HTML, JSON ou XML.

— *Opération sur la ressource :* Les ressources sont manipulées en transférant les représentations des ressources à travers une interface uniforme en utilisant l'identifiant de ressource. Chaque représentation doit implémenter une interface CRUD pour réaliser l'interface uniforme requise par l'architecture REST. Le protocole HTTP/1.1 définit huit méthodes, dont quatre permettent de définir l'interface : GET, PUT, POST et DELETE. Toute représentation d'une ressource doit accepter l'ensemble de ces quatre méthodes, bien que cela ne signifie pas qu'elles doivent obligatoirement être implémentées.

Les services Web REST sont proposés pour simplifier la publication, le développement et l'utilisation de services Web. Ces services doivent suivre les principes de l'architecture REST [PZL08]. Ils répondent aux requêtes des utilisateurs en fournissant des objets Web ("ressources") en utilisant les méthodes HTTP : Get, Head, Post, Put, Delete,etc. En revanche, les services REST présentent un ensemble de contraintes qui assure l'élimination de la nécessité d'avoir une description spécifique pour l'interface de service Web. De plus, la sélection des services Web REST est réalisée simplement en utilisant les adresses URI. En revanche, dans la pratique, cette description des services REST est généralement utile, mais elle n'est pas nécessaire. Par exemple, la description d'un service Web peut-être considérée comme une documentation de celui-ci afin que les utilisateurs de ce service puissent savoir à quoi s'attendre. Plusieurs langages ont été proposés pour décrire les services REST tels que le langage WADL ("Web Application Description Language") [Had06]. L'avantage du langage WADL réside dans le fait qu'il fournit une description à base d'XML, rendant ainsi les services REST compréhensibles et interprétables par la machine. Un service est décrit en utilisant un ensemble de balises de type « Resource ». Chaque élément « Resource » décrit une ressource fournie par un service ainsi que les méthodes appliquées sur celle-ci. Chaque méthode a des entrées (Request) et des sorties (Response).

2.2.4 Composition des services

La composition des services Web implique la combinaison d'un certain nombre de services Web existants pour produire un service plus complexe et plus utile. La composition des services Web est un sujet qui suscite l'intérêt des chercheurs. Il offre une capacité de traitement des problèmes complexes même avec de simples services Web existants tout en coopérant les uns avec les autres. La composition des services Web est une technologie importante de la SOA qui se trouve dans un environnement complexe et distribué [ME08] [TWI08] [TW08] [Yu+08]. L'une des principales cibles de la composition des services Web consiste à réutiliser les services Web existants et à les composer dans un processus. Il existe des programmes qui permettent à l'utilisateur de spécifier manuellement une composition de programmes pour effectuer une tâche, mais cela se situe déjà au-delà de la capacité humaine à gérer manuellement l'ensemble du processus. La composition des services décrit la logique d'exécution des applications basées sur les services Web en définissant leurs flux de contrôle tels que l'exécution conditionnelle, séquentielle, parallèle et exceptionnelle et en prescrivant les règles de gestion cohérente de leurs données métiers non observables. La composition peut-être considérée comme une agrégation de services Web élémentaires ou composites. La composition signifie que les services Web existants sont combinés ensemble sur la base de règles de composition pour répondre à une demande qui ne peut-être réalisée par un seul service. Les règles de composition spécifient l'ordre dans lequel les services sont appelés et les conditions dans lesquelles certains services peuvent être appelés ou non.

La composition du service Web est le processus de sélection, de combinaison et d'exécution du Web services (WS) afin de résoudre les requêtes des utilisateurs qui ne peuvent être résolues par un service individuel seul. Basé sur le degré d'implication des utilisateurs dans le processus de composition et sur le degré d'automatisation, la composition WS peut-être réalisée de trois manières différentes : manuelle (en utilisant certains langages de programmation), semi-automatique (via une série d'interactions avec l'utilisateur) et automatique. La composition de services est le mécanisme qui permet l'intégration des services dans une application. Le résultat de cette phase peut-être un nouveau service, appelé "service composite". Dans le domaine de la composition des services Web, il existe deux manières différentes qui sont : représentées par l'orchestration et par la chorégraphie des services.

— ***Orchestration :*** L'orchestration est définie comme un ensemble de processus exécutés dans un ordre prédéfini afin de répondre à un but. Ce type de composition est basé sur le principe de centraliser l'invocation des services Web composants. Chaque service est caractérisé par des termes d'actions internes. L'orchestration de services Web exige de définir un plan qui décrit l'enchaînement des services Web selon un canevas prédéfini et de les exécuter selon un script d'orchestration [Bro+04].

— ***Chorégraphie :*** La chorégraphie permet de définir la composition comme un moyen ou une méthode pour atteindre un but commun en utilisant un ensemble de services Web. La collaboration entre chaque service Web de la collection (faisant partie de la composition) est décrite par des flots de contrôle. Pour concevoir une chorégraphie, les interactions entre les différents services doivent être décrites. La logique de contrôle est supervisée par chacun des services intervenant dans la composition [Jur06], [Bon+].

2.3 Gestion d'incertitude des données

La gestion de l'incertitude a été traitée en abondance dans les communautés d'intelligence artificielle et de bases de données. Elle a été très peu étudiée dans le domaine des architectures orientées services.

2.3.1 Données incertaines

L'incertitude au niveau des données peut-être générée depuis les sources locales auxquelles les différents types d'imperfection sont impliquées (c.-à-d., l'imprécision, l'incertitude, l'incomplétude, l'incohérence, etc.). Pour remédier à ce problème, un nombre important de travaux ont été menés [NJ02a] [Ba+14b] [KS13] [BP05a][BP07] afin de modéliser et de traiter les données incertaines en se basant généralement sur des méthodes de représentation quantitative (c.-à-d. une base de données probabiliste), ou dans certains cas qualitatifs [PP14] (c'est à dire une logique floue). En outre, les bases de données probabilistes se basent sur une théorie appelée "la sémantique des mondes possibles" [VMB16] qui est un concept important pour interpréter les modèles de données incertaines. Dans la sémantique des mondes possibles, l'incertitude des données est capturée en considérant la base de données comme un ensemble de bases de données possibles. Chaque base de données représente uniquement des données certaines mais la base de données est affectée d'une probabilité d'existence. Les méthodes de traitement de requêtes sur des bases de données probabilistes ont donc été étendues pour la prise en compte de ces différents mondes possibles.

L'information est dite incertaine si l'on ne sait pas si elle est vrai ou faux. L'information élémentaire est une proposition ou une affirmation qu'un événement s'est produit. Il est modélisé par un sous-ensemble de valeurs possibles avec un marqueur d'incertitude. Ce marqueur peut-être numérique ou linguistique. Les marqueurs sont toujours des nombres

(probabilité) ou des modalités symboliques (certaines, possibles, probables). Par exemple : il est certain que Georges arrive en retard ou la probabilité que Georges arrive en retard est de 0,7.

2.3.2 Données inexactes

L'information est réellement imprécise si elle est insuffisante pour comprendre une situation. Les informations inexactes sont liées à l'idée d'informations incomplètes. En effet, supposons que les questions auxquelles il faut répondre sont les suivantes : Quelle est la valeur de la variable X ? Est-ce que la valeur v de X satisfait une certaine propriété ? Considérons, par exemple, l'âge d'une personne et le sous-ensemble S = 1, 2, ..., 100 (en nombre d'années). Le terme "jeune" est trop imprécis quand il s'agit de connaître l'âge de la personne. Une information inexacte prend la forme d'une disjonction de valeurs mutuellement exclusives [Des08]. Par exemple, considérons les informations suivantes : l'âge de Georges est compris entre 20 et 25 ans, ce qui se traduit par l'âge de George (noté X) qui appartient au sous-ensemble 20, 21, 22, 23, 24, 25, c.-à-d, X = 20 ou X = 21 ou X = 22 ou X = 23 ou X = 24 ou X = 25 (car la variable n'a qu'une seule valeur). Dans la logique classique, l'imprécision apparaît comme une disjonction. Affirmer pq signifie que l'une des propositions $pq, \neg pq, p\neg q$ est vraie.

2.3.3 Représentation d'informations imparfaites

[ADP95][AGN09] Pour modéliser et traiter des données incertaines, nous nous basons généralement sur des méthodes de représentation quantitative (c.-à-d. une base de données probabiliste), ou dans certains cas qualitatifs [Ass+10] (c.-à-d. une logique floue [Wal91], [VMB16]). Il existe également d'autres théories de gestion et de manipulation de l'imperfection qui sont apparues au cours des trente dernières années [Mol05] [Wal91]. Nous trouvons ainsi une hiérarchie de représentations basée sur des familles de probabilités convexes, incluant la théorie des probabilités imprécises [CHM94], la théorie des fonctions de croyance de Shafer [Sha76], les ensembles aléatoires [Mol05], les boîtes de probabilité, la théorie des possibilités [HP85]et plus récemment le Cloud [Neu04]. Des liens de passage existent entre l'une et l'autre de ces théories. Une étude détaillée de ces théories est explorée par S. Destercke [Des08] qui offre un aperçu des théories de l'incertitude en se concentrant sur les aspects qui unissent ces cadres. Il cherche ainsi le meilleur cadre qui va s'appliquer à une situation donnée. Nous présenterons les approches les plus utilisées qui sont l'approche possibiliste et l'approche probabiliste.

2.3.3.1 Théorie de possibilité

La théorie des possibilités a été introduite par Lotfi Zadeh [Zad99] pour traiter certaines facettes de l'incertitude dues à un état de connaissance incomplet où la théorie des probabilités est inappropriée. La théorie des possibilités offre un modèle qualitatif d'incertitude où une information est représentée au moyen d'une distribution de possibilité codant un pré-ordre complet sur les situations possibles [DP12]. Une distribution de possibilité est fréquemment attachée à une variable v prenant une seule valeur, éventuellement mal connue, sur un domaine ω.

Distribution de possibilité : Une distribution de possibilité d'une variable V, sur un domaine ω, est une fonction π_V de ω à $[0,1]$, où $\pi_V(X)$ exprime le degré auquel $x(X \in \omega)$ est une valeur possible pour V. La condition de normalisation impose qu'au moins l'une des valeurs du domaine "X_0" soit complètement possible pour toute variable V, c.-à-d. $\pi_V(X_0) = 1$ dans le cas d'une information cohérente. Lorsque le domaine est discret, une distribution de possibilité de n'importe quelle variable V de ω peut s'écrire $\pi_V = \frac{\pi_V(X_1)}{X_1}, \frac{\pi_V(X_2)}{X_2}, ..., \frac{\pi_V(X_m)}{X_m}$ où X_i est une valeur candidate et $\pi_V(X_i)$ est son degré de possibilité par rapport à la variable V.

Possibilité et nécessité : Alors que la théorie des probabilités utilise un seul nombre, la probabilité, pour décrire la probabilité qu'un événement se produise, dans la théorie des possibilités, un événement e se caractérise par deux mesures : sa possibilité et sa nécessité. La mesure de possibilité et la mesure de nécessité sont définies comme suit :

Mesure de possibilité :

La mesure de possibilité est une fonction $\pi : 2\Omega \to [0,1]$ *telle que* :

- $\pi(\phi) = 0$
- $\pi(\Omega) = 1$
- $\pi(e_1 \cup e_2) = max(\pi(e_1), \pi(e_2))$

Mesure de nécessité :

La mesure de nécessité est définie par $N(e) = 1 - \pi(e)$ où e est l'événement opposé à e. De cette formule, il est facile de montrer que :

- $N(e) \leq \pi(e)$
- $\pi(e_1 \cap e_2) = min(N(e_1), N(e_2))$
- $\pi(e) + \pi(\overline{e}) \geq 1$

2.3.3.2 Théorie de probabilité

La théorie mathématique de la probabilité a ses racines dans les tentatives d'analyse de jeux de hasard par GerolamoCardano au XVI^e siècle, et par Pierre de Fermat et Blaise Pascal au $XVII^e$ siècle. Christiaan Huygens publia un livre sur le sujet en 1657. Enfin, XIX^e siècle, Pierre Laplace acheva ce qui est aujourd'hui considéré comme l'interprétation classique [FC82]. La théorie des probabilités fournit un langage de représentation mathématiquement solide et un calcul formel pour des degrés de croyance rationnels, ce qui donne à différents agents la liberté d'avoir des croyances différentes à propos d'une hypothèse donnée. C'est le résultat d'un événement aléatoire qui ne peut-être déterminé avant qu'il ne se produise, mais il peut s'agir de plusieurs résultats possibles [Bor60]. Cela fournit un cadre convaincant pour représenter une connaissance incertaine et imparfaite qui peut provenir de divers agents. Il existe de nombreuses approches distinctes utilisant la probabilité pour le Web sémantique [GO15]. Les objets centraux de la théorie des probabilités sont des variables aléatoires, des processus stochastiques et des événements : des abstractions mathématiques d'événements non déterministes ou des quantités mesurées qui peuvent être des occurrences uniques ou évoluer dans le temps de manière apparemment aléatoire.

2.4 Approche de gestion d'incertitude

2.4.1 Prise en compte de l'incertitude dans l'intégration de données

Les systèmes d'intégration de données s'appuient sur les Mappings afin de spécifier les relations sémantiques entre les schémas des sources locales et le schéma médiateur. Toutefois, créer un tel Mapping entre un schéma local et un schéma médiateur ne peut-être réalisé avec exactitude et certitude. [DHY07] propose un modèle probabiliste qui permet de spécifier non pas un seul Mapping entre les schémas des sources locales et le schéma global, mais plutôt plusieurs Mappings avec différentes probabilités surmontant l'incertitude des correspondances approximatives. Ces mappings incertains sont ensuite exploités dans le mécanisme de réécriture de requêtes multi-sources [Gal+09] [Pan08].

2.4.2 Incertitude dans les services Web

Certains travaux se sont intéressés à l'extension des concepts de base des services (représentation, invocation et composition des services) pour la prise en compte de l'incerti-

tude des données input et output [CHM94] [Amd+14]. Ainsi, une extension WSDL a été proposée pour y inclure l'incertitude des données en termes de probabilité. Les données des services devenant incertaines, une méthode d'invocation probabiliste a aussi été proposée ainsi que la composition de services incertains. Ces extensions adoptent le modèle probabiliste des données incertaines. A notre connaissance, il n'existe à l'heure actuelle aucun travail de recherche qui utilise le modèle possibiliste des données incertaines dans le contexte où des architectures sont orientées vers les services. Dans un contexte d'open data, la sémantique des données n'est pas nécessairement explicitée. Des travaux ont proposé d'expliciter les sémantiques possibles des services d'accès à l'open data [Mal+15], [Mal+16]. Ces travaux ont également considéré le modèle probabiliste de représentations des différentes sémantiques d'interprétations des résultats d'un service.

2.4.3 Incertitude dans les bases de données

Récemment, des données incertaines ont déjà largement existé dans de nombreuses applications pratiques. La gestion de données incertaines n'est pas bien prise en charge par les systèmes de bases de données conventionnels. Un certain nombre de problèmes techniques dans les bases de données traditionnelles ont été réexaminés récemment sous une sémantique incertaine, y compris l'incertitude de modélisation, l'évaluation des requêtes, l'indexation, le traitement des requêtes par rapport aux données incertaines relationnelles et spatiales. Plusieurs approches [Bis79], [Cod79],[Vas80] ont été proposées pour le traitement de valeurs dites nulles. Grant [Gra79] a examiné le cas de valeurs partiellement connues, et Lipski [Jr.79], [Jr.81] a développé un modèle pour traiter des informations incomplètes. Ces informations sont totalement ou partiellement inconnues. Wong [Won82] a proposé une approche statistique pour modéliser l'information incertaine dans les bases de données. Plus récemment, Baldwin [RM88], Buckles et Petry [BP95], [BP82] et Umano [Jr.81] ont introduit divers fuzzifications de l'algèbre relationnelle de base de données pour tenir compte des incertitudes non probabilistes. Le premier travail a porté sur l'extension du modèle relationnel avec un champ de probabilité supplémentaire ou deux champs de possibilité et de nécessité pour gérer les requêtes de type SQL. Les bases de données probabilistes ont été étudiées presque continuellement depuis la fin des années 80. Il s'agit par exemple, des deux articles de Dalvi et Suciu [DS07] et d'Aggarwal et Yu [AY09] sur l'interrogation et l'extraction de bases de données incertaines ou la mise en place d'un tutoriel sur l'interrogation de données incertaines présenté par Pei et al dans [Pei+08].

En termes de bases de données incertaines, les c-tables [GNP92] aident à la fondation de modèles complets pour des bases de données incertaines. Un système pour les bases de données probabilistes est discuté dans [Lak+97], mais la simplification des hypothèses sur les requêtes restreint les stratégies de combinaison. Un modèle pour les données incertaines est

décrit dans [BGP92], où les probabilités peuvent être associées aux valeurs des attributs ; l'article montre comment effectuer des opérations mais suppose la présence de clés déterministes dans chaque relation. Une ligne de travail plus récente [DS07] étudie le problème de répondre efficacement aux requêtes sur des bases de données probabilistes, renvoyant une liste de tuples classés par leurs probabilités retournées dans le résultat. Concernant les bases de données possibilistes, des travaux de recherche antérieurs [BLP06],[DP02] traitent l'interrogation de bases de données relationnelles possibilistes, au moyen de requêtes dites "oui / non" généralisées dont la forme générale est formulée de la manière suivante : "Dans quelle mesure la réponse à Q satisfait la propriété P ? ". Une technique de traitement de telles requêtes est proposée, ce qui évite de calculer les mondes possibles attachés à la base de données possibiliste.

Dans [BP04], les auteurs étudient l'extension du langage de requête au cas où les requêtes peuvent impliquer des termes flous et ils étudient l'impact d'une telle extension sur le modèle. Un modèle de données possibiliste étendu approprié a été décrit, dans lequel des relations imbriquées peuvent apparaître afin de représenter des distributions de possibilité définies sur plusieurs attributs. Dans cet article [BP05b][BP03], [BP07] les auteurs proposent un langage de requête incluant trois opérateurs qui sont bien définis sur des relations possibilistes (compactes) étendues, ce qui est la clé de la traçabilité. L'originalité de l'approche est double : un mécanisme d'imbrication est introduit dans le modèle de données afin de supporter l'expression du résultat de certaines opérations autorisées, et une opération de jointure permet de composer des relations possibilistes sous certaines hypothèses raisonnables. L'intérêt de cette approche [BP02] est d'assurer à la fois la solidité du calcul associé aux opérateurs et sa traçabilité puisque les mondes sous-jacents à l'interprétation d'une base de données possibiliste ne sont pas rendus explicites. La plupart des approches existantes sont limitées pour les requêtes "select", "project" et "join".

2.4.4 Incertitude dans les services Cloud et Web

Le problème de modélisation, de découverte, d'invocation, de sélection et de composition des services Web basé sur la qualité de service et en particulier le traitement de l'incertitude des données a reçu une attention considérable dans la communauté des services Web ces dernières années. Pour surmonter le problème de la modélisation des services Web, les chercheurs ont proposé de nombreuses approches. Ces approches fournissent des services composites fiables et peu coûteux, et quelques approches ou schémas notables ont été proposés. Wang et al. [Wan+11a], [Wan+11b] proposent une approche efficace da la sélection des services basée sur la QoS. Cette approche utilise d'abord un modèle de Cloud pour calculer l'incertitude de qualité de service afin d'élaguer les services redondants tout en extrayant des services fiables. Ensuite, ils utilisent une programmation mixte pour iden-

tifier les services Web les plus appropriés. Cette approche garantit la fiabilité et les besoins en temps réel de la sélection des services Web. Dans cet article [AGN09], ils ont examiné les contributions pertinentes dans les domaines de la découverte des Cloud et de la composition des services, inspirées par la recherche sur l'incertitude croisée. Un intérêt particulier est accordé à trois théories connues : la théorie des probabilités, la théorie des fonctions de croyance et la théorie des possibilités. Le but de cette recherche [FY15] est de donner un aperçu des nouveaux développements de la découverte et de la composition des services dans le Cloud en présence d'une incertitude.

L'objectif principal est de mettre en évidence l'intérêt pour l'analyse des risques par la gestion des incertitudes car la recherche sur les risques et des incertitudes peut aider à prendre les décisions appropriées sur les actions nécessaires en cas de méconnaissance du statut du système des risques. La qualité du service fourni par un service Web est intrinsèquement incertaine. Benouaret et al. [BBH12] abordent le problème de skyline sur QoS incertain. Ils représentent chaque attribut QoS d'un service Web en utilisant une distribution de possibilité et introduisent deux extensions de skyline sur une QoS incertaine appelée "skyline pos-dominant" et "skyline nec-dominant". L'objectif de cette approche [Mal15] est de proposer une méthode de gestion d'incertitude associée à l'interprétation des services DaaS. La gestion de l'incertitude comporte plusieurs aspects, c.-à-d. savoir la représentation de services incertains, l'évaluation des requêtes en présence de services incertains, la corrélation entre services, etc. Cet article propose une approche probabiliste permettant la représentation et la composition de services DaaS incertains. Leurs principales contributions sont résumées comme suit : ils proposent un modèle probabiliste pour les services à la sémantique incertaine. Ce modèle s'associe aux différentes vues sémantiques possibles du même service en se basant sur les probabilités qui qualifient leur degré de certitude. Ils proposent également [Mal+15] une définition formelle de l'interprétation probabiliste d'une composition de services à sémantique incertaines. Ils proposent [MB13] une représentation des corrélations entre les vues sémantiques incertaines des services d'un registre PSR (probabilistic service registry) génériquement utilisant les réseaux bayésiens. Ils proposent [Mal+16] une approche pour la génération automatique de compositions de services avec une sémantique incertaine pour le traitement des requêtes. Cette approche est définie pour le cas des services non corrélés et des services corrélés.

Le traitement d'une requête d'utilisateur Q en utilisant un registre de services probabilistes revient à évaluer la requête sur chacun des registres de service possibles issus de la théorie des mondes possibles. Dans cet article [MZ13], deux approches basées sur le MORL (Multi-Objective Reinforcement Learning) sont proposées pour traiter la composition de services multi-objectifs dans des environnements incertains. À cette fin, les approches proposées adoptent un système d'apprentissage en ligne où l'agent d'apprentissage explore

continuellement l'environnement de service dans une séquence d'épisodes. Par conséquent, les approches proposées peuvent être adaptées aux situations dans lesquelles plusieurs fournisseurs rejoignent ou quittent l'environnement de service pendant l'exécution. Tout d'abord, une nouvelle méthode est proposée pour gérer la composition de services multi-objectifs dans des environnements incertains et dynamiques, en utilisant l'apprentissage par renforcement. Deuxièmement, une approche innovante est conçue pour la composition d'un seul service polyvalent. Cette approche [MZ13] adopte un mécanisme d'auto-organisation qui exploite la structure du problème pour dériver les poids des différents objectifs de QoS. Par conséquent, les utilisateurs n'ont pas besoin d'attribuer explicitement un poids pour chaque objectif de QoS. Troisièmement, une approche innovante est proposée pour la composition de services multi-politiques et multi-objectifs. Cette approche [MZ13] profite de l'opérateur de l'enveloppe (shell) convexe pour trouver simultanément l'ensemble des solutions optimales qui satisfont tous les compromis entre les différents paramètres de QoS.

Dans l'article [RL16], un modèle de composition de service est construit sur la base de POMDP (partially observable Markov décision process) nommé "service composition based on POMDP" et un algorithme de Q-learning est conçu pour résoudre le modèle. SC-POMDP. Cet algorithme peut sélectionner dynamiquement les services de composants avec une qualité de service (QoS) exceptionnelle pendant l'exécution de la composition du service, ce qui permet d'assurer l'adaptabilité de la composition du service. Différent de la plupart des méthodes existantes, le SC-POMDP proposé considère que l'environnement de la composition du service est incertain et la compatibilité entre les services de composants est prise en considération. SC-POMDP est donc plus conforme à la situation réelle. Des expériences de simulation démontrent que la méthode proposée peut résoudre avec succès les problèmes de composition de service des différentes tailles. Spécialement, en cas d'échec du service, SC-POMDP peut toujours sélectionner les services de composants alternatifs optimaux pour assurer l'exécution réussie du service composite.

2.5 Système de recherche d'information

2.5.1 Recherche d'information

Recherche d'Information (RI) [Sau05] est une branche de l'informatique qui s'intéresse à l'acquisition, l'organisation, le stockage, la recherche et la sélection d'information. Du point de vue de l'utilisateur, l'accès à l'information peut-être effectué de manière délibérée à travers un Système de Recherche d'Information (SRI) (on parle alors de recherche $ad-hoc$ ou de collecte active de l'information) ou de manière passive à travers un système de filtrage d'information. Un SRI est un ensemble de programmes informatiques qui a pour

but de sélectionner des informations pertinentes répondant aux des besoins des utilisateurs, exprimés sous forme de requêtes. Le principe du SRI est présenté par la figure suivante :

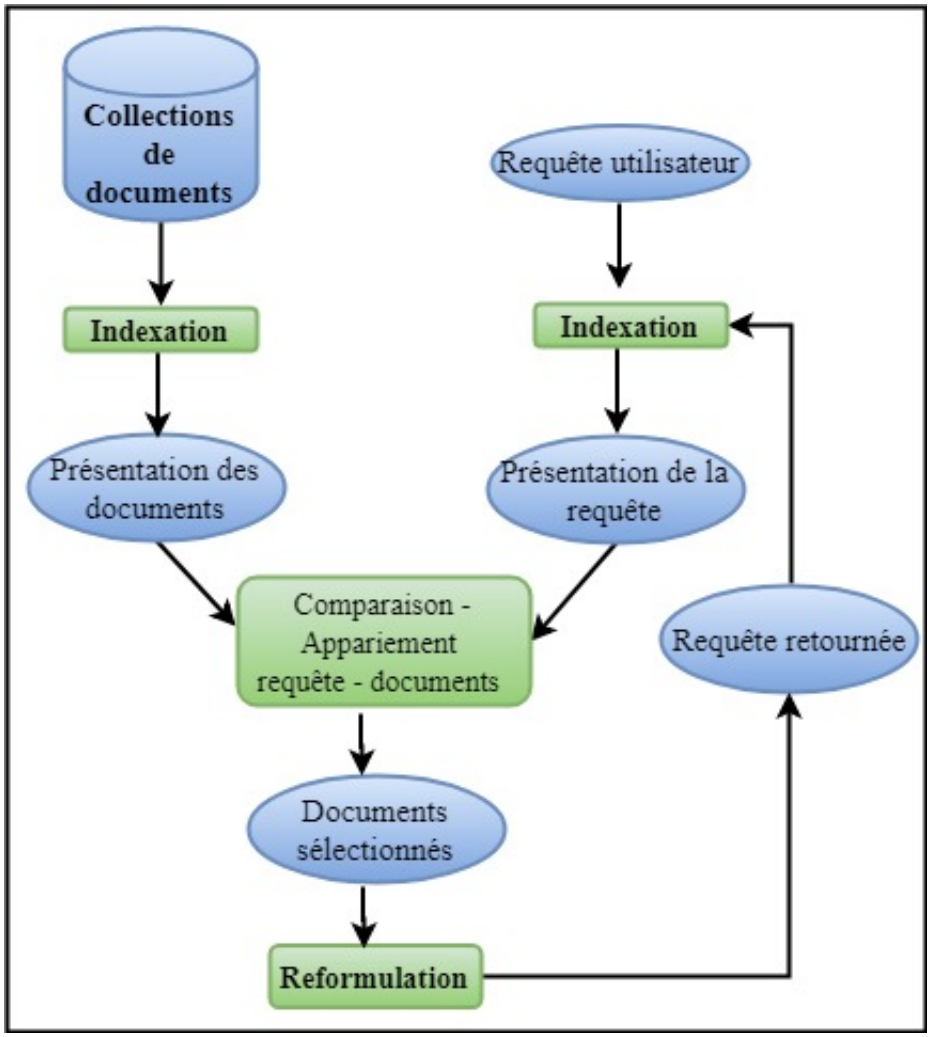


FIGURE 2.2 – Principe du SRI [Ba+14a].

Ce processus est composé de trois fonctions principales :

— L'indexation des requêtes et des documents.

— L'appariement requête-document, qui permet de comparer la requête et le document.

— La reformulation de requêtes s'est intéressée à enrichir la requête initiale par extraction et par réinjection des termes pertinents [Sau05]. La reformulation de la requête se poursuit en deux étapes principales : la première étape consiste à trouver des

termes d'extension dans la requête initiale, et la deuxième étape est de répondre ces termes dans la nouvelle requête. La reformulation de la requête peut-être interactive ou automatique.

2.5.2 Processus d'indexation

2.5.2.1 Définition d'indexation

Afin que la recherche d'information se réalise à des coûts acceptables, il convient d'effectuer une opération fondamentale sur les documents de la collection. Cette opération est nommée "indexation" [MPK14]. Elle consiste à associer à chaque document une liste de mots-clés également appelée "descripteur", susceptible de représenter au mieux le contenu sémantique des documents.

Indexer un document, c'est élire ses termes représentatifs afin de générer une liste des termes d'indexation et ajouter chacun de ces termes et la liste des références de chaque document le contenant à l'index de la collection. Par référence, on entend un identifiant, c.-à-d. un moyen de retrouver de façon non ambigüe des documents ou un document ou une partie du document où le terme apparaît [Sau05][MPK14]. L'indexation des documents est une étape primordiale et très importante car elle détermine la manière dont les connaissances contenues dans les documents fournis sont représentées. Elle a lieu à chaque ajout d'un document dans l'ensemble des documents étudiés.

L'ensemble de tous les termes d'indexation constitue le langage d'indexation. Nous pouvons distinguer deux types de langage : libre ou contrôlé. Un langage d'indexation libre est composé par un ensemble de termes extraits du document analysé. Un langage d'indexation contrôlé est composé d'un ensemble de termes. Ces termes sont préalablement définis et organisés généralement dans ce que l'on appelle un "thésaurus". Lorsqu'un document est analysé, nous gardons seulement les mots clés qui appartiennent à ce thésaurus.

L'indexation doit résoudre deux principaux problèmes, l'évaluation de leur pouvoir de représentation et le choix des termes qui représentent chaque document et chaque approche d'indexation. Il existe plusieurs types d'indexation : l'indexation manuelle, l'indexation automatique et l'indexation semi-automatique [HA].

— Manuelle : C'est un opérateur humain. Chaque document de la collection est analysé

[HA] par un documentaliste ou un spécialiste du domaine afin d'identifier les descripteurs. L'indexation manuelle assure une meilleure précision dans les documents restitués par le SRI en réponse aux requêtes des utilisateurs [HA][Gan+16].

— Automatique : L'indexation est complètement automatisée. Elle se charge d'extraire les termes caractéristiques du document. Elle est réalisée par un programme informatique et elle est composée par un ensemble d'étapes afin de construire de façon automatique l'index. L'intérêt de cette approche réside dans sa capacité à traiter de manière beaucoup plus rapide les textes que dans l'approche précédente. Elle est aussi particulièrement adaptée aux corpus volumineux.

— Semi-automatique [HA] [Sau05] : Egalement appelée "indexation supervisée", c'est une composition des deux approches précédentes, c.-à-d. de l'indexation automatique et de l'indexation manuelle. Dans ce type d'indexation, les indexeurs utilisent un vocabulaire contrôlé qui est formulé sous la forme de base terminologique ou de thésaurus. Le choix final des termes d'indexation construits à partir du vocabulaire fourni est ainsi laissé à l'indexeur humain qui est généralement un spécialiste du domaine.

2.5.2.2 Processus d'indexation

Analyse lexicale

L'analyse lexicale est l'étape qui consiste à transformer un document textuel en un ensemble de termes. Un terme est défini comme un groupe de caractères composant un mot significatif [HBA09]. Pendant cette première phase, la casse, la mise en page et la ponctuation sont supprimées. L'analyse lexicale permet de reconnaître les espaces de séparation des mots, les chiffres, les ponctuations, etc

L'élimination des mots vides

Au cours de cette étape les mots d'usage général et grammatical (les mots vides) sont éliminés. Du fait que ces mots apparaissent de manière uniforme dans les documents, ils ne sont pas utiles pour l'indexation et ils doivent être éliminés [Cha09].

Nous distinguons deux techniques principales pour éliminer les mots vides :

— L'utilisation d'une liste préétablie de mots vides (également appelée "anti-dictionnaire" ou "stop-list").

— L'élimination des mots ayant une fréquence qui dépasse un certain seuil dans la collection.

Lemmatisation (La normalisation)

La lemmatisation consiste à prendre la forme canonique du mot. Dans le document, un mot donné peut avoir différentes formes dans un texte, mais il garde le même sens ou celui-ci reste en tout cas très similaire. Par exemple, citer, citation, citations, etc. La lemmatisation permet de substituer chaque mot par sa racine (lemme). La racine d'un mot correspond soit à la forme infinitive si le mot est un verbe, soit à la forme singulière masculin si le mot est un nom. L'utilisation de la lemmatisation contribue à l'amélioration des performances des SRI (Boubekeur, 2008) puisqu'il réduit la taille de l'index. Il existe plusieurs stratégies de normalisation : la troncature, la table de correspondance, l'utilisation des N-grammes [Cha09] et l'élimination des affixes comme l'algorithme de Porter [HBA09]. Dans certains cas, cette opération présente un inconvénient majeur puisqu'elle supprime la sémantique des termes originaux.

Analyse syntaxique et morphosyntaxique

L'objectif de cette étape est de repérer des mots composés. Cette analyse syntaxique se base sur des patrons ("templates") pour extraire les mots composés. Dans ce processus une catégorie grammaticale est associée à chaque mot (ou groupe de mots) et des patrons sont manuellement construits. Ces patrons sont ensuite projetés dans les documents afin de détecter les séquences qui satisfont ces patrons syntaxiques.

Nous signalons qu'il existe d'autres méthodes dites "méthodes statistiques" qui sont utilisées dans la littérature pour l'extraction des mots composés. Ces méthodes utilisent des mesures statistiques et n'utilisent pas d'analyse linguistique [Cha09].

Pondération

La pondération des termes permet de mesurer l'importance d'un terme contenu dans un document. Cette importance est souvent calculée à partir de considérations et d'interprétations statistiques (ou parfois linguistiques). L'objectif est de trouver les termes qui représentent de la meilleure façon le contenu d'un document.

Si le nous dressons une liste de l'ensemble des mots différents appartenant à un texte quelconque ordonnés par ordre de fréquences décroissantes, nous remarquons que la fréquence d'un mot est inversement par rapport à son rang de classement dans la liste. Cette constatation est énoncée formellement par la loi de Zipf [Bou09] :

Rang * fréquence = constante.

La loi de distributions des termes suit alors la courbe présentée sur la figure 3. Zipf explique la courbe hyperbolique de la distribution des termes par ce qu'il appelle "le principe du moindre effort". Il considère ainsi qu'il est plus facile pour un auteur d'un document de répéter certains termes que d'en utiliser de nouveaux.

La relation entre la fréquence et le rang des termes permet de sélectionner les termes représentatifs d'un document. Nous éliminons de ce fait respectivement les termes de fréquences très élevées car ils ne sont pas représentatifs du document (nous pouvons par exemple citer les mots outils), et les termes de fréquences très faibles (ce qui permet d'éliminer les fautes de frappes et les néologismes). Ce processus est illustré sur la figure 3. En utilisant cette approche, le nombre de termes faisant partie de l'index d'une collection peut-être réduit considérablement.

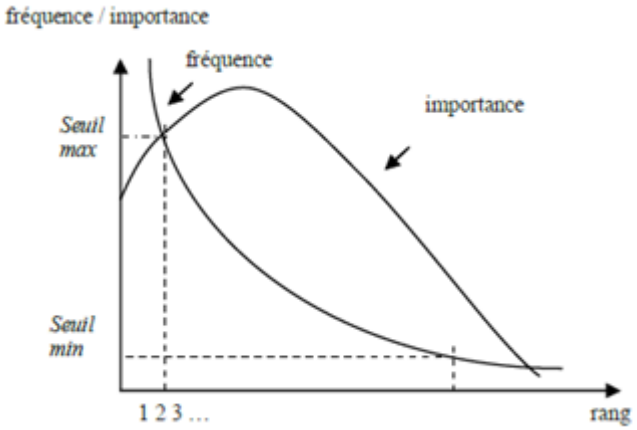


FIGURE 2.3 – Importance d'un terme en fonction de sa fréquence [Car07].

A partir de ces constatations, des techniques de pondération ont vu le jour. La plupart de ces techniques sont basées sur les facteurs tf et idf, qui permettent de combiner les pondérations locales et globales d'un terme :

— tf (Term Frequency) : Cette mesure représente la fréquence du terme dans le document (pondération locale). Elle consiste simplement à calculer le nombre de fois

que chaque mot est apparu dans chaque document (pondération locale). Elle peut-être utilisée de deux façons différentes : telle quelle ou selon plusieurs déclinaisons (présence/absence, log(tf), . . .).

— idf (Inverse of Document Frequency) : Cette mesure représente une pondération globale. Ce facteur mesure l'importance d'un terme dans une collection. Un terme qui apparaît souvent dans la base documentaire ne doit pas avoir le même impact qu'un terme moins fréquent. Il est généralement exprimé comme suit : log (N/df), où df est le nombre de documents contenant le terme et N représente le nombre total de documents contenus dans la base documentaire.

La formule tf * idf calcule le poids et la valeur de la pertinence d'un document par rapport à un terme. Cette mesure donne une bonne approximation de l'importance du terme dans le document, particulièrement dans les corpus de documents de taille homogène. Cependant, elle ne tient pas compte d'un aspect important du document : sa longueur. En général, les documents les plus longs ont tendance à utiliser les mêmes termes de façon répétée ou à utiliser plus de termes pour décrire un sujet. Par conséquent, les fréquences des termes dans les documents seront plus élevées, et les similarités à la requête seront également plus grandes.

Pour remédier à cet inconvénient, Robertson et Singhal et al. proposent d'intégrer la taille des documents à la formule de pondération : nous parlons de facteur de normalisation.

2.5.2.3 Méthodes d'indexation

Nous décrivons dans cette section un ensemble de techniques d'indexation. Nous pouvons les classer selon le type de données qu'elles traitent [Bou09]. Nous pouvons citer trois classes :

— Les méthodes d'indexation des données Web : Ce sont des techniques d'indexation automatiques. De nombreuses méthodes (linguistiques, statistiques, par assignation, etc.) ont été proposées pour concevoir ou améliorer dans certains cas les systèmes ou les logiciels d'indexation automatique [Zar04]. Il existe plusieurs méthodes d'indexation telles que la méthode linguistique, la méthode statique, la méthode mixte, la méthode par assignation, l'index par méthode sémantiques, l'indexation orientée "document" et l'indexation orientée "requêtes".

— Les méthodes d'indexation des bases de données et des entrepôts de données : Dans

les systèmes de gestion de bases de données (SGBD), l'accès aux données est d'autant plus lent que la taille de la base de données est grande. Un parcours séquentiel des données est une opération pénalisante et lente pour l'exécution des requêtes. La création d'un index permet d'améliorer considérablement le temps d'accès aux données en créant des chemins d'accès directs [Aou05]. Nous pouvons citer parmi eux les techniques suivantes telles que l'index en B-arbre, l'index de Hachage, l'index Bitmap (Binaire), l'index de jointure, l'index de jointure en étoile, l'index bitmap de jointure, l'index de jointure de dimensions, l'index de projection,... Il existe deux principaux types d'index :

— Les index primaires (clustered), ou index groupant : Les adresses contenues dans ce type d'index sont triées suivant le positionnement physique sur disque des n-uplets contenus dans la table indexée. Ce type d'index présente un inconvénient majeur qui est le coût de maintenance très élevé car il faut effectuer l'ordre du tri.

— Les index secondaires (non-clustered), ou index non-groupant : les index secondaires sont moins efficaces que les index primaires mais ils sont aussi beaucoup moins coûteux au niveau de la maintenance.

— Les méthodes d'indexation des données XML.

Nous pouvons les classer en deux types, présentés par la figure suivante :

2.6 Conclusion

Dans ce chapitre, nous avons présenté un état de l'art portant sur trois notions : les services Web, l'incertitude et l'indexation.

Dans la première partie de ce chapitre, après avoir introduit la notion de services Web, nous avons fait un tour d'horizon des service Web qui ont été proposés et nous avons décrit les concepts concernant la technologie du service Web. La deuxième partie de ce chapitre a été consacrée à l'étude des différentes définitions proposées dans la littérature afin de décrire les données incertaines. Nous avons également intéressé à l'étude des différentes approches proposées pour la gestion d'incertitude dans le Web. Cette étude nous a permis de constater que les chercheurs ne sont pas intéressés à l'incertitude de données traitées par les services et ressources Web et par les données utilisées pour indexer des documents textuels. Enfin, comme nous proposons une approche d'indexation de données incertaines en utilisant une approche probabiliste, il était également nécessaire d'étudier les notions

de base de recherche d'informations et d'indexation présentées dans la littérature.

Chapitre 3

Approche de Modélisation, de Composition et d'Invocation de Services Web Possibilistes.

Sommaire

3.1 Introduction

Les entreprises modernes s'orientent de plus en plus vers une architecture orientée services pour le partage des données en plaçant leurs sources de données derrière des services, [HOB17] offrant ainsi une manière interopérable d'interagir avec leurs données. Cette classe de services est connue sous le nom de services de données.

La composition des services Web est une solution puissante pour répondre aux requêtes complexes des utilisateurs en combinant des services Web primitifs. Comme les données manipulées par les services deviennent de plus en plus diverses, complexes et moins structurées, plusieurs travaux [VK11], aabordent différents aspects du cycle de vie du service Web, notamment la création, l'invocation, la découverte, la sélection et la composition [Yu+08] et cela dans le but de faire face au problème du volume de données, d'hétérogénéité des données ou de vitesse d'évolution des données.

Cependant, l'incertitude des données reste un aspect important et pas assez traité dans les architectures orientées services, étant donné que les données renvoyées par un service Web ne sont pas toujours certaines pour de diverses raisons, par exemple, le service accède à différentes sources de données, contraintes de confidentialité, etc. [Ben+16]. Les services Web qui renvoient des données incertaines sont considérés comme des services Web incertains [Amd+14]. Des recherches approfondies ont conduit à une compréhension générale des questions d'incertitude dans divers domaines allant de la biologie computationnelle à la prise de décision économique. Cependant, une étude de l'incertitude sur les systèmes informatiques à grande échelle et les systèmes de service Web est limitée. La majeure partie du travail examine les incertitudes du côté utilisateur tel que la qualité, les intentions et les actions des fournisseurs de services Web. Au mieux de nos connaissances, seules quelques études considèrent l'incertitude des services [Amd+14], [LDB16], [Mal+16]. Ces approches sont basées sur un modèle probabiliste. Cependant, dans de nombreux scénarios de la vie réelle, un modèle possibiliste offre une modélisation plus appropriée [VK11].

3.1.1 Exemple de motivation

Le tableau 3.1 ci-dessous donne des exemples de services Web incertains du domaine juridique et police. Le service S_1 renvoie les voitures vues par un témoin donné, alors que les services S_2 et S_3 renvoient les propriétaires d'une voiture donnée. Les symboles "$" et " ?" dénotent, respectivement, les entrées et les sorties d'un service Web.

Un service Web peut être incertain car il intègre différentes sources de données en

adoptant différentes conventions pour nommer le même ensemble d'objets. Par exemple, S_2 fournit des informations complètes sur des propriétaires de voitures en intégrant deux sources de données bien déterminées. Par contre, S_3 joint les propriétaires en intégrant une autre source de données. Cependant, le nom du même propriétaire peut être stocké différemment dans les deux sources. Les correspondances imprécises (généralement quantifiées en nombre entre 0 et 1) entre les propriétaires des deux sources peuvent être interprétées comme le degré d'incertitude que les deux propriétaires correspondent ; c'est-à-dire que S_2 retournera pour chaque tuple obtenu à partir de la correspondance, la probabilité que le tuple existe. Ce type d'incertitude est très courant dans le domaine de l'intégration de données Web [Agr+10] , [SSI10]. Il existe une littérature abondante sur le couplage d'enregistrements (également connu sous le nom de déduplication, ou fusion-purge) [MW11a], qui offre une excellente collection de techniques pour calculer des correspondances.

Services	Sémantiques	Type de service
$S_1(\$witness, ?cars)$	Retourner les voitures vues par un témoin donné	Incertain
$S_2(\$car, ?owners)$	Retourner les propriétaires d'une voiture donnée	Incertain
$S_3(\$car, ?owners)$	Retourner les propriétaires d'une voiture donnée	Incertain

TABLE 3.1 – Exemples de services Web incertains

3.1.2 Challenges

Supposons qu'un utilisateur X veut connaître le propriétaire de la voiture vue par le témoin Y. La requête de X .c-à-d. Q_1 : « Retourner les propriétaires de la voiture vue par la personne Y » peut être résolue en utilisant les services de notre exemple de motivation. Cependant, il est nécessaire de composer le service S_1 avec le service S_2, S_3 ou les deux ensembles. En d'autres termes, il faut invoquer le premier service S_1 pour récupérer les voitures vues par X. Ensuite, les voitures obtenues seront utilisées pour invoquer le service S_2, S_3 ou les deux pour obtenir le résultat final. Cependant, composer des services à sémantique incertaine pour répondre à une requête comme Q_1 soulève les challenges suivants :

— **Comment présenter et modéliser la notion de service Web incertain :** L'incertitude associée aux sorties renvoyées par un service Web doit être explicitement modélisée, car elle est nécessaire à l'interprétation de ces sorties par les consommateurs de services. La modélisation proposée doit être compatible avec les normes de service Web actuelles (par exemple, WSDL, SOAP, etc.), car elles sont largement

adoptées par la communauté de développement de services Web.

— **Comment invoquer un service Web incertain :** Nous devons définir un opérateur d'invocation générique qui pourra invoquer un service Web incertain et récupérer le degré de confiance de sa sortie. Un service incertain peut être invoqué avec des données d'entrée à la fois certaines et incertaines ; dans ce dernier cas, l'opérateur d'invocation doit prendre en compte l'incertitude de l'entrée pour calculer le degré de confiance de la sortie.

— **Comment composer des services en traitant l'incertitude des données :** Le modèle de composition de service Web doit être étendu pour permettre de calculer les degrés d'incertitude des sorties de la composition pour aider les utilisateurs à les comprendre et à les interpréter correctement. Un service incertain devrait être compatible avec des services classiques et incertains ; c'est-à-dire qu'une composition qui n'est pas consciente de l'incertitude devrait être capable d'utiliser des services incertains sans affecter son exécution normale.

3.1.3 Contributions

Dans ce qui suit de ce chapitre, nous allons nous baser sur l'approche possibiliste parce qu'elle représente à la fois une approche quantitative et qualitative. Elle est également plus précise que l'approche probabiliste. La théorie des possibilités nous permet de distinguer l'incertitude de l'imprécision, ce qui n'est pas le cas avec l'approche probabiliste. Et finalement, elle nous permet de déterminer le degré de conflits avec d'autres valeurs "possible". Dans ce chapitre, nous proposons une approche possibiliste globale d'invocation et de composition des services Web incertains. Cette approche s'articule sur les principales contributions suivantes :

— **Modèle possibiliste pour les services Web incertains :** Nous proposons une approche possibiliste pour modéliser l'incertitude des sorties renvoyées par un service de données incertain. Nous étendons la norme WSDL de description de service pour accommoder les degrés de confiance de sorties.

— **Modèle d'invocation de services Web possibiliste :** Nous proposons un modèle d'invocation qui permet l'invocation de services Web avec une entrée certaine et incertaine. Dans le premier cas, le processus d'invocation récupère les degrés d'incertitude des sorties du service. Dans la seconde, le processus d'invocation calcule les degrés d'incertitude des résultats renvoyés en fonction des degrés de confiance

renvoyées par le service et de degrés de confiance de l'entrée.

— **Modèle de composition de services Web possibiliste :** Nous définissons la sémantique d'une composition de service incertaine basée sur la théorie du monde possible. Pour ce faire, nous proposons une algèbre de composition sensible aux valeurs de possibilité, nécessité et provenance pour calculer les degrés d'incertitude des sorties de composition.

Le reste de ce chapitre est organisé comme suit : La section 3.2 est réservée à une présentation des notions de base de notre chapitre telles que les propriétés des bases de données possibilistes et de provenance. Ensuite, nous présenterons dans la section 3.3 notre modèle possibiliste d'invocation et de composition des services Web incertains. La section 3.4 sera consacrée à la présentation du modèle possibiliste pour la représentation des services Web avec provenance. Ce chapitre sera clôturé par une conclusion.

3.2 Background

3.2.1 Bases de données possibilistes

Dans cette section, nous allons, d'abord, fournir les notions sur les bases de données possibilistes. Dans ce type de base de données, les données incertaines sont modélisées comme une distribution de possibilités sur les valeurs d'attributs mal connues. Chaque valeur x d'un attribut X est affectée d'un nombre $\pi X(x)$ à partir de l'intervalle unitaire qui exprime le degré de possibilité de son occurrence. La distribution de possibilité prend la forme : $\pi X(x_1)/x_1, \pi X(x_2)/x_2, ..., \pi X(x_n)/x_n$, où x_i est un élément du domaine de X. Au moins une valeur doit être complètement possible, c'est-à-dire que son degré de possibilité est égal à 1. Cette exigence est appelée condition de normalisation.

Les principaux concepts de la théorie des possibilités sont les mesures de la possibilité et de la nécessité, notées $\pi et N$, respectivement. Une mesure de possibilité est une fonction $P(U) \rightarrow [0, 1]$, où $P(U)$ désigne l'ensemble de puissances d'un univers de discours U, tel que $\pi(\phi) = 0, \pi(U) = 1 \quad et \quad \forall A, B \in P(U), \quad \pi(A \cup B) = max(\pi(A), \pi(B))$. Sur la base de la distribution des possibilités, on peut dériver un degré de possibilité $\pi(A)$ pour tout événement $A \in P(U) : \pi(A) = sup_{x \in A} \pi X(x)$. Cependant, ayant P(A) nous ne pouvons pas conclure dans quelle mesure A est certain. Afin d'améliorer la description de A, la mesure de nécessité a été introduite. Sa valeur est égale à l'impossibilité de l'événement opposé $\overline{A} : N(A) = 1 - \pi(\overline{A})$. Ainsi, le degré de certitude de A peut être égal à 0 même si A est complètement possible ($\pi(A) = 1$). La mesure de nécessité ne peut pas être supérieure à la mesure de possibilité. Ainsi, $N(A) \leq \pi(A)$. De plus, $\pi(A) < 1 \Rightarrow N(A) = 0$.

Chaque attribut possible est caractérisé par un degré de nécessité pour tous les mondes possibles dans ce tuple. Au début, la nécessité de tous les tuples est 1. Par exemple, supposons une relation "Saw" définie par les attributs "Witness" et "Car" (voir tableau 3.2). Alice a vu une "Audi A4" avec possibilité 1 et une "Audi A5" avec possibilité 0,8, tandis qu'Amy a vu une "Audi A4" avec possibilité 1 et une "Audi A6" avec possibilité 0,5.

Witness	Car	Nécessité
Alice	1/Audi A4, 0.8/Audi A5	1
Amy	1/Audi A4, 0.4/Audi A6	1

TABLE 3.2 – La relation Saw

Le modèle possibiliste est un modèle à la fois quantitatif et qualitatif. Il adopte la sémantique du monde possible où une relation incertaine est représentée comme un ensemble de mondes possibles. Chaque instance représente une base de données caractérisée par un degré de possibilité qui représente le minimum du degré de possibilité des attributs sélectionnés. La relation "Saw" comprend les mondes possibles présentés dans le tableau 3.3.

PW_1		
Witness	**Car**	**N**
Alice	1/Audi A4	0.2
Amy	1/Audi A4	0.6
$\Pi(PW_1) = 1,\ N(PW_1) = 0.2$		

PW_2		
Witness	**Car**	**N**
Alice	1/Audi A4	0.2
Amy	0.4/Audi A6	0
$\Pi(PW_2) = 0.4,\ N(PW_2) = 0$		

PW_3		
Witness	**Car**	**N**
Alice	0.8/Audi A5	0
Amy	1/Audi A4	0.6
$\Pi(PW_3) = 0.8,\ N(PW_3) = 0$		

PW_4		
Witness	**Car**	**N**
Alice	0.8/Audi A5	0
Amy	0.4/Audi A6	0
$\Pi(PW_4) = 0.4,\ N(PW_4) = 0$		

TABLE 3.3 – Les mondes possibles de la relation "Saw"

Le tableau 3.3 montre également les valeurs de possibilité et de nécessité de chaque monde possible PW_i. La valeur de possibilité d'un monde possible PW_i est égale au minimum des valeurs de possibilités d'attributs. Sa nécessité est égale à 1 moins la possibilité maximale des possibilités de valeurs d'attributs qui n'appartiennent pas à ce monde. Par exemple, $\pi(PW_1) = min(1, 1) = 1, \quad et \quad N(PW_1) = 1 - max(0, 8, 0, 4) = 0, 2.$

3.2.2 Provenance

La provenance a récemment été reconnue comme un concept important pour annoter, intégrer des données et pour mettre en place les bases de données probabilistes [Mor10]. En informatique, ce concept - également appelée lignage - décrit la source et la dérivation des données. Il comprend une description de la façon dont l'objet a été dérivé [MS09]. Elle est utilisée pour se référer à l'historique d'un document et pour augmenter la valeur d'un objet. L'origine d'un élément de données est la lignée de cet élément de données, décrivant ce que c'est et comment il est apparu. La source des données peut exister sous différentes formes. Par exemple, la source peut se référer à des bases de données entières, des tuples dans la base de données ou des fichiers. Dans ce cadre, Buneman et al [MS09] définissent la provenance des données dans le contexte des systèmes de bases de données comme la description des origines des données et le processus par lequel elles sont arrivées à la base de données. L'importance de l'enregistrement de la provenance, ou lignée, sur les informations significatives est désormais largement reconnue, et de nombreuses recherches ont été réalisées sur la gestion des provenances dans les bases de données scientifiques [MS09][BKT01], [CCT09].

3.3 Interrogation de services Web incertains

Comme expliqué précédemment dans l'exemple de motivation, il n'est pas toujours possible de définir la sémantique d'un service Web avec exactitude et certitude. Cela arrive souvent lorsque le fournisseur de services ne fournit pas suffisamment d'informations (exemple : informations textuelles, métadonnées, etc.) sur les sources de données accédées par ses services, ce qui mène à différentes interprétations possibles pour le même service. Ces interprétations d'un service Web doivent être exploitées pour améliorer la description du service (e.g., le fichier WSDL pour les service SOAP, fichier WADL pour les service REST, etc.), afin que les consommateurs de ce dernier puissent l'interpréter et l'utiliser correctement.

Dans cette section, nous représenterons une nouvelle approche de modélisation de services Web incertains. Notre approche adopte un modèle possibiliste pour décrire l'incertitude associée au service. Dans notre proposition, nous ne changerons pas le principe de fonctionnement des services Web, c'est-à-dire que les services ont une sémantique certaine, mais ils peuvent traiter des données incertaines, retourner des résultats incertains accompagnés de degrés qui représentent la possibilité d'avoir ce résultat.

3.3.1 Modèle possibiliste pour les services Web incertains

La gestion de l'incertitude des données a fait l'objet d'une attention considérable de la part de la communauté de la recherche sur les bases de données au cours de la dernière décennie. Deux principaux défis ont été relevés : la modélisation des incertitudes et le traitement des requêtes sur des données incertaines. Différentes approches ont été proposées pour modéliser l'incertitude des données [Sad91],[BP10], [AKG91]. Parmi ces modèles, nous évoquons notamment les modèles probabilistes et les modèles possibilistes qui sont les plus adoptés en raison de leur simplicité. Dans le premier type de modèles, l'incertitude des données est modélisée comme une distribution de probabilité sur les valeurs possibles des tuple/attributs [MW11b], [BP10] ; c'est-à-dire que l'on attribue à chaque valeur de tuple/attribut possible un degré de confiance, en quantifiant sa probabilité. Dans le deuxième modèle, on attribue à chaque valeur de tuple/attribut possible un degré (normalisé) représentant la mesure dans laquelle cette valeur est possible. Le modèle possibiliste est un modèle numérique qui adopte la sémantique des mondes possibles, où une relation incertaine est considérée comme un ensemble d'instances possibles (les mondes). Chaque instance représente le monde réel avec un degré de confiance. La structure de ces mondes pourrait être régie par des règles de génération sous-jacentes (par exemple exclusion mutuelle des tuples qui représentent la même entité du monde réel).

Dans cette section, nous présentons notre modèle pour les services de données incertains. Notre modèle adopte une approche possibiliste permettant la description de l'incertitude associée avec des services de données. Dans cette thèse nous considérons qu'un service incertain a une certaine sémantique et un certain comportement, mais les services peuvent retourner des résultats incertains. Un service incertain peut avoir une ou plusieurs opérations. Chaque opération peut renvoyer une ou plusieurs sorties. Pour cette raison, nous devons traiter l'incertitude au sein de toutes ces opérations.

Définition : (Service Web Incertain) Un service Web incertain est défini par la représentation suivante :

$$S^P(I, O^{\pi N}), \text{ où}$$

— I et $O^{\pi N}$ sont respectivement les vecteurs de paramètres d'entrée et de sortie du service S^P.
— π : degré de possibilité associé au vecteur de sortie O.
— N : degré de nécessité associé au vecteur de sortie O.

Dans cette définition, l'entrée I représente une valeur certaine. Dans l'approche possibiliste, il peut y avoir des services incertains caractérisés par une valeur de possibilité égale à 1 et une valeur de nécessité égale à 1.

Exemple Le service S_1^P $(city, ?Cars)$ renvoie l'ensemble des voitures vues par un témoin spécifique. Ce service est appelé par la valeur de witness = "Amy".

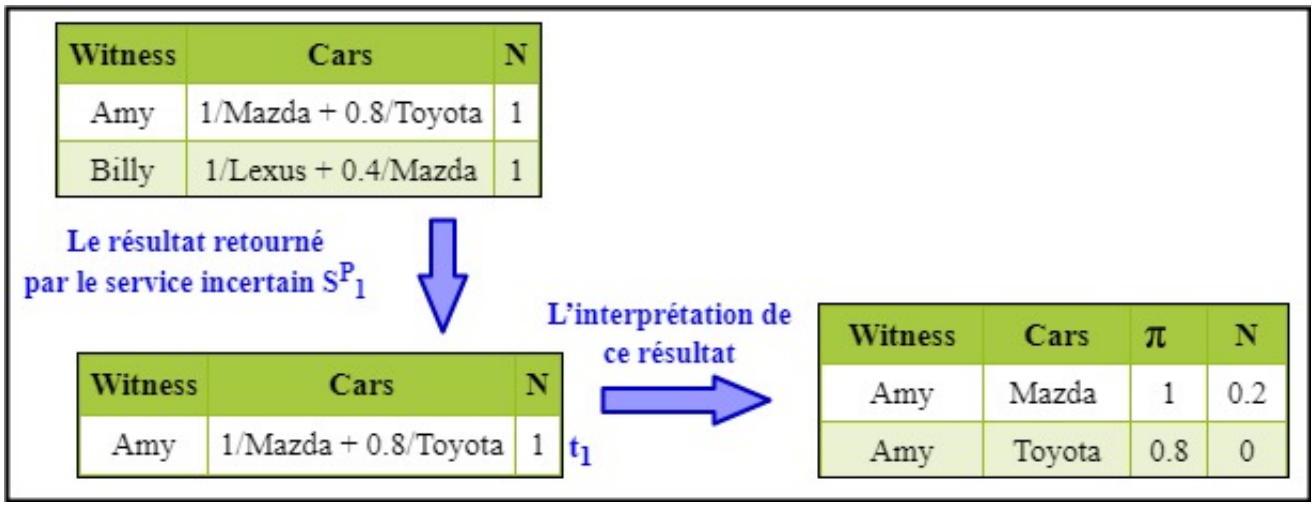


FIGURE 3.1 – Exemple de résultats renvoyés par S_1^P et les mondes possibles correspondants.

Le tuple t_1 est caractérisé par une valeur de nécessité N qui représente le degré de certitude d'avoir ce tuple. L'attribut "Cars" est caractérisé par une valeur de possibilité qui représente le degré de possibilité d'avoir cette valeur. Ces valeurs ne font pas partie de la sortie retournée par le service. Ils représentent simplement des métadonnées. La sémantique des services Web incertains peut-être expliquée en utilisant la théorie des mondes possibles. La sortie possibiliste renvoyée par une invocation peut-être interprétée comme un ensemble de mondes possibles $(PW_1, PW_2, ...PW_n)$ et chaque monde possible PW_i contient des tuples caractérisés par π_{pw_i}. Nous supposons que les tuples retournés par les services sont des tuples indépendants. Ensuite, nous avons obtenu deux mondes possibles correspondant à la combinaison différente d'attributs incertains. Par exemple, la nécessité de PW_1 est calculée par la méthode suivante $N_{PW_1} = 1 - \pi_{Attributssupprimés}$.

3.3.2 Invocation de services Web possibilistes

Dans cette section, nous nous intéressons à l'incertitude présentée au niveau du processus d'invocation des services Web. Notre but est de représenter la sémantique d'invocation possibiliste et d'expliquer comment l'incertitude des données peut être traitée.

Notations Soit S^P un service Web qui encapsule une source de données incertaine.

Ce service est caractérisé par deux paramètres :

— Les paramètres d'entrée qui peuvent être soit des données certaines I soit des données incertaines $I^P = \langle I, \pi, N \rangle$.

— Paramètres de sortie : qui peuvent être soit des données certaines O, soit incertaines $O^P = \langle I, \pi, N \rangle$.

Où : π : représente la possibilité d'avoir cette valeur, N : représente la nécessité, c.-à-d. le degré de confiance que nous pouvons donner à cette valeur. Comme expliqué dans la section précédente, le principe du raisonnement incertain des services est basé sur la théorie des mondes possibles. Ainsi, pour expliquer la sémantique de l'invocation possibiliste, nous nous baserons sur le même principe. La figure 3.2 représente le principe d'invocation possibiliste. Soit S_1^P le service invoqué par le tuple d'entrée <Amy, 1, 1>. Les résultats renvoyés par ce service sont représentés par la figure 3.a. Si nous voulons invoquer le service S2 en utilisant ce résultat, nous devons d'abord générer tous les mondes possibles, et ensuite nous devons déterminer la valeur de possibilité et de nécessité pour chaque tuple. Pour Invoquer le service S_2^P (nombre d'invocations = nombre de mondes possibles), il faut tout d'abord calculer les valeurs d'incertitude, en fonction des valeurs d'entrée, puis effectuer une agrégation afin de renvoyer le résultat final caractérisé par les degrés de possibilité et de nécessité. Ce principe est illustré par la figure 2.b.

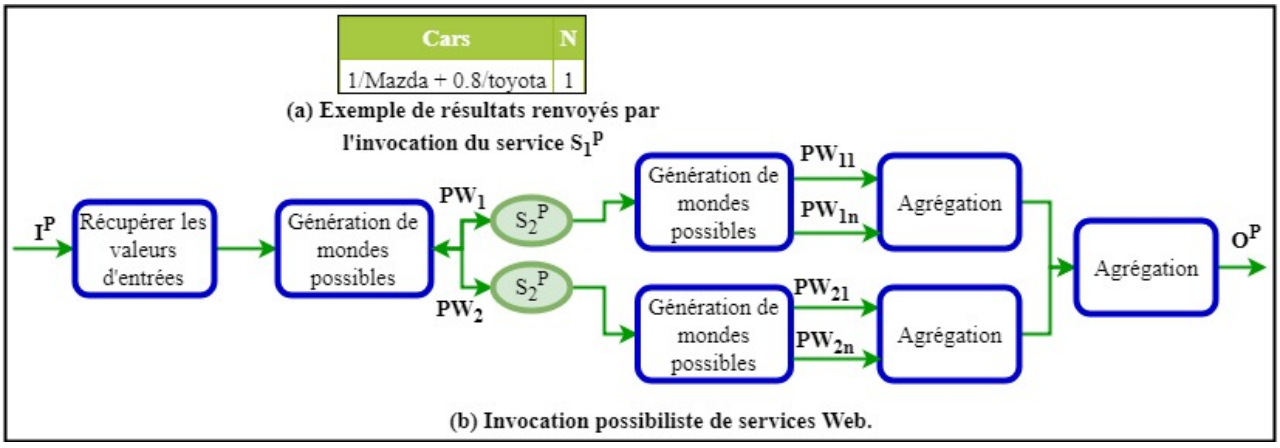


FIGURE 3.2 – Sémantique de l'invocation possibiliste.

Ainsi, cette invocation peut-être interprétée comme suit : I^P peut-être interprété comme un ensemble de mondes possibles PW_1, PW_2. Chaque monde est caractérisé par une valeur de possibilité. Les degrés de possibilité et de nécessité sont dérivés des valeurs d'incertitude d'entrée par exemple $\pi_{pW_{11}} = Max(\pi_{PW_1}, \pi_{PW11}), N_{PW_{11}} = Min(\pi_{PW_1}, \pi_{PW11})$. Ainsi, de cette manière, les valeurs d'incertitude renvoyées par le résultat final sont basées sur les relations avec toutes les valeurs utilisées lors du processus d'invocation. Pour cela, on doit utiliser une fonction d'agrégation qui nous permet de déterminer cette valeur. Et de cette

manière, nous pouvons formaliser l'invocation possibiliste par la définition suivante :

Définition : (Invocation possibiliste) L'invocation possibiliste $Invoke^P(S^P, I^P)$ peut-être définie d'une manière extensionnelle sans l'implication de la théorie des mondes possibles. Elle est définie par l'équation suivante :

$$Invoke^P\left(S_i^P, I^P\right) = \left\{\left\langle O_i^{P1}, \pi\left(O_i^{P1}\right), N\left(O_i^{P1}\right)\right\rangle, ..., \left\langle O_i^{Pn}, \pi\left(O_i^{Pn}\right), N\left(O_i^{Pn}\right)\right\rangle\right\}, o\text{ù}$$
$$O_i^{Pj} \; est\; un\; sortie\; de\; S_i^P \; satisfaisant\; I. \quad (3.1)$$

où,

- I^P : le vecteur de paramètre d'entrée du service.
- O_i^{Pn} **:** le vecteur de paramètre de sortie du service.
- S_i^P **:** service invoqué.
- π **:** degré de possibilité
- **N :** degré de nécessité

Limites de l'approche proposée L'approche proposée est limitée aux opérations de sélection et de projection. Par exemple, si nous avons une dépendance entre les attributs des mêmes tables ou une requête qui nécessite une comparaison entre ces attributs, dans ce cas-là, nous sommes obligés de renvoyer de fausses réponses aux utilisateurs. Par conséquent, cette réaction impacte la qualité de notre approche. En outre, cette proposition pose un problème au niveau de l'opération de jointure qui est une opération de base qui détermine le processus d'invocation possibiliste et rend difficile la détermination de la partie composition des services Web possibilistes. Par exemple, pour répondre à une requête qui effectue une jointure entre deux tables T1 et T2 sous la condition A = D. Cette requête est représentée par la figure 3.3.

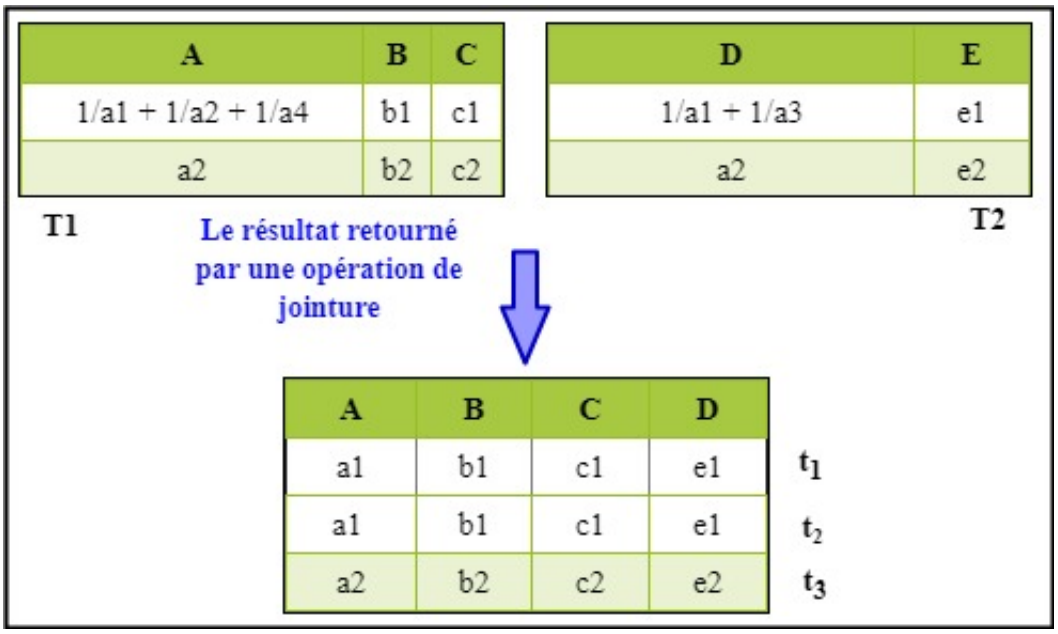


FIGURE 3.3 – Limitation de l'opération de jointure possibiliste.

Le résultat de cette jointure est résumé au niveau de la figure 3.3. Ce résultat pose un problème car il est impossible d'avoir à la fois les tuple t_1 et t_2. Par conséquent, il peut être nécessaire d'exprimer l'exclusivité de certains tuples. Ce modèle élimine également les valeurs « candidates » qui ne satisfont pas les conditions de sélection. Il se peut que certaines interprétations correctes ne puissent être trouvées à partir du résultat de la sélection. Et enfin, nous avons le problème du nombre de mondes possibles que nous devons traiter. Ce nombre peut être gigantesque car il représente le produit des valeurs candidates dans toutes les distributions apparaissant dans une relation impliquée et il n'est pas donc raisonnable de considérer un tel calcul en se basant sur l'explication des mondes possibles. Pour résoudre tous ces problèmes, nous proposerons, dans la section suivante, un modèle des services Web incertains basé sur l'approche possibiliste avec provenance.

3.4 Interrogation des services Web incertains avec provenance

Dans cette section, nous présenterons, dans premier temps, notre modèle de services Web possibilistes avec provenance, ensuite, et dans un deuxième temps, nous montrerons comment invoquer et utiliser des services Web possibilistes dans une composition.

3.4.1 Modèle Possibiliste pour les Services Web avec provenance

L'interrogation de données possibilistes en utilisant la notion de mondes possibles n'est pas efficace, car le calcul des mondes possibles d'une base de données possibiliste prend beaucoup de temps et la taille des données générées peut être énorme. Pour cette raison, dans notre approche, nous nous basons sur la notion de base de données possibilistes avec provenance [VK11].

Notre modèle est basé sur deux concepts : Le premier est la notion de x-tuples et x-relations qui a pour objectif d'exprimer la sémantique de l'invocation possibiliste. Un x-tuple est un ensemble de tuples appelés aussi alternatives. Ces alternatives représentent des valeurs mutuellement exclusives à ce tuple. Le modèle de x-relation représente un ensemble d'instances possibles qui peuvent être construites comme suit : choisissez exactement une alternative de chaque x-tuple de la relation qui est un x-tuple certain et sélectionnez zéro ou un seul tuple pour chaque x-tuble incertain. La x-relation possibiliste peut représenter chaque tuple d'une base de données possibiliste caractérisée par sa valeur de possibilité et de nécessité. Le deuxième principe est la notion de provenance qui permet de déterminer l'origine de chaque résultat obtenu dans une requête. La provenance des tuples dérivés consiste à référencer d'autres tuples dans une base de données tout en utilisant leurs identifiants uniques. Elle résout le problème de l'indépendance et de la dépendance entre les tuples. Elle facilite également la corrélation et la coordination de l'incertitude dans les résultats de la requête avec des données d'entrées incertaines. Formellement, si une alternative avec la donnée t est un résultat de deux autres alternatives t_1 et t_2, mais peut aussi être le résultat combinant des alternatives t_3 et t_4, nous obtenons l'équation suivante : $\lambda(t) = (id(t_1) \wedge id(t_2)) \vee (id(t_3) \wedge id(t_4))$, où $id(t_i)$ est l'identifiant de l'alternative t_i. La relation "Saw" est représentée en utilisant le modèle de x-relation dans le tableau 3.4.

Id	**Witness**, **Car**	**N**
11	1/⟨Alice, Audi A4⟩, 0.8/⟨Alice, Audi A5⟩	1
12	1/⟨Amy, Audi A4⟩, 0.4/⟨Amy, Audi A6⟩	1

TABLE 3.4 – x-relation Saw

Cette x-relation possibiliste peut être interprétée par tous les x-uplets possibilistes où chaque x-uplet est caractérisé par un degré de possibilité et un degré de nécessité. Cette interprétation est illustrée dans le tableau suivant :

Id (λ)	Witness	Car	Π	N
11.1	Alice	Audi A4	1	0.2
11.2	Alice	Audi A5	0.8	0
12.1	Amy	Audi A4	1	0.6
12.2	Amy	Audi A6	0.4	0

TABLE 3.5 – Interprétation de la x-relation Saw

Nous sommes maintenant prêts à définir formellement les services Web possibilistes. Un service Web possibiliste S_i^P est défini comme un tuple $S_i^P\langle I_i^P, O_i^P\rangle$. Où I_i^P *et* O_i^P sont respectivement l'ensemble des paramètres d'entrée et de sortie du service S_i^P, et $O_i^P = \{\langle o_i^{P1}, \Pi(o_i^{P1}), N(o_i^{P1}), \lambda(o_i^{P1})\rangle, \ldots, \langle o_i^{Pn}, \Pi(o_i^{Pn}), N(o_i^{Pn}), \lambda(o_i^{Pn})\rangle\}$, où n est le nombre de tuples que le S_i^P manipule, o_i^{Pj} est la j-ième sortie de S_i^P, et $\Pi(o_i^{Pj})$, $N(o_i^{Pj})$, $\lambda(o_i^{Pj})$ sont respectivement la possibilité, la nécessité et la lignée de o_i^{Pj}.

Par exemple, supposons que le service S_1 du tableau 3.1 manipule les données du tableau 3.5. $S_1\langle Witness, Car\rangle$ renvoie les voitures vues par un témoin donné, ainsi que leurs valeurs de possibilité, nécessité et provenance, c.-à-d., la sortie est un sous-ensemble de $\{\langle Audi\ A4, 1, 0.2, 11.1\rangle, \langle Audi\ A5, 0.8, 0, 11.2\rangle, \langle Audi\ A4, 1, 0.6, 12.1\rangle, \langle Audi\ A6, 0.4, 0, 12.2\rangle\}$..

3.4.2 Invocation des Services Web Possibilistes avec provenance

Dans cette section, nous définissons la fonctionnalité d'invocation du service Web possibiliste. Plus précisément, nous distinguons :

1. Invocation conventionnelle, où les entrées utilisées pour appeler le service Web sont certaines.
2. Invocation possibiliste, où les entrées utilisées sont possibilistes.

Le processus d'invocation est illustré ci-dessous (figure 3.4). Au début, l'utilisateur envoie une requête spécifique qui est composée des valeurs d'entrées caractérisées par le degré de possibilité, de nécessité et de provenance. La première étape consiste à extraire des informations décrivant l'incertitude des données. Ensuite, nous envoyons une requête simple au service Web puisque son fonctionnement incertain reste inchangeable. Dès qu'on reçoit la réponse du service, nous devons passer à l'étape de l'agrégation qui permet de déterminer les caractéristiques d'incertitude de cette réponse, et donc, de calculer le degré de possibilité, de nécessité et de provenance. Par la suite, nous obtenons la réponse finale qui peut être envoyée à l'utilisateur.

Soit $S_i^P\langle I_i, O_i^P\rangle$ un service Web possibiliste, et I une entrée certaine utilisée pour invoquer S_i^P. La fonction d'invocation conventionnelle est définie comme suit :

$$Invoke(S_i^P, I) = \{\langle o_i^{P1}, \Pi(o_i^{P1}), N(o_i^{P1}), \lambda(o_i^{P1})\rangle, \ldots, \langle o_i^{Pm}, \Pi(o_i^{Pm}), N(o_i^{Pm}), \lambda(o_i^{Pm})\rangle\}$$
$$(m < n), o\grave{u}\ o_i^{Pj}\ est\ une\ sortie\ of\ S_i^P\ satisfaisant\ I. \quad (3.2)$$

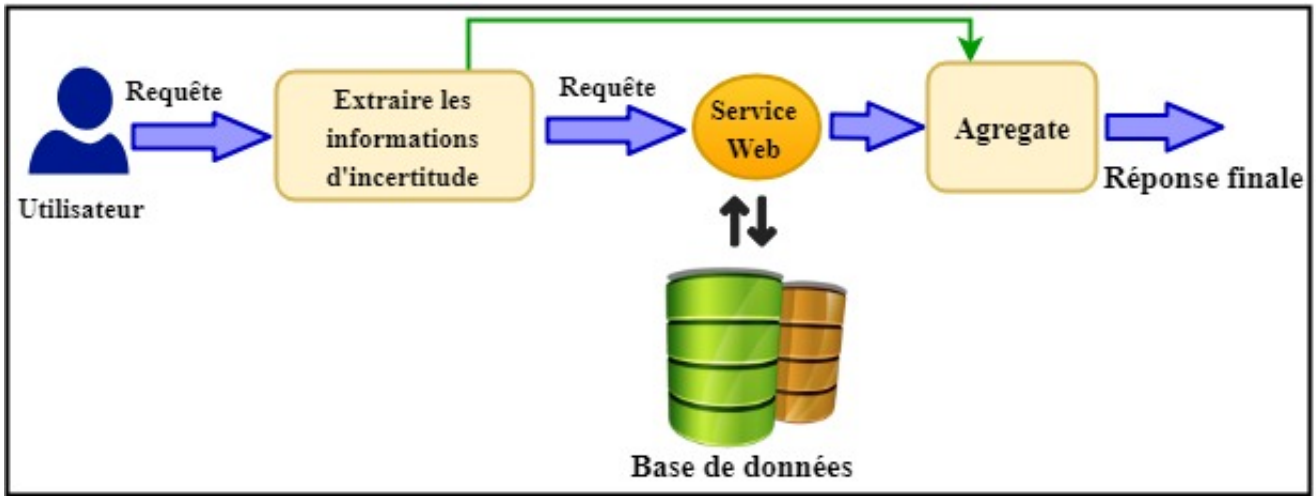


FIGURE 3.4 – Le processus d'invocation.

Par exemple, l'invocation du service S_1 avec le témoin « Alice » renvoie (retourne) les résultats présentés dans le tableau 3.6.

Car	Π	**N**	λ
Audi A4	1	0.2	11.1
Audi A5	0.8	0	11.2

TABLE 3.6 – Exemple d'invocation conventionnelle.

Maintenant, nous introduisons la fonctionnalité d'invocation possibiliste des services Web possibilistes. Nous considérons une entrée possibiliste $\langle I, \Pi(I), N(I), \lambda(I)\rangle$ utilisée pour invoquer un service Web possibiliste $S_i^P\langle I_i, O_i^P\rangle$, où $\Pi(I)$, $N(I)$, $\lambda(I)$ sont respectivement la possibilité, la nécessité et la provenance de I. La fonction d'invocation possibiliste est définie comme suit :

$$Invoke(S_i^P, \langle I, \Pi(I), N(I), \lambda(I)\rangle) = \{\langle o_i^{P1}, min(\Pi(I), \Pi(o_i^{P1})), min(N(I)), N(o_i^{P1})),$$
$$\lambda(I) \wedge \lambda(o_i^{P1})\rangle, \ldots, \langle o_i^{Pm}, min(\Pi(I), \Pi(o_i^{Pm})), min(N(I)), N(o_i^{Pm})), \lambda(I) \wedge$$
$$\lambda(o_i^{Pm})\rangle\}, o\text{ù } o_i^{Pj} \textit{ est une sortie de } S_i^P \textit{ satisfaisant } I. \quad (3.3)$$

Dans ce qui suit, nous proposons une technique pour invoquer le service incertain. L'idée est de traiter les données incertaines qualifiées par les services Web et de proposer une approche basée sur un ensemble d'équations algébriques permettant de déterminer l'incertitude. L'exécution d'une invocation renvoie un ensemble de tuples caractérisés par des degrés d'incertitude. Les données obtenues sont associées à des valeurs de possibilité, de nécessité et de provenance de l'invocation associée.

Algorithm 1 Invocation de services Web possibiliste avec provenance

Require: User : Utilisateur de Web
CS : un service Web prêt à être appelé par l'utilisateur
Ensure: O : Sortie
PC : Paramètres d'incertitude

1: Récupérer les paramètres d'entrée $(I, \pi_{input}, N_{input}, \lambda_{input})$
2: Obtenir des paramètres d'incertitude $\pi, Nand\lambda$ from service provider
3: Extraire la valeur d'entrée certaine I
4: Processus d'invocation certaine
5: Récupérer la sortie de l'invocation
6: Extraire le degré d'incertitude du résultat obtenu par l'invocation
7: Déterminer $Ext(\pi)$
8: Déterminer$Ext(N)$
9: Déterminer $Ext(\lambda)$
10: Calculer les valeurs finales de possibilité, de nécessité et de provenance en fonction des valeurs d'incertitude des entrées
11: Déterminer $\pi_{final}= F(\pi_{input}, \pi_{output})$
12: Déterminer $N_{final} = F(N_{input}, N_{output})$
13: Déterminer $\lambda_{final} = F(\lambda_{input}, \lambda_{output})$
14: Combiner les résultats $PC = f(\pi, N, \lambda)$
15: Renvoie le résultat final à l'utilisateur IN, π, λ
16: Fin

Cette invocation possibiliste proposée est décrite dans cet algorithme. Il est composé par 4 parties. La première partie consiste à traiter les paramètres d'entrée (1-3). Nous extrayons l'entrée certaine et les paramètres d'incertitude. Dans la seconde partie (4-5), nous faisons une invocation normale (certaine) et nous obtenons une sortie. Ensuite, dans

la partie suivante (6,9), nous déterminons le degré d'incertitude en utilisant les fonctions mathématiques proposées. Et enfin, et dans la dernière partie (10,16) nous renvoyons le résultat à l'utilisateur.

Id (λ)	**Owner**	**Car**	Π	**N**
12.1	Tom	Audi A4	1	0.4
12.2	Tom	Audi A5	0.6	0

TABLE 3.7 – Interprétation du propriétaire de la x-relation(S_2)

Par exemple, supposons que le service S_2 du tableau 3.1 manipule les données du tableau 3.7. L'invocation de S_2 avec $\langle Audi\ A4, 1, 0.2, 11.1\rangle$(une sortie du service S_1) renvoie $\langle Tom, 1, 0.2, 11.1 \wedge 12.1\rangle$, alors que son invocation avec $\langle Audi\ A5, 0.8, 0, 11.2\rangle$ (l'autre sortie du service S_1) renvoie $\langle Tom, 0.6, 0, 11.2 \wedge 12.2\rangle$.

Id (λ)	**Owner**	**Car**	Π	**N**
13.1	Tom	Audi A4	1	0.5
13.2	Tom	Audi A5	0.5	0

TABLE 3.8 – Interprétation de la x-relation propriétaire (S_3)

3.4.3 Composition de Services Web Possibiliste avec provenance

Dans la section précédente, nous avons proposé un modèle possibiliste pour les services incertains. Dans cette section, nous nous intéressons au problème selon lequel une composition implique des services dont la sémantique est incertaine. Dans ce cas, nous nous demandons : quelles sont les interprétations possibles de cette composition ? Et quelles sont leurs valeurs de possibilité, nécessité et provenance ?

Une composition décrite l'interaction entre deux ou plusieurs services afin de répondre à une requête complexe qui ne peut pas être résolue par un seul service. L'interprétation d'une composition (i.e., sa sémantique) doit être définie explicitement pour permettre sa réutilisation comme étant un nouveau service. L'interprétation d'une composition décrit la sémantique des données retournées par l'exécution de cette dernière. Une composition de services est caractérisée par deux facteurs : les services impliqués, et le plan de composition qui décrit l'orchestration des services composés. Informellement, un plan de composition est un graphe orienté acyclique , dans lequel les nœuds représentent les services composés

et les opérateurs de traitement de données (e.g., sélection, projection, jointure, etc.). Les arcs représentent les échanges de données faits entre ces nœuds. Deux compositions qui impliquent les mêmes services n'ont pas nécessairement le même plan de composition. Cela est dû aux différents opérateurs algébriques qui sont impliqués dans les plans des compositions.

Rappelez-vous que certaines requêtes des utilisateurs nécessitent souvent une composition de plusieurs services Web pour obtenir une réponse. Pour revenir à notre exemple, la requête de Bob implique les services Web S_1, S_2 *et* S_3 (voir Tableau 3.1). Pour trouver le propriétaire de la voiture vue par le témoin *« Alice »*, S_1 est d'abord invoqué avec l'entrée *« Alice »*. Les sorties obtenues sont indiquées dans le tableau 6. Ces sorties sont ensuite utilisées pour invoquer S_2 et S_3 (supposons que S_3 manipule les données du Tableau 3.8). Les résultats obtenus de S_2 sont donnés dans le Tableau 3.9 (voir Section 5.2 pour plus de détails sur l'invocation). De même, l'invocation de S_3 avec les sorties de S_1 renvoie les résultats du tableau 3.10.

Owner	Π	**N**	λ
Tom	1	0.2	$11.1 \wedge 12.1$
Tom	0.6	0	$11.2 \wedge 12.2$

TABLE 3.9 – Invocation de S_2

Pour calculer le résultat final, nous devons agréger les données renvoyées par ces services et calculer les valeurs de possibilité, de nécessité et de provenance du résultat. Pour calculer ces valeurs, nous avons développé un ensemble d'opérateurs algébriques qui nous permettent de calculer ces degrés. Ces opérateurs sont présentés dans la partie suivante.

Owner	Π	**N**	λ
Tom	1	0.2	$11.1 \wedge 13.1$
Tom	0.5	0	$11.2 \wedge 13.2$

TABLE 3.10 – Invocation de S_3

Les opérateurs algébriques Une composition peut inclure plusieurs services Web possibilistes. Lorsque les sorties de ces services sont agrégées, les valeurs de possibilité, nécessité et provenance des résultats obtenus doivent être calculées. Ces valeurs peuvent être importantes pour plusieurs raisons : calculer les meilleurs résultats, évaluer la qualité des résultats, prendre les bonnes décisions, etc. Le calcul de degrés de confiance des résultats finaux nécessite d'explorer différentes combinaisons de tuples possibles pour évaluer

la composition. Calculer tous les mondes possibles après l'invocation de chaque service est inefficace car le nombre de ces mondes est exponentiel avec le nombre de tuples. Pour résoudre ce problème, nous optons pour une approche extensionnelle – qui ne nécessite pas la matérialisation des mondes possibles – et définissons un ensemble d'opérateurs de composition nécessaires pour formuler les plans d'orchestration des compositions de services [Yu+08] y compris les services Web possibiliste. Ces opérateurs supposent que les tuples traités peuvent être soit corrélés soit indépendants.

— $Invoke^P$: La définition de cet opérateur a été donnée dans la section 3.4.2.

— $Project^P(I^P, \bar{a})$ **:** Soit I^P un vecteur de tuples possibilistes, et $\bar{a}$ un ensemble d'attributs. L'opérateur du project projette le vecteur sur les attributs $\bar{a}$ et les valeurs de possibilité, nécessité et provenance d'un tuple t dans l'ensemble de sortie sont calculées comme suit :

$$\begin{aligned} \pi(t) &= max(\pi_{t_i}) \\ N(t) &= max(N_{t_i}) \\ \lambda(t) &= \vee(\lambda(t_i)) \\ & \qquad where\ t_i\ \in I^P \end{aligned} \tag{3.4}$$

— $Select^P(I^P, \bar{c})$ **:**Soit $\bar{c}$ un ensemble de conditions et alt (t) représente les alternatives de $x-tuple$ t. La possibilité, la nécessité et la provenance d'un tuple t dans l'ensemble des valeurs de sortie sont calculées comme suit :

$$\begin{gathered} \pi(t) = \pi(t_i) \\ N(t) = min(1 - \max_{B}(\pi(t_j)), N(t_i)) \\ \lambda(t) = \lambda(t_i) \\ where\ B\ :\ t_j\ \in\ alt(t)\ and\ t_j\ satisfied\ the\ condition\ \bar{c} \\ and\ t_i\ \in\ alt(t)\ and\ t_i\ does\ not\ satisfy\ the\ condition\ \bar{c} \end{gathered} \tag{3.5}$$

— $Aggregate^p(I_1^p, ..., I_n^p, \bar{a})$ **:** Soit I_i^p (où$11 \leq i \leq n$) un vecteur de tuples possibilistes produits par un service donné S_i, et $\bar{a}$ un ensemble d'attributs ; l'opérateur « aggregate » joint les vecteurs $I_1^p, ..., I_n^p$ sur $\bar{a}$ et alt (t) représentent les alternatives de x-tuple t. La possibilité, la nécessité et la provenance d'un tuple agrégé t sont

calculées comme suit :

$$\begin{gathered}\pi(t) = min(\pi(t_1), ...\pi(t_n)) \\ N(t) = min(1 - max(\pi(t_1^{'}), ..., 1 - max(\pi(t_n^{'}), N(t_1), ..., N(t_n)) \\ \lambda(t) = \wedge(\lambda(t_1), ..., \lambda(t_n)) \\ where\ t_1 \in alt(I_1^p), ..., t_n \in alt(I_n^p)\ and \\ t_1^{'} \in alt(I_1^p)\ and\ t_1^{'}\ does\ not\ satisfy\ the\ condition\ of\ join\ \bar{a}, ...\end{gathered} \tag{3.6}$$

Maintenant, nous sommes prêts à calculer le résultat final de la requête. La figure 3.5 ffiche le plan d'exécution de la composition. Notre exemple suppose que les services incertains S_1, S_2 et S_3 sont impliqués dans une composition.

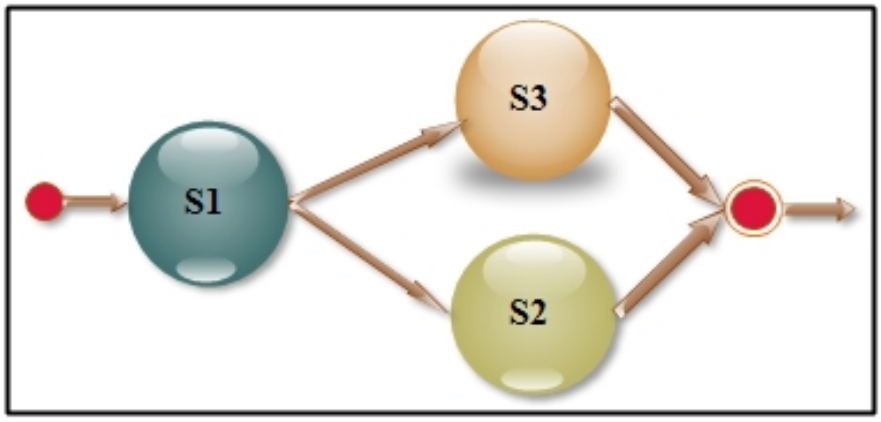


FIGURE 3.5 – Composition des services.

Calculer le degré final de possibilité, de nécessité et de provenance nécessite d'explorer les différentes combinaisons de tuples possibles pour évaluer la composition. Ce calcul est fait en utilisant les opérateurs que nous avons déjà défini. La figure 3.5 représente une interprétation de la composition. Chaque interprétation a les valeurs de possibilité, de nécessité et de provenance et représentée par une branche dans cette interprétation. Dans ces branches, nous pouvons utiliser les opérateurs algébriques (project, select,). L'opérateur d'invocation calcule la possibilité, la nécessité et la provenance des tuples en sortie en choisissant le minimum de possibilité et de nécessité et en concaténant l'expression de la provenance. Une composition correspond à un ensemble de composition possible. Dans cette composition, le même tuple peut exister en plusieurs mondes possibles. Par exemple, le tuple <Tom> existe dans toutes les branches. L'opérateur d'agrégation à la fin de chaque branche calcule la possibilité, la nécessité et la provenance de chaque tuple par la formule de l'agrégation. Par exemple, le résultat final retourné par les services est le tuple <Tom> avec possibilité = $min(1, 1, 0.5, 06) = 0.5, nécessité = min(0.4, 0.5, 0, 0) = 0$ et provenance = $(11.1 \wedge 12.1) \wedge (11.1 \wedge 13.1) \wedge (11.2 \wedge 12.2) \wedge (11.2 \wedge 13.2)$.

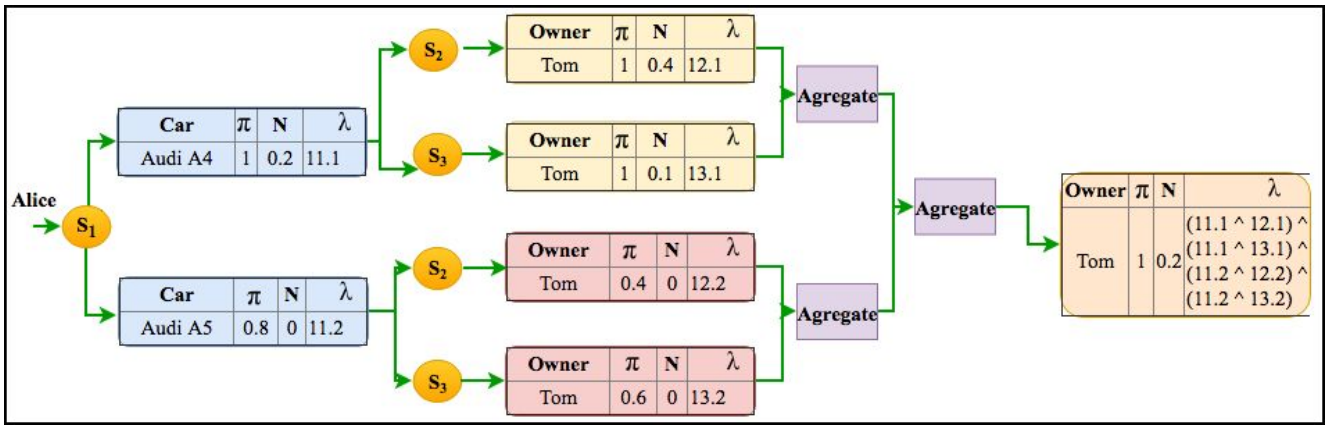


FIGURE 3.6 – Composition et exécution des services.

D'une façon plus générale, nous devons proposer un algorithme d'évaluation des requêtes au sein des services Web incertains possibilistes qui peut répondre à tout type de requête. Cet algorithme repose sur plusieurs blocs : le premier représente la partie qui permet la génération des tuples possibles s'il s'agit d'une entrée qui peut être certaine ou incertaine et nous permet de trouver le service à invoquer. Avant de procéder à l'opération d'invocation, l'extraction des valeurs d'incertitude à savoir les valeurs de possibilité, de nécessité et de provenance. L'opération d'invocation est ensuite réalisée en considérant uniquement l'entrée certaine. A ce stade, le fonctionnement du service est considéré comme standard les opérateurs algébriques sont ensuite pris en compte pour déterminer les valeurs d'incertitude des données retournées par ce service. Ces opérations sont répétées jusqu'à épuiser tous les services impliqués dans cette composition. Le résultat retourné à l'issue de cette opération d'invocation est un ensemble de tuples possibiliste qui vont subir une opération d'agrégation qui sera le résultat final retourné à l'utilisateur.

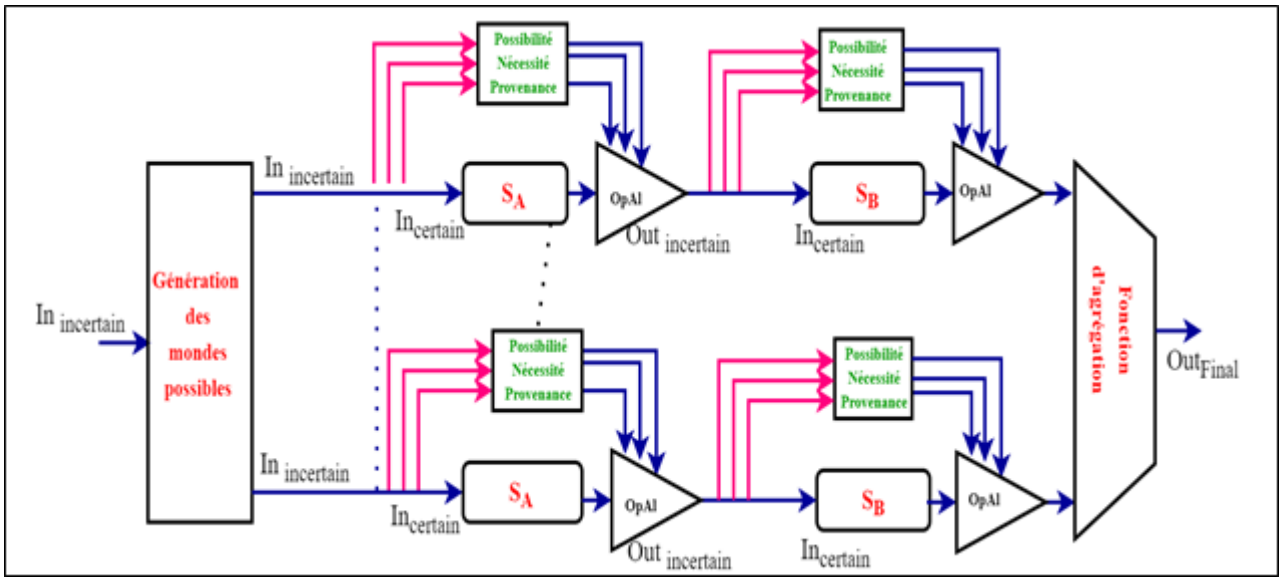


FIGURE 3.7 – Composition et exécution des services.

L'algorithme 2 décrit le principe de composition des services pour répondre à une requête à travers les services Web possibilistes. Il s'est articulé autour de trois parties principales. La première partie (lignes 1-3) décrit le traitement de la requête et la manière d'extraire les informations d'incertitude. La deuxième partie (lignes 4-7) est consacrée à la phase de composition des services. Cette partie décrit la génération de l'ensemble des tuples possibles correspondant à chaque entrée. Elle décrit aussi l'invocation de services, le calcul des valeurs d'incertitude de chaque réponse retournée par ces services. Elle détaille également le processus d'extraction des degrés d'incertitude qui seront pris en compte parmi les entrées aux services suivants. Ces étapes sont répétées jusqu'à la fin des services impliqués dans cette composition. A la fin de cette étape, nous obtenons n tuples auxquels sont associés des degrés de possibilité, de nécessité et de provenance. La troisième et dernière partie de cet algorithme (lignes 8-11) détaille la phase d'agrégation chargée de la construction d'une réponse finale qui sera retournée à l'utilisateur. Il s'agit donc de récupérer les n tuples obtenus par les services invoqués, d'appliquer les opérations algébriques pour calculer les valeurs finales de possibilité, nécessité et provenance et de déterminer et retourner la réponse finale.

Algorithm 2 Composition de services Web possibiliste avec provenance

Require: User : Utilisateur de Web
CS : un service Web prêt à être appelé par l'utilisateur
Ensure: O : Sortie
PC : Paramètres d'incertitude

1: Recevoir la requête R
2: Traitement de requête
3: Extraire les valeurs d'entrée
4: Générer les différents tuple possible T_i
5: Appeler chaque service S_i (nombre d'appel correspond au nombre de tuple généré)
6: **for** chaque service Si de la base B **do**
7: Définir les variables d'invocation de chaque service
8: Extraire les informations d'incertitude de chaque tuple $Inf(p_i, \lambda, N)$
9: Invoquer le service S_i
10: Récupérer la sortie de $S_i(Out_{S_i})$
11: Appliquer la fonction de jointure $Join(Out_{S_i}, Inf(p_i, \lambda, N))$
12: **end for**
13: Retourner à l'étape 6 pour le service $S_i + 1$
14: **if** fin des services de composition **then**
15: Récupérer toutes les réponses retournées par les services
16: Appliquer une fonction d'agrégation
17: Récupérer le résultat final
18: Envoyer la réponse accompagnée par ces valeurs de possibilité, nécessité et provenance.
19: **end if**
20: Fin

3.5 Conclusion

Dans ce chapitre, nous avons proposé une approche possibiliste d'Invocation et de composition de services Web à sémantique incertaine. En particulier, nous avons proposé le concept de services possibilistes qui permet de représenter un ensemble de services avec leurs mondes possibles. Ensuite, nous avons défini la façon dont les différentes interprétations possibles d'une composition existante (impliquant des services Web incertains) peuvent être calculées. Nous avons également défini deux algorithmes qui assurent l'invocation des services possibilistes et l'évaluation efficace des requêtes utilisateurs. Les services possibilistes sont formellement décrits par le modèle possibiliste avec provenance, de sorte que les services incertains sont considérés comme des mondes possibilistes indépendants (pas de corrélation entre elles). L'implémentation du protocole développé, confirme la faisabilité de l'approche proposée.

Dans le chapitre suivant, nous proposons une approche probabiliste pour les ressources qui permettent de représenter les différentes corrélations existantes entre les ressources incertaines. Nous proposons également une méthode efficace pour l'évaluation des requêtes à travers les ressources à sémantique incertaine et une algèbre pour traiter le problème de la composition.

Chapitre 4

Approche probabiliste de composition des ressources Web incertaines.

Sommaire

4.1 Introduction

De nos jours, les systèmes d'information modernes permettent aux individus, aux objets connectés et aux organisations de produire et de publier une énorme quantité de données sur le Web via des API Web et des points de terminaison publics [MPD10]. Les données sont ensuite collectées à partir de différentes sources et combinées dans des mashups [BDS08] pour produire de la valeur ajoutée. Cependant, le processus d'intégration pose plusieurs problèmes, car les données peuvent provenir de sources hétérogènes, contradictoires ou incomplètes [HRO06].

Les données extraites du Web sont pleines d'incertitudes : elles peuvent contenir des contradictions ou résulter de processus intrinsèquement incertains, tels que l'intégration ou l'extraction de données. L'examen de l'origine de l'incertitude - comme de légères différences dans la mesure d'événements physiques ou des différences dans les informations fournies par différentes sources de données décrivant la même entité - peut aider à comprendre sa nature. Cependant, le principal défi du Web d'aujourd'hui est fourni une solution pour faire face à cette incertitude. Au lieu de choisir une version unique mais arbitraire de l'information, nous croyons que les utilisateurs devraient avoir toute la gamme de possibilités pour décrire une entité. L'état actuel du Web donne aux utilisateurs la possibilité de sélectionner une seule représentation pour une entité.

Dans Ce chapitre, nous présentons la notion de ressource Web incertaine probabiliste, comment elles peuvent être utilisés pour représenter les données du Web, et la complexité de l'évaluation des requêtes au sein de ce type de ressources. Les données réelles (par exemple, World Wide Web) sont souvent incertaines en raison de l'incertitude inhérente aux données elles-mêmes ou aux processus de collecte, d'intégration et de gestion des données. L'objectif principal de ce chapitre est de proposer un cadre théorique pour décrire, manipuler et évaluer des données incertaines sur le Web. Nous proposons un algorithme qui peut aider à répondre aux requêtes et déterminer le degré d'incertitude de chaque réponse. Nous présentons un modèle pour définir et interpréter les ressources Web incertaines. Nous définissons un modèle d'interprétation et une algèbre pour calculer l'incertitude dans le contexte de la navigation hypertexte classique et dans le contexte de l'évaluation des requêtes de données.

4.1.1 Exemple de motivation

Le Web d'aujourd'hui exige aux utilisateurs de sélectionner une seule représentation à partir d'un ensemble de représentations d'une entité. A titre d'exemple, considérons un utilisateur essayant de répondre à la question suivante : « Quel est le nom du directeur

de l'université de Paris ? ». En règle générale, lors de la navigation sur les pages Web, un utilisateur tape le nom de la personne dans un moteur de recherche, puis il choisit l'un des liens proposés dans la page de résultats et poursuit sa navigation de ce point à la réponse. Une telle méthode ne fournit qu'une seule représentation à la fois. Le choix de la représentation à sélectionner est laissé à l'utilisateur qui clique sur les liens jugés les plus appropriés.

Considérons également la même requête émise vers une API Web traditionnelle. Le processus mashup rejette généralement les données identifiées comme incorrectes au cours du processus d'intégration pour fournir à l'utilisateur uniquement les informations supposées correctes. Alors que l'objectif principal est de permettre au Web de traiter des données incertaines, ce qui nécessite de fournir un cadre théorique permettant de décrire, de manipuler et d'exposer des données incertaines aux utilisateurs, nous identifions plusieurs défis dans ce contexte. Premièrement, les ressources Web devraient pouvoir donner accès à des représentations multiples et simultanées qui laissent la possibilité de décrire la même entité de différentes manières : il est nécessaire de concevoir et de développer des ressources Web incertaines. Deuxièmement, la navigation Web et les mashups de données devraient traiter à la fois des ressources Web incertaines et certaines. Il est nécessaire de fournir une solution pour combiner des informations issues de ressources Web incertaines afin de répondre aux requêtes des utilisateurs.

4.1.2 Challenges

Supposons qu'un utilisateur Alice veut trouver le nom du directeur de l'université de Paris. La requête d'Alice, i.e., Q1 : "Retourner le nom du directeur de l'université de Paris" peut-être résolue en utilisant les ressources Web de notre exemple de motivation. Cependant, composer des ressources à sémantique incertaine pour répondre à une requête comme Q1 soulève les challenges suivants :

— **Comment modéliser la sémantique incertaine des ressources Web REST :** Lorsque les ressources Web ont un contenu incertain, un nouveau modèle de représentation pour les ressources Web est nécessaire. Ce modèle doit être capable de représenter les ressources Web incertaines avec leurs degrés de certitude ou d'exactitude en se basant sur un modèle probabiliste. De plus, il doit prendre en considération les dépendances qui peuvent exister entre les interprétations possibles.

— **Comment analyser la représentation programmatique des ressources Web incertaines :** La représentation standard (JSON, . . .) nécessite d'être modifiée ou adaptée pour la prise en compte de cette incertitude. Un nouveau Parser doit

être proposé pour valider le JSON probabiliste proposé pour la représentation des ressources Web incertaines. Ce Parser doit valider ou non les représentations de ces ressources.

— **Comment évaluer une requête utilisateur à travers des ressources Web incertaines :** Contrairement aux techniques d'évaluation classiques où les ressources et les compositions sont certaines, quand la sémantique des ressources devient incertaine, les techniques d'évaluation sont insuffisantes dans le cas de ressources incertaines. Ces techniques classiques nécessitent donc une adaptation pour la prise en compte de la part d'incertitude dans la description des ressources. La composition de ressources certaines et incertaines devient elle-même une composition incertaine, et devra retourner non plus un résultat mais différents résultats accompagnés de leurs degrés d'incertitude.

4.1.3 Contributions

Dans ce chapitre, nous proposons une approche probabiliste pour les ressources Web incertaines. Les principales contributions de ce chapitre sont les suivantes :

— **Modèle de représentation des ressources Web incertaines :** Nous proposons un modèle probabiliste pour représenter les ressources Web à sémantique incertaine. Le principe de cette solution repose sur le fait que chaque interprétation possible d'une ressource sera représentée par un monde possible caractérisé par un degré de confiance.

— **Parser pour JSON probabiliste :** pour analyser le modèle probabiliste qu'on a proposé ci-dessus, nous avons développé un Parser probabiliste permettant de reconnaitre syntaxiquement le modèle d'interprétation proposé et de valider les ressources certaines et incertaines.

— **Modèle d'interprétation d'une composition des ressources Web incertaines :** Le principe général de ce modèle est basé sur le concept des mondes possibles probabiliste où l'interprétation d'une composition existante est définie comme étant un ensemble d'interprétations possibles obtenues en évaluant la même composition de ressources incertaines. Il est clair que l'évaluation de l'interprétation d'une composition est intractable, étant donné que les algorithmes proposés, pour simplifier l'interprétation possible, présentent une complexité exponentielle. Dans ce sens, nous proposons un théorème qui permet de calculer efficacement (i.e., dans un temps polynomial) la probabilité d'une interprétation possible.

— **Algèbre d'évaluation d'incertitude :** Cette algèbre sera utilisée dans un contexte d'une navigation hypertexte classique qui permet de combiner des données provenant de plusieurs ressources Web incertaines. Nous nous appuyons sur les modèles d'XML probabiliste pour développer un ensemble d'opérateurs et d'algorithmes spécifiques d'évaluation des requêtes de données incertaines.

— **Algorithme d'évaluation de ressources Web incertaines :** Évaluer une requête utilisateur Q à travers les ressources Web probabilistes revient à l'évaluer sur chaque représentation de ressources incertaines possible. Comme le nombre de Web possibles peut-être extrêmement large, par conséquent l'évaluation des requêtes devient difficile voire impossible. Dans ce sens, nous proposons un algorithme qui permet d'évaluer efficacement les requêtes utilisateurs (i.e., dans un temps raisonnable), sans avoir besoin de matérialiser l'ensemble des mondes possibles. De plus, le résultat obtenu (i.e., compositions et leurs probabilités), par notre solution, est en concordance avec la sémantique des compositions possibles.

Le reste de ce chapitre est organisé comme suit. Dans la section 2, nous décrivons les différentes notions de base de notre proposition. Dans la section 3, nous présentons le concept de ressource Web probabiliste et nous définissons sa sémantique. Dans la section 4, nous présentons la représentation programmatique de notre modèle probabiliste. La section 5 est réservée pour présenter le principe du Parser de JSON probabiliste proposé. La section 6 présente et détaille l'approche proposée pour l'évaluation des requêtes. Dans cette section nous décrivons aussi l'interprétation d'une composition dont les ressources impliquées sont incertaines. La septième et dernière section conclue ce chapitre en invoquant les perspectives des différentes contributions présentées dans ce chapitre.

4.2 Background

4.2.1 XML-Probabiliste

Le XML probabiliste [Ben+10] [Abi+09b] [AS13] est l'un des concepts proposés pour modéliser et gérer les différents types de données incertaines. En substance, un document XML probabiliste est une représentation compacte d'une distribution de probabilité sur des documents XML ordinaires. Différents modèles de XML probabiliste fournissent des langages différents, avec différents degrés d'expressivité, pour de telles représentations compactes.

Dans cette section, Nous présentons d'abord les concepts de base d'XML-probabiliste, nous décrivons le cadre formel de ce chapitre, et en particulier les définitions formelles des concepts de base : un document XML (ordinaire), un espace XML probabiliste et la représentation p-document d'un XML probabiliste [SS12].

4.2.1.1 Document XML

Nous supposons qu'un ensemble infini d'étiquettes (de labels), où une étiquette dans $\sum$ peut représenter une balise XML, un attribut XML, une valeur textuelle incorporée dans un élément XML ou la valeur d'un attribut. L'hypothèse que $\sum$ est infinie est faite dans l'intérêt de l'analyse de complexité. Un document XML [SS13] [Abi+09b] (ou simplement un document) est un arbre (fini) ordonné et non ordonné, où chaque nœud a une étiquette de $\sum$. L'étiquette d'un nœud de document V est désignée par label (v). On note $D_{\sum}$ l'ensemble (infini) de tous les documents. Les documents XML (ou les documents en abrégé) sont classés, non classés, étiquetés. Étant donné un document d, l'ensemble des nœuds et l'ensemble des arêtes de l'enfant sont désignées respectivement par $V(d)$ *et* $C(d)$. Nous utilisons l'opérateur $\prec$ pour désigner l'ordre total strict sur les nœuds d'un document, Root(d) pour la racine de d, L pour l'ensemble fini d'étiquettes (des labels), et $lbl(v) \in L$ pour l'étiquette (label) d'un nœud v. Nous ne faisons pas la distinction entre les étiquettes et les valeurs ici. Les notions d'enfant, parent, feuille, descendant, ancêtre ont leurs significations standard [SS13]. Deux documents d_1 *et* d_2 sont isomorphes, notés par $d1 \sim d2$, si l'un peut-être obtenu de l'autre en remplaçant les nœuds par d'autres nœuds tout en préservant les étiquettes et l'ordre des nœuds. Formellement, $d1 \sim d2$ s'il y a une bijection $\phi : V(d_1) \to V(d_2)$, tel que pour tout $v_1, v_2 \in V(d_1)$,

1. $lbl(v_1) = lbl(\phi(v_1))$;
2. $(v_1, v_2) \in C(d_1)$ *si et seulement si* $(\phi(v_1), \phi(v_2)) \in C(d_2)$; *et*
3. $v_1 \prec v_2$ *si et seulement si* $\phi(v_1) \prec \phi(v_2)$.

À titre d'exemple, la partie inférieure de la figure 4.1 montre un document d. Dans cette figure, les étiquettes représentent des valeurs textuelles (par exemple « financement automobile ») sont écrites en italique. Notez que la direction des arêtes n'est pas explicitement spécifiée et est supposée être descendante. De même, l'ordre parmi les frères et sœurs est supposé être de gauche à droite.

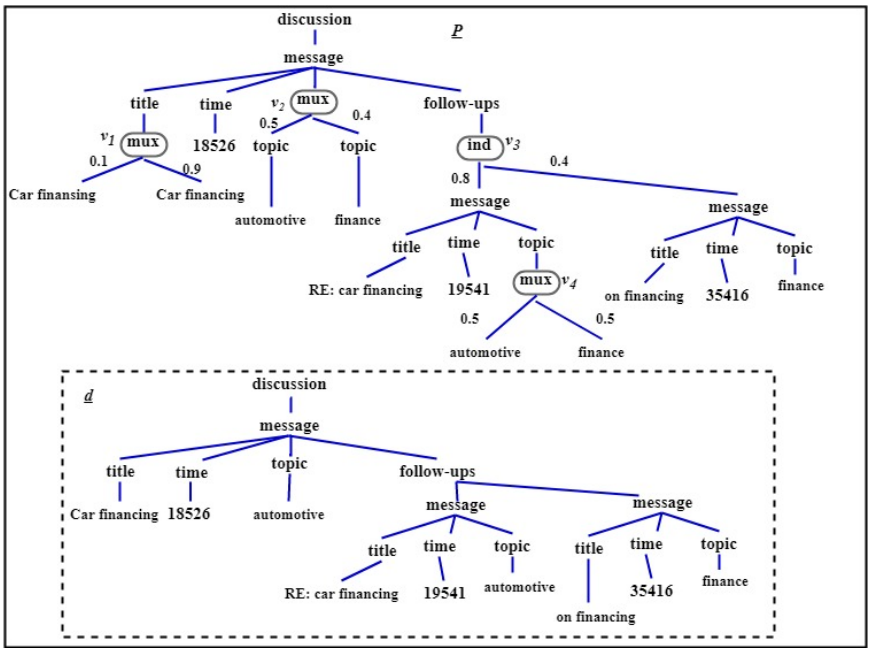


FIGURE 4.1 – Exemple d'un modèle d'XML probabiliste.

4.2.1.2 Espace d'Xml probabiliste

L'espace XML probabiliste, abrégé px-espace, est un espace de probabilité sur les documents. Bien que nous examinions brièvement les px-espaces continus, nous nous intéressons principalement aux px-espaces discrets [SS13] [Abi+10]. Donc, sauf indication contraire, nous supposerons implicitement qu'un px-espace est discret. Plus précisément, nous considérons un px-espace comme une paire $X = (D, p)$, *où* D est un ensemble de documents fini ou dénombrable infini, et $P : D \to [0, 1]$ est une fonction de probabilité satisfaisante $\sum_{d \in D}(P(d)) = 1$[Abi+09a]. Le support d'un px-espace $X = (D, p)$ est l'ensemble des documents $d \in D$, tel que $p(d) > 0$. On dit que le px-espace X est fini si X a un support fini ; sinon, X est infini. Quand il n'y a pas de risque d'ambiguïté, on peut abuser de notre notation et identifier un px-espace X par la variable aléatoire qui obtient un document choisi en fonction de la distribution de X. Ainsi, par exemple, si $X = (D, p)$ et d est un document, alors $Pr(X = d)$ (en mots, la probabilité que X soit égal à d) est $p(d)$ si $d \in D$, *et* 0 *sinon.*

4.2.1.3 P-documents

Un px-espace est codé au moyen d'une représentation compacte. La notion de base de la plupart de ces modèles de représentation est celle d'un P-document [AS06] [KNS11].

Formellement, un p-document est un arbre P qui est similaire à un document XML, sauf que P possède un ensemble distingué de nœuds distributifs en plus des nœuds ordinaires (qui ont des étiquettes (labels) de $\sum$). Les nœuds ordinaires de P peuvent appartenir à des documents dans le px-espace codé [SA07] [KNS10]. D'autre part, les nœuds distributifs sont utilisés uniquement pour définir le processus probabiliste qui génère des documents aléatoires (mais ils ne se produisent pas réellement dans ces documents). A titre d'exemple, la figure 4.1 montre un P-document, où les nœuds distributifs sont ceux représentés par des cases avec des coins arrondis (et désignés par v_1, v_2, etc.). Les mots "ind" et "mux" à l'intérieur de ces boîtes seront discutés plus tard. Chaque nœud distributionnel spécifie une distribution de probabilité. Dans le processus probabiliste qui génère un document aléatoire, un nœud distributionnel choisit au hasard un sous-ensemble de ses enfants en fonction de la distribution spécifiée pour ce nœud. La racine et les feuilles d'un P-document doivent être des nœuds ordinaires.

4.3 Ressources Web incertaines

Avant de présenter les ressources Web incertaines, il est d'abord nécessaire de rappeler certaines notions de base. Commençant par le concept des ressources Web. En général, il existe deux types de ressources Web qui sont les plus répandues et les plus utilisées : SOAP et RESTful. Dans ce chapitre, nous allons nous concentrer sur le principe des ressources Web de type RESTful [SS12].

Une ressource Web est tout ce qui peut-être obtenu à partir du World Wide Web (WWW) [Brz+11][Ros+08] par exemple les pages Web, le courrier électronique, les informations provenant des bases de données et les services Web [Eck+14]. Le concept d'une ressource Web a été évolué au cours de l'histoire du Web, de la notion initiale de documents ou de fichiers adressables statiques, à une définition plus générique et abstraite, englobant une entité qui peut-être identifiée, nommée, adressée ou traitée sur le Web en général ou dans tout système d'information en Web. Une ressource est un objet avec un type, des données associées, des relations avec d'autres ressources et un ensemble de méthodes qui opèrent dessus. Il est similaire à une instance d'objet dans un langage de programmation orienté objet, avec la différence importante que seules quelques méthodes standard sont définies pour la ressource (correspondant aux méthodes standard HTTP GET, POST, PUT et DELETE), tandis qu'une instance d'objet a généralement beaucoup de méthodes [RR07]. Les ressources peuvent être regroupées en collections. Chaque collection est homogène de sorte qu'elle ne contient qu'un seul type de ressource et qu'elle n'est pas ordonnée. Les ressources peuvent également exister en dehors de toute collection. Dans ce cas, nous nous référons à ces ressources en tant que ressources singleton. Une ressource peut four-

nir un objet unique, un ensemble de sous-ressources ou une notion abstraite, par ex. un concept de l'ontologie. Les ressources sont identifiées par un identifiant de ressource. Un identifiant est un localisateur de ressources uniformes URI (Uniform Resource Locators).

Pour mieux représenter les ressources Web, nous proposons la notation suivante qui définit une ressource par :

$$R = \{URI_R, \langle A_j, V_j \rangle\}$$

Où

— $A_j : Nom\ \ d'attribut$
— $V_j : Valeur\ \ de\ \ type\ \ :\ \ String|Number|Objet|Array|False|nullj \in [1, n]$
— $URI_R :\ \ l'URI\ \ qui\ \ identifie\ \ la\ \ ressource\ \ R.$

Chaque URI doit identifier une seule ressource, mais une ressource peut avoir plusieurs ressources (sous ressources) et l'expression $\langle A_j, V_j \rangle$ définit la représentation d'une ressource Web dans un serveur. Pour accéder à ces ressources en utilisant les méthodes HTTP, nous utilisons POST pour créer une ressource dans le serveur, GET pour récupérer une ressource, PUT pour changer le statut d'une ressource ou pour modifier la ressource et DELETE pour supprimer une ressource.

4.3.1 Définition

Dans le contexte des données incertaines, la sémantique des ressources Web incertaines est la façon dont une ressource est représentée sur le Web. Dans notre approche, les ressources Web incertaines adoptent la sémantique des Web possibles qui est basée sur la théorie du monde possible [KNS10]. Une ressource incertaine a plusieurs représentations possibles qui peuvent potentiellement et individuellement être interprétées comme vraies. Ces possibilités peuvent être interprétées comme un ensemble de mondes possibles (PW1, ..., PWn) avec une valeur de probabilité (PWi). Nous les appelons des Webs possibles, et dans ces Webs possibles, les données sont considérées comme certaines. Comme mentionné ci-dessus, le principe de ressource REST sera adopté. D'autre part, et afin de définir la notion de ressources incertaines, nous avons proposé certaines hypothèses que nous devons respecter :

1. En raison des principes REST, plusieurs représentations d'une URI (c'est-à-dire une

ressource) ne peuvent pas coexister, de sorte que les représentations possibles d'une ressource doivent être mutuellement exclusives.

2. Puisque nous traitons des ressources, l'incertitude ne devrait affecter que les représentations des ressources, donc, à l'intérieur d'une représentation possible donnée, chaque donnée est considérée comme certaine.
3. En raison des principes d'XML probabiliste, on a deux types d'incertitude : des nœuds de type d'incertitude indépendant et des nœuds de type d'incertitude mutuellement exclusive. Pour le premier type, si une propriété de ressource a plusieurs valeurs possibles, elle peut apparaître dans des représentations séparées ou dans la même représentation. Par contre le deuxième type exige l'apparition de ces valeurs dans des représentations séparées.
4. Par souci de simplicité, dans cette approche, nous considérons que chaque représentation possible d'une ressource donnée est représentée selon le même modèle.

Basé sur la définition de ressource Web [RR07], nous proposons une représentation spécifique qui aide à définir des ressources incertaines. Une ressource incertaine est représentée par la notation suivante :

$$R^P = \{URI_R, \langle A_j, V_j \rangle\}$$

Avec

— $A_j : Nom \ \ d'attribut$
— $V_i : V|IND(\langle V_1, P_1\rangle, ..., \langle V_k, P_k\rangle)|MUX(\langle V_1, P_1\rangle, ..., \langle V_k, P_k\rangle)$
— $i \in [1, n]$.
— $V_1, ..., V_k : \ \ Valeur \ \ de \ \ type \ \ : String|Number|Objet|Array|False|null$

Dans cette équation, l'expression $\langle A_i, V_i\rangle$ représente l'ensemble des représentations possibles de R^P. Puisque plusieurs représentations d'une ressource ne peuvent pas coexister au même URI, ces représentations sont mutuellement exclusives, et nous avons $P_i \in]0, 1]$. Dans la section suivante, nous allons présenter un nouveau modèle pour les ressource Web probabiliste.

4.3.2 Modèle d'une ressource Web incertaine

Nous proposons un modèle des ressources probabilistes qui permet de représenter efficacement les ressources Web incertaines. Ce modèle est basé sur le principe d'XML probabiliste spécialement sur l'approche ProTDB [NJ02b] qui est un modèle XML probabiliste proposé par Nieman et Jagadish en 2002. Cette approche diffère des travaux précédents dans le

développement des systèmes relationnels probabilistes dans la mesure où elle construit une base de données XML probabilistes. Cette conception est motivée par des besoins applicatifs impliquant des données qui ne sont pas facilement disponibles pour une représentation relationnelle.

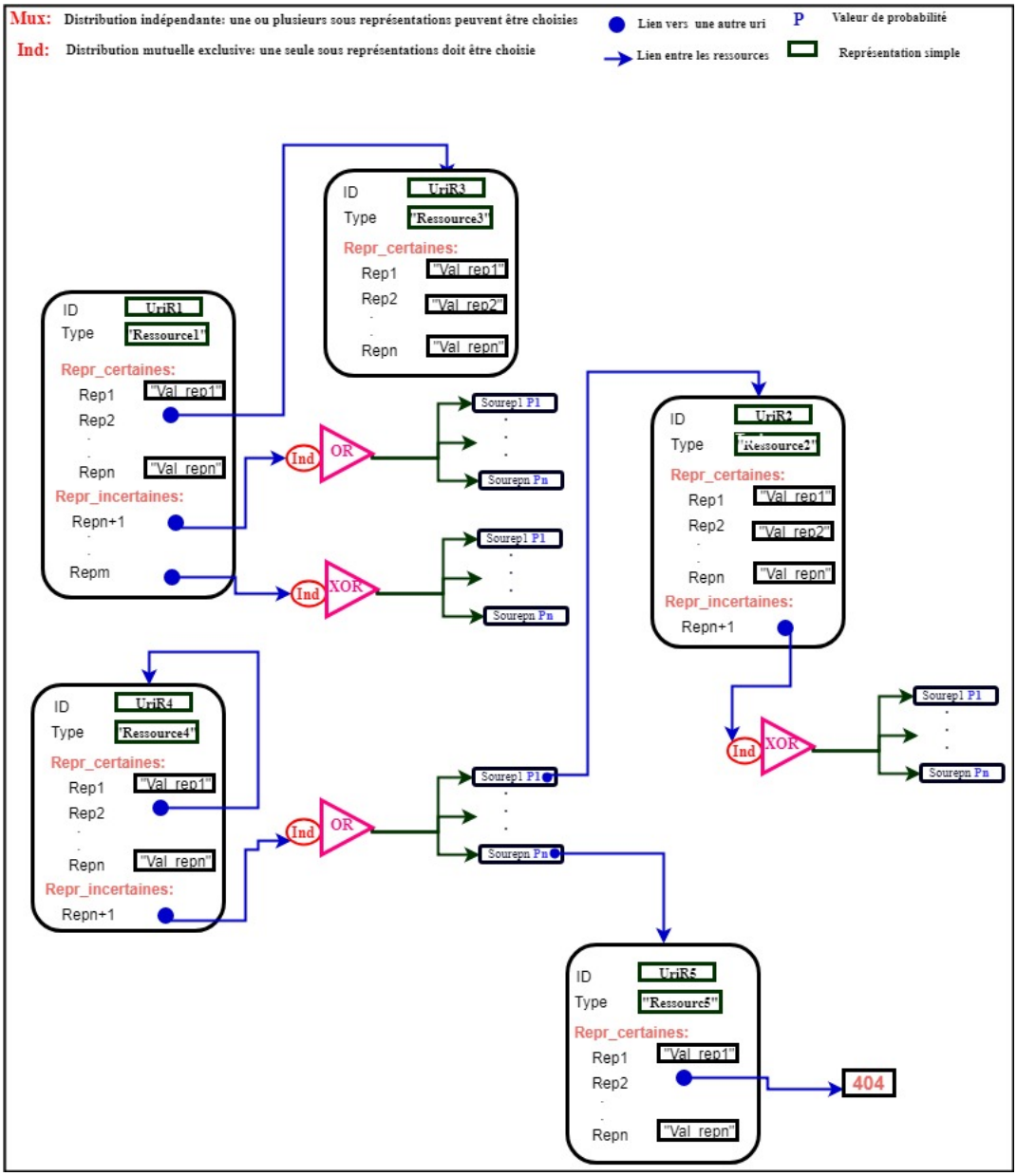


FIGURE 4.2 – Modèle général d'une ressource incertaine.

La figure 4.2 représente notre modèle pour les ressources Web incertaines. Nous définissons la ressource par un rectangle arrondi en noir dont chacun est identifié par un unique URI qui est l'identifiant. Les liens sont modélisés à travers des grands points et des flèches

en bleu. Ces symboles en bleu peuvent indiquer deux différents types de liens : soit un lien vers une nouvelle ressource certaine ou incertaine, soit vers un opérateur de type "Mux" ou "Ind". Ces deux opérateurs indiquent les types d'incertitude. Nous allons expliquerons le principe de ces deux nœuds ultérieurement. Comme nous avons déjà expliqué chaque ressource peut-être définie par une ou plusieurs représentations. Nous pouvons distinguer deux types de représentation : une représentation certaine qui est représentée par un simple rectangle vert et une représentation incertaine qui est représentée par un grand point bleu. Et enfin "P" représente la valeur d'incertitude d'une représentation c.-à-d. la valeur de probabilité d'avoir cette valeur. La valeur de cette probabilité P doit appartenir à l'intervalle]0,1].

Dans notre approche proposée pour les ressources Web incertaines, nous pouvons distinguer deux types de nœuds. Nous devrions expliquer le principe des deux types d'incertitude qu'on a mentionnés ci-dessus : "Mux" et " ind ". Nous devons spécifier pour chaque nœud distributionnel v, la distribution de probabilité de choisir un sous-ensemble des enfants de v. Nous définissons deux types de nœuds de distribution, chacun avec une manière différente de décrire cette distribution de probabilité.

4.3.2.1 Nœud de type ind (pour indépendant)

Un nœud v de type "ind" spécifie pour tout enfant W, la probabilité $Pv(W)$ de choisir W ; ce choix est indépendant de tout autre choix d'enfants (de v ou d'autres nœuds distributifs). Par conséquent, la probabilité de choisir un sous-ensemble C des enfants de v est

$$\prod_{W \in C} P^v(W) \prod_{W \in \overline{C}} (1 - P^v(W))$$

Où $\overline{C}$ est l'ensemble des enfants de v qui ne sont pas en C. Le fonctionnement de ce nœud semble beaucoup à l'opération mathématique OR. Dans notre modèle, ce nœud est présenté par le symbole présenté dans la figure suivante :

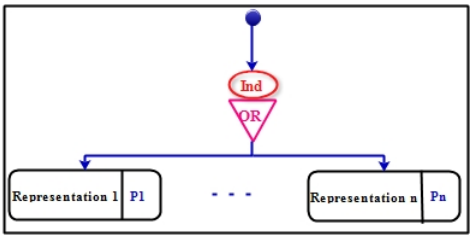


FIGURE 4.3 – Principe de nœud " Ind ".

4.3.2.2 Nœud de type Mux (pour mutuellement exclusifs)

Un nœud v de type "mux" spécifie les probabilités $Pv(w_1), ..., Pv(w_k)$ pour ses enfants $w_1, ..., w_k$, respectivement. Le nœud v choisit au maximum un enfant w_i avec la probabilité $Pv(w_i)$, indépendamment des autres nœuds distributifs. Nous exigeons que $\sum_{i=1}^{K} P^v(W_i) \leq 1$. La probabilité que v ne choisisse aucun de ses enfants est $1 - \sum_{i=1}^{K} P^v(W_i)$. Son principe de fonctionnement peut-être résumé par l'opération mathématique logique XOR. Ce nœud distributionnel est présenté par la figure suivante :

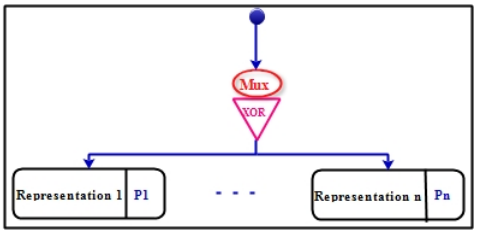


FIGURE 4.4 – Principe de nœud " Mux ".

Exemple La figure 4.5.a représente un exemple d'une ressource Web incertaine probabiliste qui contient à la fois les deux types de nœuds expliqués dans la partie précédente. Nous remarquons que les représentations possibles de notre scénario de ressource Web " Personne " représenté par la figure 4.5.b représentent les différents Web possibles dans lesquels la représentation est certaine. Dans chacun de ces sites, les ressources sont utilisables en tant que ressources classiques. Notez que l'interprétation de la représentation probabiliste est complètement indépendante d'une ressource à l'autre. Le modèle associé que nous avons défini est le suivant : chaque ressource est indépendante, mais chaque URI identifie une ressource unique, qui ne peut avoir qu'une seule représentation.

Afin d'interpréter nos ressources incertaines, nous adaptons le modèle d'XML probabiliste ProTDB [NJ02a]. Notre modèle spécifie, d'une part, que toutes les représentations possibles d'une ressource sont disjointes, et d'autre part, que les interprétations de ressources sont indépendantes les unes des autres. La figure 4.5.b montre comment nous interprétons les ressources incertaines comme un ensemble de représentations probables, chacune de ces représentations créant un Web possible dans lequel elle est la seule représentation de cette ressource. Dans cette figure, nous représentons chaque ressource sous forme d'un ensemble de représentations, le nombre en haut à droite représente la probabilité de cette représentation. Par exemple. Dans le Web possible W_2, la ressource ***« Nom »*** a une représentation unique qui contient un lien vers la représentation ***« Alice »*** ; les représentations ***« Jean »*** et ***« Isabelle »*** existent toujours mais ne sont pas connectées à la ressource ***« Nom »***.

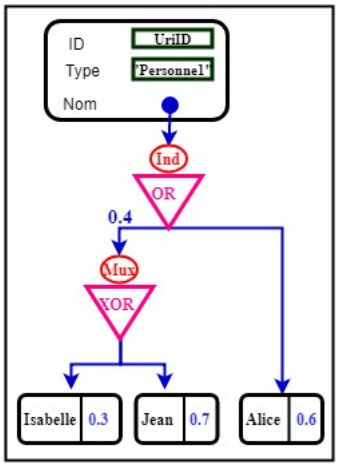


Figure 4.5.a. Exemple d'une ressource incertaine.

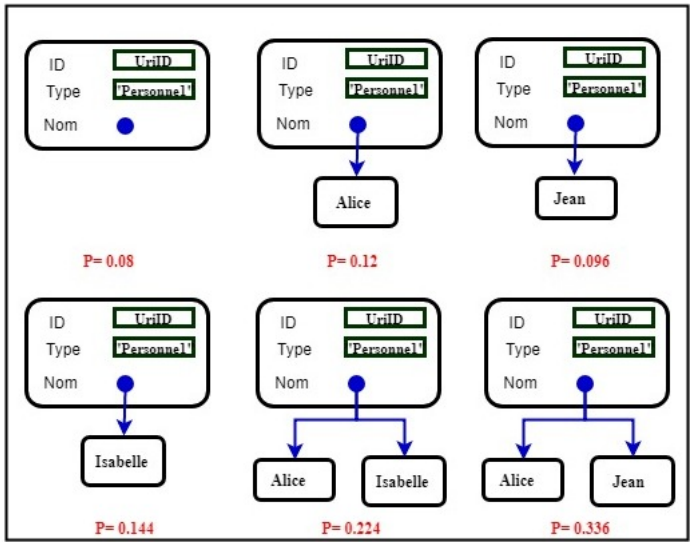


FIGURE 4.5 – Exemple de ressources Web incertaines.

Dans le Web possible W_1, la ressource incertaine *« **Nom** »* ne pointe à rien, elle est considérée comme une ressource inexistante qui n'a aucune représentation, dans notre formalisme, cette ressource inconnue est notée $\emptyset$. Comme le dit T. Fielding dans sa définition de REST [Fie00] : "Une ressource peut correspondre à l'ensemble vide, ce qui permet de faire référence à un concept avant que la réalisation de ce concept existe". Une ressource peut-être inconnue pour plusieurs raisons. Le serveur qui héberge la ressource aurait pu être arrêté, la ressource aurait pu être supprimée de son emplacement, l'URI pourrait être mal formée. Techniquement, une requête GET sur une telle ressource indéfinie conduit à une erreur HTTP, par exemple 404 non trouvé ou 503 état d'erreur Service non disponible. Cependant, lorsque nous demandons la ressource incertaine A, nous obtenons seulement la représentation connue.

4.4 Représentation programmatique de ressources incertaines

Afin de fournir un moyen pour gérer ces ressources incertaines, nous avons proposé un formalisme pour les représenter physiquement. Nous proposons un modèle de représentation, où nous fournissons toutes les représentations possibles d'une ressource incertaine. Dans ce modèle, nous donnons une probabilité à toutes ces représentations.

4.4.1 Modèle de JSON Standard

Avec REST, la communication se base sur des technologies Web et plus exactement sur le protocole HTTP (Hypertext Transfer Protocol) et les URI utilisés par le Web. REST utilise diverses techniques pour représenter une ressource. Les techniques les plus populaires sont XML et JSON. REST n'impose aucune restriction sur le format d'une représentation de ressource. Un client peut demander une représentation JSON alors qu'un autre client peut demander une représentation XML de la même ressource sur le serveur, etc. Il est de la responsabilité du serveur REST de transmettre au client la ressource dans le format que le client comprend. Les messages sont transmis dans un format standardisé. Pour l'intégration des réponses, nous utilisons généralement le format JSON (JavaScript Object Notation), plus léger et moins verbeux que le XML (eXtensible Markup Language). Des nombreuses bibliothèques et outils sont disponibles pour comprendre, analyser et modifier les données JSON. Il est facile pour les machines d'analyser et de générer. En JSON, il existe seulement trois types de données : scalaire (nombre, chaîne, booléen, nul), tableau et objet. A titre illustratif, voici un modèle de représentation d'une ressource Web certaine en JSON :

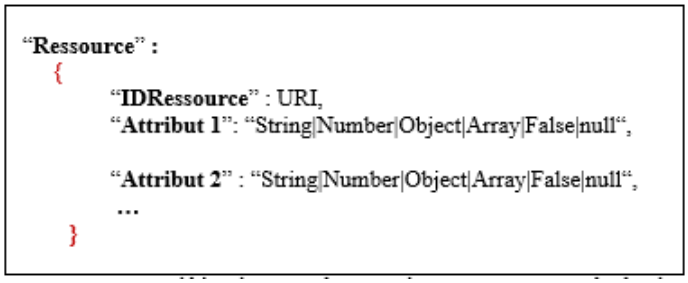


FIGURE 4.6 – Modèle de représentation JSON en général.

4.4.2 Modèle de JSON Probabiliste proposé

Nous avons défini des ressources ainsi que les données qui leur sont associées en termes de modèle de données JSON. Cependant, ces ressources sont toujours considérées comme

étant des entités abstraites. Avant de pouvoir être communiques, à un client, via une connexion HTTP, ces entités doivent être sérialisées en une représentation textuelle. Cette représentation peut ensuite être incluse en tant qu'entité dans un corps de message HTTP. Comme nous l'avons déjà expliqué précédemment, nous allons nous baser sur le principe de JSON standard. Pour cela, nous devons effectuer quelques modifications pour qu'il soit applicable au modèle que nous avons proposé. Le modèle proposé de JSON probabiliste est décrit par l'extrait présenté par la figure 4.7.

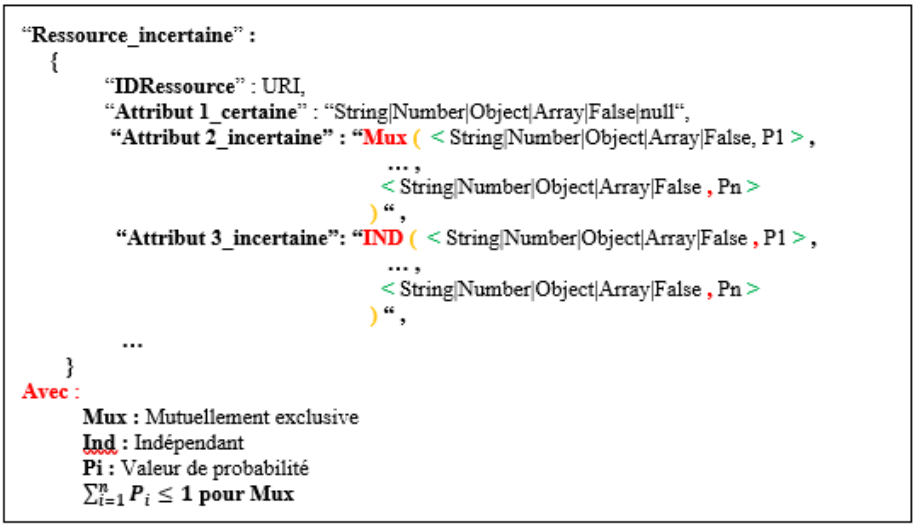


FIGURE 4.7 – Modèle de représentation JSON probabiliste.

D'après cette représentation nous pouvons distinguer deux types de ressources (attributs) : certaines et incertaines. Les attributs certains suit le modèle de JSON standard et les attributs incertains suit le modèle probabiliste. Ce dernier modèle présente deux types d'incertitude : mutuellement exclusive et indépendant. Les figures 4.8 et 4.9 représentent un exemple de ressource incertaine « laboratoire » et sa représentation sous le modèle de JSON probabiliste proposé.

Exemple de ressource incertaine

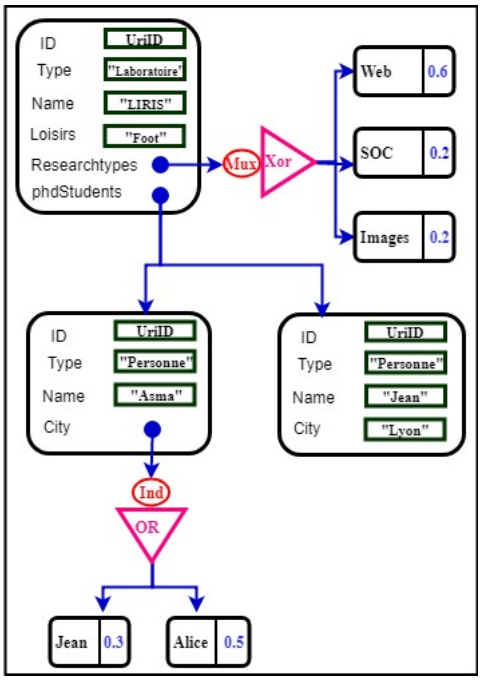


FIGURE 4.8 – Exemple d'une ressource incertaine.

```
{
  "Name" : "LIRIS",
  "Loisirs" : "foot",
  "Researchtypes": "Mux(< "WEB",0.6>, <"SOC",02>, <"IMAGES", 0.2>)",
  "phdStudents": [  {
                      "Name": "Asma",
                      "City": "Ind(<"Sousse", 0.3>, <"Monastir", 0.5>)",
                    },
                    {
                       "Name": "Jean"
                       "City":" Lyon",
                    },
                  ]
}
```

FIGURE 4.9 – La représentation d'une ressource Web probabiliste.

4.4.3 Modèle Parser JSON probabiliste

Dans la technologie informatique, un analyseur est un programme, généralement intégré à un compilateur. Il reçoit des informations sous la forme d'instructions de programme source séquentielles, de commandes interactives en ligne, de balises ou d'autres interfaces définies (par exemple, les noms (objets), les verbes (méthodes) et leurs attributs ou options). Ces informations peuvent, ensuite, être gérées par une autre programmation (par

exemple, d'autres composants dans un compilateur). Un analyseur peut également vérifier que toutes les entrées nécessaires ont été fournies. C'est un composant logiciel qui prend des données en entrées (texte fréquemment) et construit une structure de données, souvent une sorte d'arbre d'analyse, sous forme d'arbre de syntaxe abstraite ou autre structure hiérarchique, donnant une représentation structurelle de l'entrée, vérifiant la syntaxe correcte.

Comme nous l'avons déjà présenté, le modèle JSON probabiliste que nous avons proposé considère 11 types de données à savoir nombre, nombre probabiliste, chaine, chaine probabiliste, booléen, booléen probabiliste, tableau, tableau probabiliste, objet, objet probabiliste et nul. Par contre le modèle JSON standard, seulement trois types de données sont prises en compte, il s'agit du type scalaire (nombre, chaîne, booléen, nul), du type tableau et du type objet. Les types scalaires ont juste une valeur unique, par contre les tableaux contiennent une liste de valeurs de type arbitraire. Les objets constituent un ensemble non ordonné de paires (clé, valeur), également appelées attributs différents de ceux d'XML, où la clé est une chaîne et la valeur peut avoir un type arbitraire. Les types ajoutés doivent aussi être modélisés.

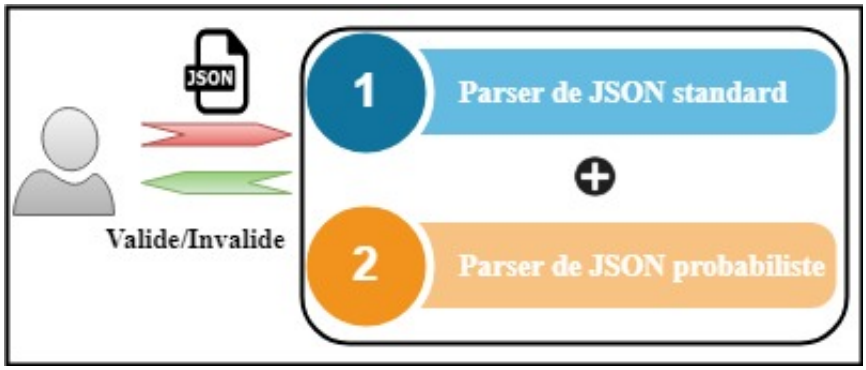


FIGURE 4.10 – Modèle Parser JSON probabiliste.

Dans cette partie, nous avons construit un analyseur JSON en utilisant la bibliothèque d'analyseurs que nous avons présenté dans la section précédente. L'analyseur est de nature similaire au tokenizer, sauf qu'il prend des jetons en entrée et sort les indices des éléments. Tout comme avec les jetons, un élément est marqué par sa position (index de départ), sa longueur et éventuellement son type d'élément. Ces numéros sont stockés dans la même structure que celle utilisée pour stocker les jetons. La grammaire de JSON définit le format correct pour le texte JSON. Ce sont les règles que nous devons suivre pour analyser correctement le texte JSON. Si le texte ne suit pas la grammaire correcte, alors l'analyseur doit lancer un message d'erreur pour indiquer que le texte JSON est impossible à analyser. La grammaire JSON peut-être (partiellement) définie comme ceci :

<JSON> ::= <value>

<Value> ::= <object> | <array> | <boolean> | <string> | <number> | <null>

<Array> ::= "[" [<value>] {"," <value>}* "]"

<Object> ::= "{" [<property>] {"," <property>}* "}"

<PropPrty> ::= <string> ":" <value>

Aussi, on doit ajouter les types incertains qui sont tableau probabiliste, nombre probabiliste, booléen probabiliste, String probabiliste et objet probabiliste. Nous devons aussi vérifier le format de deux nœud *Mux* et *Ind* et nous devons calculer la somme des valeurs de probabilité pour le nœud de type indépendant et nous devons vérifier que cette somme doit être inférieure à 1.

< Probabilistic JSON > ::= <value>|< probabilistic Value >

< probabilistic Value > ::= <object> | <array> | <boolean> | <string> | <number> | <null>|< probabilistic object > | < probabilistic array > | < probabilistic boolean > | < probabilistic string > | < probabilistic number >

<Array> ::= "[" [<probabilistic value >] {"," < probabilistic value >}* "]"

<Object> ::= "{" [<probabilistic property >] {"," < probabilistic property >}* "}"

< probabilistic Property> ::= <string> ":" Mux|Ind " (" "<" <value>, probability value"> "{"," "<"<value>, probability value">"}*")"

4.5 Requête HTTP sur des ressources incertaines

Afin de gérer les ressources Web incertaines, il est nécessaire que le client, qui interroge une ressource incertaine, soit capable de comprendre cette ressource. Nous appelons client incertain un client capable de demander et de comprendre des ressources incertaines. De la même manière, un client qui ne sait pas ce qui est une ressource incertaine devrait être capable de travailler avec cette ressource. Afin de respecter les principes du Web et de fournir une possibilité de rendre le client conscient de l'incertitude, nous comptons sur la négociation de contenu pour servir nos ressources incertaines.

La négociation de contenu est un mécanisme fourni par HTTP qui permet de servir

différentes versions de la même représentation de ressource (c.-à-d. au même URI), pour s'adapter au client. Grâce à la négociation de contenu, un client peut informer le serveur qu'il est capable de communiquer avec des ressources incertaines. De ce fait, nous sommes en mesure d'enrichir les ressources classiques et de manipuler à la fois leurs représentations certaines et incertaines. Chaque représentation doit mettre en œuvre une interface CRUD pour réaliser l'interface uniforme requise par l'architecture REST. Le protocole HTTP/1.1 définit huit méthodes, dont quatre permettent de définir l'interface : GET, PUT, POST et DELETE. Dans notre travail nous allons nous intéresser seulement à la méthode GET. Nous faisons la différence entre les requêtes GET classiques et incertaines. Afin de différencier les requêtes GET standard et les requêtes GET probabiliste des données incertaines, nous proposons la notation GET^P, qui décrit une requête GET provenant d'un client conscient incertain (c.-à-d. utilisant des en-têtes spécifiques pour demander des représentations incertaines).
Soit R^Pune ressource incertaine déployée sur URI_{R^P}, nous avons défini les représentations attendues suivantes :

$$GET(URI_R) := Rep_R$$

$$GET^P(URI_{R^P}) := \langle A_i, V_i \rangle$$

Avec
— $V_i : V|IND(\langle V_1, P_1\rangle, \ldots, \langle V_k, P_k\rangle)|MUX(\langle V_1, P_1\rangle, \ldots, \langle V_k, P_k\rangle)$
— $i \in [1, n]$
Où
— $V_1, \ldots, V_k : valeur\ \ de\ \ type\ \ String|Number|Objet|Array|False|null$

Les deux figures 4.11 et 4.12 représentent deux exemples d'une réponse de GET standard et de GET probabiliste :

```
HTTP/1.1 200 OK
Date: Thu, 11 Jan 2007 14:00:36 GMT
Server: Apache/2.0.54 (Debian GNU/Linux) DAV/2 SVN/1.1.4

{
        "espèce": "Dog",
        "race": "Labrador Retriever",
        "couleur": "Yellow",
        "âge": 6
    }
```

FIGURE 4.11 – Exemple de réponse GET standard.

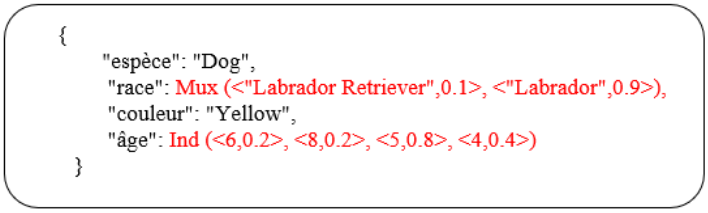


FIGURE 4.12 – Exemple de réponse GET probabiliste.

Dans notre approche, GET^P ne définit pas une nouvelle méthode HTTP. GET^P est seulement une notation plus complexe d'un GET standard avec des en-têtes spécifiques. La méthode GET agit comme un GET standard avec un en-tête HTTP spécifique que nous définissons comme ***Uncertainty-Processing : 1*** et ***Degree-Uncertainty : 0.2***. Nous choisissons de définir un en-tête spécifique pour éviter toute interférence avec l'utilisation normalisée de l'en-tête standard. En effet, l'en-tête ***« Uncertainty-Processing : 1 »*** est utilisé pour informer l'utilisateur que nous pouvons accepter le traitement des ressources Web incertaine et que la réponse que nous lui avons envoyée n'est pas certaine. Et l'en-tête ***« Degree-Uncertainty : 0.2 »*** spécifie le degré d'incertitude ou d'exactitude de la réponse envoyée. La bonne pratique consiste alors à spécifier un en-tête spécifique ad hoc pour respecter les standards HTTP .

La figure 4.13 représente un exemple illustratif d'un entête d'une réponse retournée par des ressources Web incertaines :

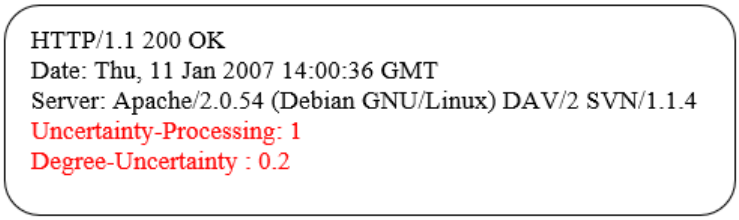


FIGURE 4.13 – Exemple d'un entête probabiliste.

4.6 Composition des ressources Web incertaines

Il est important de noter que les ressources incertaines peuvent être liées à d'autres ressources, qui peuvent aussi être des ressources incertaines. Dans de tels cas, nous générons une composition incertaine entre les ressources. Soit G une composition des ressources Web R_i, où chaque ressource R_i est identifiée par une adresse URIi et chaque représentation Ri est caractérisée par un ensemble de représentations possibles de la ressource PW_{R_i}. Dans

cette composition, chaque représentation possible conduit à la génération d'un nouveau Web possible. La probabilité de ce Web possible est dérivée des probabilités de ses représentations impliquées Dans ce calcul, nous devons prendre en compte, les représentations des ressources impliquées. Puisque nous considérons que nos ressources sont indépendantes, nous calculons la probabilité résultante en suivant quelques opérations algébriques. Soit un Web possible Wi impliquant l'ensemble des représentations $\langle A_1, V_1\rangle, \ldots, \langle A_n, V_n\rangle$, la probabilité du Web résultant est calculée comme suit :

$$PW_i = P_{Mux} * P_{Ind}$$

avec

— $P_{ind} = \prod_{W \in C} P^v(W) \prod_{W \in \overline{C}} = (1 - P^v(W))$
— $P_{Mux} = 1 - \sum_{i=1}^{K} P^v(W_i)$
Où $\overline{C}$ est l'ensemble des enfants de v qui ne sont pas en C.

Exemple

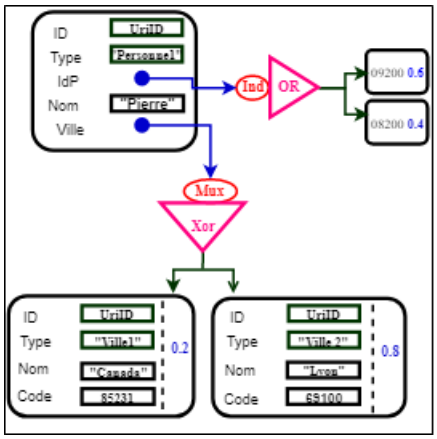


FIGURE 4.14 – Exemple d'une composition des ressources probabilistes.

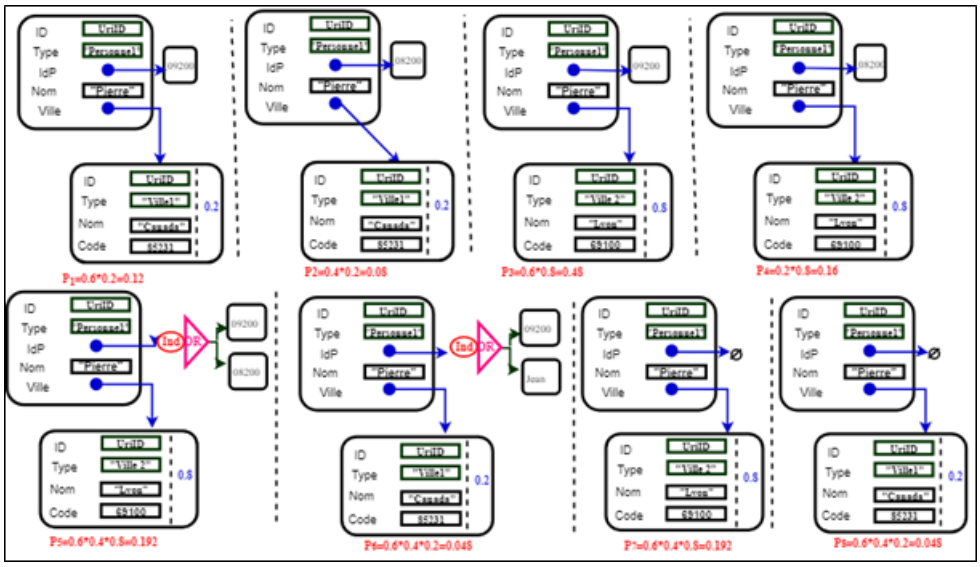


FIGURE 4.15 – Exemple d'Interprétation de cette composition

À partir de ce scénario, comme le montre la figure 4.15, nous générons tous les Web possibles qui sont dérivés des ressources impliquées. Dans cette figure, l'attribut ***« IdP »*** et la ressource ***« Ville »*** sont incertains. Pour l'attribut ***« IdP »*** dans les deux Web possibles PW_8 et PW_9 représentent la partie inconnue de cet attribut. . C.-à-d., techniquement, c'est que ces URI ne pointent rien. Chaque probabilité est calculée grâce à la formule de la sous-section précédente. A titre d'exemple, la probabilité de Web possible PW_4 et PW_6 est $prob(W_4) = prob(09200) * prob(08200) * prob(Canada) = 0.6 * 0.4 * 0.2 = 0.048$.

Afin de gérer les ressources Web incertaines, le client a besoin de connaître qu'il a interrogé une ressource incertaine pour être en mesure de comprendre la réponse envoyée. Nous appelons un client incertain le client qui est capable d'interroger et de comprendre des ressources incertaines. De la même manière, un client qui ne sait pas ce qui est une ressource incertaine devrait être capable de travailler avec cette ressource. Afin de respecter les principes du Web et de fournir une possibilité de rendre le client conscient de l'incertitude des ressources, nous comptons sur la négociation de contenu pour servir nos ressources incertaines.

Pour évaluer une requête au sein d'un ensemble des ressources incertaines. Nous avons proposé trois différents algorithmes de GET probabiliste. La principale différence entre ces trois algorithmes est le degré de complexité de chacun. Dans la partie suivante, nous allons détailler le principe de ces méthodes tout en présentant les différences entre eux. Tout d'abord, nous allons supposer que les ressources à parcourir sont enregistrés sous forme de

fichiers de type JSON.

4.6.1 Évaluation des requêtes : Algorithme 1

Dans cette section, nous allons présenter le principe d'algorithme le plus simple. Le cas où la requête va retourner tout un fichier JSON. Le principe de cette méthode est présenté par la figure suivante :

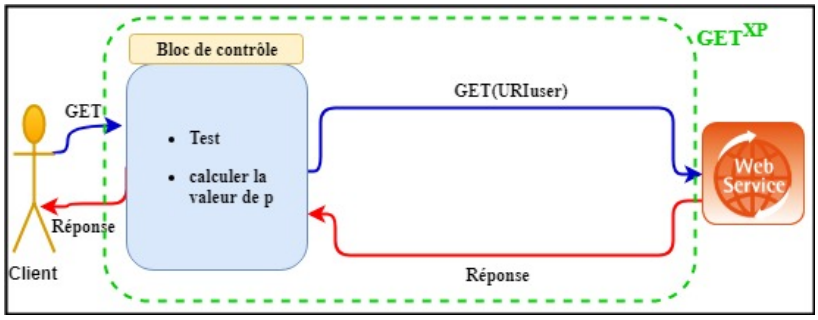


FIGURE 4.16 – Principe de la méthode GET 1.

Cette méthode est composée principalement par deux couches : une couche utilisateur et une couche serveur. Dans la première couche, nous trouvons l'utilisateur qui peut-être soit un client certain soit un client incertain. Ce client doit envoyer une requête GET qui est normalement sous la forme d'une adresse d'une représentation d'une ressource demandée. Cette requête ne contient aucune information sur l'incertitude ou la certitude de cette représentation. Et à la fin de cette méthode il reçoit la réponse de sa requête accompagnée par un entête qui contient le degré d'incertitude de la réponse. Pour la deuxième couche, nous avons la partie côté serveur, tout d'abord la requête doit être envoyée au bloc de contrôle qui va tout simplement envoyer la requête au service concerné par l'adresse dégagée de la requête utilisateur. Ce service retourne la représentation de la ressource demandée qui est sous forme d'un fichier JSON. Cette réponse doit être envoyée au bloc de contrôle afin de l'examiner, puis calculer la valeur de probabilité et ensuite enrichir l'entête de la réponse pour enfin envoyer la réponse à l'utilisateur.

L'algorithme 1 décrit l'évaluation des requêtes à travers des ressources Web probabilistes, et permet de trouver efficacement l'ensemble des représentations possibles présentées dans un seul fichier JSON. Cet algorithme est divisé en trois parties. La première partie (lignes 1) l'utilisateur envoie sa requête au serveur. La deuxième partie (ligne2, Lignes 10-20) consiste à décrire la partie de bloc de contrôle. Cette partie est divisée en deux principales étapes : la première étape est représentée par un bloc qui reçoit la requête en-

voyée par l'utilisateur et l'envoie au service pour chercher les représentations de ressource demandée. Pour la deuxième étape (ligne 10- 20) : le bloc de contrôle récupère la réponse de la requête, la parcourt pour dégager les nœuds Mux et Ind et pour calculer le degré de probabilité de la réponse. Puis Ajoute la valeur de P à l'entête de la réponse et ajoute une information qui indique une incertitude dans cette ressource et enfin envoie la réponse à l'utilisateur. Et la troisième partie présentée par les lignes (4-9) représente le fonctionnent du service responsable de la recherche de la représentation. Tout d'abord, il doit parcourir tous les documents JSON existant pour chercher la ressource demandée, récupère la représention demandée et l'envoie au bloc de contrôle.

Algorithm 3 GET 1 probabiliste

Require: URI
Ensure: $Prob_GET_Cas1$
1: Envoyer la requête (Req) au service de contrôle ($Cont\_Serv$).
2: Recevoir une Req de type GET.
3: Envoyer la Req au service correspondant.
4: **for** tous les documents.JSON (Doc_josn) **do**
5: **if** $Doci\_json == Doc\_demamndé(Doc\_Dem)$ **then**
6: Recupérer le Doc_Dem,JSON (Res).
7: **end if**
8: **end for**
9: Envoyer Res au $Cont\_Serv$).
10: Procourir Res.
11: **if** on a trouvé Mux **then**
12: $P = P \ \ XOR \ \ P\_Mux.$
13: **else if** on a trouvé Ind **then**
14: $P = P * P\_Ind.$
15: **else if** fin Res **then**
16: Ajouter la valeur de P à l'entête.
17: Ajouter une indication qu'il a une incertitude à l'entête.
18: Envoyer la réponse à l'utilisateur.
19: **end if**
20: Fin.

4.6.2 Évaluation des requêtes : Algorithme 2

Dans cette partie, nous allons présenter le principe du deuxième algorithme proposé. Cet algorithme est plus complexe que le premier puisque la réponse à une requête est sous la forme d'un ensemble de fichiers JSON reliés entre eux par des adresses(URL). Le principe de cet algorithme est présenté par la figure suivante :

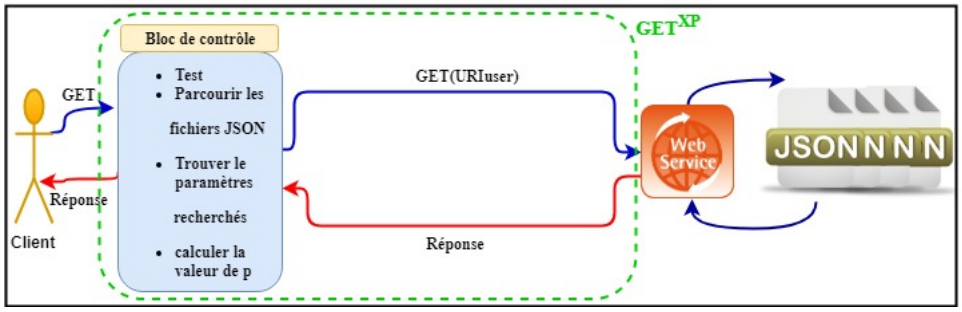


FIGURE 4.17 – Principe de GET (2eme méthode).

Cet algorithme est composé principalement de deux grandes parties : partie client et partie serveur. La première partie est identique à celle de l'algorithme précédent, par contre la deuxième partie présente deux services celui du bloc de contrôle et celui qui se charge de trouver la réponse à une requête. Le bloc reçoit la requête, l'envoie au service responsable pour trouver la réponse, récupère cette réponse, parcourir l'ensemble des fichiers, calculer la valeur de probabilité, enrichir la réponse et envoyer la réponse à l'utilisateur. Concernant le service chargé de trouver une représentation, il reçoit la requête envoyée par le bloc de contrôle, parcourt tous les fichiers JSON, regroupe celles correspondant à la requête et envoie cette réponse au bloc.

Dans ce qui suit, nous proposons un algorithme GET probabiliste qui permet de répondre à une requête utilisateur à laquelle est associé la valeur de probabilité. L'algorithme 2 permet de répondre efficacement à une requête au sein des ressources Web incertaines. Dans ce qui suit, nous décrivons les principales étapes de cet algorithme proposé.

— ***Étape 1 Envoie de la requête (lignes 1-3) :*** Dans cette phase, le client qui peut-être soit un client certain soit un client incertain envoie une requête au bloc de contrôle. Ce bloc récupère cette requête et l'envoie au service responsable pour chercher la réponse.

— ***Étape 2 Génération des représentations demandées (lignes 4-15) :*** pour chaque document JSON nous devons tester s'il correspond à la requête demandée (Doci-json = Doc-demandé (Doc-Dem)). Ensuite, pour chaque fichier json correspondant à la requête, nous devons le sauvegarder. Et enfin, nous devons envoyer l'ensemble des fichiers au bloc de contrôle.

— ***Étape 3 Calcul de degré de probabilité (lignes 16-29).*** Nous récupérons l'ensemble des fichiers. Nous devons les parcourir et à chaque fois que nous trouvons un nœud Mux ou Ind nous devons calculer la valeur de probabilité. Dès que nous terminons le parcours de tous les fichiers, nous devons ajouter la valeur de probabilité

et l'indication que la ressource est incertaine. Enfin, nous envoyons la réponse à l'utilisateur.

Algorithm 4 GET 2 probabiliste

Require: URI
Ensure: *Prob_GET_Cas*2
1: Envoyer la requête (Req) au service de contrôle (*Cont_Serv*).
2: Recevoir une Req de type GET.
3: Envoyer la Req au service correspondant.
4: **for** tous les documents.JSON (*Doc_josn*) **do**
5: **if** *Doci_json* == *Doc_demamndé*(*Doc_Dem*) **then**
6: /*Chercher le paramètre (*Param_Dem*)*demandé**/
7: **for** tous les Paramètre(Param) dans *Doc_Dem* **do**
8: **if** *Parami* == *Param_Dem* **then**
9: Récupérer la valeur demandée (Resp)
10: Récupérer sa valeur de probabilité (*Resp_prob*)
11: **end if**
12: **end for**
13: **end if**
14: **end for**
15: Envoyer Res au *Cont_Serv*).
16: Parcourir Res.
17: **for** tous les fichiers JSON **do**
18: **if** on a trouvé Mux **then**
19: $P = P \ \ XOR \ \ P\_Mux.$
20: **else if** trouvé Ind **then**
21: $P = P * P\_Ind.$
22: **end if**
23: **end for**
24: **if** fin Res **then**
25: Ajouter la valeur de *Resp_Prob* à l'entête.
26: Ajouter une indication qu'il a une incertitude à l'entête.
27: Envoyer la réponse à l'utilisateur.
28: **end if**
29: Fin.

4.6.3 Algorithme GET probabiliste général

D'une façon plus générale, nous devons proposer un algorithme d'évaluation des requêtes au sein des ressources Web incertaines probabilistes. Cet algorithme peut répondre à n'importe quelle requête. Cette méthode est composée de cinq blocs. Le premier repré-

sente la partie client qui peut-être de deux types : certain et incertain. Puis, nous trouvons le bloc de contrôle qui est normalement un service qui reçoit la requête utilisateur, il joue principalement le rôle d'un intermédiaire entre le client et le serveur. Il envoie la requête de l'utilisateur au serveur. Et en dernière étape il reçoit la réponse, parcourt la totalité des réponses (un seul fichier JSON ou plusieurs fichiers) pour calculer la valeur de probabilité qui précise le degré d'incertitude de la réponse, enrichit l'entête en ajoutant les deux entêtes que nous avons déjà expliqués dans les parties précédentes et enfin envoie la réponse à l'utilisateur. Le bloc suivant est le serveur qui s'engage de rechercher la représentation de la ressource demandée par le client. Tout d'abord, il récupère la requête envoyée par le bloc de contrôle, envoie la requête au bloc suivant pour générer l'ensemble des mondes possibles puis nous devons évaluer la requête dans chaque monde. Au sein de chaque bloc, l'évaluation de la requête est le même fonctionnement d'un GET standard puisque chaque monde représente une représentation certaine puis nous passons à l'étape suivante qui prend comme entrées l'ensemble des réponses obtenues par les différents mondes possibles, il doit agréger ces réponses pour nous donner comme résultat une seule sortie. Enfin, nous envoyons la réponse au serveur qui envoie cette réponse au bloc de contrôle.

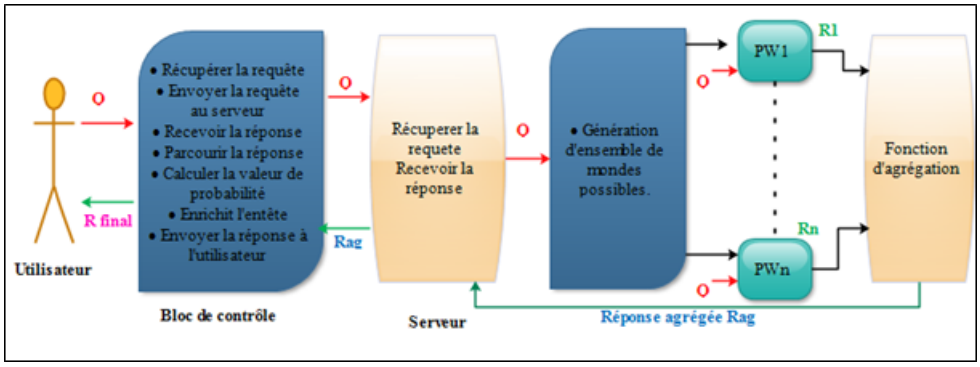


FIGURE 4.18 – Principe de GET probabiliste en général.

Pour plus de détail, nous avons proposé l'algorithme 3 qui décrit le principe de GET probabiliste général. Cet algorithme est composé de quatre parties. La première partie (Ligne1) représente le client qui va envoyer la requête. Cette partie est identique aux approches précédentes. La deuxième partie (Ligne 2 et Ligne 3) représente le fonctionnement du bloc de contrôle dès qu'il reçoit une requête à ce niveau-là ce bloc joue le rôle d'un acteur intermédiaire entre l'utilisateur et le serveur. Puis la partie suivante (ligne 4- Ligne 12) qui décrit le foncièrement du serveur. Au début il reçoit la requête envoyée par le bloc, il génère tous les mondes possibles ($PW_1, \ldots, PW_n$) s'il s'agit d'une ressource incertaine, il doit évaluer la requête sur chacun de ces Web possibles. Cette évaluation est indépendante d'un Web à un autre. Au sein de chaque Web, l'évaluation s'effectue en utilisant la méthode GET standard. Après cette phase, nous obtenons n réponses et chaque réponse est caractérisée par une valeur de probabilité. Nous devons passer à l'étape suivante qui consiste

à prendre comme entrées l'ensemble des réponses et leurs degrés de probabilité et nous donne comme sortie une seule sortie accompagnée d'une une seule valeur de probabilité. Ce résultat doit être envoyé au serveur qui l'envoie au bloc de contrôle. Et enfin la dernière partie (13-25), qui représente la partie dans laquelle nous allons enrichir la réponse. Tout d'abord le bloc récupère la réponse, il doit la parcourir et tester à chaque fois s'il y a un nœud Mux ou Ind , si nous avons l'un de ces deux nœuds nous devons calculer le degré de probabilité en utilisant les opérateurs expliqués. Après le parcours de totalité de réponse, nous devons l'enrichir par les deux entêtes et enfin nous envoyons la réponse à l'utilisateur.

Algorithm 5 Algorithme général de GET probabiliste

Require: URI
Ensure: *Prob_GET_Cas_General*
1: Envoyer la requête (Req) au service de contrôle (*Cont_Serv*).
2: Recevoir une Req de type GET.
3: Envoyer la Req au serveur.
4: Générer tous les mondes possibles
5: **for** tous les mondes Possibles PWI **do**
6: Evaluer la requête
7: Recevoir la réponse de PWI : Ri
8: **end for**
9: Envoyer les réponses Ri à la fonction d'agrégation
10: Appliquer les principes d'agrégation
11: Envoyer la réponse agrégée au serveur
12: Le serveur envoie la réponse au bloc de contrôle Rag
13: Récupérer la réponse
14: Parcourir Rag.
15: **if** on a trouvé Mux **then**
16: $P = PXORP\_Mux$.
17: **else if** on a trouvé Ind **then**
18: $P = P * P\_Ind$
19: **else if** on a trouvé URI **then**
20: GET(URI)
21: **else**
22: */fin Rag/**
23: Concaténer toutes les valeurs de probabilités
24: Ajouter la valeur de P à l'entête.
25: Ajouter une indication qu'il a une incertitude à l'entête.
26: Envoyer la réponse à l'utilisateur Rfinal.
27: **end if**
28: 25.Fin.

4.7 Conclusion

Il est nécessaire de fournir une solution permettant aux utilisateurs Web de gérer l'incertitude des données lorsqu'ils naviguent dans l'hypertexte. Le traitement de l'incertitude va

définitivement améliorer la qualité des réponses envoyées : nous traitons et faisons confiance à l'énorme quantité d'informations disponibles sur le Web. Dans ce chapitre, nous abordons la nécessité d'une solution pour gérer l'incertitude des données lors du référencement et de la navigation des ressources sur le Web. Dans ce chapitre, nous proposons un modèle pour représenter les ressources Web incertaines, en considérant les ressources incertaines comme des ressources spécifiques qui pourraient avoir plusieurs représentations possibles avec une probabilité. Nous définissons ces représentations possibles comme mutuellement exclusives et nous nous appuyons sur la théorie du monde possible pour interpréter ces représentations comme faisant partie d'un ensemble de sites Web possibles.

Aussi, nous avons proposé un Parser pour valider le modèle probabiliste que nous avons proposé et pour permettre aux développeurs d'utiliser cette approche sur le navigateur http. En plus de cela, nous nous appuyons sur le modèle de ressource Web incertain pour proposer une algèbre pour l'interprétation et l'évaluation des requêtes de données dans des compositions de ressources incertaines. Dans ce contexte, nous définissons les requêtes de données comme des chemins de ressources Web, et proposons des algorithmes pour évaluer les requêtes au sein des ressources incertaines. Tout en respectant notre modèle de probabilité, nous fournissons un algorithme de calcul, permettant de récupérer et de calculer l'incertitude associée à la réponse à une requête de données.

Chapitre 5

Approche d'indexation de documents textuels en présence de données incertaines

Sommaire

5.1 Introduction

Avec l'accumulation rapide des données de divers types, les systèmes de base de données modernes sont confrontés au problème de la gestion de grand volume de données, l'évolution rapide des données, l'hétérogénéité et l'incertitude de ces données. La récupération d'informations représente un pont entre l'utilisateur et cet univers qui est toujours en cours d'évolution. Les systèmes de récupération d'informations sont destinés à être des messagers dont l'objectif principal est de répondre aux requêtes des utilisateurs [BCP97]. Cependant, comme tout système construit en se basant sur d'autres systèmes, il est nécessaire d'améliorer l'efficacité du processus d'indexation pour nous donner une représentation fidèle, compacte et accessible de l'information recherchée. Pour améliorer la vie quotidienne de chacun en mettant à sa disposition des informations, il est souvent difficile de trouver des systèmes de récupération de l'information qui doivent être rendu d'une façon plus efficace et plus flexible. Pour cette raison, l'un des axes qui devrait être le plus intéressant est celui de l'indexation [Tam07].

Les techniques d'indexation ont été largement étudiées et constituent une option très importante pour le Big Data. En fait, l'indexation dans le contexte de l'entrepôt de données est un problème important en raison du grand volume de données traitées et du nombre d'attributs candidats qui peuvent également être très importants [SYY74]. La plupart des études de recherche proposées considèrent que l'indexation est définie sur un seul problème de données. Les données d'aujourd'hui présentent un ensemble de risques et de défis que les chercheurs veulent résoudre. L'un de ces risques est l'incertitude des données et des sources de données. L'indexation de données offre la possibilité de partager des données de tous types et de toutes valeurs, le partage de données est donc totalement ou partiellement certain ? En d'autres termes, l'indexation peut-elle assurer la protection contre l'incertitude des données sans perdre l'utilité de ces données lors de leur partage ? L'objectif principal de ce chapitre est de concevoir une approche de recherche d'informations pour collecter les informations les plus pertinentes en présence de données incertaines.

Cependant, l'information dans son intégralité est éparpillée entre plusieurs sources de données, privées ou publiques. ces sources de données sont conçues indépendamment les unes des autres à l'aide de structures de données, de schémas et de sémantiques spécifiques. Par conséquent, les sources de données présentent structurellement et sémantiquement des données hétérogènes. L'hétérogénéité structurelle n'est plus problématique avec l'utilisation croissante de RDF comme format de données du Web. D'autre part, l'hétérogénéité sémantique est un problème complexe et fondamental dans le processus de recherche d'information distribuée. Pour corréler les données distribuées d'une manière pertinente, il

est nécessaire de concilier leur sémantique. Ce chapitre vise à proposer une approche de collecte d'information sémantique en présence de données incertaines issues de données ouvertes. Notre principal défi est de concevoir un mécanisme d'indexation hautement efficace et qui peut supporter les caractéristiques de données d'aujourd'hui et surtout l'incertitude de données.

5.1.1 Exemple de motivation

Afin de construire notre proposition, nous nous appuyons sur un scénario d'une application dans le domaine de recherche d'information. Pour chercher l'ensemble des documents répondant à une requête, le système de recherche d'informations s'appuie sur une méthodologie formelle ou opérationnelle pour vérifier la correspondance entre les termes de chaque document ou les termes de la requête de l'utilisateur. La plupart des systèmes s'opposent que les termes composants des documents ont été parfaitement reconnus ou identifiés. L'étape qui s'occupe de l'extraction de ces termes est l'étape d'indexation , où les fondements des systèmes imposent que les termes extraits des documents sont des termes certains. Après le développement des technologies et réseaux sociaux, les données deviennent incertaines pour plusieurs raisons. Soit l'exemple présenté dans la figure 5.1 dans lequel les informations sur les documents audio (des conversations par exemple) sont extraites par des outils de reconnaissance de la parole. Ces outils fournissent pour chaque document audio une liste séquentielle de mots liés à la confiance de leur reconnaissance. Traiter ces documents revient donc à traiter des documents dont les mots sont incertains et associés des incertitudes liées au processus d'extraction. Au-delà de l'extraction des mots, ces systèmes abordent le problème de l'ambiguïté des mots : savoir si 'porte', par exemple, est un mot certain ou incertain dans un document permet d'assurer une meilleure précision face à une requête spécifiant l'une seule des deux possibilités. A partir de cet exemple, nous constatons qu'il existe effectivement des contextes dans lesquels l'hypothèse qu'un document du corpus est une séquence de mots certaines. Dans ces contextes, le document doit être considéré comme une suite de mots associés à des hypothèses d'extraction. De ce fait, le système de recherche d'informations qui s'appuie sur de tels documents doit intégrer cette nouvelle dimension qui nous permet de déterminer si les mots extraits sont des mots certains ou incertains. L'objectif de ce chapitre est de montrer une première approche où nous proposons tout d'abord une approche pour l'indexation syntaxique en présence de données incertaines, une approche d'indexation sémantique dans un environnement incertain et une approche hybride pour l'indexation de données et spécialement en prenant en considération l'incertitude de données.

La figure 5.1 présente un outil d'échange entre les différents utilisateurs d'une telle application peut s'effectuer en utilisant deux types de documents à savoir des documents

écrits et oraux. Les transcriptions automatiques de conversation sont des données à sémantique incertaines. Un système de recherche d'information capable de gérer les données incertaines s'avère nécessaire dans ce genre d'application. Dans ce contexte, nous allons s'intéresser à l'indexation de système de recherche d'information qui sera adaptée aux données incertaines.

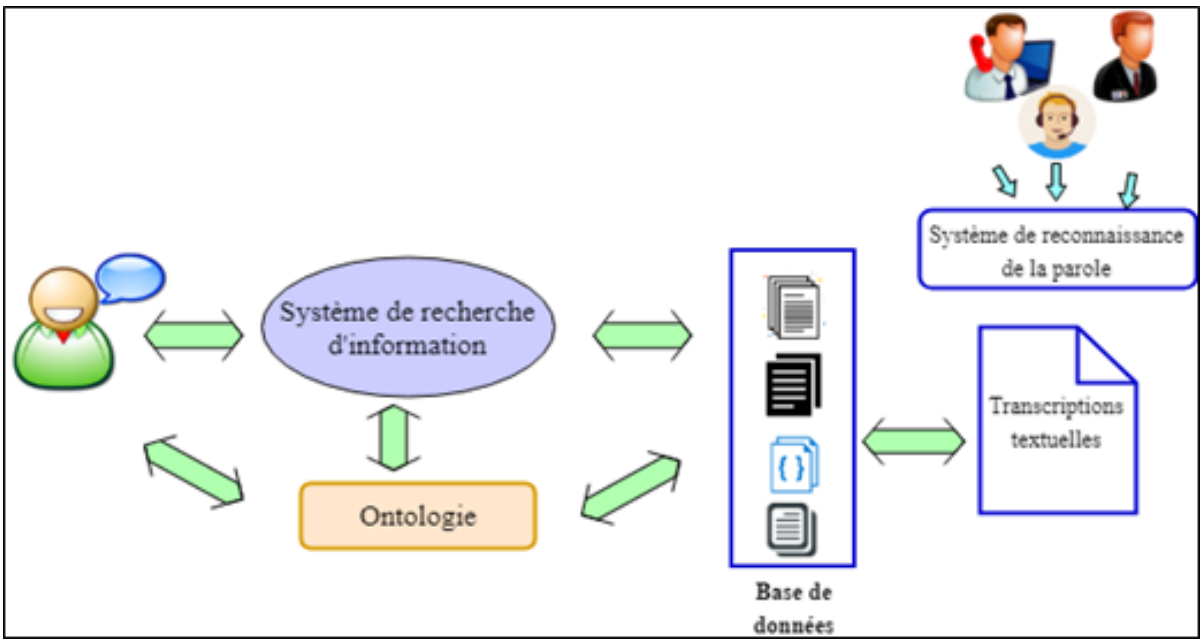


FIGURE 5.1 – Exemple de motivation

5.1.2 Challenges

Dans cette section, nous présenterons le modèle que nous avons proposé pour l'indexation des données en prenant en compte le traitement de l'incertitude ou la véracité des données. Dans cette section, nous nous concentrons sur la représentation et le traitement des incertitudes relatives à la valeur que peut prendre une variable. Cette incertitude peut provenir de la variabilité intrinsèque des phénomènes influençant la valeur de cette variable, de l'imprécision ou de la non-fiabilité des variables. Nous nous concentrons sur la phase d'indexation en présence de données incertaines. Pour atteindre notre objectif, nous devons répondre aux questions suivantes :

— ***Comment calculer l'incertitude des termes :*** la mise en place d'un nouveau modèle de représentation des termes demeure indispensable. Ce modèle doit être capable de déterminer le degré de certitude de chaque terme extrait d'un document dans le domaine de recherche d'information. Il sera articulé autour d'un module

basé sur la théorie des probabilités où à chaque terme sera associé une valeur P qui exprime son degré d'exactitude au sein d'un document bien déterminé.

— ***Développer un module de calcul du degré d'incertitude dans la partie indexation syntaxique :*** une nouvelle méthode d'indexation syntaxique, sous la forme d'un module, doit être proposée pour faire face aux données incertaines. Cette méthode doit être capable de retourner l'ensemble des termes extraits d'un document en prenant en considération leurs valeurs de probabilité. Il ne s'agit pas de retourner ces valeurs d'incertitude mais plutôt de proposer une méthode qui permet de construire le document indexé dans un environnement incertain.

— ***Ajouter un module de calcul d'incertitude dans la partie indexation sémantique :*** Une nouvelle approche d'indexation sémantique incertaine doit être aussi proposée. Cette méthode nous permet de déterminer le degré d'incertitude des concepts extraits de l'ontologie, en greffant un module à base de logique probabiliste.

— ***Mettre en place une solution hybride pour l'indexation de données incertaines :*** une nouvelle approche d'indexation qui combine les deux approches précédentes incertitude et hétérogénéité des données, doit être proposée.

Dans ce qui suit, nous essaierons d'expliquer le principe de chaque phase dans notre modèle proposé.

5.1.3 Contributions

Dans ce chapitre, nous proposons une approche probabiliste pour la représentation des termes d'un document dans le domaine de la recherche d'information. Nous allons nous intéresser spécialement à la phase d'indexation des données au sein d'un environnement incertain. Ci-dessous, nous résumons nos principales contributions :

— **Approche de modélisation des termes contenus dans un document textuel :** Une méthode de représentation de termes incertains extraits d'un document qui soit basée soit sur une approche possibiliste, soit une approche probabiliste. Pour des raisons de simplicité, notre méthode est basée sur la théorie de probabilité ou à chaque terme t, appartenant à un document D, est associé un degré de probabilité. Ce degré exprime à quel ce terme indexe le document D en question.

— **Approche d'indexation syntaxique des données incertaine :** Pour traiter les données incertaines au niveau de domaine de recherche d'information, une nouvelle méthode d'indexation syntaxique a été proposée. Cette méthode prendre en considération les degrés de probabilité associés à chaque terme. Cette valeur doit être soit impliquer au niveau des phases de pondération des termes, soit utiliser dans une nouvelle équation.

— **Approche d'indexation sémantique de données incertaines :** Pour traiter les données incertaines d'un point de vue sémantique, nous devons intégrer un module de calcul d'incertitude dans la partie d'indexation sémantique. Après avoir extrait les termes des documents, nous calculerons les degrés d'incertitude à associer à chaque terme t_i relativement à un document D_j.

— **Approche hybride d'indexation de données incertaines :** Une troisième approche qui combine les deux précédentes sera développée et proposée. Cette approche permet de résoudre deux problèmes à la fois : le premier problème est l'incertitude de données en ajoutant un module pour la détermination d'incertitude des termes au niveau de l'indexation syntaxique et la détermination du degré d'incertitude de chaque concept au niveau de l'indexation sémantique. Le deuxième problème est l'hétérogénéité de données en combinant les deux approches ensemble.

Le reste de ce chapitre est organisé comme suit : d'abord, dans la section 2, nous décrierons quelques notions de base sur le processus d'analyse syntaxique. La section 3 est réservée à la présentation des différentes notations utilisées tout au long de ce chapitre. Dans la section 4 nous détaillerons le principe de la méthode que nous avons proposée pour le calcul des degrés d'incertitude associés aux termes contenus dans un document textuel. Dans la section 5, nous présenterons le principe de l'approche proposée pour l'indexation syntaxique de données dans un environnement incertain. Ensuite, dans la section 6, nous exposerons la nouvelle approche proposée d'indexation sémantique en présence de données incertaines également. La section 7 sera consacré à la présentation d'une description détaillée de l'approche hybride proposée. La huitième et dernière section, de ce chapitre, sera réservée à une conclusion sur les principaux points forts des approches proposées ainsi qu'aux limités soulignées.

5.2 Background

5.2.1 Étiqueteur syntaxique

Dans le modèle de « analyseur syntaxique » (Figure 5.2), chaque mot du document, fourni en entrée d'un processus d'analyseur syntaxique, correspond au même mot en sortie du processus associé à une étiquette syntaxique [Pal90][MK60]. Pour chaque mot, nous disposons de toutes les étiquettes syntaxiques possibles, chacune est caractérisée par une valeur de certitude.

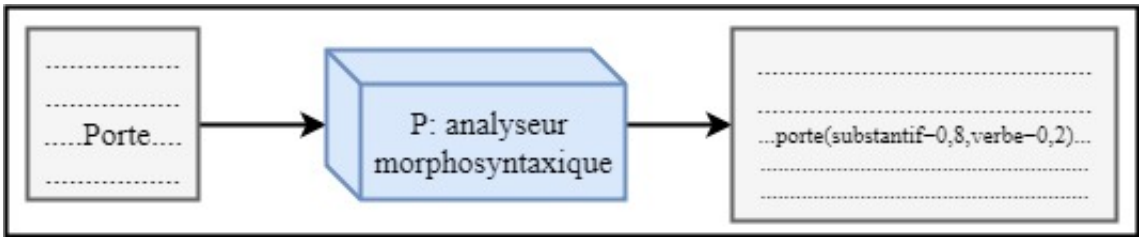


FIGURE 5.2 – Principe d'étiqueteur morphosyntaxique.

Le processus de la phase d'analyse syntaxique prend en entrée une donnée « simple », qui représente un terme extrait d'un document, et fournit, en sortie, un terme sous la forme d'un ensemble de couples $(Catégorie\_terme1 - poids\_Catégorie\_terme1, Catégorie\_terme2 - poids\_Catégorie\_terme2, \ldots, Catégorie\_termei - poids\_Catégorie\_termei)$. Dans l'exemple, en figure 5.2, l'analyseur donne comme sortie : $\acute{}porte(substantif - 0,8, verbe - 0,2)\check{}$. Cette valeur de sortie signifie qu'il existe une ambiguïté sur le mot extrait « porte » : ce terme est de type substantif avec une certitude de 80% ou de type verbe conjugué avec une certitude de 20%[DL96]. Dans ce contexte, les termes s'avèrent parfaitement identifiés. Par contre, il existe une incertitude au niveau de leur catégorie morphosyntaxique, le système d'étiquetage détermine une ou plusieurs catégories pour chaque terme. Ainsi, en sortie d'étiqueteur, nous obtenons un document sous la forme d'une liste de termes accompagnés par une ou plusieurs catégories et à chaque catégorie à est associé une valeur de probabilité pondérée [SB88]. Les expérimentations s'intéressent à la pondération de chaque type de données : tout document est une séquence de termes, dont chacun est caractérisé par une liste de catégories pondérées. Prenant l'exemple présenté par la figure 5.2, le terme "porte" est caractérisé par la liste suivante : $((Substantif, 0, 8), (VerbeConjugué, 0, 2))$. Dans un même document, le même terme peut apparaître plusieurs fois. Dans ce dernier cas, il existe un lien entre un terme et ses différentes listes pondérées de catégories. Par exemple, dans un même document, le mot "porte" peut se trouver présent 3 fois, 1 fois associé à la liste $(Substantif, 1)$, une fois à la liste $(VerbeConjugué, 1)$, et une fois à la

liste $((Substantif, 0, 8), (VerbeConjugué, 0, 2))$.

5.3 Notations

Dans cette section, nous présenterons les principales notations qui seront utilisées dans ce chapitre :

- D : Document
- C : Corpus
- N = nombre de documents dans le corpus
- tf (t,d) = term frequency = nombre d'occurrences du terme t dans le document d
- idf (t) = inverse document frequency = le nombre de documents indexés par le terme t dans une collection
- Cf : la fréquence de concept
- P : degré de probabilité

5.4 Calcul d'incertitude

La plupart des approches existantes se basent sur l'hypothèse que les mots extraits des documents ont été parfaitement identifiés ou reconnus. Par contre, dans certains cas, les mots extraits sont incertains, c'est-à-dire qu'ils ont été caractérisés par une identification incertaine [Bro+93]. De ce fait, la fonction d'égalité entre terme extrait du document et terme extrait de la requête n'est plus vraie. Dans ce cadre, nous allons nous concentrer sur la représentation et le traitement des incertitudes relatives à la valeur que peut prendre une variable. Cette incertitude pouvant provenir soit de la variabilité intrinsèque des phénomènes influençant la valeur de cette variable, soit de l'imprécision ou du manque de fiabilité des informations disponibles. Nous nous intéresserons à la phase d'indexation en présence de données incertaines. Pour atteindre notre objectif, la première étape consiste à procéder à une étude comparative sur les principales méthodes de calcul d'incertitude dans la littérature [BKN04]. Nous pouvons citer la théorie des probabilités, la théorie des possibilités, la logique floue, la théorie des possibilités, les P-boxes, et la théorie de Dempster-Shafer , ... Dans notre approche nous allons nous intéresser à l'incertitude au niveau de la langue basé sur le contexte de *« modèle de langue »*.

Une autre méthode de modélisation des documents basée sur une approche probabiliste se présente dans l'utilisation des modèles de langue. Ces modèles de langue fournissent une représentation d'un langage. Les modèles de recherche d'information utilisent, généralement

ces représentations de langues. Le principe de cette méthode de modélisation consiste à calculer la certitude de chaque terme ou phrase en se basant sur les outils et moyens offerts par la théorie de probabilité. Ce modèle de langue détermine une probabilité à chaque expression extraite d'un document. Il joue aussi un rôle central dans le domaine de la traduction automatique statistique [Bro+93]. Un modèle de langue se construit en se basant sur un le principe suivant : « une fonction de probabilité P qui attribue une probabilité $P(S)$ à un terme ou une séquence de termes S dans une langue » [BKN04][MK60] (Figure 5.3). La langue dans ce modèle représente un corpus de documents. Cette fonction de probabilité consiste à estimer la probabilité d'une séquence quelconque de termes dans la langue modélisée ou de façon plus générale, à estimer la probabilité de générer cette séquence de termes à partir du modèle de langue.

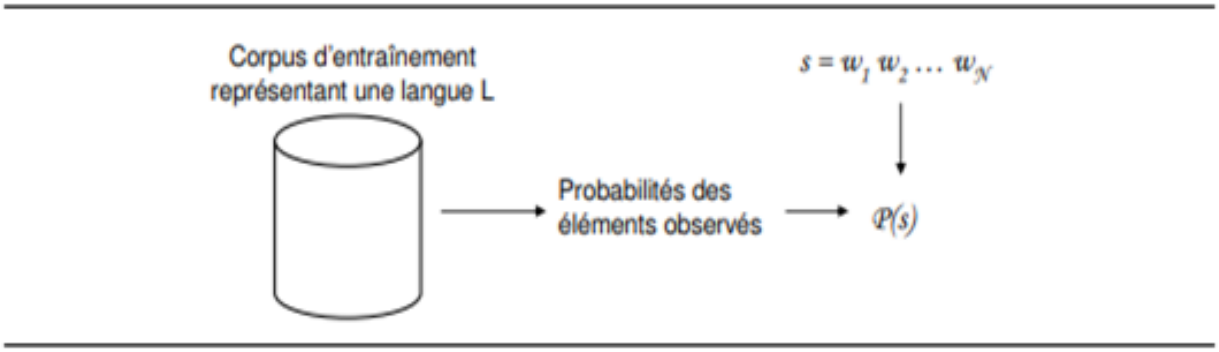


FIGURE 5.3 – Principe de fonctionnement des modèles de langue.

Pour calculer la probabilité d'appartenance à la langue L d'une phrase présentée par l'expression suivante : $Ph = m_1 \quad m_2 \ldots m_n$, nous devons utiliser l'équation suivante qui nous permet d'estimer la probabilité de la phrase d'être dans le modèle ML de la langue L [PC98] :

$$P(Ph) = \prod_{i=1}^{n} P_{ML}(M_i|M_1, ..., M_{i-1})$$

Nous utilisons un corpus de documents qui nous permet de représenter la langue à modéliser [MK60] [PC98] pour estimer la valeur de probabilité. Le calcul de la probabilité d'un terme m dans un corpus de document C est basé sur l'estimation de vraisemblance maximale du mot m et est donnée par :

$$P_{ML}(M) = \frac{|M|}{\sum_{M \in C}} = \frac{Nombre \quad d'occurence \quad du \quad terme \quad M \quad dans \quad le \quad corpus \quad C}{Nombre \quad de \quad n-gramme \quad de \quad C}$$

Par simplification d'écriture, nous remplaçons la notation $P_{ML}(t)$ par $P(t)$ en donnant au préalable la langue modélisée. Le principe général des modèles de langue est basé principalement sur deux grandes étapes : la première étape est l'apprentissage du modèle de langue.

La deuxième étape est l'évaluation de la probabilité d'appartenance d'un document à ce modèle (Figure 5.4). Ainsi, les paramètres du modèle de langue M_{Lx} s'estiment (2) en se basant sur les caractéristiques statistiques de langue extraites du corpus d'entraînement (1). A partir de ce modèle de langue MLx, la probabilité du document D d'appartenir à la langue X s'évalue par $P(D|M_{Lx})$ (3).

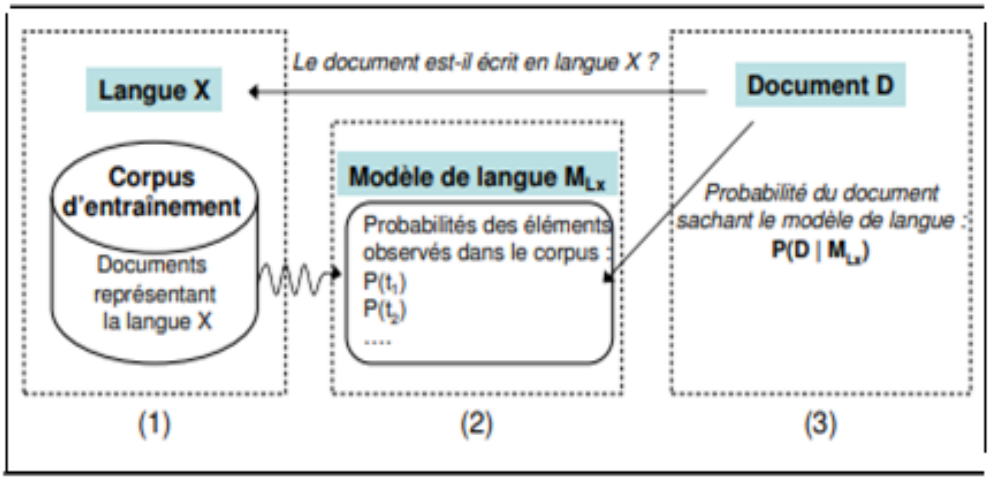


FIGURE 5.4 – Principe de fonctionnement des modèles de langue.

Pour le calcul de l'incertitude des mots extraits d'un document, nous allons nous baser sur le principe de modèle de langue de recherche d'information.

Calcul de certitude d'un terme Soit une valeur de certitude c associée à chaque terme $a \in V_c$. Nous représentons cette valeur par la fonction de certitude Fcert suivante :
$Fcert : V_c \rightarrow \Re^+$
$a \rightarrow P$

Pour mieux comprendre le principe, prenons comme exemple la phrase suivante dite à l'oral : *Ph = « Isabelle est à l'honneur. Cet honneur ... »*. Nous supposons qu'on a un processus d'extraction des données tel qu'un système de reconnaissance automatique de la parole qui fournit en sortie une phrase écrite dans un document. Le principe général de modèle que nous allons proposer est présenté par la figure suivante :

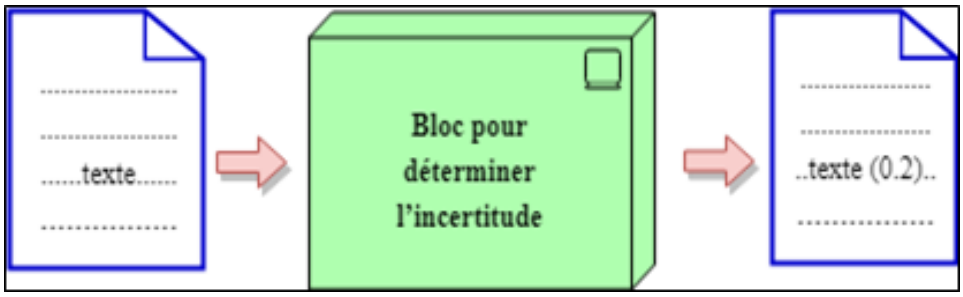


FIGURE 5.5 – Principe de calcul d'incertitude.

Après avoir pris l'entrée Ph, ce modèle nous donne comme sortie :
Psortie = « Isabelle est talonneur. Cet honneur . . . » Avec les indications suivantes :

- Cert (Isabelle) = 0.7
- Cert(est) = 0.2
- Cert(talonneur) = 0.5
- Cert(cet) = 0.18
- Cert(honneur) = 0.8

O.7, 0.2, 0.5, 0.18, 0.8, correspondent respectivement aux valeurs d'incertitude associées aux termes *isabelle, est, talonneur, cet, honneur*. Dans ce qui suit de ce chapitre, nous allons expliquer le principe de chaque approche d'indexation dans un environnement incertain.

5.5 Approche d'Indexation syntaxique de données incertaines proposée

Cette partie consiste à ajouter une étape de calcul d'incertitude dans la phase d'indexation syntaxique. Le principe de cette méthode est présenté par la figure suivante :

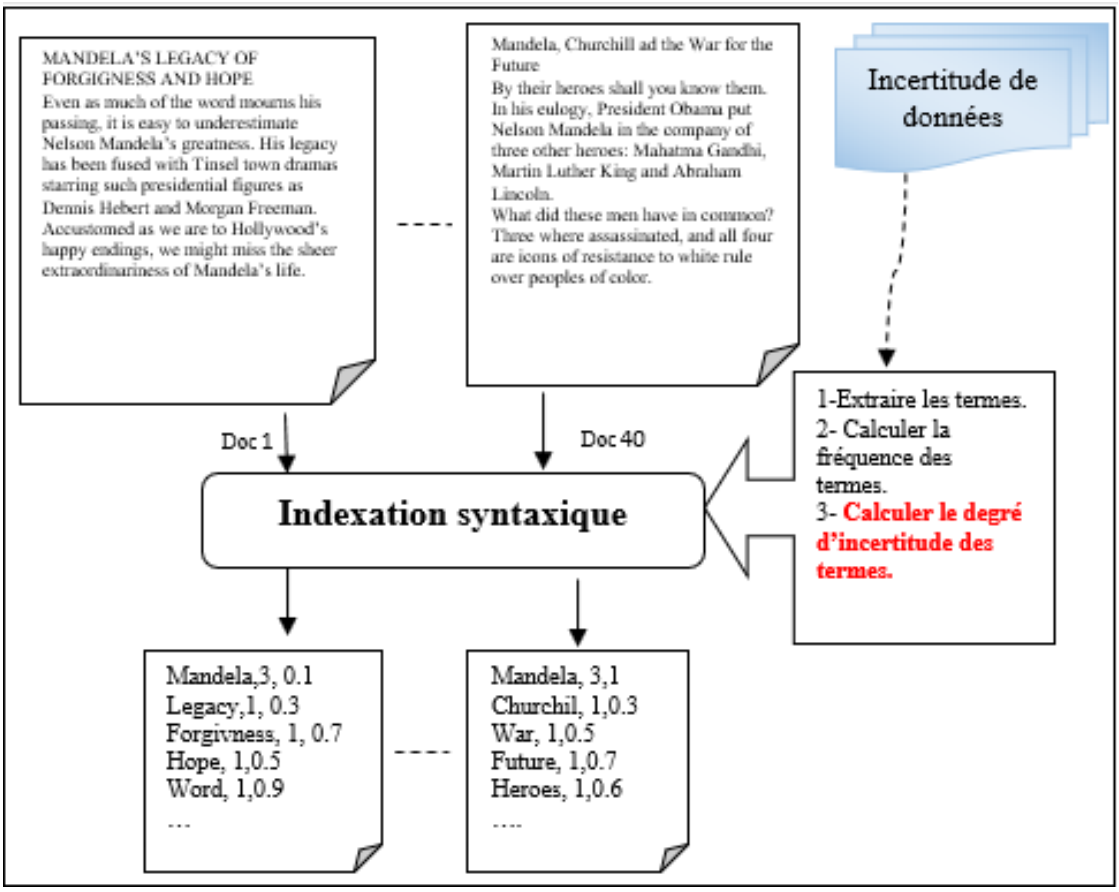


FIGURE 5.6 – Principe d'indexation syntaxique incertaine.

Cette approche est basée essentiellement sur le principe standard d'indexation qui prend comme entrée un ensemble de documents textuels et nous donne un ensemble de documents indexés. Après avoir extrait les termes des documents, nous calculerons le degré d'incertitude de ces termes en utilisant la méthode de calcul d'incertitude ($Incert_{ij}$:représente le degré d'incertitude de terme i dans un document j) expliqué dans la partie précédente. Cette méthode estime un degré d'incertitudes à tous les termes i dans un document j. Finalement, nous calculerons la fréquence de chaque terme. Puis nous devons soit concaténer la valeur d'incertitude de chaque terme et le poids de chaque terme soit trouver une relation entre le poids et le degré d'incertitude.

Algorithm 6 Indexation syntaxique probabiliste

Require: D : document.
C : Corpus.
Ensure: *D_Index_Syn_Pro* : Documents d'indexation syntaxique probabiliste.

```
1: for Chaque document D dans Corpus C do
2:    2. /*pour tous les documents D dans le corpus, D est le document courant*/
3:    Create(DPro)
4:    /* Création du document probabiliste*/
5:    for chaque ligne L dans D do
6:       */Pour toutes les lignes du document courant DPro, Ligne courant : L*/
7:       for Chaque mot w dans L do
8:          /* pour tous les mots d'une ligne L*/
9:          if fin_mot(w) == false then
10:            /* si le mot n'est pas égal à la fin_mot*/
11:            if  DPro.Contain(w) == false  then
12:               /*Le document probabiliste ne contient pas le mot w*/
13:               DPro.Add(W,Cert(t))
14:               /* ajouter le mot au document probabiliste avec une valeur d'incertitude de mot p = Cert(t)
    */
15:            else
16:               /* Le document probabiliste contient le mot w*/
17:               /* Mettre à jour la valeur de P*/
18:               DPro.UpdateP(P * w.P)
19:            end if
20:         end if
21:      end for
22:   end for
23: end for
24: for Chaque document DPro dans Corpus C do
25:    /*pour tous les documents DPro dans le corpus, DPro est le document courant*/
26:    Create(D_Ind_Synt_Pro)
27:    /*Créer (Indexation syntaxique probabiliste Ind_Synt_Pro du cahque document courant D*/
28:    for Chaque ligne L dans DPro do
29:       */Pour tous les lignes du document courant DPro, Ligne courant : L*/
30:       for chaque mot w dans L do
31:          /* pour tous les mots d'une ligne L*/
32:          if fin_mot(w) == false then
33:             /* si le mot n'est pas égal à la fin_mot*/
34:             if D_index_syn.Pro.Contain(w) == false  then
35:                /*Le document d'indexation syntaxique probabiliste ne contient pas le mot
36:
37:                                    w*/
38:                D.Index_Syn_Pro.Add(W, 1, 1)
39:                /* ajouter le mot au document d'indexation syntaxique probabiliste avec
40:
41:                                          une fréquence de mot tf =1 une incertitude Incert =1*/
42:             else
```

43: /*Index_Syn contain the word : increment tf*/;
44: /* Le document d'indexation syntaxique probabiliste contient le mot w*/
45: /* incrémenter tf*/
46: D. $Index\_Syn\_Pro.IncreTF(w.Tf)$
47: /*Extraire la valeur de probabilité de chaque terme Incert*/
48: ./* Modifier la valeur d'incertitude */
49: $D.Index\_Syn\_Pro.UpdateP(P * Incert)$
50: **end if**
51: **end if**
52: **end for**
53: **end for**
54: **end for**

L'algorithme 1 décrit le principe d'indexation des documents textuels en présence de données incertaines en se basant sur une approche probabiliste. Il permet de retourner efficacement l'ensemble des mots présentés dans un document en calculant leurs degrés de probabilité qui détermine leurs certitudes. L'objectif de cette indexation syntaxique probabiliste est d'extraire les termes des documents et de les stocker avec leurs nombres d'occurrences et leurs degrés de certitude dans un index physique. Cet algorithme est divisé en 3 parties : (1) la première partie (lignes 1-23) consiste à construire l'ensemble des documents probabilistes. Cette phase prend comme entrée un ensemble de documents du corpus, nous parcourons tous les documents de corpus. Pour chaque document, nous devons parcourir toutes les lignes de chaque document et pour chaque ligne, nous devons parcourir tous ses termes. Pour chaque terme nous devons déterminer le degré de probabilité qui lui est associé. Nous obtenons comme résultat un ensemble de documents ou chaque terme est caractérisé par son degré de certitude. La deuxième phase (lignes 24-41) consiste à construire d'index, ensuite extraire les termes signifiant du document sauf le terme qui indique la fin. La troisième phase (lignes 42-54) consiste à calculer l'occurrence de chaque terme et le degré de probabilité de chaque terme. Pour chaque terme signifiant, la fréquence d'occurrence doit être calculée et la probabilité doit être déterminée. A chaque fois qu'un terme est trouvé, nous incrémentons sa fréquence et nous calculons le degré de probabilité qui lui est associé. Le résultat de cette phase est stocké dans un fichier $´index - Syn_{P}ro˘$ pour chaque document.

Pour chaque document, nous obtenons tous les termes extraits avec leurs valeurs de fréquence et leurs degrés de probabilité :

$$Document\_ID \Rightarrow \{(term_1, tf_1, P_1), ..., (term_n, tf_n, P_n)\}$$

Avec

— n : nombre des mots signifiants de document D.
— P : probabilité de chaque terme.
— Tf : fréquence d'occurrence de chaque terme.

5.6 Approche d'indexation sémantique de données incertaines proposée

Dans cette section, nous allons nous intéresser à l'indexation sémantique dans un environnement incertain. Le module à mettre en place, sous forme d'un algorithme, prend en entrée la sortie de l'algorithme d'indexation proposé dans la section précédente à savoir un ensemble de documents indexés. Les termes de ces documents sont accompagnés par leurs degrés de probabilité et leurs fréquences d'occurrence. Dans cette approche, nous proposons d'ajouter une phase de calcul d'incertitude de données dans un contexte sémantique dont le principe est présenté dans la figure 5.7 :

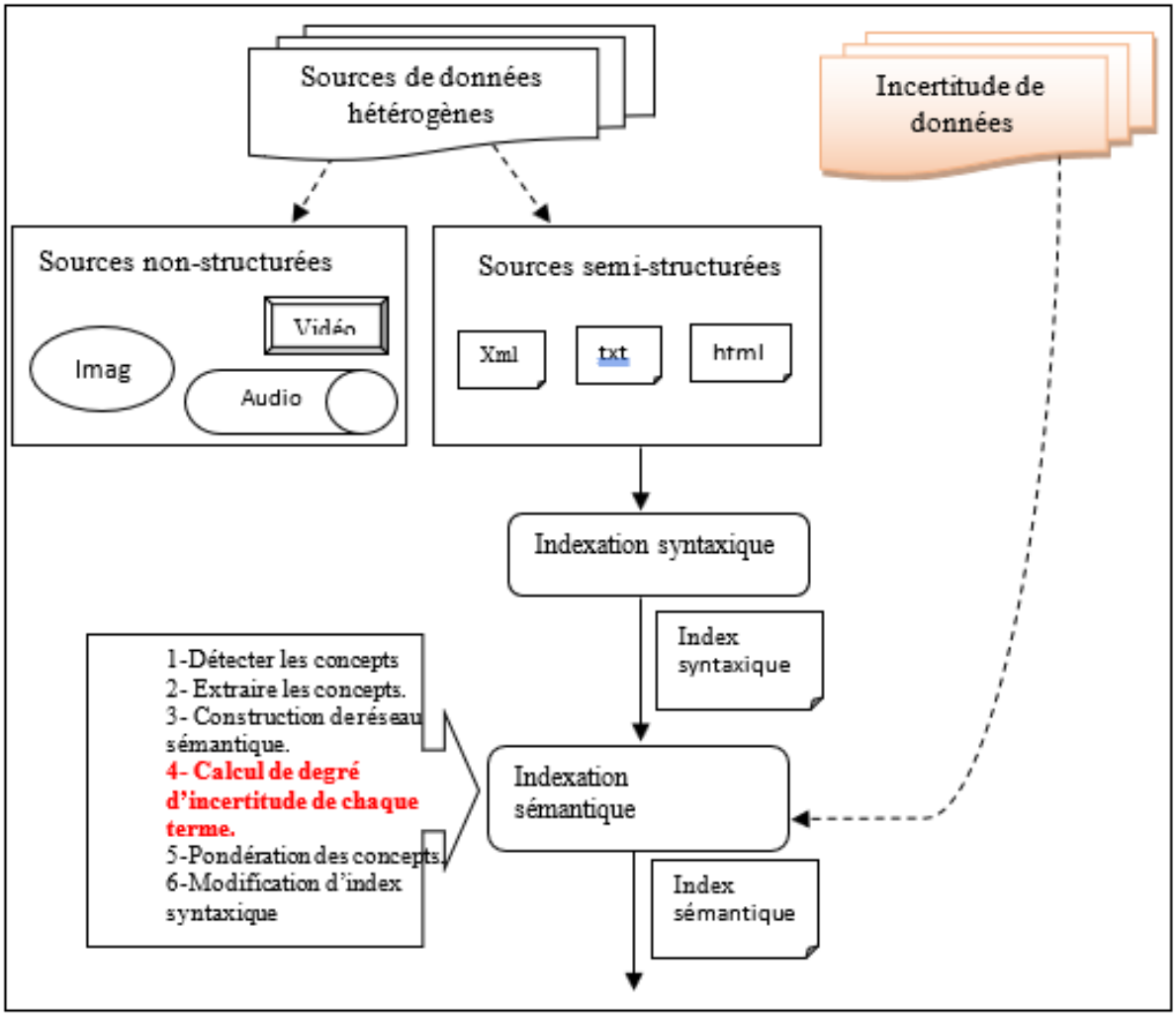


FIGURE 5.7 – Principe d'indexation sémantique incertaine.

L'algorithme d'indexation sémantique, admet comme entrée le résultat issu de l'algorithme de l'indexation syntaxique comme le montre la figure 5.7. Le processus de l'indexation sémantique probabiliste est articulé autour des étapes suivantes : (1) la détection des concepts, un terme est un concept s'il appartient à au moins une entrée dans l'ontologie. La détection de concept se fait en traitant les termes un par un et en les projetant sur l'ontologie. (2) L'étape suivante consiste à considérer chaque concept et détecter ses liens avec les différents concepts de l'ontologie. L'ensemble de ces concepts sont ensuite étendus par leurs synonymes et hyponymes existante au sein du même document. Ainsi, une liste de synonymes et de sens sera associée à chaque concept pour former un réseau montrant les liens et les relations entre ces différents concepts. Cette étape est la projection de l'ontologie sur le document. (3) Etape de construction du réseau sémantique du terme : Dans les deux étapes précédentes, nous avons utilisé l'ontologie pour représenter le contenu des

documents sous forme des concepts et des relations entre ces concepts. Puis, nous devons récupérer les valeurs d'incertitude de chaque terme. Ces valeurs sont présentées dans l'ensembles des paramètres d'entrées. Pour chaque terme du document, nous devons extraire la valeur d'incertitude. Cette valeur va être utilisé pour calculer le poids de chaque terme. L'étape suivante est la pondération des concepts : il existe plusieurs approches pour pondérer les termes importants d'un document ou d'une requête. Beaucoup d'entre eux sont basés sur les facteurs tf (fréquence de terme) et idf (fréquence de document inverse) qui permettent la prise en compte des poids locaux et globaux d'un terme. La mesure de $tf*idf$ permet de rapprocher la représentativité d'un terme dans un document.

Dans notre approche, nous allons utiliser une variante de cette mesure, qui est le $idf*CF$ où CF n'est pas la fréquence de terme mais la fréquence de concept. Pour calculer cette valeur, il faut ajouter le tf de terme initial à tous les tf des termes qui ont été associés dans le même document dans la phase précédente à savoir le tf des termes de réseau sémantique :

$$CF = Tf(t) + \sum Tf(réseau \ \ sémantique)$$

Avec

— t : terme actuel

L'étape suivante consiste à calculer la valeur d'*idf*. Dans cette approche, nous distinguons la fréquence d'occurrence d'un terme dans un document tf déjà calculé dans la phase d'indexation syntaxique et *la fréquence d'occurrence pour le même terme dans toute la collection (idf).*

$$idf = \log \frac{Document \ \ total}{Nombre \ \ de \ \ document \ \ avec \ \ ceterme}$$

Avant de calculer le produit de $CF * idf$, nous allons ajouter une phase qui permet de calculer le degré d'incertitude d'un terme dans un document. Nous notons par $Incert_{ij}$ le degré d'incertitude qu'un terme i représente un document j. Cette valeur sera, ensuite, ajoutée au produit de $CF \ \ et \ \ idf$. La phase finale consiste à modifier l'indexation syntaxique en prenant en compte les concepts avec une entrée de l'ontologie. Les termes qui appartiennent à leur réseau seront prises comme étant des concepts avec le même poids. Dans cette phase nous enrichissons l'index syntaxique avec les termes du réseau crée et nous supprimons les termes qui ne sont pas des concepts. Pour plus de détails, nous avons proposé l'algorithme 2 d'indexation sémantique probabiliste.

Algorithm 7 Indexation sémantique probabiliste

Require: *Index_Syn_Pro* : index probabiliste
Ensure: *Index_Sem_Pro* : index sémantique probabiliste

```
1: for Chaque Index_Syn_Pro do
2:    /* Pour tous les index syntaxiques des documents Index_Syn_Pro */
3:    for chaque terme t dans Index_Syn_Pro do
4:       /* Pour tous les termes t d'Index_Syn_Pro*/
5:       if t.EstConcept() then
6:          /*treme t correspond à une entrée de l'ontologie*/
7:          /*Projection dans l'ontologie*/
8:          Expansion_with_ontology();
9:          /*Détecter les liens entre les concepts et les étendre*/
10:         Construction_semantic_network();
11:         * Concepts et termes sont associées entre eux*/
12:         /*Récupérer la valeur de probabilité P de chaque terme t de Index_syn_Pro*/
13:         P= t.RecupereT();
14:         Ponderation_Concept();
15:         /*concepts are weighted according to Cf * idf * P*/;
16:      else
17:         /* terme t ignoré */
18:      end if
19:      Update_Index_Syn_Pro();
20:      * Modifier Index_Syn_Pro càd Seulement les concepts sont considérés*/
21:   end for
22: end for
```

L'algorithme 2 permet d'indexer efficacement un ensemble de documents en présence de données incertaines. Dans ce qui suit, nous décrivons les principales étapes de notre algorithme.

— Étape 1 : Détection des concepts (lignes 1-6) : Dans cette phase, nous devons tout d'abord extraire les termes d'index syntaxique probabiliste. Puis, nous devons faire une projection de ces termes sur l'ontologie. Cette projection nous permet de détecter les concepts. Par exemple : "président" est un concept car il correspond à une entrée dans l'ontologie qui est du type "personne". Cette étape est la projection du document sur l'ontologie. La détection de concept est faite en utilisant WordNet.

— Étape 2 : Expansion de concept en utilisant l'ontologie (lignes 7-13). Dans cette phase, nous devons détecter les liens entre les différents concepts et les relier en utilisant WordNet.

— Étape 3 : Construction de réseau sémantique (lignes 15-21). L'ensemble des termes $S_n = \{t_1...t_p\}$ forme un réseau avec un terme t d'un document doc, cela signifie que :

  — S_n est formé par l'union des éléments ei tels que ei sont des termes en relation avec le terme t, où la relation est = synonymie, dérivation, même famille, ...

— et $\forall t(ei) \in S_n, t(ei) \in doc$

— Étape 4 : Pondération des concepts : Nous devons calculer le poids de chaque concept. Ce poids est incertain puisque nous allons utiliser les valeurs de probabilité que nous avons déjà récupérées des paramètres d'entrées.

— Mise à jour d'index syntaxique probabiliste : Dans cette phase, nous enrichissons $l'Index\_Syn\_Pro$ avec les termes du réseau créé et nous supprimons les termes qui ne sont pas des concepts.Le résultat sera, un index sémantique ($Index\_Sem\_pro$) qui est composé par un ensemble de concepts.

5.7 Approche hybride d'indexation de données incertaines

Dans cette section, nous allons combiner les deux approches précédentes, à savoir l'approche d'indexation syntaxique incertaine et indexation sémantique incertaine, pour proposer une nouvelle approche hybride d'indexation probabiliste.

Le principe de cette approche est présenté par la figure 5.8. Nous remarquons que cette approche permet de traiter l'hétérogénéité et la quantité énorme de données et l'incertitude en utilisant deux types d'indexation : l'indexation syntaxique et l'indexation sémantique. Dans cette approche, nous allons hybrider les deux approches précédentes en ajoutant une phase qui permet de calculer le degré d'incertitude dans l'indexation syntaxique et une autre dans la partie d'indexation sémantique. Le principe de cette approche est décrit par la figure 5.8.

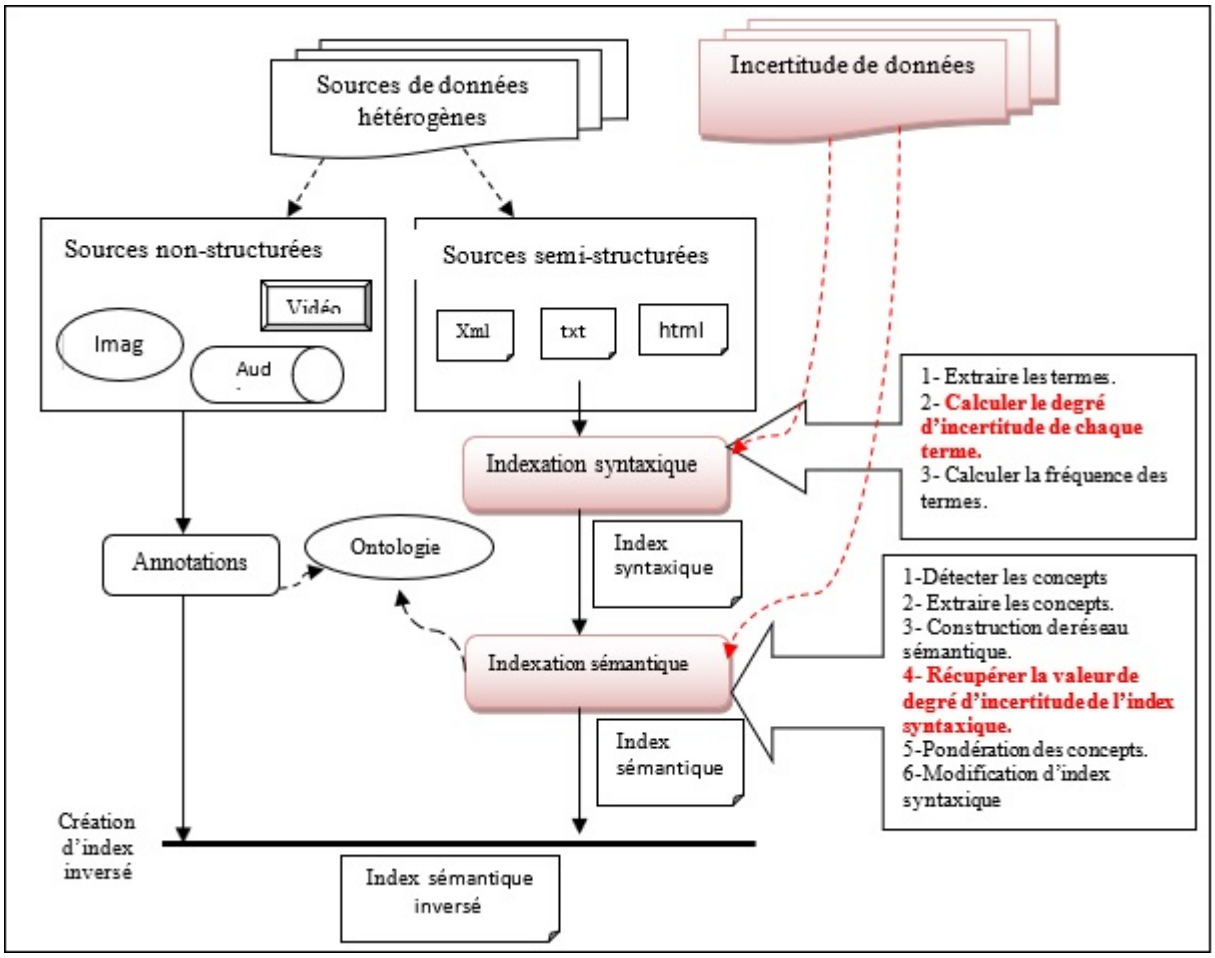


FIGURE 5.8 – Principe de l'approche d'indexation hybride dans un environnement incertain.

Cette approche est composée de deux parties : la première partie consiste à indexer syntaxiquement les documents textuels. Cette étape consiste : (1) à parcourir l'ensemble des documents, (2) à extraire les termes de chaque document, (3) à calculer le degré d'incertitude de chaque document, (4) à déterminer la fréquence d'occurrence de chaque terme et de construire un index physique qui contient l'ensemble des termes accompagnés par leurs valeurs de fréquence d'occurrence et leurs degrés de probabilité. La deuxième partie consiste à déterminer l'indexation sémantique de ces documents. Elle prend comme entrée le résultat obtenu de l'indexation sémantique. Un parcourir de ces documents est nécessaire pour extraire les termes, à procéder à une projection sur l'ontologie, à détecter les concepts, à récupérer la valeur de probabilité, à calculer le poids de chaque terme en fonction de sa valeur de probabilité et à modifier les documents obtenus par l'indexation syntaxique probabiliste.

Algorithm 8 Indexation hybride probabiliste

Require: D : document
C : corps
Ensure: $Index\_hybride\_Pro$: index hybride probabiliste
1: **for** Chaque document D_i dans le corpus C **do**
2: */* Création d'index syntaxique probabiliste*/*
3: $D\_Index\_Syn\_Pro_i = Index\_Syn\_Pro(D_i)$
4: **end for**
5: **for** Chaque document indexé syntaxiquement $D\_Index\_Syn\_Pro_i$ **do**
6: /* Pour tous les index syntaxiques probabiliste des documents de corpus C */
7: /* Création d'index sémantique probabiliste */
8: $D\_Index\_Sem\_Pro_i = Index\_Sem\_Pro(D\_Index\_Syn\_Pro_i)$
9: **end for**
10: Retourner les documents D indexés syntaxiquement et sémantique ment selon une approche probabiliste
11: Fin

L'algorithme 3 décrit le principe d'indexation hybride probabiliste. Il est composé principalement par deux grandes parties. La première partie décrit la phase d'indexation syntaxique probabiliste (lignes 1-3). Cette partie prend comme entrées un ensembles des documents textuels et un corpus C et consiste à construire des documents indexés syntaxiquement en se basant sur l'approche probabiliste décrite dans la section précédentes. La deuxième partie (ligne 4- 10) consiste à déterminer la phase d'indexation sémantique probabiliste. Elle prend comme entrées l'ensemble d'index syntaxique accompagnées par leur degré de probabilité et retourne des documents indexés d'une façon sémantique probabiliste.

5.8 Conclusion

Dans ce chapitre, nous avons proposé deux techniques d'indexation automatique, une première technique sémantique et une deuxième technique syntaxique sur des sources hétérogènes et dans un environnement incertain. L'approche que nous avons proposé permet d'indexer des sources semi-structurées et non structurées. L'objectif de l'intégration des techniques, pour diverses sources, est de gérer l'hétérogénéité et l'incertitude dans des sources des données de différents. En effet, nous avons essayé, également, de prendre en considération toutes les sources d'information du Web. Notre contribution se résume en trois algorithmes principaux : un premier algorithme s d'indexation syntaxique, un deuxième algorithme pour l'indexation sémantique et un troisième algorithme pour l'hybridation des deux premiers types d'indexation.

Dans un proche avenir, nous allons définir un modèle de traitement de données incertain

adapté au Big Data et mettre en œuvre notre approche. En conséquence, notre approche sera testée sur de vrais exemples et comparée les uns aux autres pour nous permettre de choisir l'approche la plus efficace qui sera comparée avec les travaux connexes. Enfin, des mesures de performance seront nécessaires pour évaluer la robustesse et l'efficacité de notre approche.

Chapitre 6

Etude expérimentale et analyse des résultats

Sommaire

6.1 Introduction

Dans la première partie de ce chapitre, nous allons présenter la mise en œuvre et l'étude des performances de l'approche que nous avons proposée pour la composition des services Web avec prise en compte de données incertaines. Nous présentons ainsi les systèmes et les architectures que nous avons considérées pour implémenter cette première approche. Nous présentons ensuite les expérimentations effectuées et nous analysons les résultats obtenus. Dans la deuxième partie de ce chapitre, nous mettrons en œuvre notre deuxième contribution. Nous présentons ainsi une étude de performance du cadre de composition de ressources Web incertaines que nous avons adopté et les performances de l'algorithme GET probabiliste que nous avons proposé. Dans la troisième partie, nous analysons la complexité des différents algorithmes proposés dans le cadre d'indexation de données incertaines.

6.2 Implémentation des services Web possibilistes

Dans cette section, nous présentons le système de composition de services Web. Ce système implémente respectivement les approches proposées dans le chapitre 3 (i.e., composition des services incertains). Le système a été développé sous l'environnement de développement *JavaEE* en utilisant comme éditeur *Eclipse version Oxygène*. Les services Web sont déployés dans le serveur d'application *Apache version 8.0.*

6.2.1 La mise en œuvre

Pour classer notre approche possibiliste de modélisation, d'invocation et de composition de services Web incertains, nous avons développé le système de composition de service Web [BGP92], [BBH12], en considérant une requête Q, formulée par rapport à une ontologie de domaine, et un ensemble de services Web dont la sémantique est modélisée en tant que vues RDF sur la même ontologie de domaine.

6.2.2 Architecture du système de mise en œuvre

L'architecture du système que nous avons mis en œuvre, pour interroger et composer des services Web incertains illustrée à la figure 6.1, est organisée en trois couches : la première couche est constituée d'un ensemble de bases de données Oracle / MySQL qui stockent les données. La deuxième couche comprend un ensemble d'applications propriétaires développées en Java ; chaque application accède aux bases de données à partir de la première couche (c.-à-d., elle exécute des requêtes de base de données). Ces applications propriétaires sont exportées en tant que services Web incertains vers le système. Nous avons utilisé le kit de déploiement fourni avec le serveur Web Apache pour créer et déployer nos services Web sur un ensemble de serveurs Web Apache fonctionnant sur un ensemble de machines PC (exécutant Windows XP). La troisième couche considérée comme couche supérieure comprend une interface utilisateur graphique (GUI) ainsi que notre système de composition. Les utilisateurs accèdent au système via une interface graphique. Ils peuvent soumettre des requêtes spécifiques ou paramétrées au système de composition.

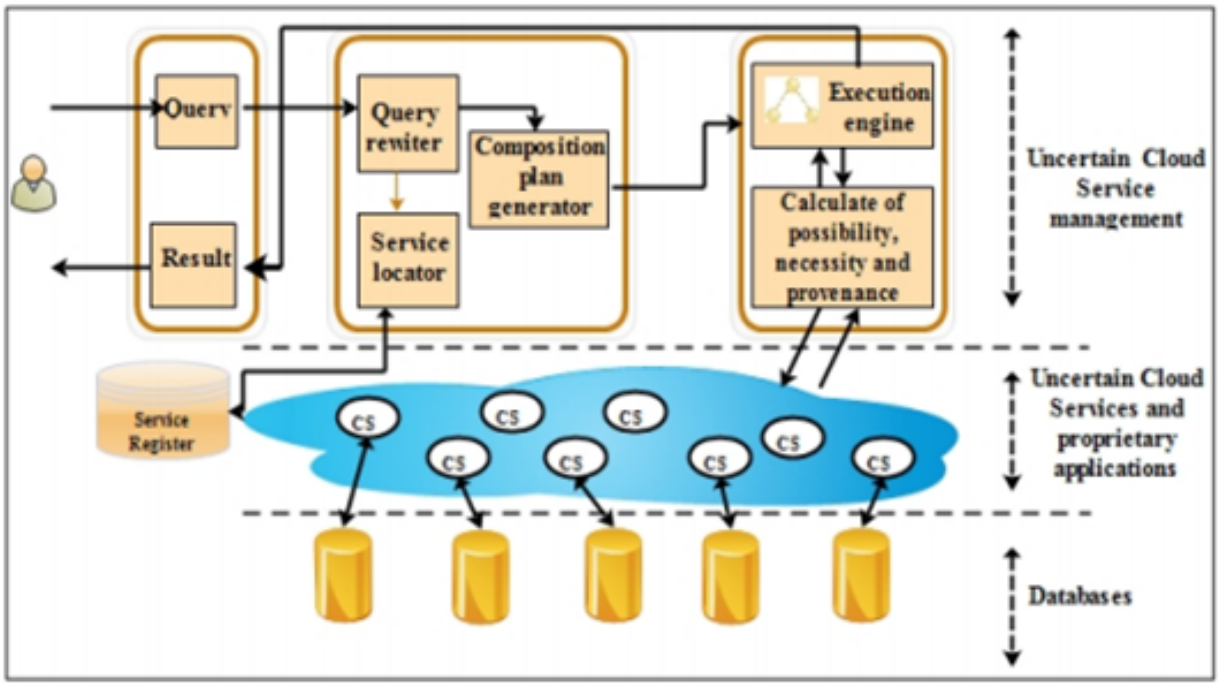


FIGURE 6.1 – Architecture du système.

Au premier niveau, le système sélectionne les services pertinents pour la requête Q, réécrit cette requête en fonction de ces services et exécute les requêtes modifiées pour renvoyer les résultats par rapport à la requête Q. Nous avons étendu ce système comme suit :

6.2.2.1 Description des services possibilistes

Pour répondre à notre modèle de service incertain, nous avons étendu la norme de service Web (WSDL, SOAP). WSDL est utilisé pour décrire l'interface d'un service Web. 2.0 alors que WSDL 2.0 fournit un modèle et un format XML pour décrire les services Web. WSDL 2.0 décrit un service Web en deux étapes fondamentales : la première étape est abstraite et deuxième étape est concrète. Il définit plusieurs éléments d'extensibilité qui peuvent être utilisés pour annoter les fichiers de descriptions de service avec des métadonnées et des informations sémantiques. Ces éléments peuvent être ajoutés aux attributs ou aux éléments XML de la description du service, l'exigence principale étant que les éléments d'extensibilité soient définis dans leur propre espace de noms. Nous avons utilisé ces éléments de WSDL2.0 et défini les quatre attributs suivants sur les éléments de message de sortie : (1) possibilité pour spécifier le degré de possibilité associé à chaque élément de sortie, (2) nécessité pour spécifier le degré d'incertitude associé à chaque élément de sortie, (3) provenance pour déterminer l'origine des attributs et (4) la clé pour spécifier qu'un paramètre de sortie représente l'identifiant, c.-à-d. une clé primaire..

FIGURE 6.2 – Exemple de partie d'un fichier WSDL étendu d'un service possibiliste.

6.2.2.2 Invocation des services Possibilistes

Le système de composition [BGP92], [BBH12], est basé sur une API Java standard pour l'invocation de services Web (Jax-ws.java.net). Il fournit un modèle de programmation pour produire (côté serveur) ou consommer (côté client) des services Web qui communiquent par type de message XML /SOAP. L'interface Dispatch (javax.xml.ws.Dispatch) prend en charge l'appel dynamique d'une opération de point de terminaison de service. C'est un client orienté messagerie XML destiné aux développeurs XML avancés qui préfèrent travailler au niveau XML en utilisant des constructions XML. Pour écrire un client Dispatch, vous devez maîtriser les API client Dispatch, les types d'objets pris en charge et la connaissance des représentations de messages pour le fichier WSDL (Web Services Descrip-

tion Language) associé. Il permet au développeur de construire les messages d'invocation manuellement en utilisant l'API XML souhaitée ou en utilisant l'architecture Java pour XMLBinding (JAXP jawp.java.net/) pour communiquer entre les messages XML et les objets Java internes qui constituent l'application SOA. Le type d'entrée Ip est Java Object (La possibilité, la nécessité et la provenance sont simplement les champs de la classe Java correspondante) et sont les arguments du processus d'invocation. Ensuite, le bloc Input Message Constructor construit le message d'appel XML d'entrée, ce processus peut-être exécuté manuellement ou automatiquement en utilisant respectivement des API XML et le mappage JAXB / Java / XML. Le message de résultat est ensuite encapsulé par une enveloppe SOAP et envoyé au service Web. Le fournisseur de service de bloc a renvoyé un message SOAP qui est désencapsulé pour extraire le message XML de sortie. Ce message retourné est alors lu et le résultat varie en fonction du mode de lecture. Si le message XML de sortie est lu manuellement par une API XML, le code peut alors lire les valeurs de possibilité, nécessité et provenance de l'attribut puis les définir dans la description du service WSDL, sinon, la possibilité, la nécessité et la provenance de l'attribut doivent être lié aux variables (possibilité, nécessité et provenance) de l'objet Java par le mappage JAXB Java / XML. Enfin, la possibilité, la nécessité et la provenance seront mises à jour en tenant compte de la caractéristique de l'entrée.

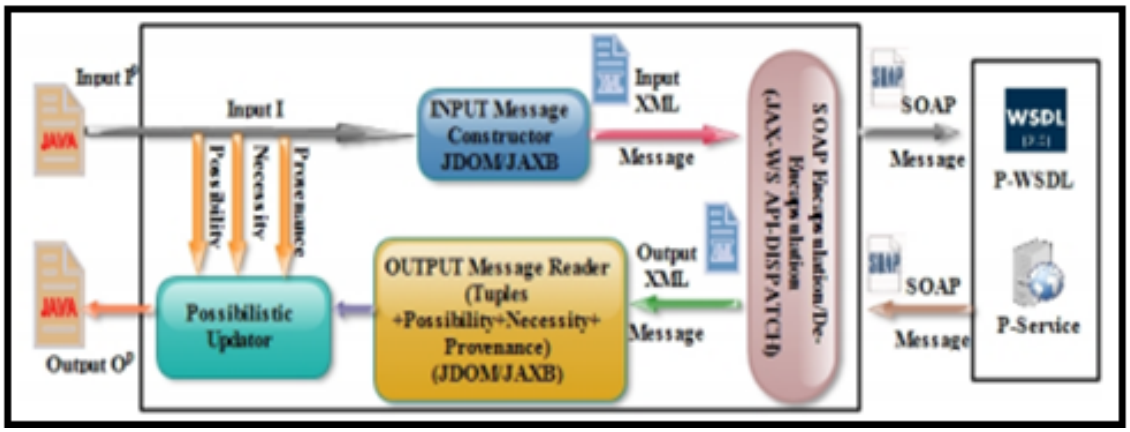


FIGURE 6.3 – Implémentation de l'Invocation Possibiliste.

6.2.2.3 Composition de services possibilistes

Nous avons implémenté les opérateurs de notre modèle de composition de services possibilistes et l'invocation de services possibilistes. La figure 6.4 suivante montre un exemple d'appel de service S_1. Nous notons que les valeurs de possibilité, de nécessité et de provenance sont calculées et renvoyées à l'utilisateur.

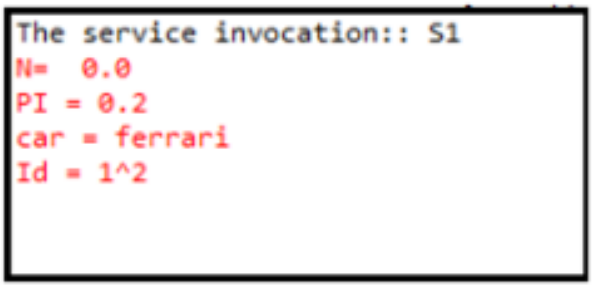


FIGURE 6.4 – Exemple d'Invocation de service : S1.

6.2.3 Évaluation expérimentale et analyse des résultats

Toutes les expériences sont effectuées sur un ordinateur fonctionnant sous Système Windows 7 et un processeur Intel Core i5 avec 6 Go de RAM. Afin d'analyser les résultats retournés par notre approche et étudier ses performances, nous avons considéré le temps d'exécution. Nous présenterons dans cette section quelques résultats et comparerons notre approche incertaine avec l'approche à sémantique certaine. Le but de cette expérience est de montrer que malgré la grande complexité de notre invocation incertaine et l'algorithme de composition de service, les requêtes peuvent encore être résolues dans un délai raisonnable. En ce sens, nous comparons les performances de l'algorithme de composition certain [BLP06] qui considère les services avec une sémantique certaine, avec notre proposition tout en augmentant le nombre de services de 1 à 40 et en variant le nombre de tuples par service de 1000 à 5000. Dans cette section, nous présentons les deux parties.

6.2.3.1 Effet de la variation du service Web

Dans cette section, nous évaluons l'effet de la variation du service Web sur le temps d'exécution. Dans cette série d'expériences, nous avons mesuré le temps d'exécution de la composition des services. Nous avons lancé les deux approches (avec et sans incertitude) et calculé le temps. Nous définissons la taille des données traitées par service à 100 Mo pour les deux approches. Cette expérience nous permet d'évaluer le coût encouru en calculant la possibilité, la nécessité et la provenance de notre algèbre de composition. Pour cette raison, nous avons mis en place 40 services Web sur une base de données incertaine, d'une taille de 100 Mo, stockant des données synthétiques sur les produits, sur les consommateurs et sur les commerciaux. Nous avons mesuré les temps d'exécution d'une composition avec et sans le calcul des degrés de possibilité, de nécessité et de provenance. Ces résultats de performance sont illustrés par la figure 6.5 :

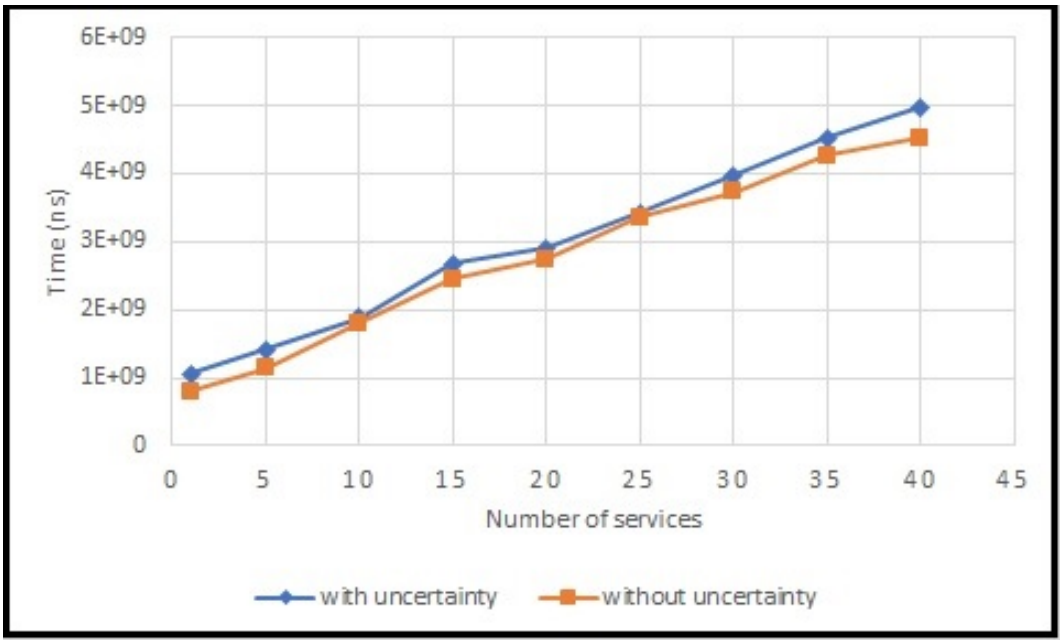


FIGURE 6.5 – Résultats de performance en termes de temps d'exécution avec et sans calcul de possibilité, de nécessité et de provenance.

La figure 6.5 montre les résultats du temps d'exécution nécessaire des deux approches en faisant varier le nombre de services impliqués pour répondre à une requête. Le résultat issu de cette expérimentation a montré que le temps pris pour calculer ces valeurs est

négligeable. Le temps d'exécution (temps de réponse à une requête), donné par notre approche, est de 1888732216 ns. Par exemple, lorsque le nombre de services est de 10, l'approche certaine donne un temps de réponse de 1810796017 ns. Dans tous les cas, le temps requis pour la composition des services Web de notre approche reste négligeable par rapport à l'approche 2 (voir Fig. 6.5). Nous pouvons déduire de ces résultats que notre approche donne de meilleurs résultats, car elle nous permet de gérer l'incertitude des données sans pour autant trop augmenter le temps de réponse.

6.2.3.2 Effet de la variation de la taille des données

Cette deuxième expérimentation consiste également à étudier le temps d'exécution consommé par les services Web en faisant varier la taille des données. Nous avons fixé le nombre de services utilisés pour répondre à une demande à 10 services en faisant varier la taille des données traitées par service pour l'approche possibiliste de composition de service et l'approche sémantique certaine de composition de services. La figure 6.6 résume les résultats obtenus par cette expérimentation dans laquelle nous avons varié la taille du corpus de 1000 Mo à 4000 Mo.

FIGURE 6.6 – Résultats de performance en termes de temps d'exécution avec variation du volume de données avec et sans calcul de possibilité, de nécessité et de provenance.

D'après les résultats illustrés par la figure 6.6, l'approche possibiliste donne un temps d'exécution de 2038630356 ns pour une taille de données de 2500 Mo, alors que pour l'approche certaine le temps d'exécution est de 2136551019 ns. Nous avons remarqué qu'il n y a pas une augmentation importante quant au temps d'exécution. Cette légère augmentation du temps d'exécution est considérée comme raisonnable étant donné que nous avons ajouté une phase possibiliste pour le calcul de l'incertitude.

6.3 Implémentation du Parser JSON probabiliste proposé

Dans cette section, nous proposons une implémentation pour l'analyseur JSON probabiliste que nous avons proposé ainsi qu'un exemple du code de notre analyseur. Comme présenté précédemment, puisque notre approche repose sur les principes d'XML probabiliste, nous sommes donc en mesure d'utiliser n'importe quels paramètres d'entrées de type JSON standard ou JSON probabiliste afin de valider les représentations des ressources incertaines que nous avons proposées. Pour développer une approche d'analyseur de JSON probabiliste, différentes techniques d'implémentation sont été utilisées telle que, node.Js et java script pour le développement. Pour implémenter ce Parser, nous devons vérifier la validité de tous les types, le début et la fin de chaque structure ainsi que la somme des valeurs de probabilité, ...

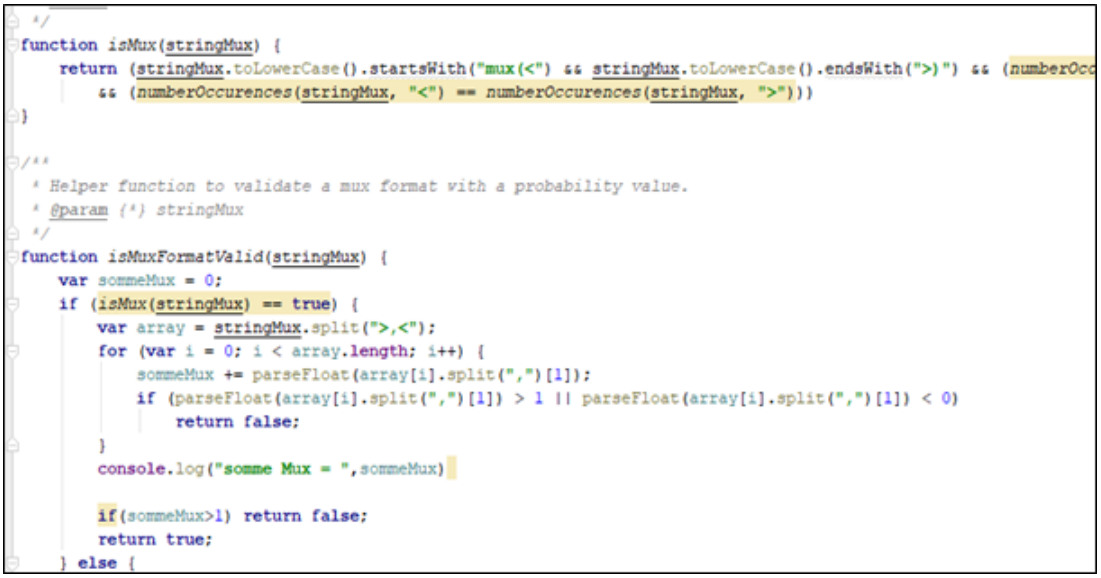
La figure 6.7 illustre deux exemples de code de Parser qui nous permettent de tester tous les types de données JSON. Le premier exemple nous présente les types JSON standard Tableaux.

```
function isArrayString(arrayStringValue) {
    try{
        return (arrayStringValue instanceof Array);
    }catch(exp){
        return false;
    }
}

function parse(StringJson) {
    var json = null;
    try {
        json = toEffectifJson(StringJson);
        for (var key in json) {
            if (isArrayString(json[key])) {
                if (json[key].length == 1) {
                    return parse(JSON.stringify(json[key][0]));
                } else {
                    var newArray = json[key].splice(0, 1);
                    return parse(JSON.stringify(json[key][0])) && parse(JSON.stringify(newArray));
                }
            } else if (typeof (json[key]) == "string") {
                if (isBooleanString(json[key]) || isNumberString(json[key])) {
                    //the value is number or boolan
```

FIGURE 6.7 – Premier exemple de code de Parser JSON probabiliste.

Le deuxième exemple, nous permet de tester la somme des valeurs de probabilité des nœuds de type « Mux ». Cette somme ne doit pas dépasser la valeur 1. Ce deuxième exemple nous permet aussi de tester le début, la fin et les séparateurs entre les données dans un type de donnes de type String probabiliste.



```
 */
function isMux(stringMux) {
    return (stringMux.toLowerCase().startsWith("mux(<") && stringMux.toLowerCase().endsWith(">)") && (numberOcc
        && (numberOccurences(stringMux, "<") == numberOccurences(stringMux, ">")))
}

/**
 * Helper function to validate a mux format with a probability value.
 * @param {*} stringMux
 */
function isMuxFormatValid(stringMux) {
    var sommeMux = 0;
    if (isMux(stringMux) == true) {
        var array = stringMux.split(">,<");
        for (var i = 0; i < array.length; i++) {
            sommeMux += parseFloat(array[i].split(",")[1]);
            if (parseFloat(array[i].split(",")[1]) > 1 || parseFloat(array[i].split(",")[1]) < 0)
                return false;
        }
        console.log("somme Mux = ",sommeMux)

        if(sommeMux>1) return false;
        return true;
    } else {
```

FIGURE 6.8 – Deuxième exemple de code de Parser de JSON probabiliste.

Pour évaluer et valider notre analyseur, nous avons implémenté tous les types que nous avons proposé à savoir les types standards et les types probabilistes. La figure 6.9 représente l'interface d'accueil de notre Parser JSON probabiliste dans lequel l'éditeur de texte nous permet de saisir le code à valider.

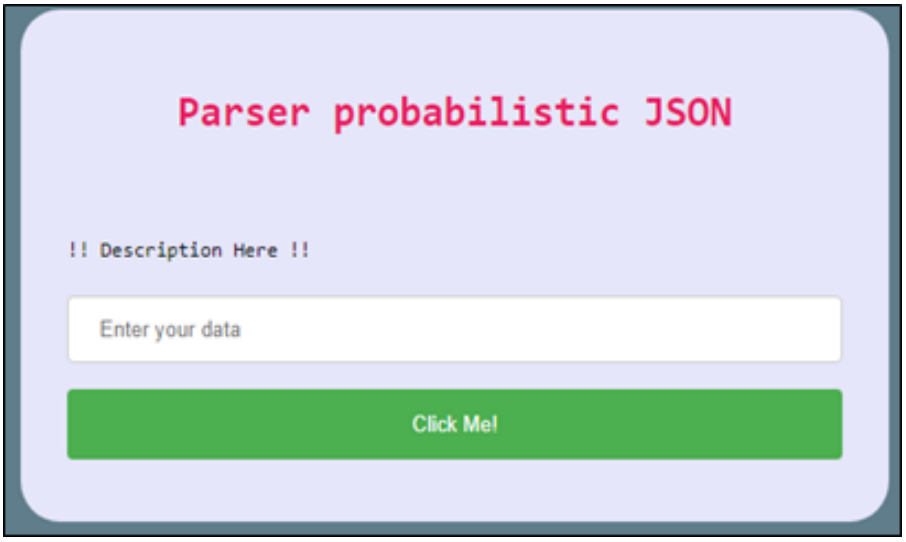


FIGURE 6.9 – Page d'accueil du Parser Probabiliste JSON.

Le bouton de confirmation « Click Me » permet à notre application de nous retourner soit une ressource validée, soit une ressource non validée. Dans le cas d'une ressource incertaine, l'application nous retourne la description des fautes telles que « Somme de probabilité supérieur à 1 », ... La figure 6.10 représente un exemple de réponse retournée par le Parser qui indique que cette ressource est non validée.

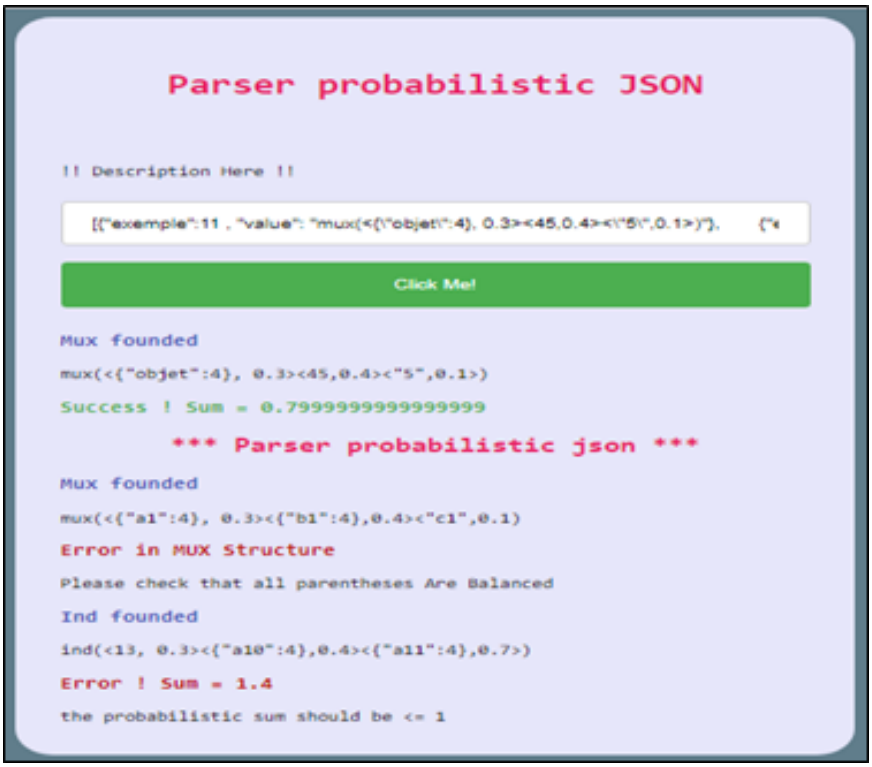


FIGURE 6.10 – Exemple d'une ressource non validée.

FIGURE 6.11 – Exemple d'une ressource validée.

La figure 6.11 représente un exemple d'une réponse indiquant une ressource validée.

6.4 Implémentation de GET probabiliste proposé

Nous proposons une implémentation pour l'algorithme GET probabiliste que nous avons proposé. Nous allons commencer par implémenter l'algorithme le plus simple à savoir l'algorithme 1 de GET. Comme présenté précédemment, notre approche repose sur les principes REST, nous sommes donc en mesure d'utiliser n'importe quel client HTTP pour accéder à nos ressources incertaines. Ce que nous proposons ici, est une implémentation de l'algorithme 1 de GET, qui va les requêtes HTTP nécessaires. Afin d'assurer la réutilisabilité de notre approche et permettre son intégration avec d'autres approches, nous avons implémenté les algorithmes GET probabiliste, en tant que services RESTful. Les appels de service sont effectués via POST et GET afin de récupérer une description conviviale du service. Notre implémentation de l'algorithme de calcul va transformer cette requête en sous la forme d'une liste de concepts à extraire de ressource en ressource, en créant notre chemin descendant à travers l'arbre de possibilité. Voici un exemple de requête http (voir Figure 6.12), utilisant la négociation de contenu, vers une ressource incertaine : la représentation incertaine d'une ressource :

FIGURE 6.12 – Exemple d'une réponse GET probabiliste.

Afin d'évaluer notre approche, nous nous concentrons sur le temps de traitement consommé par nos algorithmes. À cette fin, nous avons hébergé des services RESTful desservant des ressources Web incertaines. Cette représentation, de la ressource présentée par la figure 6.12, est accompagnée par un entête spécifique présenté par la figure 6.13 :

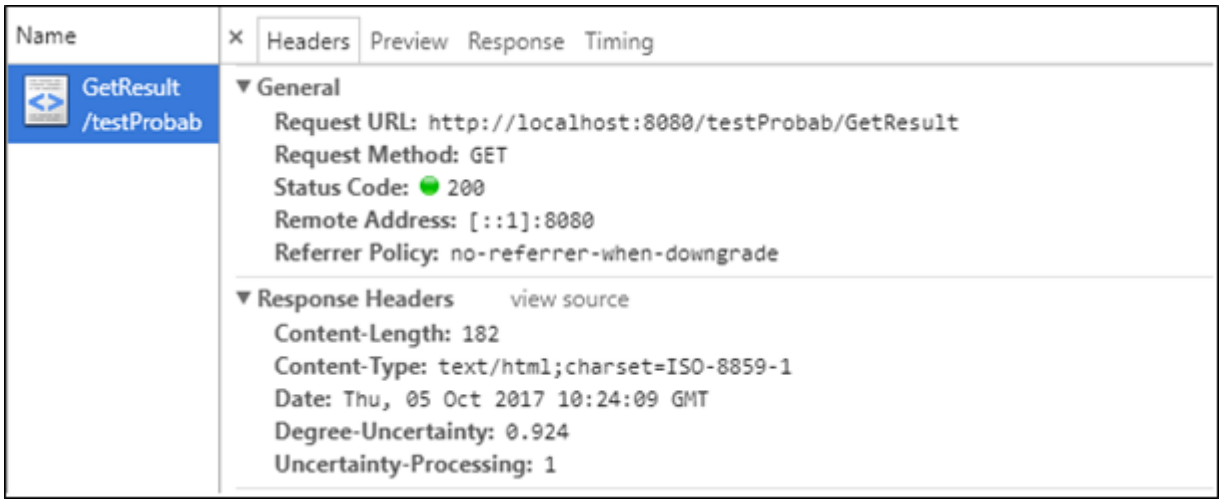


FIGURE 6.13 – Exemple d'un entête d'une réponse GET probabiliste.

Cette figure représente l'entête standard enrichi par les lignes proposées : « degree-Uncertainty :0.924 » qui représentent le degré de certitude de la réponse envoyée à l'utilisateur et « Uncertainty-Processing :1 » pour informer l'utilisateur que le traitement d'incertitude des ressources a été accepté.

Dans cette section, nous étudions l'effet de la variation du nombre des services sur le temps d'exécution. Dans cette série d'expérimentations, nous allons mesurer le temps d'exécution nécessaire pour la composition des services en implémentant les deux approches, avec et sans calcul de probabilité. Ces expérimentations nous permettent d'évaluer le coût encouru en calculant la probabilité de composition effectuée pour répondre à une requête utilisateur. Pour cette raison, nous avons mis en place 16 services Resfull sur une base de données incertaine, de taille 100 Mo, stockant des données synthétiques sur les étudiants, sur les enseignants et sur les universités. Nous avons mesuré les temps d'exécution nécessaire pour la composition avec et sans le calcul des degrés de probabilité. Ces résultats sont illustrés par la figure 6.14 :

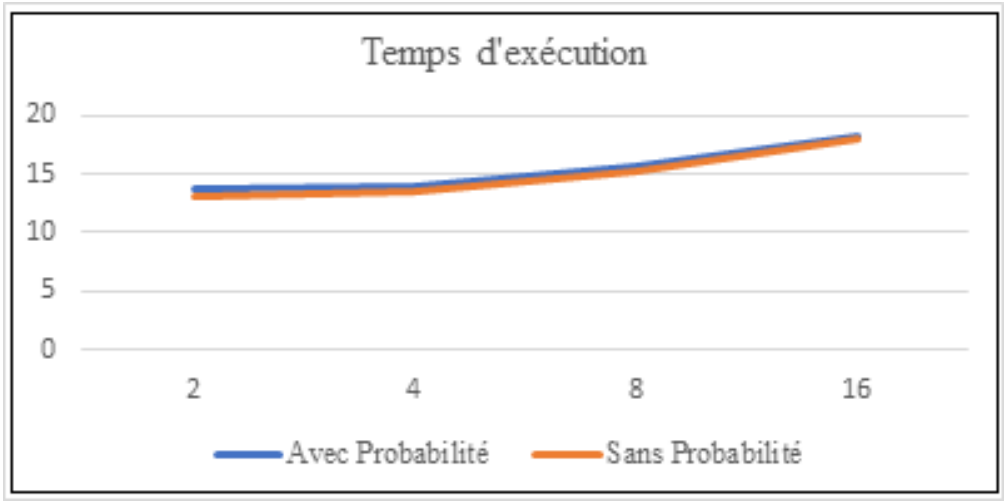


FIGURE 6.14 – Résultats de performance : en termes de temps d'exécution avec et sans calcul de probabilité.

La figure 6.14 montre les résultats en termes de temps d'exécution effectués par les deux approches en faisant varier le nombre de services impliqués pour répondre à une requête. Le résultat de cette expérimentation montre que le temps nécessaire pour retourner les réponses à une requête, estimé à 13.72 ms, est négligeable. Lorsque le nombre de services est de 2, le temps d'exécution nécessaire mis par l'approche certaine est de 13.04 ms. Dans tous les cas, le temps requis pour la composition des services de notre approche est considéré comme négligeable par rapport à l'approche 2 (voir Fig. 6.14). Nous pouvons déduire de ces résultats que notre approche donne de meilleurs résultats, puisqu'elle nous permet de gérer l'incertitude des données sans pour autant consommer davantage de temps.

L'algorithme GET probabiliste permet de parcourir plusieurs fichiers JSON afin de trouver la représentation d'une ressource demandée par l'utilisateur. La réponse à une requête permet, par exemple, de récupérer tous les animaux qui partagent les mêmes mots-clés comme représenté par la figure 6.15 :

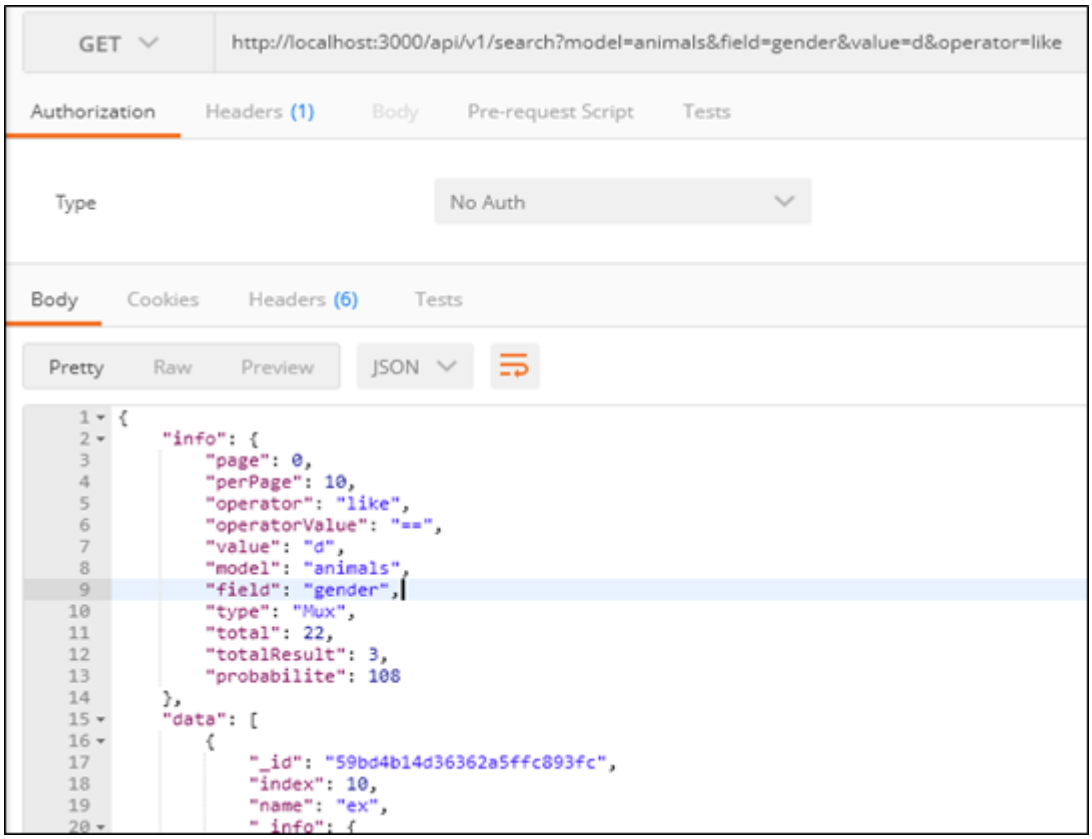


FIGURE 6.15 – Exemple d'une réponse d'algorithme 2 de GET probabiliste.

6.5 Etude et analyse de la complexité des algorithmes d'indexation

6.5.1 Algorithme d'indexation syntaxique probabiliste

Dans cette partie, nous étudions la complexité de l'algorithme d'indexation syntaxique probabiliste (cf. au chapitre 5) en analysant la complexité de ses différentes phases.

— Dans la première partie, chaque document textuel D est transformé en un document textuel probabiliste. Supposons que N est le nombre de document à indexer dans C (c.-à-d. une collection de document), $LMax$ est le nombre maximal de Lignes possibles par document et $MMax$ est le nombre maximal de mots possibles par ligne appartenant à un document D. La complexité de la première phase est donnée

par la quantité $O(N * LMax * MMax)$.

— Dans la deuxième partie, les document probabilistes *Dpro* sont déjà récupérés. Cette phase s'intéresse à la construction d'index, nous devons extraire les termes signifiant du document sauf le terme qui indique la fin de terme. Un parcourir de tous ces documents est nécessaire pour les indexés.

— La troisième phase, consiste à calculer l'occurrence et la probabilité de chaque terme. Pour chaque signifiant terme, la fréquence d'occurrence doit être calculée et la probabilité doit être déterminée. A chaque fois qu'un terme est trouvé, nous incrémentons sa fréquence ce, nous calculons ensuite le produit des probabilités. Nous supposons que nous avons un document indexé DI., chaque document contient $LDIMax$ lignes et ligne contient $MDIMax$ mots, alors la complexité de cette phase est donnée par $O(LDIMax * MDIMax)$.

En résumé, les différentes phases de notre algorithme ont une complexité polynomiale.

6.5.2 Complexité de l'algorithme d'indexation sémantique probabiliste

Dans cette partie, nous étudions la complexité de l'algorithme d'indexation sémantique probabiliste (cf. au chapitre 5) en analysant la complexité de ses différentes phases.

— Dans la première phase, nous nous intéressons à la détection des concepts. Nous devons, tout d'abord, extraire les termes d'index syntaxique probabiliste. Puis, faire une projection de ces termes sur l'ontologie. Cette projection nous permet de détecter les concepts. Supposons que N est le nombre de documents indexés syntaxiquement noté DIS. Chaque DIS est caractérisé par le nombres de lignes par document LDIS et par le nombre des mots par lignes MLDI. Soit M le nombre de concepts pour chaque ontologie et NO le nombre d'ontologie, alors la complexité de la première phase est $O(N * LDIS * MLDI + M * NO)$.

— Dans la deuxième phase, nous nous intéressons à l'expansion de concept en utilisant l'ontologie. Pour cela, nous devons détecter les liens entre les différents concepts et ensuite les relier entre eux. La complexité cette phase est $O(M * NO)$.

— La troisième phase consiste à procéder à, la construction du réseau sémantique.

L'ensemble des termes $Sn = t1...tp$ forme un réseau avec un terme t d'un document doc, cela signifie que : Sn est formé par l'union des éléments ei qui sont des termes en relation avec le terme t, où la complexité de cette phase est $O(P)$.

— Dans la quatrième étape nous nous intéressons à la pondération des concepts : en calculant le poids de chaque concept. Ce poids est incertain puisque nous allons utiliser les valeurs de probabilité que nous avons déjà récupérées des paramètres d'entrées. La probabilité de chaque composition est calculée en multipliant les probabilités des vues sémantiques impliquées dans son interprétation. Supposons que H est le nombre de compositions correctes et $VMax$ est le nombre maximal de vues sémantiques par interprétation, alors la complexité de cette phase est donnée par $O(H * VMax)$.

— Dans la cinquième et dernière phase, nous devons mettre à jour d'index syntaxique probabiliste : Le résultat sera, un index sémantique ($Index_Sem$) qui est composé comme suit : i, j : Pondération du concept ($Cf * idf$ déjà calculé). $k : 1...m/m$: la taille du réseau sémantique du concept.. La complexité de cette phase est$O(m*NO)$.

En résumé, les différentes phases de notre algorithme ont une complexité polynomiale.

6.6 Conclusion

Dans ce chapitre, nous avons brièvement présenté le système que nous avons mis en place pour évaluer notre approche de la composition des services de données dans un environnement incertain. Nous avons également effectué une analyse de performance sur un large ensemble de données afin d'évaluer l'efficacité de notre proposition. Et pour évaluer l'efficacité et la faisabilité de nos solutions proposées, nous avons réalisé un ensemble d'expérimentation sur un benchmark contenant un nombre important de jeux de données sous forme de services Web. Nous avons aussi présenté et analysé les résultats, obtenus par les algorithmes de GET probabiliste. Nous avons également présenté une implémentation de Parser JSON probabiliste. Une synthèse des résultats trouvés montre que notre système peut gérer des ressources Web incertaines en un temps raisonnable, même avec les calculs de degrés d'incertitude.

Chapitre 7

Conclusion générale

Sommaire

7.1 Synthèse

Ces dernières années ont connu un intérêt croissant pour le partage et l'ouverture de données au Web. En plus, de nombreuses sources de données sont alimentées chaque jour par les utilisateurs pour diffuser les connaissances et la culture ou pour organiser les intérêts et les passe-temps sur les plateformes destinées à cet effet. Cette nouvelle façon d'utiliser le Web, associée au nombre croissant d'API et de terminaux rendant ces données disponibles, via des formats lisibles par machine, a conduit les fournisseurs à essayer d'ajouter des valeurs à d'énormes quantités de données disponibles gratuitement. Ils collectent, organisent et réutilisent ces données dans des mashups et des services pour fournir des outils et des plateformes utiles. Ces nouvelles possibilités ont obligé les entreprises à adopter de nouvelles stratégies basées sur les données, ajoutant plus de structure et de sémantique pour tirer profit des données sur le Web en les combinant avec leurs données internes. Un grand nombre d'approches et de modèles a été proposé tel que le Web sémantique ou l'initiative de données liées pour relier les données, afin de combiner ces données et d'extraire des informations de grande valeur, en réponse à un besoin spécifique.

En raison de l'importance du service Web, de nombreuses entreprises ont investi massivement dans les technologies de services Web nous citons à titre indicatif, Microsoft .NET, Websphere d'IBM, J2EE de SUN. Ces efforts ont abouti à un nombre croissant de services Web déployés sur le Web. Par conséquent, améliorer les capacités des moteurs de recherche actuels avec des techniques efficaces de récupération et de sélection de services Web devient un problème important.

Dans cette thèse, nous avons abordé les problèmes d'incertitude des services Web, des ressources Web et leur composition. Aussi, nous sommes nous intéressées au problème d'incertitude au niveau d'indexation de données. Premièrement, nous avons proposé une approche pour répondre aux requêtes sur les services Web incertains. Notre approche nous permet de traiter l'incertitude qui peut-être associée aux données consultées des services en proposant une modélisation probabiliste pour les services.

Deuxièmement, nous avons proposé une approche pour la modélisation et la composi-

tion des ressources Web dans un environnement incertain. Cette proposition nous permet d'évaluer des requêtes au sein de ces ressources incertaines.

Et enfin, nous avons proposé une approche d'indexation de données en traitant le problème d'incertitude de données à indexer. Cette thèse couvre les différents aspects du problème ci-dessus, à savoir la modélisation des services de données incertains, la sélection des services, la composition, la modélisation des ressources Web incertaines, la composition, l'évaluation des ressources incertaines et l'indexation de données incertaines. Nous résumons ci-dessous nos contributions majeures :

— ***Un modèle possibiliste pour les services Web incertains :*** : Nous avons proposé une approche possibiliste pour modéliser l'incertitude des résultats renvoyés par un service Web incertain. Le modèle suppose qu'un service Web incertain a une certaine sémantique et un certain comportement. Seuls les résultats retournés sont incertains. Nous avons proposé un modèle d'invocation qui permet l'invocation de services de données avec une entrée certaine et incertaine.

  Dans le premier cas, le processus d'invocation récupère les degrés de possibilité, de nécessité et de provenance des sorties du service. Dans le second cas, le processus d'invocation calcule les degrés de l'incertitude des résultats renvoyés en fonction des possibilités, de la nécessité et de la provenance renvoyées par le service et par les degrés d'incertitude de l'entrée.

— ***Un modèle de composition pour les services de données incertains :*** Nous avons défini la sémantique d'une composition de service incertaine basée sur la théorie du monde possible [Bosc 2010]. Calculer les valeurs de possibilité, la nécessité et la provenance de la sortie d'une composition basée sur la théorie du monde possible est inefficace car le nombre de mondes possibles est exponentiel avec le nombre de tuples. Ainsi, nous avons opté pour une approche extensionnelle et nous avons proposé une algèbre de composition sensible aux valeurs de possibilités, de nécessité et de provenance pour calculer les degrés finals des sorties de composition. Ces valeurs sont importantes pour calculer les meilleurs résultats, évaluer la qualité des résultats, prendre les bonnes décisions, etc.

— ***Ensemble des opérations algébriques :*** Nous définissons un ensemble d'opérateurs de compositions nécessaires pour formuler les plans d'orchestration des compositions de services y compris les services Web possibilistes.

— ***La proposition de la notion de ressources Web probabilistes :*** Nous proposons un modèle incertain pour les ressources Web en nous basant sur la théorie de probabilité. Dans notre modèle, chaque ressource Web incertaine est caractérisée par un ensemble d'interprétations. Ces interprétations sont déterminées par l'explication des mondes possibles (Web possible). Chaque Web possible est caractérisé par une probabilité qui détermine son degré de confiance.

— ***La proposition d'un analyseur JSON incertain :*** Pour analyser le modèle de représentation de ressources Web probabilistes que nous avons proposé, nous avons implémenté un Parser probabiliste qui consiste à analyser les représentations des ressources incertaines. Cet analyseur est capable de parcourir les représentations écrites en JSON et de valider les ressources certaines et les ressources incertaines.

— ***L'interprétation des ressources Web incertaines :*** En se basant sur le concept des mondes possibles probabilistes, l'interprétation d'une composition existante est définie comme étant un ensemble d'interprétations possibles obtenues en évaluant la même composition de ressources incertaines. Il est clair que l'évaluation de l'interprétation d'une composition est intractable. Ceci s'explique par le fait que le nombre des algorithmes proposé en avance pour simplifier l'interprétation possible est exponentiel. Dans ce sens, nous proposons une théorème qui permet de calculer efficacement (i.e., dans un temps polynomial) la probabilité d'une interprétation possible. Pour évaluer cette incertitude dans un contexte d'une navigation hypertexte classique qui permet de combiner des données provenant de plusieurs ressources Web incertaines. Nous nous appuyons sur ces modèles (modèle d'XML probabiliste) pour développer un ensemble d'opérateurs et d'algorithmes spécifiques afin d'évaluer les requêtes de données incertaines.

— ***La proposition d'un algorithme d'évaluation des ressources incertaines :*** Nous proposons un algorithme pour évaluer une requête au sein des ressources Web incertaines. L'évaluation de la requête en général consiste à déterminer tous les Web

possibles, puis à évaluer la requête sur chaque Web possible et enfin à retourner le résultat final accompagné par son degré d'incertitude. Comme le nombre de Web possible peut-être exponentiel, dans ce cas l'évaluation des requêtes devient difficile voire impossible. Pour résoudre ce problème, nous proposons un ensemble d'algorithmes qui permet d'évaluer efficacement les requêtes utilisateurs dans un temps raisonnable et sans avoir besoin de déterminer l'ensemble des Webs possibles.

— ***Approche de modélisation des documents probabilistes :*** Nous proposons une méthode de représentation de mots incertains extraits d'un document. Cette méthode est basée sur une approche possibiliste où chaque mot t appartenant à un document D doit être caractérisé par un degré de certitude P (valeur de probabilité).

— ***Approche d'indexation syntaxique incertaine :*** Nous proposons une nouvelle méthode d'indexation syntaxique pour traiter les données incertaines. Cette méthode est basée sur la théorie de probabilité. Chaque terme est caractérisé par une valeur de probabilité. Cette valeur est impliquée aux phases de pondération des termes.

— ***Approche d'indexation sémantique incertaine :*** Nous proposons une méthode d'indexation sémantique pour le traitement des données incertaines. Cette méthode consiste à ajouter une phase de calcul d'incertitude des mots extraits des documents textuels. Après avoir extrait les termes des documents, nous calculerons le degré d'incertitude de ces termes en utilisant la méthode de calcul qui a toutes les incertitudes des termes i dans un document j.

— ***Approche hybride d'indexation incertaine :*** Nous proposons une approche qui combine les deux méthodes précédentes : indexation syntaxique et indexation sémantique. Cette approche permet de résoudre le problème d'incertitude de données et le problème d'hétérogénéité de données.

— ***Mise en œuvre et étude de performance :*** Nous avons présenté le système que nous avons mis en place pour évaluer notre approche de la composition des services Web, ressources Web dans l'incertitude. Nous avons également effectué une analyse de performance sur un large ensemble de données afin d'évaluer l'efficacité de notre

proposition. Les résultats ont montré que notre système peut gérer des centaines de services Web de données dans un délai raisonnable, même avec des calculs de possibilité, de nécessité et de provenance.

7.2 Travaux futurs

Cette thèse conduit à divers terrains fertiles pour de futures recherches. Nous identifions les directions principales suivantes pour les travaux futurs :

— ***Optimisation de la composition des services en fonction de des valeurs d'incertitude :*** Une composition de services de données peut accepter différents plans qui respectent tout d'abord son graphe de dépendance. Certains de ces plans calculent les degrés d'incertitude correctes alors que d'autres ne le font pas. Ces plans ont des coûts d'évaluation différents qui peuvent dépendre de l'ordre de leurs différents opérateurs (par exemple invocations, sélections, jointures, etc.) ainsi que des services (par exemple la sélection des services, c.-à-d. le nombre moyen de tuples de sortie et de tuples d'entrée, sa capacité à être invoqué avec des blocs de tuples, etc.).

Des efforts de recherche plus poussés sont nécessaires pour étudier le problème de l'inférence du meilleur plan de composition qui calcule toujours correctement les degrés de possibilité, de nécessité et de provenance des sorties. Dans certaines applications comme le classement des objets Web, le plus important est de classer efficacement les objets (en fonction de leurs valeurs) plutôt que de connaître leurs valeurs de possibilité, de nécessité et de provenance exactes. Par conséquent, un plan de composition complexe, mais efficace, qui calculerait ces degrés approximatives (mais suffisamment précises pour le classement) serait parfois préféré à un plan de composition sûr, mais inefficace. Par conséquent, davantage d'efforts de recherche sont nécessaires pour quantifier les limites d'erreur de ces valeurs qui pourraient être produites par un plan de composition complexe. Le même objectif de recherche est bénéfique pour les compositions dures (c.-à-d., les compositions qui n'acceptent pas un plan sûr).

— ***Le classement de services de données incertains :*** (ou leur composition) ren-

voie souvent à un nombre écrasant de résultats (par exemple, des tuples de données), conduisant ainsi les consommateurs de données à manquer ceux qui correspondent le mieux à leurs besoins. Les requêtes Top-k sont une approche courante pour rapporter les k meilleures réponses (d'une requête) basées sur la correspondance des tuples traités aux préférences des utilisateurs. Dans le contexte de services de données incertains, les tuples produits doivent être classés non seulement en fonction de leurs degrés correspondants avec les préférences des utilisateurs, mais aussi en fonction de leurs valeurs de possibilité, de nécessité et de provenance et de ces degrés des tuples intermédiaires corrélés. Les scores de tuple et l'incertitude interagissent pour décider des données sorties en top-k.

L'interaction entre l'incertitude des données et le «top-k» donne lieu à différentes interprétations possibles des requêtes top-k incertaines : (i) les tuples "top-k" dans le monde "le plus probable" ; (ii) les tuples «top-k les plus probables» qui appartiennent à des mondes possibles valides ; (iii) l'ensemble des tuples "top ith" les plus probables dans tous les mondes possibles, où i = 1 ... k, etc. Des efforts de recherche semblent nécessaires pour concevoir de nouvelles méthodes et techniques de classement efficaces qui mettent en œuvre ces interprétations et qui considèrent les valeurs d'incertitude (nécessité, possibilité, provenance) des données comme une dimension de classement importante. Des recherches sont également nécessaires pour optimiser l'exécution de compositions répondant à des requêtes top-k de manière à arrêter l'exécution de la composition dès que des réponses top-k sont produites.

— ***Algorithme GET général :*** Implémentation d'algorithme GET probabiliste en général proposé dans le chapitre 4. Nous devons faire une évaluation avec d'autres travaux connexes. Aussi, nous devons proposer des algorithmes ou des méthodes pour l'optimisation de cet algorithme. Cet algorithme exige la détermination de l'ensemble des mondes possibles et l'évaluation de la requête sur chacun de ces mondes possibles (Webs possibles). De cette façon, la complexité de cet algorithme est n-compet puisque le nombre de Web possibles est exponentiel et dans certains cas cet algorithme est intraitable.

— ***Implémentation des algorithmes d'indexation :*** Implémenter les algorithmes d'indexation proposées dans le chapitre 5 et les évaluer avec d'autres travaux dans

la littérature. Calculer le temps d'exécution de ces algorithmes et déterminer l'effet de calcul d'incertitude de données

Bibliographie

[Abi+09a] Serge Abiteboul et al. « Agrégation de documents XML probabilistes ». In : *Proc. BDA*. Conference without formal proceedings. Namur, Belgium, oct. 2009 (cf. p. 65).

[Abi+09b] Serge Abiteboul et al. « On the expressiveness of probabilistic XML models ». In : *VLDB J.* 18.5 (2009), p. 1041–1064 (cf. p. 63, 64).

[Abi+10] Serge Abiteboul et al. « Aggregate queries for discrete and continuous probabilistic XML ». In : *Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, March 23-25, 2010, Proceedings*. 2010, p. 50–61 (cf. p. 65).

[ADP95] A. M. Anile, S. Deodato et G. Privitera. « Implementing Fuzzy Arithmetic ». In : *Fuzzy Sets Syst.* 72.2 (juin 1995), p. 239–250 (cf. p. 19).

[AGN09] Zainab Assaghir, Philippe Girardin et Amedeo Napoli. « Fuzzy Logic Approach to Represent and Propagate Imprecision in Agri-Environmental Indicator Assessment ». In : *Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference, Lisbon, Portugal, July 20-24, 2009*. 2009, p. 707–712 (cf. p. 19, 24).

[Agr+10] Parag Agrawal et al. « Foundations of Uncertain-Data Integration ». In : *PVLDB* 3.1 (2010), p. 1080–1090 (cf. p. 37).

[AKG91] Serge Abiteboul, Paris C. Kanellakis et Gösta Grahne. « On the Representation and Querying of Sets of Possible Worlds ». In : *Theor. Comput. Sci.* 78.1 (1991), p. 158–187 (cf. p. 42).

[Alo+04] Gustavo Alonso et al. *Web Services : Concepts, Architectures and Applications*. Data-Centric Systems and Applications. Berlin : Springer, 2004 (cf. p. 2).

[Amd+14] Soumaya Amdouni et al. « Handling Uncertainty in Data Services Composition ». In : *IEEE International Conference on Services Computing, SCC 2014, Anchorage, AK, USA, June 27 - July 2, 2014*. 2014, p. 653–660 (cf. p. 22, 36).

[Aou05] Kamel Aouiche. « Techniques de fouille de données pour l'optimisation automatique des performances des entrepôts de données ». Thèse de doctorat dirigée par Zighed, Djamel Abdelkader et Darmont, Jérôme Informatique Lyon 2 2005. Thèse de doct. 2005, 1 vol. (240 p.) (Cf. p. 32).

[AS06] Serge Abiteboul et Pierre Senellart. « Querying and Updating Probabilistic Information in XML ». In : *Advances in Database Technology - EDBT 2006, 10th International Conference on Extending Database Technology, Munich, Germany, March 26-31, 2006, Proceedings*. 2006, p. 1059–1068 (cf. p. 65).

[AS13] Antoine Amarilli et Pierre Senellart. « On the Connections between Relational and XML Probabilistic Data Models ». In : *Big Data - 29th British National Conference on Databases, BNCOD 2013, Oxford, UK, July 8-10, 2013. Proceedings*. 2013, p. 121–134 (cf. p. 63).

[Ass+10] Zainab Assaghir et al. « Managing Information Fusion with Formal Concept Analysis ». In : *Modeling Decisions for Artificial Intelligence - 7th International Conference, MDAI 2010, Perpignan, France, October 27-29, 2010. Proceedings*. 2010, p. 104–115 (cf. p. 19).

[AY09] Charu C. Aggarwal et Philip S. Yu. « A Survey of Uncertain Data Algorithms and Applications ». In : *IEEE Trans. Knowl. Data Eng.* 21.5 (2009), p. 609–623 (cf. p. 22).

[Ba+14a] Mouhamadou Lamine Ba et al. « Integration of Web Sources Under Uncertainty and Dependencies Using Probabilistic XML ». In : *Database Systems for Advanced Applications - 19th International Conference, DASFAA 2014, International Workshops : BDMA, DaMEN, SIM - 3 - , UnCrowd ; Bali, Indonesia, April 21-24, 2014, Revised Selected Papers*. 2014, p. 360–375 (cf. p. 26).

[Ba+14b] Mouhamadou Lamine Ba et al. « Integration of Web Sources Under Uncertainty and Dependencies Using Probabilistic XML. » In : *DASFAA Workshops*. Sous la dir. de Wook-Shin Han et al. T. 8505. Lecture Notes in Computer Science. Springer, 2014, p. 360–375 (cf. p. 18).

[Bar13] Douglas K. Barry. *Web Services, Service-Oriented Architectures, and Cloud Computing, Second Edition : The Savvy Manager's Guide*. 2nd. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2013 (cf. p. 13).

[BBH12] Karim Benouaret, Djamal Benslimane et Allel HadjAli. « Selecting Skyline Web Services from Uncertain QoS ». In : *2012 IEEE Ninth International Conference on Services Computing, Honolulu, HI, USA, June 24-29, 2012*. 2012, p. 523–530 (cf. p. 24, 112, 115).

[BCP97] Marie-France Bruandet, Jean-Pierre Chevallet et François Paradis. « Construction de thesaurus dans le systeme de recherche d'information IOTA : application a l'extraction de la terminologie ». In : *1eres Journees Scientifiques et Techniques du Reseau Francophone de l'Ingerierie de la Langue de l'AUPELF-URF*. Avignon, France, 1997, p. 537–544 (cf. p. 90).

[BDS08] Djamal Benslimane, Schahram Dustdar et Amit P. Sheth. « Services Mashups : The New Generation of Web Applications ». In : *IEEE Internet Computing* 12.5 (2008), p. 13–15 (cf. p. 60).

[Ben+10] Michael Benedikt et al. « Probabilistic XML via Markov Chains ». In : *PVLDB* 3.1 (2010), p. 770–781 (cf. p. 63).

[Ben+16] Djamal Benslimane et al. « The Uncertain Web : Concepts, Challenges, and Current Solutions ». In : *ACM Trans. Internet Techn.* 16.1 (2016), 1 :1–1 :6 (cf. p. 36).

[BGP92] Daniel Barbará, Hector Garcia-Molina et Daryl Porter. « The Management of Probabilistic Data ». In : *IEEE Trans. Knowl. Data Eng.* 4.5 (1992), p. 487–502 (cf. p. 23, 112, 115).

[BHB04] D. Booth, H. Haas et A. Brown. *Web Services Glossary - W3C Working Group Note 11 February 2004*. Rapp. tech. World Wide Web Consortium (W3C), 2004 (cf. p. 12).

[Bis79] Joachim Biskup. « A Formal Approach to Null Values in Database Relations ». In : *Advances in Data Base Theory, Vol. 1, Based on the Proceedings of the Workshop on Formal Bases for Data Bases, December 12-14, 1979, Centre d'Etudes et de Recherches de l'Ecole Nationale Supérieure de l'Aéronautique et de l'Espace de Toulouse (CERT), France*. 1979, p. 299–341 (cf. p. 22).

[BKN04] Mohand Boughanem, Wessel Kraaij et Jian-Yun Nie. « Modèles de langue pour la recherche d'information ». In : *Les systèmes de recherche d'informations*. Sous la dir. de majid Ihadjadene. Lavoisier, 11, rue Lavoisier 75008 : Hermes-Lavoisier, 2004, p. 163–182 (cf. p. 96, 97).

[BKT01] Peter Buneman, Sanjeev Khanna et Wang Chiew Tan. « Why and Where : A Characterization of Data Provenance ». In : *Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings*. 2001, p. 316–330 (cf. p. 41).

[BLP06] Patrick Bosc, Nadia Liétard et Olivier Pivert. « About Inclusion-Based Generalized Yes/No Queries in a Possibilistic Database Context ». In : *Foundations of Intelligent Systems, 16th International Symposium, ISMIS 2006, Bari, Italy, September 27-29, 2006, Proceedings*. 2006, p. 284–289 (cf. p. 23, 117).

[Bon+] S. Bonomi et al. *SWSCE – An Automatic Semantic Web Service Composition Engine* (cf. p. 18).

[Bon04] Robert Boncella. « Web Services and Web Services Security. » In : *AMCIS*. Association for Information Systems, 2004, p. 565 (cf. p. 14).

[Bor60] A. A. Borovkov. « Limit Theorems on the Distributions of Maxima of Sums of Bounded Lattice Random Variables. I ». In : *Theory of Probability & Its Applications* 5.2 (1960), p. 125–155. eprint : `https://doi.org/10.1137/1105014` (cf. p. 21).

[Bou09] Kamel Boukhalfa. « De la conception physique aux outils d'administration et de tuning des entrepôts de données ». Thèse de doct. École nationale supérieure de mécanique et d'aérotechnique, Chasseneuil-du-Poitou, Poitiers, France, 2009 (cf. p. 29, 31).

[Box+00] D. Box et al. « Simple Object Access Protocol (SOAP) 1.1 ». http ://www.w3.org/TR/soa 2000 (cf. p. 13).

[BP02] Patrick Bosc et Olivier Pivert. « Vers un modèle relationnel possibiliste à un niveau de relations imbriquées ». In : *Actes du XXème Congrès INFORSID, Nantes, France, 4-7 juin, 2002*. 2002, p. 73–88 (cf. p. 23).

[BP03] Patrick BOSC et Olivier PIVERT. « A propos de l'évaluation de requêtes possibilistes ». In : , Nancy, 54000, 2003, p. 317–332 (cf. p. 23).

[BP04] Patrick Bosc et Olivier Pivert. « From Boolean to fuzzy algebraic queries in a possibilistic database framework ». In : *IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2004, Budapest, Hungary, July 25-29, 2004*. 2004, p. 547–552 (cf. p. 23).

[BP05a] Patrick Bosc et Olivier Pivert. « About projection-selection-join queries addressed to possibilistic relational databases ». In : *IEEE Trans. Fuzzy Systems* 13.1 (2005), p. 124–139 (cf. p. 18).

[BP05b] Patrick Bosc et Olivier Pivert. « About projection-selection-join queries addressed to possibilistic relational databases ». In : *IEEE Trans. Fuzzy Systems* 13.1 (2005), p. 124–139 (cf. p. 23).

[BP07] Patrick Bosc et Olivier Pivert. « About Possibilistic Queries and Their Evaluation ». In : *IEEE Trans. Fuzzy Systems* 15.3 (2007), p. 439–452 (cf. p. 18, 23).

[BP10] Patrick Bosc et Olivier Pivert. « Modeling and Querying Uncertain Relational Databases : a Survey of Approaches Based on the Possible Worlds Semantics ». In : *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 18.5 (2010), p. 565–603 (cf. p. 42).

[BP82] Billy P Buckles et Frederick E Petry. « A fuzzy representation of data for relational databases ». In : *Fuzzy Sets and Systems* 7.3 (1982), p. 213 –226 (cf. p. 22).

[BP95] Bill P. Buckles et Frederick E. Petry. « Fuzzy databases in the new era ». In : *Proceedings of the 1995 ACM symposium on applied computing, SAC'95, Nashville, TN, USA, February 26-28, 1995*. 1995, p. 497–502 (cf. p. 22).

[Bro+04] Antonio Brogi et al. « Formalizing Web Service Choreographies ». In : *Electr. Notes Theor. Comput. Sci.* 105 (2004), p. 73–94 (cf. p. 17).

[Bro+93] Peter F. Brown et al. « The Mathematics of Statistical Machine Translation : Parameter Estimation ». In : *Computational Linguistics* 19.2 (1993), p. 263–311 (cf. p. 96, 97).

[Brz+11] Jerzy Brzezinski et al. « Workflow Engine Supporting RESTful Web Services ». In : *Intelligent Information and Database Systems - Third International Conference, ACIIDS 2011, Daegu, Korea, April 20-22, 2011, Proceedings, Part I*. 2011, p. 377–385 (cf. p. 66).

[Car07] Michael J. Carey. « Declarative Data Services : This Is Your Data on SOA ». In : *IEEE International Conference on Service-Oriented Computing and Applications, SOCA 2007, 19-20 June 2007, Newport Beach, California, USA*. 2007, p. 4 (cf. p. 30).

[CCT09] James Cheney, Laura Chiticariu et Wang Chiew Tan. « Provenance in Databases : Why, How, and Where ». In : *Foundations and Trends in Databases* 1.4 (2009), p. 379–474 (cf. p. 41).

[Cha09] Yaël Champclaux. « Un modèle de recherche d'information basé sur les graphes et les similarités structurelles pour l'amélioration du processus de recherche d'information ». Thèse de doctorat dirigée par Mothe, Josiane Informatique Toulouse 3 2009. Thèse de doct. 2009, 1 vol. (162 p.) (Cf. p. 28, 29).

[CHM94] Luis M. de Campos, Juan F. Huete et Serafín Moral. « Probability Intervals : a Tool for uncertain Reasoning ». In : *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 2.2 (1994), p. 167–196 (cf. p. 19, 22).

[Cle+04] Luc Clement et al. *UDDI Version 3.0.2*. Organization for the Advancement of Structured Information Standards, UDDI Spec Technical Committee Draft. 2004 (cf. p. 13).

[Cod79] E. F. Codd. « Extending the Database Relational Model to Capture More Meaning ». In : *ACM Trans. Database Syst.* 4.4 (1979), p. 397–434 (cf. p. 22).

[Cur+02] Francisco Curbera et al. « Unraveling the Web Services Web : An Introduction to SOAP, WSDL, and UDDI ». In : *IEEE Internet Computing* (2002), p. 86–93 (cf. p. 2).

[Des08] Sébastien Destercke. « Uncertainty representation and combination : new results with application to nuclear safety issues. (Représentation et combinaison d'informations incertaines : nouveaux résultats avec applications aux études de sûreté nucléaires) ». Thèse de doct. Paul Sabatier University, Toulouse, France, 2008 (cf. p. 19).

[DHY07] Xin Luna Dong, Alon Y. Halevy et Cong Yu. « Data Integration with Uncertainty ». In : *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*. 2007, p. 687–698 (cf. p. 21).

[DL96] Jiangying Zhou Daniel Lopresti. « Retrieval strategies for noisy text ». In : *Proceedings 1996 Symposium on Document Image Understanding Technology* (1996), p. 255–270 (cf. p. 95).

[DP02] Laurence Duval et Olivier Pivert. « A propos de requêtes possibilistes adressées à des bases de données possibilistes ». In : *18èmes Journées Bases de Données Avancées, BDA '02, 21-25 octobre 2002, Evry, Actes (Informal Proceedings)*. 2002 (cf. p. 23).

[DP12] Didier Dubois et Henri Prade. « Possibility Theory ». In : *Computational Complexity : Theory, Techniques, and Applications*. Sous la dir. de Robert A. Meyers. New York, NY : Springer New York, 2012, p. 2240–2252 (cf. p. 20).

[DS07] Nilesh N. Dalvi et Dan Suciu. « Efficient query evaluation on probabilistic databases ». In : *VLDB J.* 16.4 (2007), p. 523–544 (cf. p. 22, 23).

[Eck+14] Kai Eckert et al. « RESTful open workflows for data provenance and reuse ». In : *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume*. 2014, p. 259–260 (cf. p. 66).

[Emm00] Wolfgang Emmerich. *Engineering Distributed Objects*. John Wiley & Sons Software, 2000 (cf. p. 10).

[FC82] Hiroshi Fukuda et Tsu-Wei Chou. « A probabilistic theory of the strength of short-fibre composites with variable fibre length and orientation ». In : *Journal of Materials Science* 17.4 (1982), p. 1003–1011 (cf. p. 21).

[Fie00] Roy Thomas Fielding. « Architectural Styles and the Design of Network-based Software Architectures ». AAI9980887. Thèse de doct. 2000 (cf. p. 72).

[FT02] Roy T. Fielding et Richard N. Taylor. « Principled Design of the Modern Web Architecture ». In : *ACM Trans. Internet Technol.* 2.2 (mai 2002), p. 115–150 (cf. p. 14).

[FY15] Fatima Zohra Filali et Belabbas Yagoubi. « Classifying and Filtering Users by Similarity Measures for Trust Management in Cloud Environment ». In : *Scalable Computing : Practice and Experience* 16.3 (2015), p. 289–302 (cf. p. 24).

[Gal+09] Avigdor Gal et al. « Aggregate Query Answering under Uncertain Schema Mappings ». In : *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*. 2009, p. 940–951 (cf. p. 21).

[Gan+16] Abdullah Gani et al. « A Survey on Indexing Techniques for Big Data : Taxonomy and Performance Evaluation ». In : *Knowl. Inf. Syst.* 46.2 (fév. 2016), p. 241–284 (cf. p. 28).

[GNP92] Shashi K. Gadia, Sunil S. Nair et Yiu-Cheong Poon. « Incomplete Information in Relational Temporal Databases ». In : *18th International Conference on Very Large Data Bases, August 23-27, 1992, Vancouver, Canada, Proceedings.* 1992, p. 395–406 (cf. p. 22).

[GO15] Kamel Garrouch et Mohamed Nazih Omri. « Possibilistic Network based Information Retrieval Model ». In : *15th International Conference on Intelligent Systems Design and Applications, ISDA 2015, Marrakech, Morocco, December 14-16, 2015.* 2015, p. 25–30 (cf. p. 21).

[Gra79] John Grant. « Partial Values in a Tabular Database Model ». In : *Inf. Process. Lett.* 9.2 (1979), p. 97–99 (cf. p. 22).

[HA] Souhila Boucham Hassina Aliane. « Une Ontologie pour l'Indexation et la Recherche d'Information Multilingue ». In : *JLaboratoire Systèmes Informatiques, USTHB, Alger, Algérie* () (cf. p. 27, 28).

[Had06] Marc J. Hadley. *Web Application Description Language (WADL).* Rapp. tech. Mountain View, CA, USA, 2006 (cf. p. 16).

[HBA09] Arezki Hammache, Mohand Boughanem et Rachid Ahmed-Ouamer. « Introduction de la sémantique d'un document sous le modèle de langage ». In : *COnférence en Recherche d'Infomations et Applications - CORIA 2009, 6th French Information Retrieval Conference, Presqu'île de Giens, France, May 5-7, 2009. Proceedings.* 2009, p. 433–444 (cf. p. 28, 29).

[HOB17] Jalel Eddine Hajlaoui, Mohamed Nazih Omri et Djamal Benslimane. « Multi-tenancy Aware Configurable Service Discovery Approach in Cloud Computing ». In : *26th IEEE International Conference on Enabling Technologies : Infrastructure for Collaborative Enterprises, WETICE 2017, Poznan, Poland, June 21-23, 2017.* 2017, p. 232–237 (cf. p. 36).

[HP85] Didier Dubois Henri Prade. *Théorie des possibilités applications à la représentation des connaissances en informatique.* 1985 (cf. p. 19).

[HRO06] Alon Y. Halevy, Anand Rajaraman et Joann J. Ordille. « Data Integration : The Teenage Years ». In : *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*. 2006, p. 9–16 (cf. p. 60).

[JO02] Arthur ter Hofstede Justin O'Sullivan David Edmond. « What's in a Service ? A. Distributed and Parallel Databases ». In : *Commun. ACM* (2002) (cf. p. 13).

[Jr.79] Witold Lipski Jr. « On Semantic Issues Connected with Incomplete Information Databases ». In : *ACM Trans. Database Syst.* 4.3 (1979), p. 262–296 (cf. p. 22).

[Jr.81] Witold Lipski Jr. « On Databases with Incomplete Information ». In : *J. ACM* 28.1 (1981), p. 41–70 (cf. p. 22).

[Jur06] Matjaz B. Juric. *Business Process Execution Language for Web Services BPEL and BPEL4WS 2Nd Edition*. Packt Publishing, 2006 (cf. p. 18).

[KNS10] Evgeny Kharlamov, Werner Nutt et Pierre Senellart. « Updating probabilistic XML ». In : *Proceedings of the 2010 EDBT/ICDT Workshops, Lausanne, Switzerland, March 22-26, 2010*. 2010 (cf. p. 66, 67).

[KNS11] Evgeny Kharlamov, Werner Nutt et Pierre Senellart. « Value joins are expensive over (probabilistic) XML ». In : *Proceedings of the 4th International Workshop on Logic in Databases, Uppsala, Sweden, (EDBT/ICDT '10 joint conference), March 25, 2011, Proceedings*. 2011, p. 41–48 (cf. p. 65).

[KS13] Benny Kimelfeld et Pierre Senellart. « Probabilistic XML : Models and Complexity ». In : *Advances in Probabilistic Databases for Uncertain Information Management*. 2013, p. 39–66 (cf. p. 18).

[Lak+97] Laks V. S. Lakshmanan et al. « ProbView : A Flexible Probabilistic Database System ». In : *ACM Trans. Database Syst.* 22.3 (1997), p. 419–469 (cf. p. 22).

[LDB16] Angel Lagares Lemos, Florian Daniel et Boualem Benatallah. « Web Service Composition : A Survey of Techniques and Tools ». In : *ACM Comput. Surv.* 48.3 (2016), 33 :1–33 :41 (cf. p. 36).

[Mal+15] Abdelhamid Malki et al. « Composing Data Services with Uncertain Semantics ». In : *IEEE Trans. Knowl. Data Eng.* 27.4 (2015), p. 936–949 (cf. p. 4, 22, 24).

[Mal+16] Abdelhamid Malki et al. « Data Services with uncertain and correlated semantics ». In : *World Wide Web* 19.1 (2016), p. 157–175 (cf. p. 22, 24, 36).

[Mal15] Abdelhamid Malki. « Modélisation sémantique du cloud computing : vers une composition de services DaaS à sémantique incertaine. (Semantic modeling for cloud computing : toward Daas service composition with uncertain semantics) ». Thèse de doct. Claude Bernard University Lyon 1, France, 2015 (cf. p. 24).

[MB13] Abdelhamid Malki et Sidi Mohamed Benslimane. « Semantic Cloud : Building Dynamic Mashup in Cloud Environment ». In : *IJITWE* 8.4 (2013), p. 20–35 (cf. p. 4, 24).

[MBE03] Brahim Medjahed, Athman Bouguettaya et Ahmed K. Elmagarmid. « Composing Web Services on the Semantic Web ». In : *The VLDB Journal* 12.4 (nov. 2003), p. 333–351 (cf. p. 2).

[ME08] Abdulmotaleb El Saddik Mohamad Eid Atif Alamri. « A reference model for dynamic web service composition systems ». In : *International Journal of Web and Grid Services (IJWGS)* 4.2 (2008), p. 115–150 (cf. p. 17).

[MK60] M. E. Maron et J. L. Kuhns. « On Relevance, Probabilistic Indexing and Information Retrieval ». In : *J. ACM* 7.3 (1960), p. 216–244 (cf. p. 95, 97).

[Mol05] Ilya Molchanov. *Theory of Random Sets*. Springer London, 2005 (cf. p. 19).

[Mor10] Luc Moreau. « The Foundations for Provenance on the Web ». In : *Foundations and Trends in Web Science* 2.2-3 (2010), p. 99–241 (cf. p. 41).

[MPD10] Maria Maleshkova, Carlos Pedrinaci et John Domingue. « Investigating Web APIs on the World Wide Web ». In : *8th IEEE European Conference on Web Services (ECOWS 2010), 1-3 December 2010, Ayia Napa, Cyprus*. 2010, p. 107–114 (cf. p. 60).

[MPK14] Anita Brigit Mathew, Priyabrat Pattnaik et S. D. Madhu Kumar. « Efficient information retrieval using Lucene, LIndex and HIndex in Hadoop ». In : *11th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2014, Doha, Qatar, November 10-13, 2014*. 2014, p. 333–340 (cf. p. 27).

[MS09] Kiran-Kumar Muniswamy-Reddy et Margo I. Seltzer. « Provenance as first class cloud data ». In : *Operating Systems Review* 43.4 (2009), p. 11–16 (cf. p. 41).

[MW11a] Amélie Marian et Minji Wu. « Corroborating Information from Web Sources ». In : *IEEE Data Eng. Bull.* 34.3 (2011), p. 11–17 (cf. p. 37).

[MW11b] Amélie Marian et Minji Wu. « Corroborating Information from Web Sources ». In : *IEEE Data Eng. Bull.* 34.3 (2011), p. 11–17 (cf. p. 42).

[MZ13] Ahmed Moustafa et Minjie Zhang. « Multi-Objective Service Composition Using Reinforcement Learning ». In : *Service-Oriented Computing - 11th International Conference, ICSOC 2013, Berlin, Germany, December 2-5, 2013, Proceedings*. 2013, p. 298–312 (cf. p. 24, 25).

[Neu04] Arnold Neumaier. « Clouds, Fuzzy Sets, and Probability Intervals ». In : *Reliable Computing* 10.4 (2004), p. 249–272 (cf. p. 19).

[NJ02a] Andrew Nierman et H. V. Jagadish. « ProTDB : Probabilistic data in XML ». In : *In Proceedings of the 28th VLDB Conference*. Springer, 2002, p. 646–657 (cf. p. 18, 71).

[NJ02b] Andrew Nierman et H. V. Jagadish. « ProTDB : Probabilistic Data in XML ». In : *VLDB 2002, Proceedings of 28th International Conference on Very Large Data Bases, August 20-23, 2002, Hong Kong, China*. 2002, p. 646–657 (cf. p. 68).

[Pal90] Patrick Palmer. « Étude d'un analyseur de surface de la langue naturelle : application à l'indexation automatique de textes. (Study of a surface parser for natural language. Application to automatic indexing of text) ». Thèse de doct. Joseph Fourier University, Grenoble, France, 1990 (cf. p. 95).

[Pan08] Tadeusz Pankowski. « Reconciling Inconsistent Data in Probabilistic XML Data Integration ». In : *Sharing Data, Information and Knowledge, 25th British National Conference on Databases, BNCOD 25, Cardiff, UK, July 7-10, 2008. Proceedings*. 2008, p. 75–86 (cf. p. 21).

[Pap03] Mike P. Papazoglou. « Service -Oriented Computing : Concepts, Characteristics and Directions ». In : *Proceedings of the Fourth International Conference on Web Information Systems Engineering*. WISE '03. Washington, DC, USA : IEEE Computer Society, 2003, p. 3– (cf. p. 2).

[PC98] Jay M. Ponte et W. Bruce Croft. « A Language Modeling Approach to Information Retrieval ». In : *SIGIR '98 : Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*. 1998, p. 275–281 (cf. p. 97).

[Pei+08] Jian Pei et al. « Query answering techniques on uncertain and probabilistic data : tutorial summary ». In : *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*. 2008, p. 1357–1364 (cf. p. 22).

[PP14] Olivier Pivert et Henri Prade. « Querying Uncertain Multiple Sources ». In : *Scalable Uncertainty Management - 8th International Conference, SUM 2014, Oxford, UK, September 15-17, 2014. Proceedings*. 2014, p. 286–291 (cf. p. 18).

[PZL08] Cesare Pautasso, Olaf Zimmermann et Frank Leymann. « Restful Web Services vs. "Big"' Web Services : Making the Right Architectural Decision ». In : *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. Beijing, China : ACM, 2008, p. 805–814 (cf. p. 16).

[RL16] Xu Hang Ren Lifang Wang Wenjian. « Uncertainty-Aware Adaptive Service Composition in Cloud Computing ». In : *Journal of Computer Research and Development* 53.12, 2867 (2016), p. 2867 (cf. p. 25).

[RM88] K. V. S. V. N. Raju et Arun K. Majumdar. « Fuzzy Functional Dependencies and Lossless Join Decomposition of Fuzzy Relational Database Systems ». In : *ACM Trans. Database Syst.* 13.2 (1988), p. 129–166 (cf. p. 22).

[Ros+08] Florian Rosenberg et al. « Composing RESTful Services and Collaborative Workflows : A Lightweight Approach ». In : *IEEE Internet Computing* 12.5 (2008), p. 24–31 (cf. p. 66).

[RR07] Leonard Richardson et Sam Ruby. *Restful Web Services*. First. O'Reilly, 2007 (cf. p. 66, 68).

[SA07] Pierre Senellart et Serge Abiteboul. « On the complexity of managing probabilistic XML data ». In : *Proceedings of the Twenty-Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 11-13, 2007, Beijing, China*. 2007, p. 283–292 (cf. p. 66).

[Sad91] Fereidoon Sadri. « Modeling Uncertainty in Databases ». In : *Proceedings of the Seventh International Conference on Data Engineering, April 8-12, 1991, Kobe, Japan*. 1991, p. 122–131 (cf. p. 42).

[Sau05] Karen Sauvagnat. « Modele flexible pour la Recherche d'Information dans des corpus de documents semi-structures ». Theses. 2005 (cf. p. 25–28).

[SB88] Gerard Salton et Chris Buckley. « Term-Weighting Approaches in Automatic Text Retrieval ». In : *Inf. Process. Manage.* 24.5 (1988), p. 513–523 (cf. p. 95).

[Sha76] Glenn Shafer. *A Mathematical Theory of Evidence*. T. 79. 1976, p. 7–25 (cf. p. 19).

[Sin01] Munindar P. Singh. « Physics of Service Composition ». In : *Being Interactive* (2001) (cf. p. 2).

[SS12] Asma Souihli et Pierre Senellart. « Optimisation des approximations de probabilité des requêtes en XML probabiliste ». In : *BDA (Bases de Données Avancées)*. Clermont-Ferrand, France, oct. 2012, p. 20 (cf. p. 64, 66).

[SS13] Asma Souihli et Pierre Senellart. « Optimizing approximations of DNF query lineage in probabilistic XML ». In : *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*. 2013, p. 721–732 (cf. p. 64, 65).

[SSI10] Mohamed A. Soliman, Mina Saleeb et Ihab F. Ilyas. « MashRank : Towards uncertainty-aware and rank-aware mashups ». In : *Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA*. 2010, p. 1137–1140 (cf. p. 37).

[SYY74] Gerard Salton, Chung-Shu Yang et Clement T. Yu. « Contribution to the Theory of Indexing ». In : *IFIP Congress*. 1974, p. 584–590 (cf. p. 90).

[Tam07] Caroline Tambellini. « An information retrieval system adapted to uncertain data : adaptation of language model ». Theses. Université Joseph-Fourier - Grenoble I, déc. 2007 (cf. p. 90).

[TW08] Diana Comes Kurt Geihs Thomas Weise Steffen Bleul. « Different Approaches to Semantic Web Service Composition ». In : *Internet and Web Applications and Services, 2008. ICIW '08.* (2008) (cf. p. 17).

[TWI08] Sayed Gholam Hassan Tabatabaei, Wan M. N. Wan-Kadir et Suhaimi Ibrahim. « A Comparative Evaluation of State-of-the-Art Approaches for Web Service Composition ». In : *Proceedings of the Third International Conference on Software Engineering Advances, ICSEA 2008, October 26-31, 2008, Sliema, Malta*. 2008, p. 488–493 (cf. p. 17).

[Vas80] Yannis Vassiliou. « Functional Dependencies and Incomplete Information ». In : *Sixth International Conference on Very Large Data Bases, October 1-3, 1980, Montreal, Quebec, Canada, Proceedings.* 1980, p. 260–269 (cf. p. 22).

[VK11] Angelos Vasilakopoulos et Verena Kantere. « Efficient Query Computing for Uncertain Possibilistic Databases with Provenance ». In : *3rd Workshop on the Theory and Practice of Provenance, TaPP'11, Heraklion, Crete, Greece, June 20-21, 2011*. 2011 (cf. p. 36, 47).

[VMB16] Pierre De Vettor, Michael Mrissa et Djamal Benslimane. « Towards Definition and Composition of Uncertain RESTful Resources ». In : *Proceedings of the 4th Workshop on Services and Applications over Linked APIs and Data co-located with the 13th Extended Semantic Web Conference (ESWC 2016), Crete, Greece, May 29, 2016.* 2016 (cf. p. 18, 19).

[VN02] S. J. Vaughan-Nichols. « Web Services : Beyond the Hype ». In : *Computer* 35 (fév. 2002), p. 18–21 (cf. p. 10, 11).

[Wal91] Peter Walley. *Statistical Reasoning with Imprecise Probabilities*. 1991 (cf. p. 19).

[Wan+11a] Shangguang Wang et al. « Cloud model for service selection ». In : *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2011, p. 666–671 (cf. p. 23).

[Wan+11b] Shangguang Wang et al. « Reliable web service selection via QoS uncertainty computing ». In : *IJWGS* 7.4 (2011), p. 410–426 (cf. p. 23).

[Won82] Eugene Wong. « A Statistical Approach to Incomplete Information in Database Systems ». In : *ACM Trans. Database Syst.* 7.3 (1982), p. 470–488 (cf. p. 22).

[Yu+08] Qi Yu et al. « Deploying and Managing Web Services : Issues, Solutions, and Directions ». In : *The VLDB Journal* 17.3 (mai 2008), p. 537–572 (cf. p. 17, 36, 53).

[Zad99] L. A. Zadeh. « Fuzzy Sets As a Basis for a Theory of Possibility ». In : *Fuzzy Sets Syst.* 100 (avr. 1999), p. 9–34 (cf. p. 20).

[Zar04] Haifa Zargayouna. « Contexte et sémantique pour une indexation de documents semi-structurés. » In : IRIT, 2004, p. 161–178 (cf. p. 31).