Découverte d'annuaires de services web dans un environnement distribué

Mohamed Sellami

TELECOM SudParis

Université d'Évry Val d'Essonne

Thèse de doctorat de Télécom & Management SudParis, préparée dans le cadre de l'école doctorale S&I, en accréditation conjointe avec l'Université d'Evry-Val d'Essonne
Spécialité Informatique

Par :

Mohamed SELLAMI

Découverte d'annuaires de services Web dans un environnement distribué

Thèse présentée pour l'obtention du grade de
Docteur de TELECOM & Management SudParis

Soutenue le 18 octobre 2011 devant le jury composé de :

Rapporteurs :		
M.	Abdelkader Hameurlain	Professeur à l'Université Paul Sabatier - IRIT
M.	Philippe Lamarre	Maître de Conférences - HDR à l'Université de Nantes -LINA
Examinateurs :		
M.	Chihab Hanachi	Professeur l'Université Toulouse 1 - IRIT
M.	Bruno Defude	Professeur TÉLÉCOM SudParis - SAMOVAR
M.	Samir Tata	Professeur à TÉLÉCOM SudParis - SAMOVAR (Directeur de thèse)
M.	Walid Gaaloul	Maître de conférences à TÉLÉCOM SudParis - SAMOVAR (Encadrant)

Thèse numéro : 2011TELE0022

Résumé : Les services Web (SW) sont considérés comme étant un bloc pilier pour la réalisation de transactions électroniques entre entreprises (B2B). Ainsi, de plus en plus d'entreprises utilisent des SW pour réaliser des transactions avec leurs partenaires et/ou offrir des services en ligne. Dans un tel contexte, chaque entreprise possède un ou plusieurs référentiels privés de descriptions de SW. Pour que ses SW soient consultables et puissent être découverts par d'autres entreprises, une entreprise devra rendre ses référentiels de SW publics. La solution couramment utilisée consiste à publier les descriptions de ses SW dans des annuaires de SW. En conséquence, le nombre d'annuaires disponibles peut être aussi important que le grand nombre d'entreprises et la découverte d'un SW devient ainsi une tâche fastidieuse pour un demandeur de services et coûteuse pour un système de découverte.

L'objectif principal de cette thèse est de fournir des solutions pour assurer une découverte d'annuaires de SW adaptée à un environnement distribué. Pour ce faire, nous proposons tout d'abord d'organiser le réseau d'annuaires en communautés. Nous utilisons comme critère d'organisation les fonctionnalités offertes par les SW annoncés par chaque annuaire. Ensuite, afin de préserver la consistance de cette organisation en communautés face aux changements qui peuvent se produire (départ/arrivée d'annuaires), nous définissons les mécanismes de gestion nécessaires. Ainsi, nous avons identifié les différents événements qui peuvent se produire et induire une inconsistance au niveau de notre organisation et nous avons défini les mécanismes de gestion adéquats. Enfin, nous proposons une approche pour la découverte d'annuaires de SW qui utilise deux filtres pour limiter l'espace de recherche d'un demandeur de services. Le premier filtre consiste à utiliser les besoins fonctionnels d'un demandeur de services pour sélectionner la communauté d'annuaires adéquate. Le deuxième filtre consiste à utiliser les besoins non-fonctionnels et la trace des services déjà utilisés par le demandeur de services pour sélectionner un ou plusieurs annuaires. La sélection d'un annuaire au sein d'une communauté est basée sur une technique de recommandation. Les solutions proposées ont été testées par la mise en place d'un réseau de communautés d'annuaires, l'implémentation d'un gestionnaire de communautés et d'un système de découverte d'annuaires.

Mots clés : Services Web - SOC - Annuaires de services Web - Organisation d'annuaires de services Web - Communautés d'annuaires - Gestion de communautés d'annuaires - Découverte d'annuaires.

Web Services Registries Discovery in a Distributed Environment

Abstract : Web services can be seen as a pillar block for achieving electronic B2B transactions. More and more companies are using Web services to achieve transactions with their partners and/or offer on-line services. These companies have to make their Web services available for consultation through their own private Web service registries. As a result, the number of Web service registries that are made available for use can be as many as the large number of companies. This raises an old, search engine, problem in a new form : discovery mechanisms of Web services are not efficient both in response times and quality of results. Basically, a company interested in a service has to screen several registries to discover the service that best suits its needs. This task can be very cumbersome since the number of available registries, and also the services they advertise can be very large. In this context, if appropriate solutions are not considered, "traditional" Web services discovery mechanisms that consist of scanning all the registries would for instance slow down the increase rate of Web services.

In this thesis, we are interested in Web services discovery in a distributed registry environment. To do this, we first propose to organize a registries network into communities based on functionalities offered by the Web services of a registry. Then, to handle the dynamic nature of communities and their members, we define the needed management mechanisms to monitor changes and reconcile potential conflicts. We identify the different capabilities of a registry (joining a community, updating functionalities,...) and a community (creation, dismantling, merging,...) life-cycle and we specify the associated management operations. Finally, we propose an approach for Web services registries discovery that uses two filters to limit the search space. We first use the functional requirement of a service requester to select the appropriate registry community. Then we use the requester's non-functional requirements and his behavior to select the registries. The proposed solutions were tested by setting up a P2P network of registry communities, implementing a community manager and a discovery system.

Keywords : Web Services - SOC - Web services registry - Web services registries organization - Communities of Web service registries - Managing communities of Web service registries - Registries discovery.

Remerciements

Je voudrais exprimer mes sentiments les plus spontanés envers les personnes qui sans lesquelles ce travail de thèse n'aurait pas pu voir le jour. Leur aide, accompagnement et soutien m'ont été indispensables afin de pouvoir aboutir aux contributions de ma thèse.

Je voudrais tout d'abord exprimer ma reconnaissance envers tous les membres du jury pour la grande attention qu'ils ont bien voulu porter à mon travail.

Je suis très reconnaissant à Samir Tata, mon directeur de thèse, et Walid Gaaloul, mon encadrant, qui m'ont encadré et dirigé dans mes recherches tout au long de ces années. Je leur dis ma gratitude pour l'aide compétente qu'ils m'ont apportée, pour leurs encouragements et pour la confiance qu'ils m'ont toujours témoignée. Je les remercie aussi d'être de très bons amis, pour le café gratuit (Samir) et pour l'agréable cadre de travail qu'ils m'ont offert.

Mes plus chaleureux remerciements vont également à Bruno Defude, qui a su m'aider pour mener à bien le présent travail. Sa disponibilité et ses conseils m'ont permis de mieux cerner les problèmes et ainsi de mieux y répondre.

Je remercie très sincèrement mes rapporteurs Abdelkader Hameurlain et Philippe Lamarre pour avoir bien accepté d'être mes rapporteurs et pour avoir bien voulu lire et évaluer mon travail de thèse. Je les remercie pour leurs lectures approfondies de mon mémoire de thèse, pour tout le temps qu'ils m'ont accordé et pour les remarques très constructives qu'ils m'ont données.

Je remercie également Chihab Hanachi et Bruno Defude qui m'ont fait honneur en acceptant de faire partie de mon jury de thèse. Leur participation dans l'évaluation de ce travail est d'une trés grande valeur.

Je remercie Maelle Sivy pour ses efforts lors de la relecture de ce mémoire et pour toutes les nouvelles règles de conjugaison et de grammaire qu'elle m'a appris.

Je remercie les membres de l'équipe SIMBAD et du département informatique de Télécom SudParis : Olfa Bouchaala pour sa précieuse aide à plusieurs occasions, Brigitte Houassine qui nous simplifie énormément la vie au sein du département et Djamel Belaid pour ses encouragements et ses conseils. Je remercie aussi toutes les personnes que j'ai connu au cours de cette thèse et qui l'ont rendu plus agréable : Alda, Amel, Amine[2], Djalel, Dorsaf, Gabriel, Imen, Jérôme, Leon, Marie, Mohamed, Mouna, Mourad, Chan, Nomane, Sami, Rami, Rim, Yassine, Zahra, Zhangbing et Zied.

Je remercie mon cher ami Wissem et son épouse Nouha pour tous ce qu'ils ont faits pour moi depuis mon arrivée en France. Je remercie aussi mes amis en Tunisie qui ne cessent de m'encourager par mail et sur facebbok : Ahmed, Mehdi, Nabil, Riadh et Slim.

Un grand merci à ma famille pour son soutien tout au long de cette thèse. Mes parents, Amel et Zouhir, pour les sacrifices qu'ils ont faits en faveur de mon éducation et sans qui j'aurais pas pu réaliser cette thèse. Ma chère épouse Manel pour m'avoir supporté, encouragé et motivé ces dernières années. Je lui suis pour toujours reconnaissant. Mon fils Yassine pour qu'il puisse trouver son nom le jour où il apprendra à lire. Mes beaux parents pour leurs encouragements constants. Je tiens également à remercier mes deux frères Rami et Ramzi.

Table des matières

Chapitre 1

Introduction générale

1.1 Contexte et problématique de la thèse

Les services Web sont considérés comme étant un bloc pilier pour la réalisation de transactions électroniques entre entreprises (B2B). Ceci est particulièrement dû au fait que les services Web sont construits autour de standards XML pour la description de services (WSDL [Moreau 2007]), le référencement de services (UDDI [Bellwood 2002]) et l'échange de messages (SOAP [Mitra 2007]). Ces standards ont promu l'utilisation des services Web pour la mise en place d'applications B2B, principalement parce qu'ils permettent aux entreprises d'interagir entre elles sans pour autant nécessiter un travail de réingénierie. Ainsi, de plus en plus d'entreprises utilisent des services Web pour réaliser des transactions avec leurs partenaires et/ou offrir des services en ligne. Par exemple, dans une enquête, menée par le cabinet *McKinsey Quarterly* [The Mckinsey Quarterly 2007] en 2007, sur plus de 2800 entreprises à travers le monde, 80 % utilisent ou envisagent d'utiliser des services Web. Parmi ces entreprises, 78% affirment que la technologie des services Web est l'une des trois technologies les plus importantes pour leur activité.

Dans un tel contexte, chaque entreprise possède un ou plusieurs référentiels privés de descriptions de services Web. Pour que ses services soient consultables et puissent être découverts par d'autres entreprises, une entreprise devra rendre ses référentiels de services publics. La solution couramment utilisée consiste à publier ces descriptions de services dans des annuaires de services Web (par exemple UDDI, ebXML [Breininger 2001] consultables par les autres entreprises. En conséquence, le nombre d'annuaires de services Web disponibles peut être aussi important que le nombre d'entreprises, et la découverte d'un service Web devient ainsi une tâche fastidieuse pour un demandeur de services et coûteuse pour le système.

La découverte de services Web représente une thématique de recherche importante depuis l'émergence de la technologie des services Web. En effet, un service est le plus souvent **découvert** avant d'être utilisé. Le World Wide Web Consortium (W3C) définit un service Web par : « . . .une application logicielle identifiée par un URI, dont les interfaces et les liaisons peuvent être définies, décrites et **découvertes** en XML . . . » [1]. Pour découvrir un service Web, plusieurs techniques et approches existent. Ces approches peuvent être classées en deux catégories : approches de découverte **mono-annuaire** et approches de découverte **multi-annuaires**.

1. . . .*a software application identified by a URI, whose interfaces and bindings are capable of being defined, described, and* ***discovered*** *as XML artifacts*. . .

Dans le contexte de **découverte mono-annuaire**, un demandeur de services interagit avec un seul annuaire de services **centralisé** pour trouver un service Web. Le problème revient à trouver la (ou les) réponse(s) à une requête de découverte de services parmi l'ensemble des descriptions de services annoncées par cet annuaire. Une requête de découverte de services peut être décrite sous un format non-structuré (des mots-clés par exemple) ou structuré (une description de service syntaxique/-sémantique abstraite par exemple). Les techniques de découverte varient ainsi des simples utilisations de mots-clés sur un moteur de recherche aux approches utilisant des appariements syntaxiques/sémantiques.

Pour une découverte de services dans un contexte **multi-annuaires**, un demandeur de services doit interagir avec plusieurs annuaires de services **distribués**. La découverte de services se déroule sur deux étapes. La première étape consiste à trouver le ou les annuaire(s) de services susceptible(s) de contenir la description du service recherché. La deuxième étape revient à une découverte **mono-annuaire** appliquée aux annuaires identifiés.

Dans cette thèse, nous nous intéressons à une découverte de services dans un contexte B2B où chaque entreprise offre ses services à travers un annuaire de services exposant leurs descriptions sémantiques. Nous nous retrouvons donc dans un contexte de découverte **multi-annuaires** en utilisant comme requête une description abstraite et sémantique d'un service Web. Nous nous intéressons seulement à la première étape consistant à découvrir un ou plusieurs annuaires de services. La deuxième étape, c.-à-d. la découverte mono-annuaire, fait l'objet d'autres études menées au sein de notre équipe [Chabeb 2010, Nguyen 2010, Nguyen 2011].

Dans un tel environnement, nous souhaitons que les réponses à une requête de découverte de services soient adaptées aux besoins fonctionnels et/ou non-fonctionnels du demandeur de services. Nous avons donc besoin d'une approche permettant de **réduire le coût** d'un processus de découverte de services tout en assurant un **bon rappel** et une **bonne précision** des résultats.

Découvrir « le service recherché » dans un contexte multi-annuaires est un processus long et problématique, spécialement quand le nombre d'annuaires et de services qu'ils annoncent est important. Les approches existantes de découverte de services organisent les annuaires de services en groupes, généralement en fonction de leurs domaines métier, pour réduire l'espace de recherche d'un demandeur de services. De telles organisations sont inadéquates pour structurer un réseau d'annuaires de services comme nous allons le montrer dans le Chapitre 2. De plus, ces organisations exigent des interventions de la part du fournisseur d'annuaires (une entreprise par exemple) pour être mise en place et des interventions d'une tierce partie (un gestionnaire du réseau d'annuaires par exemple) pour les tenir à jour après les changements qui peuvent se produire. Dans ce contexte, la problématique de notre recherche est exprimée par la question suivante. **Comment assurer une découverte multi-annuaires tout en garantissant un bon compromis entre réduction de l'espace de recherche et pertinence des résultats ?**

1.2 Un aperçu sur la thèse

1.2.1 Objectifs

Afin de résoudre la problématique posée, nous visons trois objectifs. Notre premier objectif est de proposer **une approche implicite pour l'organisation d'un ensemble d'annuaires de services en groupes**. Compte tenu du contexte de notre travail et des besoins décrits ci-dessus, nous avons identifié deux éléments à considérer :

- **Le critère d'organisation** : La structuration d'un ensemble d'annuaires de services permet d'améliorer le processus de découverte de services en permettant le routage d'une requête de découverte de services vers le groupe d'annuaires adéquat. Ce routage est assuré en fonction du critère d'organisation et un critère inadéquat peut ainsi inférer un routage imprécis de la requête. Ainsi, l'approche proposée devra tenir compte de ce point.
- **La méthode d'organisation** : Un fournisseur d'annuaires n'a pas forcément les compétences nécessaires ni la volonté pour participer à l'organisation d'un réseau d'annuaires de services. Par conséquent, notre méthode d'organisation devra être automatique et implicite.

Afin de préserver la consistance d'un ensemble d'annuaires de services structurés face aux changements qui peuvent se produire (départ/arrivée d'annuaires), des mécanismes de gestion dynamiques doivent être définis. Ainsi, notre deuxième objectif est de proposer **une approche pour la gestion d'un ensemble d'annuaires de services organisés en groupes**. Pour garantir cet objectif, les différents événements qui peuvent se produire et induire une inconsistance au niveau d'une organisation d'un réseau d'annuaires doivent être **identifiés** et **gérés**. L'approche de gestion doit aussi se faire dynamiquement sans l'intervention d'une tierce partie.

Notre troisième objectif est de proposer **une approche pour la découverte d'annuaires de services Web dans ce contexte**. L'approche de découverte doit tirer profit de l'approche d'organisation des annuaires de services pour guider une requête de découverte de services vers le groupe d'annuaires de services le plus adéquat. Toutefois, ce **premier filtre** peut être inefficace pour limiter l'espace de recherche dans le cas où un groupe d'annuaires réunit un grand nombre d'annuaires. Ainsi, notre approche doit mettre en œuvre un **deuxième filtre** pour sélectionner un ou plusieurs annuaires de services à partir du groupe d'annuaires.

1.2.2 Approche

Afin d'atteindre nos deux premiers objectifs, nous proposons une approche qui repose sur les points suivants :

- Fournir un moyen pour résumer les fonctionnalités offertes par les services Web annoncés dans un annuaire. Ceci est assuré en réalisant une agrégation des éléments de descriptions de services d'un annuaire.

- Organiser un réseau d'annuaires de services en utilisant comme critère les **fonctionnalités** offertes par les services Web annoncés dans chaque annuaire. En effet, vu qu'un demandeur de services est à la recherche d'une fonctionnalité, une telle organisation est plus logique qu'une organisation basée sur le domaine métier. De cette façon, une requête de découverte de services adéquat (exprimant une fonctionnalité recherchée) permettra par un simple appariement sémantique de retrouver le groupe d'annuaires de services (représentant une fonctionnalité offerte).
- Définir des mécanismes de gestion pour conserver la consistance de l'organisation d'un réseau d'annuaires de services en groupes face aux changements pouvant survenir vu la nature dynamique des services et des annuaires de services. Pour cela, nous identifions les différentes étapes des cycles de vie d'un annuaire et d'une communauté d'annuaires et nous précisons les opérations de gestion associées.

Pour assurer notre troisième objectif, nous proposons une approche de découverte de services Web tenant compte des historiques des utilisations passées. Ainsi, tout en prenant en compte l'organisation des annuaires selon les fonctionnalités, nous avons adopté une approche qui repose sur les points suivants :

- Inclure les besoins non fonctionnels et la trace des services déjà utilisés par un demandeur de services dans sa requête de découverte de services. Cette requête est sous forme d'une description abstraite d'un service Web qui exprime la fonctionnalité recherchée par le demandeur de services.
- Utiliser deux filtres pour limiter l'espace de recherche d'un demandeur de services dans un environnement multi-annuaires structuré :
  - Premier filtre : utiliser les besoins fonctionnels décrits dans la requête d'un demandeur de services pour sélectionner le groupe d'annuaires le plus adéquat.
  - Deuxième filtre : utiliser les besoins non fonctionnels et la trace des services déjà utilisés par un demandeur de services pour sélectionner un ou plusieurs annuaires. La sélection est assurée en utilisant une technique de recommandation tenant compte des utilisations passées des services Web annoncés par un annuaire.

1.2.3 Contributions

Les contributions de notre thèse sont :

- La définition d'un modèle sémantique (WSRD) pour la description d'annuaires de services et une approche automatique et implicite pour sa construction. Ces descriptions WSRD nous permettront d'organiser un réseau d'annuaires de services en fonction des fonctionnalités offertes par les services Web annoncés dans chaque annuaire. De plus, en associant une description WSRD à un annuaire, il ne sera plus considéré par un demandeur de services comme une « boîte noire ».
- La proposition du concept de communautés d'annuaires de services et d'une

approche automatique et implicite pour organiser un réseau d'annuaires de services en communautés. Une communauté d'annuaires est définie comme étant un groupe d'annuaires offrant des services Web ayant des fonctionnalités similaires. Nous utilisons les communautés comme un moyen pour organiser automatiquement un environnement distribué d'annuaires selon leurs WSRD.
- Des mécanismes de gestion pour les communautés d'annuaires de services. Ces mécanismes permettent de préserver la consistance d'une organisation en communautés conçue selon notre approche de création de communautés face aux différents changements qui peuvent se produire.
- Une nouvelle approche pour la découverte de services Web dans un contexte multi-annuaires structuré. Plus précisément, cette approche permet de découvrir les annuaires de services susceptibles de contenir une description du service recherché au sein d'une communauté d'annuaires.
- La définition des concepts de caractérisation d'utilisateur (RC) et de caractérisation globale d'annuaire (GRC). Une RC représente les besoins non fonctionnels et la trace des services déjà utilisés d'un demandeur de services et une GRC la liste des caractérisations des demandeurs de services précédents d'un annuaire.
- L'utilisation d'une technique de recommandation implicite pour sélectionner un ou plusieurs annuaire(s) à partir d'une communauté d'annuaires. Cette recommandation est réalisée sur la base de la RC d'un demandeur de services et la liste des GRC des annuaires d'une communauté. Notre technique de recommandation permet de filtrer, efficacement et avec un coût beaucoup plus faible que celui des techniques d'appariements sémantiques, un ensemble d'annuaires pour n'en retenir que ceux qui sont en adéquation avec la caractérisation d'un demandeur de services.

1.3 Organisation de la thèse

Cette thèse se compose d'une introduction générale, de quatre chapitres et d'une conclusion générale.

Le chapitre 2 propose un état de l'art des travaux relatifs à notre travail de thèse. Nous étudions dans ce chapitre des travaux sur la découverte de services Web dans un environnement distribué d'annuaires et nous nous focalisons sur les mécanismes d'organisation des annuaires et leurs limites. Nous présentons, par la suite, des travaux qui se sont intéressés à la gestion de communautés de catalogues et de services Web et nous identifions les points pouvant être projetés sur notre contexte. Enfin, nous étudions les limites de travaux utilisant des techniques de recommandations pour améliorer la découverte de services Web.

Les chapitres 3, 4 et 5 constituent le cœur de notre travail. Nous détaillons respectivement nos approches de construction de communautés, de gestion de communautés et de découverte d'annuaires dans un réseau de communautés dans ces chapitres.

Dans le chapitre 3, nous présentons notre approche pour la construction de communautés d'annuaires. Nous proposons un modèle sémantique (WSRD) pour la description d'annuaires de services et son approche de création automatique. Nous introduisons le concept de communauté que nous utilisons pour organiser un réseau d'annuaires. Par la suite, nous décrivons notre approche automatique et implicite de construction de communautés d'annuaires en utilisant les descriptions WSRD.

Dans le chapitre 4, nous présentons notre approche de gestion de communautés d'annuaires. Nous spécifions le cycle de vie d'un annuaire au sein d'une communauté (rejoindre une communauté, mise à jour des fonctionnalités d'un annuaire, etc.) ainsi que les opérations de gestion nécessaires à chacune des étapes identifiées. De même, nous décrivons les étapes du cycle de vie d'une communauté dans un réseau de communautés (création, fusion, division, etc.) et nous exposons les opérations de gestion nécessaires à chacune de ces étapes.

Dans le chapitre 5, nous présentons les étapes de sélection de communauté et d'annuaires de notre approche de découverte d'annuaires de services dans un réseau d'annuaires organisés en communautés. Nous définissons une requête permettant de représenter la fonctionnalité recherchée par un demandeur de services ainsi que sa caractérisation (ses besoins non fonctionnels et sa trace d'exécution). Nous présentons enfin dans ce chapitre la technique de recommandation que nous proposons pour assurer l'étape de sélection d'annuaires.

Dans le chapitre 6, nous nous sommes intéressés à la mise en œuvre des différentes contributions présentées dans les chapitres précédents. Finalement, le chapitre 7 résume nos contributions et dégage les perspectives directes de nos travaux de recherche.

Chapitre 2

État de l'art

Sommaire

2.1 Introduction

Comme suite logique au succès des architectures orientées services au cours de ces dernières années, un nombre important de services Web est disponible en ligne [The Mckinsey Quarterly 2007, Bentahar 2007]. Avec cette augmentation, plusieurs problèmes ont été soulevés, notamment liées à la découverte, la disponibilité, la qualité de services, etc.

Conformément au modèle orienté services, la découverte d'un service par un demandeur de services se fait comme le montre la Figure 2.1. D'un côté, un fournisseur de services publie les descriptions des fonctionnalités de ses services dans un annuaire de services. De l'autre côté, un demandeur de services utilise cet annuaire pour localiser un service et par la suite l'invoquer.

Les techniques de découverte de services traditionnelles, comme décrite dans ce **modèle**, peuvent être inadéquates dans des contextes **réels** (un grand nombre de services et d'annuaires, manque d'expressivité des descriptions de services, etc.).

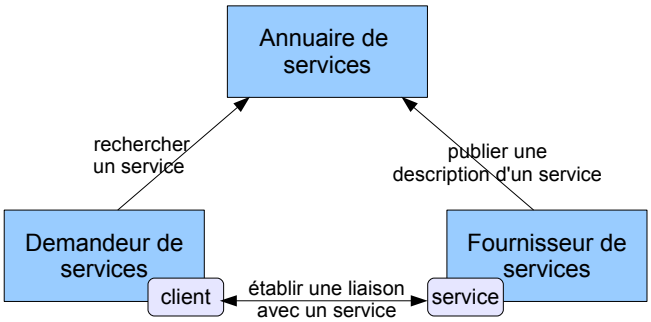


FIGURE 2.1 – Modèle d'une architecture orientée services

Plusieurs travaux ont proposé des solutions pour améliorer le processus de découverte de services Web. Ces solutions sont principalement élaborées autour de la distribution d'annuaires de services [Verma 2005, Pilioura 2004], l'utilisation de la sémantique pour la description de services [Lausen 2007, Roman 2005, Martin 2004b] et la prise en compte des critères de qualité de services [Xu 2007b, Maximilien 2004, Ran 2003].

Dans ce chapitre, nous introduisons différentes approches de découverte de services Web traitant d'une façon ou d'une autre nos objectifs abordés dans le chapitre précédent à savoir : proposer « une approche implicite pour organiser un ensemble d'annuaires », « une approche pour la gestion d'un ensemble d'annuaires organisés en groupes » et « une approche pour la découverte d'annuaires dans un réseau d'annuaires organisés en communautés ». Nous commençons par étudier des approches de découverte de services dans un réseau d'annuaires distribués et nous nous penchons sur l'aspect organisationnel d'un tel réseau dans la section 2.2. Nous étudions par la suite, dans la section 2.3, des travaux qui se sont intéressés à la gestion de communautés de catalogues et de services Web et nous identifions les points pouvant être projetés sur notre contexte. Dans la section 2.4, nous présentons des travaux utilisant des techniques de recommandations dans le but d'améliorer la découverte de services Web, et leurs limites. Enfin, nous concluons ce chapitre dans la section 2.5.

2.2 Découverte de services Web dans un environnement distribué d'annuaires

Plusieurs approches de découverte de services Web se sont intéressées à la découverte de services dans un environnement distribué d'annuaires de services [Verma 2005, Pilioura 2004, Sivashanmugam 2004, Pilioura 2009, Du 2005, Ayorak 2007, Xu 2007a]. L'utilisation de plusieurs annuaires dans un processus de découverte offre principalement l'avantage de dépasser le problème du « *single point of failure* » et permet d'éviter les goulots d'étranglements comme lors de l'utilisation d'un annuaire unique centralisé. De plus, il n'est pas réaliste de considérer un seul annuaire et un deman-

deur de services a de plus grandes chances pour retrouver « exactement » le service recherché en ayant accès à des annuaires de fournisseurs de services différents. En contrepartie, si le nombre de ces annuaires devient trop important, la découverte de services devient une tâche encombrante et son coût peut devenir pesant.

Afin d'améliorer le processus de découverte de services, les approches citées ci-dessus proposent de structurer leurs réseaux d'annuaires. Dans cette section, nous présentons les mécanismes d'organisation d'annuaires (section 2.2.1) et les mécanismes de découverte de services (section 2.2.2) proposées dans ces travaux. A travers l'étude de ces travaux, nous visons à répondre à ces questions :

- ***Quels sont les critères utilisés pour organiser un réseau d'annuaires*** ?
- ***Comment organise-t-on un réseau d'annuaires*** ?
- ***Quels sont les profits tirés pour la découverte de services Web*** ?

2.2.1 Mécanismes d'organisation d'un environnement distribué d'annuaires

Dans [Verma 2005], les auteurs présentent MWSDI : une infrastructure d'annuaires distribués pour la découverte de services Web. Pour faire face à l'augmentation du nombre d'annuaires et pour assurer un bon passage à l'échelle, MWSDI adopte une approche pair-à-pair (P2P) hybride. Dans ce travail, le réseau d'annuaires est structuré, sous forme de fédérations, en utilisant une ontologie d'annuaires appelée *Registries Ontology*. Cette ontologie permet de regrouper des annuaires sur la base de leur domaine métier. Le domaine métier d'un annuaire est spécifié par le fournisseur de l'annuaire qui devra l'associer à un ou plusieurs concepts de la *Registries Ontology*. Cette ontologie définit aussi les relations de type annuaire-annuaire pouvant exister entre les différents annuaires d'une fédération [Sivashanmugam 2004].

Dans la Figure 2.2, nous donnons un exemple d'utilisation de la *Registries Ontology* pour organiser un réseau d'annuaires. Chaque annuaire, membre de l'infrastructure MWSDI, doit être associé à une ou plusieurs classes (représentants des domaines métiers) de l'ontologie (*Registry 1* est associé à *Air Travel* par exemple). Cette association doit être faite par le fournisseur d'un annuaire lorsqu'il rejoint l'infrastructure. Si le fournisseur désire associer son annuaire à un domaine qui n'est pas défini par l'ontologie il devra mettre à jour l'ontologie pour pouvoir le faire. Dans MWSDI, les auteurs définissent un super pair (*Gateway Peer*) comme étant le « point d'entrée » à l'infrastructure et permettant aux fournisseurs de services de mettre à jour l'ontologie.

Dans ce même exemple, les deux annuaires *Registry 1* et *Registry 2* appartiennent à deux compagnies aériennes partenaires ainsi leurs fournisseurs peuvent définir une relation « *partnerOf* » pour l'exprimer. De telles relations permettent d'assurer une meilleure collaboration entre les différents annuaires et ainsi d'améliorer le processus de découverte de services Web.

Les auteurs dans [Pilioura 2004, Pilioura 2009], proposent une autre infrastructure d'annuaires distribués pour la découverte de services appelée PYRAMID-S.

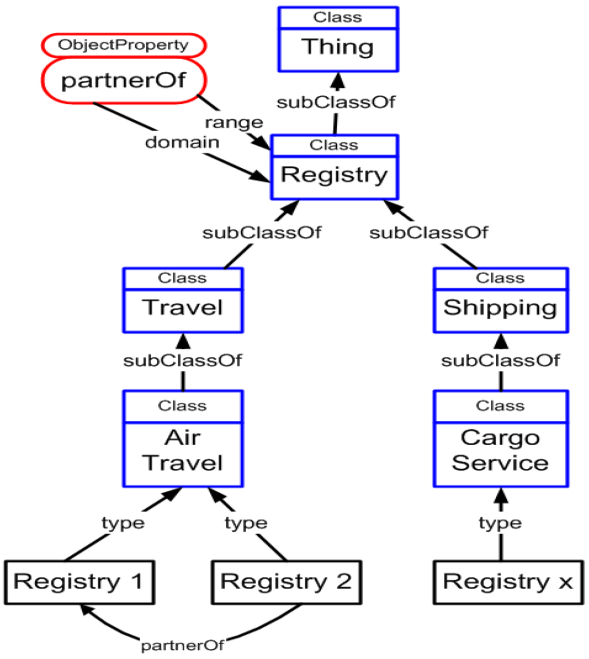


FIGURE 2.2 – Un extrait de la *Registries Ontology* [Verma 2005]

PYRAMID-S se base aussi sur une ontologie pour classifier un réseau d'annuaires. A la différence de MWSDI, PYRAMID-S permet aux fournisseurs d'annuaires d'éditer leurs classification à travers plusieurs pairs, à la place d'un seul, assurant ainsi une meilleure tolérance aux fautes.

PYRAMID-S est construite au-dessus de trois couches différentes. La *Registries Layer* qui contient les annuaires de services (UDDI, ebXML, etc.). La *Gateway Layer* formée par plusieurs pairs qui représentent les points d'accès au système PYRAMID-S. La *Routers Layer* formée par des pairs de routage dont la fonctionnalité est d'assister les *gateway peers* pour transmettre les requêtes/descriptions de services à l'annuaire adéquat.

PYRAMID-S utilise aussi une ontologie de classification de domaines (*DCO*) qui sert à définir les relations entre les différents domaines métiers et établir les association entre les annuaires et leurs domaines métiers (voir Figure 2.3). Un annuaire est représenté par un tuple $T_i =< R_i, D_i, A_i >$ dans la *DCO*. R_i représente l'URL de l'annuaire, D_i le (ou les) domaine(s) métier(s) auquel est associé l'annuaire (par exemple, *Registry Y* est associé à *Loan Service* et *Investment Service*) et A_i les propriétés de l'annuaire (type, fournisseur, etc.). L'organisation des annuaires dans PYRAMID-S est aussi assurée par les fournisseurs d'annuaires. L'ajout d'un nouvel annuaire par un fournisseur dans PYRAMID-S est fait à travers un *gateway peer*. Le fournisseur d'annuaire doit alors spécifier le domaine métier associé à son annuaire en parcourant la *DCO*.

Une autre infrastructure distribué d'annuaires est proposée dans [Du 2005]. Dans ce travail, les auteurs présentent Ad-UDDI ; une infrastructure active et dis-

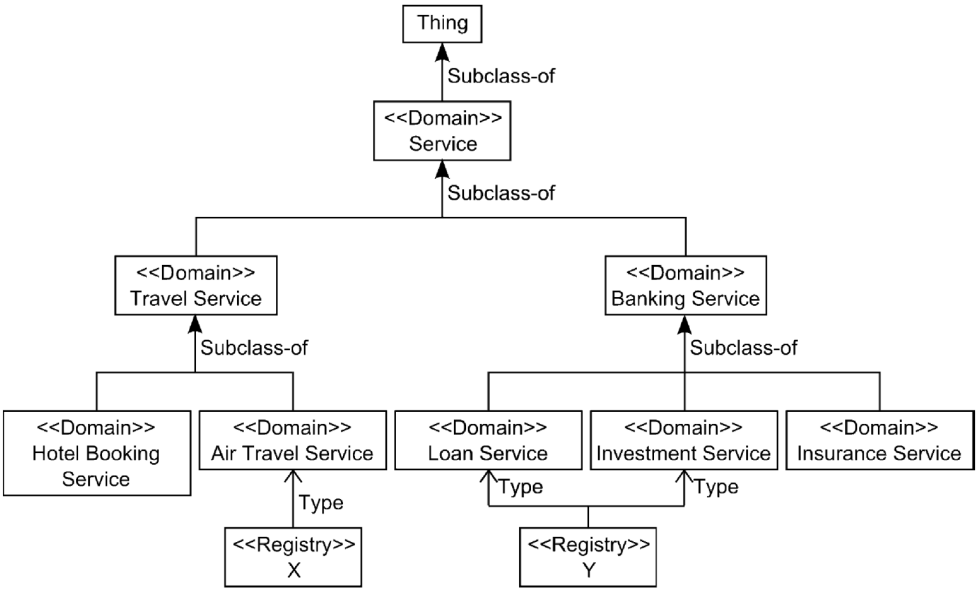


FIGURE 2.3 – Un extrait de l'ontologie de classification de domaines *DCO* [Pilioura 2009]

tribuée d'annuaires UDDI pour l'organisation d'un réseau d'annuaires privés et semi-privés. Ad-UDDI est composée de trois couches (voir Figure 2.4). La première couche, la *root registry layer*, contient un seul annuaire Ad-UDDI (*root registry*) cataloguant les informations concernant les annuaires formant le réseau d'annuaires. La deuxième couche, la *business service registry layer*, regroupe les annuaires Ad-UDDI offerts par les fournisseurs d'annuaires. Ces annuaires sont organisés selon la taxonomie GICS [1] classifiant divers secteurs d'activités industrielles, industries, sous-industries et groupes d'industries réelles. La troisième et dernière couche, *service layer*, contient les services énoncés par les annuaires de la *business service registry layer*.

Lorsqu'un nouvel annuaire Ad-UDDI rejoint le réseau, il commence par s'inscrire auprès du *root registry*. Son fournisseur devra spécifier la classe à laquelle il appartient selon la taxonomie utilisée. Le nouvel annuaire demande par la suite au *root registry* la liste des annuaires existants dans la même classe. Cette liste permet d'établir des relations entre les annuaires appartenant à une même classe.

Dans les infrastructures présentées ci-dessus (MWSDI, PYRAMID-S et Ad-UDDI), les annuaires de services utilisés sont privés ou semi-privés. Vu que ces annuaires offrent généralement des services d'un même domaine métier, l'organisation du réseau d'annuaires se fait en fonction des domaines associés aux annuaires indépendamment des services qu'ils offrent. Toutefois, ces approches d'organisation nécessitent une intervention humaine pour spécifier les domaines auxquels sont associés les annuaires et éventuellement pour mettre à jour l'ontologie afin de définir

1. GICS : Global Industry Classification Standard. http://www.mscibarra.com/products/indices/gics/gics_structure.html

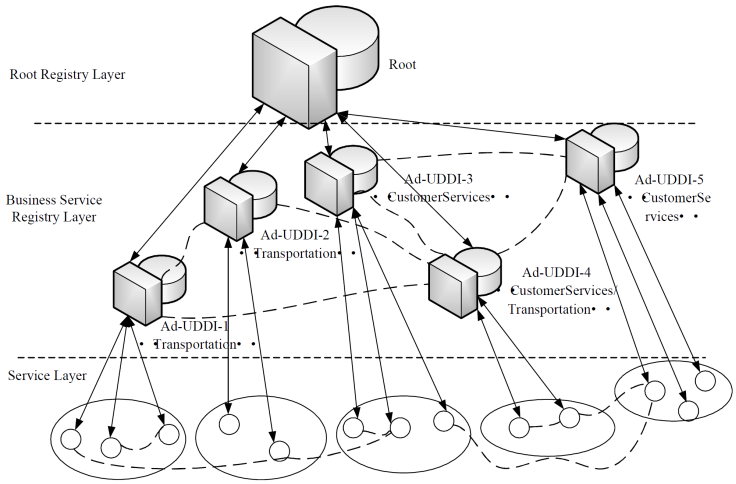


FIGURE 2.4 – L'architecture distribuée d'Ad-UDDI [Du 2005]

un nouveau domaine métier inexistant.

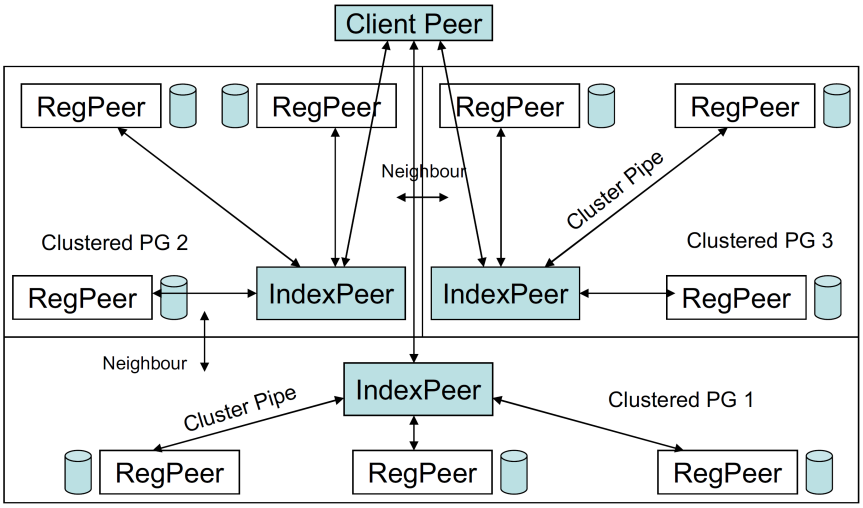
Dans [Ayorak 2007], une architecture hybride pour la découverte de services dans un réseau d'annuaires publics, appelé SPWSDA, est proposée. Contrairement aux approches précédentes, la structure du réseau est crée sur la base du domaine métier des services : les services opérant dans des domaines similaire sont placés dans le même cluster. SPWSDA est basée sur un réseau P2P structuré de *registry peers* (représentants des répertoires de descriptions de services Web).

La structure de l'architecture proposée est illustrée à travers la Figure 2.5(a). L'organisation en groupes, ici appelés *clusters*, est assurée à travers une ontology de domaine appelée *Vocabulary classification tree* décrite en OWL [McGuinness 2004] (voir Figure 2.5(b)). La mise en place de cette organisation impose aux fournisseurs de services : (i) d'utiliser OWL-S [Martin 2004a] pour la description des services Web et (ii) de définir une *classe mère* pour chacun de leurs services à partir des classes de services définis dans la *Vocabulary classification tree*.

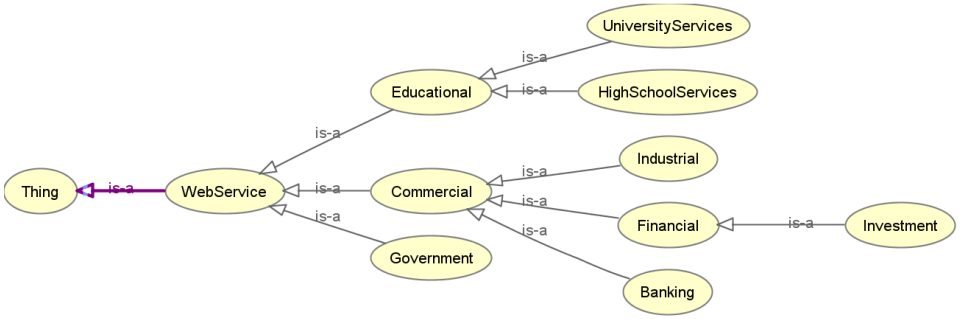
L'architecture proposée est formée par trois types de pairs (voir Figure 2.5(a)) :

- *Index peers.* Servent de super pairs dans cette architecture et indexent les informations contenues dans les *registry peers* localisés dans leurs cluster.
- *Registry peers.* Possèdent le rôle de dépôt pour les descriptions sémantiques (en OWL-S) et les informations d'invocation (en WSDL) des services. Ces données sont gardées dans des bases de données relationnelles.
- *Client peers.* Se sont des petites applications pour la publication et la recherche des services Web.

L'organisation des annuaires est initiée par la création de clusters selon la classification attribuée aux services. Chaque cluster est indexé par un *index peer* et formé par un ou plusieurs *registry peer*. L'*index peer* utilise une structure de données arborescente (permettant des recherches plus rapides) pour stocker les données d'indexations des *registry peer* de son groupe. A l'arrivée d'un nouveau service, le fournisseur lance une requête d'inscription pour son service à travers un *client peer*.



(a) Architecture SPWSDA



(b) *Vocabulary classification tree*

FIGURE 2.5 – Illustrations de l'architecture SPWSDA [Ayorak 2007]

Cette requête est transmise à l'un des *index peer*, qui la transmet aussi à ses voisins. Chaque *index peer* calcule alors le degré d'appartenance du service à son cluster selon le domaine auquel est associé le service. Le service est ainsi dirigé vers le groupe le plus adéquat selon les degrés calculés.

Les approches présentées ci-dessus, utilisent des annuaires *UDDI-like* pour contenir les descriptions de services. Nous définissons un annuaire de services *UDDI-like* comme étant une entité centralisée contenant des descriptions de services Web. Ces descriptions sont stockées dans l'annuaire selon un modèle de données prédéfini et sont consultables et interrogeables par tout demandeur de services. Dans ces approches, les annuaires sont organisés en fonction de leurs domaines métiers (MWSDI, PYRAMID-S et Ad-UDDI) ou celui des services (SPWSDA).

Dans [Xu 2007a], et contrairement aux approches précédentes, les auteurs proposent d'utiliser un annuaire de services décentralisé dont les descriptions de services sont contenues dans différents pairs d'un réseau P2P. Ils implémentent une architecture P2P où les différents pairs jouent à la fois le rôle de fournisseur de services et d'annuaire de services. Ces pairs sont organisés en fonction de mots-clés extraits à

partir de ces descriptions. Les descriptions de services ne sont pas publiées dans des annuaires de services (*UDDI-like*) mais elles sont disponibles sur le même serveur sur lequel leurs services sont déployés. Ce choix assure une meilleure disponibilité du service selon le raisonnement : si le service n'est plus disponible, sa description ne l'est plus non plus. Ces serveurs représentent les pairs d'un réseau P2P et sont organisés en groupes.

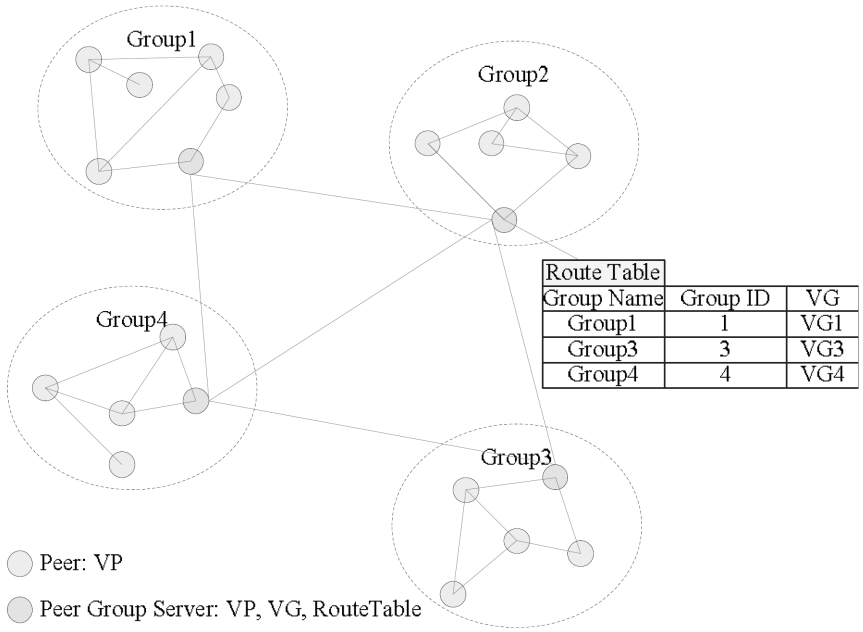


FIGURE 2.6 – Organisation P2P proposée par [Xu 2007a]

Dans ce travail, un service Web est décrit par une description de services en OWL associée à une description textuelle. L'organisation du réseau est faite en fonction des termes contenus dans la description textuelle. La Figure 2.6 illustre cette organisation. Pour mettre en place cette organisation, les auteurs associent un vecteur VP à chaque pair, construit à partir des termes contenus dans la description, et un vecteur VG à chaque groupe de pairs. Toutefois, les détails sur la l'identification et la création de groupe de pairs n'ont pas été fournis. L'appartenance d'un pair à un groupe est établi selon le degré de similarité entre son vecteur VP et le vecteur VG associé au groupe. Dans chaque groupe, le rôle de super pair est associé à l'un des pairs qui garde en plus de son vecteur VP, le vecteur VG associé au groupe et une table de routage (voir Figure 2.6). Cette table représente les « connaissances » d'un groupe et contient une liste des groupes les plus similaires.

Contrairement aux différentes approches d'organisation d'annuaires présentées dans cette section, l'organisation proposée dans [Xu 2007a] n'est pas basée sur les domaines métiers d'annuaires (décrits par une taxonomie). Dans [Xu 2007a], l'utilisation de mots-clés associés aux services Web permet d'acquitter les fournisseurs de l'entretien et de la mise à jour d'une taxonomie vu que l'organisation est assurée automatiquement à travers ces mots-clés. Toutefois, ils devront toujours intervenir

pour définir ces mots-clés.

La majorité des approches présentées dans cette section, à l'exception de [Xu 2007a], organisent un réseau d'annuaires en fonction de leurs domaines métiers. Nous montrerons dans la section 2.2.3 qu'une telle organisation est inadéquate et peut être inefficace dans certains cas et inférer un routage imprécis de la requête d'un demandeur de services.

2.2.2 Mécanismes de découverte de services

Dans cette section, nous étudions les approches de découvertes de services utilisées par les travaux précédents. Nous étudions principalement les profits tirés des mécanismes d'organisations d'annuaires présentées précédemment.

Dans MWSDI [Verma 2005], le fait d'associer à chaque annuaire un ou plusieurs concept(s) de la *Registries Ontology* permet d'améliorer le processus de découverte d'un service Web. En effet, une requête de découverte de services Web peut être guidée vers l'annuaire le plus adéquat en fonction des domaines d'intérêt du demandeur de services. Pour cela, le demandeur de services commence par formuler une requête pour la sélection d'annuaires. Dans cette requête l'utilisateur spécifie les noms des fédérations et les domaines métiers auxquels il s'intéresse, les relations annuaires-domaines utilisées, l'ensemble des ontologies supportées par les annuaires et l'ensemble des relations inter-annuaires. Une fois les annuaires sélectionnés, le demandeur de services formule un *Search Template* sémantique [Sivashanmugam 2003] dans lequel il spécifie ses besoins. Ce *Search Template* est utilisé pour la sélection de services et est propagé vers les annuaires sélectionnés dans les différentes fédérations. Finalement, les services Web sélectionnés seront transmis à l'utilisateur.

Dans PYRAMID-S [Pilioura 2004], l'approche de découverte de services proposée est semblable à celle proposée dans MWSDI. En effet, la structuration des annuaires selon leurs domaines métiers, permet d'appliquer un filtre et de sélectionner les annuaires adéquats aux besoins d'un demandeur de services. Un processus de découverte de services dans PYRAMID-S est initié par un demandeur de services en contactant un *gateway peer*. Il sélectionne son domaine métier à partir de la *DCO* et spécifie sa requête en utilisant la *SDO* associée au domaine choisi. Le système identifie alors les annuaires adéquats, selon le domaine métier du demandeur de services, et exécute la requête sur ces annuaires. De plus, par rapport à MWSDI, les auteurs utilisent des métriques de qualité de services (QdS) pour trier les résultats (services Web retournés) de la requête de découverte de services.

De même, la classification des annuaires dans [Du 2005] permet de réduire l'espace de recherche d'un demandeur de services en guidant sa requête vers les annuaires selon leurs classes. De plus, par rapport aux autres approches et afin d'améliorer l'efficacité du processus de découverte, chaque annuaire Ad-UDDI garde en cache les résultats des découvertes passés. En recevant une requête de découverte de service, un annuaire commence par vérifier le cache des découvertes passés. Si le service recherché est trouvé, l'annuaire envoie le résultat au demandeur de services et met fin à la requête. Sinon, il interroge son référentiel local et les annuaires voi-

sins. Une fois la requête résolue, l'annuaire transmet la réponse au demandeur de services et met à jour son cache.

Les auteurs dans [Ayorak 2007] tirent profit de l'architecture P2P (SPWSDA) utilisée pour distribuer les annuaires en se basant sur des mécanismes de routage P2P pour améliorer le coût du processus de découverte. Un processus de découverte est initié par un *client peer* qui transmet la requête d'un demandeur de services à un *index peer*. Ce dernier recherche les *registry peers* adéquats dans son cluster et leur transmet la requête. Il transmet aussi la requête à ses voisins. Les résultats trouvés seront transmis directement au *client peer*. Les *index peer* communiquent en utilisant le protocole de routage P2P CAN [Ratnasamy 2001] afin d'éviter de saturer le réseau P2P avec des messages inutiles. Vu que ces *index peer* représentent des super pairs pour les groupes d'annuaires de SPWSDA, les requêtes de découverte de services sont routées par les *index peer* vers les annuaires adéquats.

Dans [Xu 2007a], une requête de découverte de services est routée vers le groupe de pairs le plus similaire en utilisant des mots-clés qui devront être spécifiés avec la requête. Par la suite, un appariement sémantique entre la requête et l'ensemble des descriptions de services offertes par ce groupe est réalisé pour répondre à la requête. Ceci est assuré en diffusant la requête à tous les annuaires du groupe sélectionné.

A travers les approches de découverte de services Web présentées, il est clair que le fait d'organiser un réseau d'annuaires permet d'améliorer le processus de découverte en limitant l'espace de recherche d'un demandeur de services. Toutefois, ce « filtrage » est fait sur la base des domaines métiers des annuaires et ne prend pas en considération les besoins fonctionnels et non fonctionnels du demandeur de services.

2.2.3 Synthèse

Nous avons présenté différents systèmes et approches de découverte de services Web dans un environnement distribué d'annuaires. Notre étude permet de répondre ainsi aux trois questions posées plus haut :

- ***Quels sont les critères utilisés pour organiser un réseau d'annuaires*** ?
  Toutes les approches étudiées, à l'exception de [Xu 2007a], proposent d'organiser un réseau d'annuaires en fonction de leurs **domaines métiers** ou celui des services Web (par exemple e-travel, marketing, etc.). Ces domaines métiers, représentés par des concepts sémantiques d'une ontologie [Verma 2005, Sivashanmugam 2004] ou des classes d'une classification [Du 2005], sont associés par le fournisseur de services à son annuaire ou bien aux services Web qu'il propose. Dans [Xu 2007a], l'organisation du réseau est faite en fonction de **mots-clés** qu'un fournisseur de services associe aux services Web qu'il offre.
- ***Comment organise-t-on un réseau d'annuaires*** ?
  L'organisation des annuaires dans ces approches est **semi-automatique** et nécessite une intervention humaine. Le fournisseur de services ou d'annuaires devra spécifier le domaine métier associé aux ressources, annuaires de services

Web ou services Web, qu'il propose. De plus, la taxonomie ou l'ontologie utilisée pour structurer le réseau d'annuaires devra être mise à jour chaque fois qu'un nouveau domaine métier est identifié. A titre d'exemple, la classification utilisé dans [Du 2005] contient 256 classes dont seulement 10 représente des secteurs d'activités. Dans [Xu 2007a], l'organisation est assurée automatiquement mais toutefois le fournisseur de services devra intervenir pour spécifier les mots-clés associés à chaque service offert.

– ***Quels sont les profits tirés pour la découverte de services Web*** ?
Le fait d'organiser les annuaires de cette façon permet d'améliorer le processus de découverte en limitant l'espace de recherche d'un demandeur de services. En effet, organiser un réseau d'annuaires en groupes permet de router directement une requête de découverte vers le groupe qui lui est adéquat. Ce routage est fait sur la base des **domaines métiers des annuaires** [Verma 2005, Pilioura 2009, Du 2005], des **domaines métiers des services Web** [Ayorak 2007, Papazoglou 2003] ou de **mots-clés** associés aux services Web [Xu 2007a].

Cependant, une organisation d'un réseau d'annuaires en fonction de leurs domaines métiers peut être inefficace dans certains cas et inférer un routage imprécis de la requête d'un demandeur de services. Ceci pourra être l'exemple d'un demandeur de services cherchant un service lui permettant de regarder un match de football en ligne et qui est dirigé vers un annuaire offrant des services de billetterie en ligne pour ce match. En effet, une organisation par **domaine métier** est plus efficace en recherche d'information mais en recherche de services c'est surtout une **fonctionnalité** qui est recherchée.

De plus, les approches proposées pour organiser ces réseaux d'annuaires **ne sont pas automatiques** et **nécessitent une intervention humaine** ce qui rend ces organisations prédisposées aux erreurs. Ceci constitue un obstacle à la prolifération de ces approches dans un contexte B2B où les fournisseurs d'annuaires peuvent ne pas avoir les compétences et/ou la volonté pour participer à l'organisation des annuaires.

Aussi, en structurant leurs réseaux d'annuaires ces travaux ont pu limiter l'espace de recherche d'un demandeur de services. Toutefois, même en limitant la recherche à un seul groupe d'annuaires, la découverte d'un service reste une tache encombrante spécialement si le nombre d'annuaires dans le groupe est important. Dans MWSDI [Verma 2005], pour éviter qu'un demandeur de services diffuse sa requête vers tous les annuaires d'un groupe, ce dernier doit intégrer dans sa requête les domaines métiers auxquels il s'intéresse et les relations à considérer entre annuaires [Sivashanmugam 2004]. En se basant sur des ontologies de domaines et l'ontologie des annuaires, les annuaires adéquats aux domaines métiers spécifiés par le demandeur de services sont sélectionnés. Pour SPWSDA [Ayorak 2007], les auteurs utilisent un *index peer* dans chaque *cluster* pour l'indexation des *registry peers*. Ainsi, quand un *index peer* reçoit la requête d'un demandeur de services, il la transmet uniquement aux annuaires adéquats et éventuellement à ses voisins.

Dans les travaux présentés, l'espace de recherche d'un demandeur de services (c.-à-d. l'environnement distribué d'annuaires) est généralement réduit à un ou plu-

sieurs annuaires sur la base du domaine métier du service recherché. Vu qu'un demandeur de services est à la recherche d'une fonctionnalité, ses **besoins fonctionnels** devront aussi être pris en considération lors de la réduction de l'espace de recherche. Aussi, les **besoins non fonctionnels** du demandeur de services (coût du service, temps de réponse, disponibilité, etc.) représentent un critère assez important pour la sélection d'un service lors du processus de découverte. Toutefois, les travaux étudiés ici, à part [Pilioura 2004] qui utilise la QdS pour classer les résultats, ne prennent pas en considération ce critère pour la découverte d'un service.

Dans cette thèse, nous proposons une approche **automatique** et **implicite** pour organiser un réseau d'annuaires selon les **fonctionnalités** des services Web annoncés par ces annuaires. Aussi, nous proposons une approche de découverte de services en tenant compte des **besoins non fonctionnels** d'un demandeur de services en plus de ses besoins fonctionnels.

2.3 Gestion de communautés

Dans un contexte B2B, chaque compagnie possède un référentiel de services (c.-à-d. un annuaire) contenant les descriptions de services qu'elle offre. Pour alléger le processus de découverte de services dans ce contexte, l'ensemble de ces référentiels peut être organisé en groupes selon différents critères (voir section 2.2.1). Dans notre travail, nous proposons d'organiser un tel réseau d'annuaires en communautés selon les fonctionnalités offertes par les services. Vu la dynamicité d'un environnement orienté services, des mécanismes de gestion sont nécessaires pour conserver la consistance d'une telle organisation. Bien que cette problématique représente actuellement un enjeu essentiel pour la stabilité des architectures de découvertes de services Web, elle n'a pas été traitée par les approches introduites dans la section 2.2 optant pour l'organisation d'un réseau d'annuaires en groupes ou communautés. Cependant, cette problématique a été traitée par d'autres approches dans le contexte de communautés de catalogues [Paik 2005] et de communautés de services Web [Medjahed 2005, Maamar 2009]. Dans cette section, nous présentons un recueil sur les mécanismes de gestion utilisés dans ces approches.

2.3.1 Les communautés de catalogues

Dans [Paik 2005], Paik et al. présentent un système appelé WS-CatalogNet proposant de regrouper des catalogues de produits en communautés. Un catalogue est défini comme étant un ensemble de produits organisés selon une catégorisation et qui est consultable en ligne [Paik 2004]. Une communauté de catalogues est un ensemble de catalogues de produits partageant le même **domaine métier**. Dans le système proposé, des relations peuvent être définies entre deux communautés. En effet, les auteurs associent à chaque communauté des catégories et une relation est établie entre deux communautés si leurs catégories sont similaires.

Deux types de relations sont définis entre les communautés : `SubCommunity-of` et `PeerCommunity-of`. `SubCommunity-of` représente une relation de spécialisation

des catégories des deux communautés. Chaque communauté possède au moins une *superComunity*. La relation `PeerCommunity-of` entre deux communautés indique qu'une communauté peut être une alternative pour l'autre. Les différentes communautés coopèrent donc entre elles pour traiter la requête d'un utilisateur.

Dans [Paik 2002], les auteurs définissent un réseau de communautés par un graphe orienté et pondéré $G = (N, E_1, E_2, W, l)$ avec :

- N est un ensemble fini de nœuds dont chacun représente une communauté de catalogue
- E_1 et E_2 sont deux ensembles finis d'arcs représentant respectivement les relations `Subcommunity-of` et `PeerCommunity-of` entre les communautés
- $W : E_2 \longrightarrow [0, 1]$ est une fonction attribuant un poids pour chaque arc décrivant une relation `PeerCommunity-of`
- l une fonction attribuant un nom pour chaque communauté

Le système offre également des fonctionnalités de monitoring pour surveiller les interactions de l'utilisateur et des opérations de gestion pour restructurer, en fonction de ces interactions, le réseau de communautés si nécessaire. Les opérations de gestion définies, appelées opérations de restructuration dans ce travail, sont : la création, le démontage, la division et la fusion de communautés. Pour chacune de ces opérations, les auteurs définissent les préconditions nécessaires à leur exécution ainsi que les effets sur le réseau G.

Par exemple, l'opération de suppression d'une communauté de catalogues est conditionnée par son existence dans le graphe G : $c \in N \backslash \{AllCatalog\}$. En supprimant une communauté, les effets de cette opération sont spécifiés comme suit (voir Figure 2.7) :

- Supprimer la relation *SuperCommunity-of* de c.
- Supprimer toutes les relations `SubCommunity-of` de c
- Supprimer toutes les relations `PeerCommunity-of` entrantes à c.
- Supprimer toutes les relations `PeerCommunity-of` sortantes de c.
- Relier, par des relations `SubCommunity-of`, toutes les communautés isolées à la *SuperCommunity* de c.

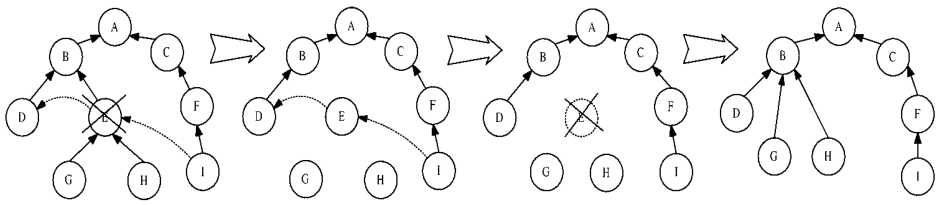


FIGURE 2.7 – Suppression d'une communauté [Paik 2002]

2.3.2 Les communautés de services Web

Dans la communauté académique des services Web, nous pouvons identifier deux visions différentes concernant le concept de communautés de services Web : la vision

de Medjahed et Bouguettaya et la vision de Maamar et al. Dans cette section, nous introduisons ces deux visions et nous présentons les mécanismes de gestion utilisés.

2.3.2.1 Vision de Medjahed et Bouguettaya

Dans [Medjahed 2005], Medjahed et Bouguettaya proposent une approche pour regrouper les services Web ayant un même **domaine d'intérêt** dans un même cluster appelé communauté. Cette organisation est assurée en utilisant une *community ontology* œuvrant comme un modèle pour la description des communautés de services Web. Les communautés représentent des instances de cette ontologie.

La Figure 2.8 illustre le processus de création d'une communauté de services Web. Une communauté de services Web C_i est « créée » par un fournisseur de communautés. Elle représente une instance de la *community ontology* et le fournisseur de communautés associe alors des valeurs aux attributs et aux concepts correspondants dans l'ontologie (Figure 2.8 étape (a)). En même temps, une communauté est « créée » en tant que service Web et publiée dans un annuaire de services pour qu'elle puisse être découverte par les fournisseurs de services (étape (b)). Les membres d'une communauté (c.-à-d. ses services Web) sont ajoutés par les fournisseurs de services. Pour cela, le fournisseur de services identifie la communauté adéquate à son intérêt (étape (d)) et publie ses services (étape (c)).

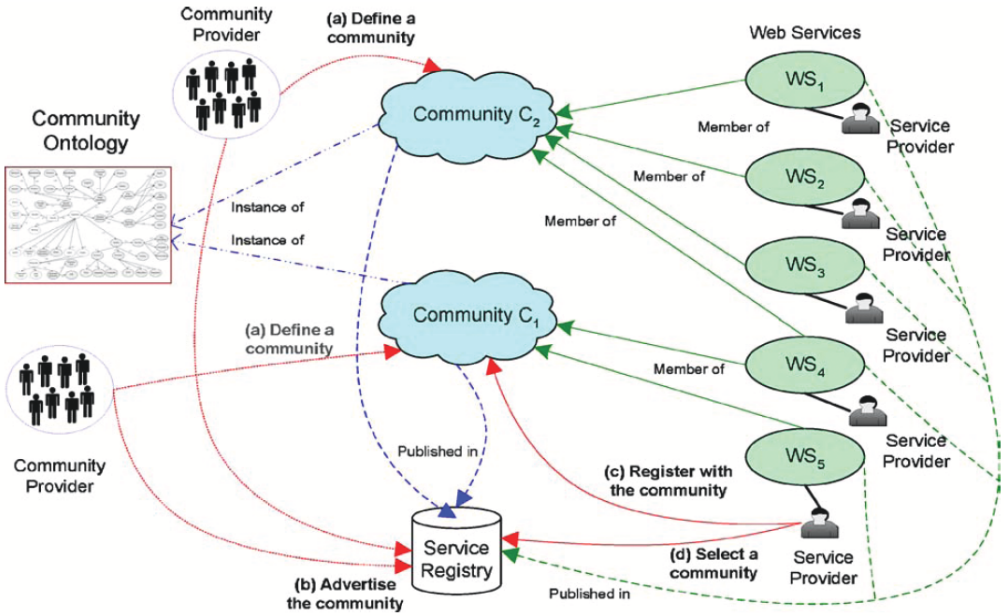


FIGURE 2.8 – Architecture de communautés de services Web [Medjahed 2005]

Une communauté C_i est décrite par un tuple ($Identifier_i$, $Category_i$, $G-operation_i$, $Members_i$). $Identifier_i$ contient un nom unique et une description textuelle décrivant les caractéristiques de la communauté. $Category_i$ décrit son

domaine d'intérêt. $G - operation_i$ est la liste des *opérations génériques* offertes par la communauté. Ces *opérations génériques* sont des opérations abstraites dont au moins une doit être offerte par un service Web pour qu'il puisse être membre de cette communauté. $Members_i$ est la liste des membres de C_i.

Les fournisseurs de communautés peuvent ajouter, supprimer ou modifier des *opérations génériques*. De même, les fournisseurs de services peuvent supprimer un service Web d'une communauté, rendre ses opérations indisponibles temporairement ou bien modifier la définition de ses opérations. Ces changements peuvent rendre inadéquate l'organisation des services Web en communautés (par exemple des services Web référençant des *opérations génériques* qui ont été supprimées) et nécessitent des mécanismes de gestion. Les auteurs proposent ainsi une approche P2P pour gérer les communautés. En effet, tous les changements qui peuvent avoir lieu sont déclenchés par les fournisseurs de communautés et de services. Un réseau P2P d'agents associés aux fournisseurs et aux communautés est établi. Ces agents interagissent automatiquement avec leurs pairs pour propager tout changement et éviter d'éventuels conflits.

2.3.2.2 Vision de Maamar et al.

Maamar et al. [Maamar 2009] définissent une communauté de services comme étant un ensemble de services Web offrant des **fonctionnalités** communes. Une communauté offre ainsi une description d'une fonctionnalité spécifique sans se référer à un service Web concret. La Figure 2.9 illustre l'architecture d'une communauté de services Web. Pour faciliter la gestion d'une communauté, les auteurs attribuent le rôle de *master* à un service Web dans chaque communauté. Le rôle principal du *master* est la gestion d'une communauté et notamment ses membres appelés *slaves*.

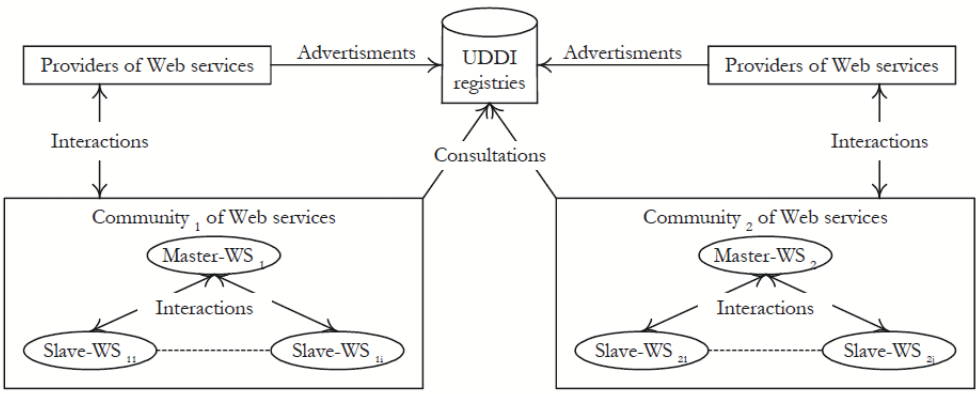


FIGURE 2.9 – Architecture de communautés de services Web [Maamar 2009]

Une communauté de services est dynamique de nature : de nouveaux services Web peuvent rejoindre une communauté, d'autres peuvent la quitter et certains services Web peuvent être temporairement indisponibles. Ces événements peuvent

causer des conflits et les auteurs présentent les opérations de gestion nécessaires. Ces opérations sont gérées par deux protocoles : *Web services Community Development Protocol* (`WSCDProtocol`) et *Enhanced Contract-Net Protocol* (`ECNProtocol`). Le `WSCDProtocol` s'intéresse à la gestion des opérations de développement d'une communauté, à savoir : la création, le démontage ainsi que l'attraction et la rétention d'un service Web dans une communauté. Le `ECNProtocol` gère les interactions lors du processus de sélection d'un service Web.

Pour l'établissement d'une communauté par exemple, le *WSCDProtocol* spécifie les opérations résultantes de cette création et définit les mécanismes de gestion conséquents. Cette création est déclenchée par le fournisseur de communauté qui identifie la nouvelle fonctionnalité de la communauté et désigne son *master*. Ce dernier se charge par la suite du reste des opérations de création de la communauté en invitant des *slaves* pour la rejoindre. Ensuite, si le *master* remarque que la communauté est devenue obsolète (ayant peu de membres), elle sera automatiquement démontée.

Dans [Bentahar 2007, Bentahar 2008], un *framework* formel pour la gestion des communautés utilisant des agents argumentatifs est proposé. Un agent argumentatif est associé à chaque service Web lui permettant ainsi de prendre des décisions et de réaliser des conversations complexes avec les autres services. Ce *framework* permet de rendre les services Web autonomes et supporte ainsi une gestion automatique des communautés de services. Ceci est assuré par l'utilisation de la logique argumentative [van Eemeren 1996] par les agents surtout utile lors des négociations entre agents. Les services Web peuvent ainsi établir une connaissance commune des engagements des uns et des autres, proposer des compromis et persuader d'autres agents de rejoindre leur communauté. Dans ce travail, les auteurs ont abordé les opérations de gestion liées à la création/le démantèlement d'une communauté et l'attraction/rétention de services Web à une communauté.

2.3.3 Synthèse

Lors de l'étude des travaux sur la gestion de communautés de catalogues et de services Web, nous avons constaté que la gestion des communautés est généralement établie suite à un changement engendré par les **fournisseurs de services**, les **fournisseurs de communautés** ou les **utilisateurs**. En effet, dans [Paik 2005] Paik et al. s'intéressent à la restructuration d'un réseau de communautés de catalogues en se basant sur l'interaction des utilisateurs avec ce réseau. Dans [Medjahed 2005] et [Maamar 2009], les auteurs s'intéressent à l'aspect dynamique des communautés (arrivée/départ d'un service) et gèrent par la suite les changements déclenchés par les fournisseurs de services Web. Medjahed et Bouguettaya [Medjahed 2005] gèrent en plus les changements déclenchés par les fournisseurs de communautés.

Pour remédier aux conflits potentiels pouvant se produire suite à ces changements, toutes les approches proposent des mécanismes de gestion. Pour faciliter la spécification de ces mécanismes, Paik et al. [Paik 2002] modélisent leur réseau de communautés de catalogues en utilisant la théorie des graphes. Les autres tra-

vaux [Medjahed 2005, Maamar 2009] n'ont pas présenté un modèle et leurs descriptions sont plutôt informelles. Les opérations de gestion proposées par Maamar et al. [Subramanian 2007] sont relatives à la création/suppression d'une communauté, l'attraction et la rétention d'un service dans une communauté. Paik et al. [Paik 2002] présentent les opérations de création, suppression, fusion et division de communautés. Ces opérations sont définies sous forme d'opérations sur le modèle du réseau de communautés de catalogues défini en utilisant la théorie des graphes.

Nous rappelons que nous nous intéressons dans notre travail à la gestion d'un réseau d'annuaires organisé en communautés afin d'assurer sa consistance. Le contexte des travaux étudiés dans cette section est certes différent mais plusieurs points peuvent être adaptés et adoptés pour notre travail. Par exemple, l'idée de modéliser les communautés en utilisant la théorie des graphes pour faciliter la spécification des mécanismes de gestion dans [Paik 2002] est aussi avantageuse dans notre contexte. Toutefois, les opérations de gestion spécifiées, notamment la fusion et la division de communautés, ne sont pas applicables dans notre contexte. En effet, dans notre travail un annuaire a une appartenance inclusive aux communautés (c.-à-d. peut appartenir à une ou plusieurs communautés) alors que dans ces travaux les membres d'une communauté (catalogues, services Web) ont des appartenances exclusives.

2.4 Systèmes de recommandation pour la découverte de services Web

Les systèmes de recommandation offrent une solution intéressante aux problèmes liés à la manipulation de grands volumes de données. Pour recommander un élément (produit, document, service, etc.), ces systèmes se basent sur des données telles que les profils d'utilisateurs, les expériences passées et les notes attribuées aux éléments par des utilisateurs précédents. Avec la prolifération des applications basées sur les services Web et l'augmentation conséquente du nombre de services Web, les systèmes de recommandation devraient aider à faire face aux questions récurrentes qui freinent l'acceptation des services Web : Comment découvrir les services Web avec un coût et un effort réduits ?, Comment suggérer/éviter le « bon »/« mauvais » service Web sur la base d'utilisations passées ?, etc.

Les systèmes de recommandation collectent des données afin de pouvoir conseiller un demandeur de services dans le choix d'un service. Selon la méthode utilisée dans la collecte de ses données, nous distinguons des techniques de recommandations **implicites** (collectent ces données sans interagir avec l'utilisateur) et **explicites** (interagissent avec l'utilisateur pour collecter ces données). Dans cette section, nous étudions des approches ayant utilisé des techniques de recommandation explicites et implicites pour la découverte des services Web.

2.4.1 Techniques de recommandations explicites

Dans [Manikrao 2005], Manikrao et al. proposent un système de découverte de services Web combinant une technique d'appariement sémantique à un système de recommandation. Dans ce système, les fournisseurs de services devront fournir la description WSDL du service qu'ils proposent ainsi qu'une description sémantique écrite en DAML-S [2] contenant le profil sémantique du service ainsi que ses besoins en terme de QdS.

Un processus de découverte de services dans ce système est initié par la réception d'une requête de découverte de la part d'un demandeur de services. Cette requête exprime sémantiquement son besoin fonctionnel et éventuellement ses besoins en termes de QdS. Un appariement sémantique est alors effectué entre la requête et les descriptions de services offertes par le système. Cette étape fournit une liste de services répondant aux besoins exprimés par la requête. Cette liste est transmise à un système de recommandation qui, sur la base des évaluations d'utilisateurs précédents, la trie et la transmet au demandeur de services. Quand un demandeur de services exécute un service, le système de recommandation lui demande de l'évaluer afin de pouvoir assurer les futures recommandations.

Dans [Mehta 2004], les auteurs s'intéressent aux problèmes liés au courtage de services Web dans des architectures en grille. Ils affirment que les descripteurs syntaxiques (comment est décrit le service) et sémantiques (que fait le service) associés aux services sont insuffisants pour découvrir un service Web qui répond le mieux aux besoins d'un demandeur de services. Ils proposent d'ajouter deux dimensions supplémentaires pour la description de services, à savoir : la qualité de services (quelle est la fiabilité du service) et les modèles d'utilisation (avec quels services est-il utilisé) (*quality and usage patterns*). Sur la base de ces quatre dimensions, les auteurs proposent une architecture pour un système de médiation de services assisté par un système de recommandation. Dans l'architecture proposée, les descripteurs syntaxiques de services sont contenus dans un annuaire de services. Les autres données (sémantique, QdS et modèles d'utilisation) sont regroupées dans un autre dépôt de données. Les auteurs définissent un *tracking component* et un *analysis component* surveillant les services Web du système afin de collecter les données de QdS et les modèles d'utilisation. Les auteurs offrent deux approches pour la découverte d'un service Web, en utilisant un appariement ou une recommandation :

– Le processus d'appariement est initié par la formulation d'une requête de découverte de service par le demandeur de services. La requête contient la description du service recherché selon les quatre dimensions proposées. Cette requête est par la suite transmise à un *mediation component* qui applique un algorithme d'appariement entre la requête (les quatre dimensions) et les services Web disponibles en utilisant les données collectées par le système. La dimension sémantique est au « centre » de l'algorithme proposé. Toutefois, la QdS et les modèles d'utilisation jouent un rôle important pour classer les services.

2. DARPA agent markup language for services, www.daml.org

– Le processus de recommandation est initié sans formulation d'une requête mais en utilisant le modèle d'utilisation du demandeur de services. Quand un demandeur de services contacte le *mediation component* pour une recommandation, ce dernier récupère son modèle d'utilisation à partir d'un *Usage pattern repository*. Le *usage pattern component* cherche alors des demandeurs de services présentant des modèles d'utilisation similaires et retourne au demandeur de services la liste des services utilisés par ces utilisateurs. La technique de recommandation proposée tient aussi compte des évaluations données par les utilisateurs passés à propos des services Web qu'ils ont exécutés. Ces évaluations sont collectées par un *feedback component*.

2.4.2 Techniques de recommandations implicites

Dans [Birukou 2007], Birukou et al. proposent d'utiliser un système de recommandation implicite pour améliorer le processus de découverte d'un service Web. La recommandation intervient lors de la sélection de services et est basée sur les expériences de découverte de services d'autres utilisateurs. Plus précisément, leur système de recommandation utilise de données prises à partir des requêtes de découverte et des invocations/exécutions de services correspondant sans aucune interaction explicite avec les utilisateurs du système.

L'*IC-service*, un service Web introduit dans ce travail, représente le cœur du système proposé. L'IC-service est responsable de la collecte des données d'observations relatives à chaque requête qui sont requises pour recommander les services adéquats à la requête d'un demandeur de services. Pour recommander un service à un utilisateur, l'*IC-service* calcule le degré de similarité entre sa requête et celles des utilisateurs précédents. Comme résultat, l'utilisateur reçoit la liste des services qui ont été utilisés dans le passé pour les requêtes similaires. Pour calculer la similarité entre les requêtes, les auteurs les représentent sous forme de vecteurs en utilisant les différents termes constituant les requêtes. Ils utilisent le modèle vectoriel VSM [Salton 1975] et la *term frequency-inverse document frequency* (TF-IDF) comme méthode de pondération. La similitude entre deux vecteurs (c.-à-d. requêtes) est alors calculée en utilisant la fonction cosinus combinée à une mesure de similarité sémantique basée sur Word-Net pour détecter les termes qui présentent un même sens [Seco 2004].

Blake et al. [Blake 2007, Blake 2008] proposent un système de recommandation permettant de recommander des services Web à un utilisateur en aval à l'expression de sa requête ou de son besoin. Ce système utilise une technique d'appariement syntaxique sur la base de données collectées pendant les sessions opérationnelles de l'utilisateur sur le système (fichiers et messages échangés au cours de ses sessions). Par la suite, des termes extraits à partir de ces données sont utilisés pour identifier les services Web proposés par le système et qui peuvent être « pertinents » pour l'utilisateur. Enfin, les services Web les plus pertinents par rapport aux données capturées sont recommandés à l'utilisateur. Les facteurs de recommandations d'un service Web pour un utilisateur sont calculés en utilisant un appariement syntaxique

entre des termes extraits à partir des données utilisateur collectées et des termes extraits à partir des descriptions de services Web offerts par le système (précisément les valeurs des attributs `name` des éléments `message` et `operation`). Le choix d'un appariement syntaxique a été « imposé » afin de permettre des recommandations anticipées en fonction des données utilisateurs tout en assurant la performance du système. Pour palier au manque de sémantique des termes utilisés pour l'appariement, le système proposé tient compte des hétérogénéités syntaxiques pouvant être observées dans les descriptions de services (par exemple des services avec des noms d'opérations différents mais offrant une fonctionnalité similaire). Les auteurs ont ainsi identifié quatre spécificités observées à partir d'une étude de descriptions de services réels :

- L'utilisation d'une partie d'un mot pour le référencer (par exemple name=username)
- L'utilisation d'abréviations (par exemple building = bldg)
- Des mots différents mais ayant une sémantique similaire peuvent être identifiés à travers l'utilisation de parties communes (par exemple first_name = username)
- L'appariement de termes formés par moins de 3 caractères ou plus de 15 ne produit aucun résultat.

L'algorithme d'appariement syntaxique qu'ils proposent tient en compte de ces spécificités lors du calcul des facteurs de recommandation et essaye ainsi de palier en quelque sorte à l'absence de la sémantique.

Zhang et al. [Zhang 2007] proposent un système de recommandation pour une découverte dynamique et implicite de services Web. Ce système permet de recommander dynamiquement des services Web à un demandeur de services durant les différentes étapes d'un processus de composition de services. Ceci permet de fournir automatiquement des services « adaptés » aux besoins du demandeur de services durant les différentes étapes de composition d'un service.

Pour cela, les auteurs définissent un modèle, *Interest model*, pour la description des intérêts en termes de services du demandeur de services. Ce modèle est construit sur la base des paramètres de sortie des services Web précédemment choisis. Les auteurs associent à chacun de ces paramètres, en fonction du nombre et de l'ordre de sa détection, un poids indiquant son importance dans l'*Interest model*.

Dans ce travail les auteurs utilisent des services Web décrits sémantiquement et chaque paramètre est ainsi représenté par le concept de l'ontologie l'annotant dans la description. Ainsi, à chaque étape du processus de composition de services, le système recommande des services à l'utilisateur en fonction de son *Interest model*. La recommandation est assurée à travers un appariement sémantique entre les intérêts de l'utilisateur (représentés par les paramètres de sortie collectés) et les paramètres d'entrée des services Web disponibles. Pour assurer une meilleure précision dans l'appariement, les descriptions de services et les *user's Interest model* sont annotés sémantiquement.

2.4.3 Synthèse

Nous avons présenté différentes approches de découverte de services Web utilisant des techniques de recommandation. Dans les approches avec recommandation implicite [Birukou 2007, Blake 2007, Zhang 2007], la recommandation d'un service est faite sur la base d'**historique d'utilisations passées**. Ces données sont collectées à partir : d'expériences de découverte de services d'autres utilisateurs [Birukou 2007], des données échangées au cours des sessions opérationnelles du demandeur de services avec le système de découverte [Blake 2007] ou des paramètres de sortie des services Web précédemment choisis par le demandeur de services [Zhang 2007]. L'utilisation de l'historique des utilisations passées permet de recommander des services Web en tenant compte du « **comportement** » du demandeur de services et/ou celui des demandeurs de services passés. Cependant, ce critère peut **induire** la recommandation de « **faux positifs** » dans le cas où des utilisateurs auraient bien utilisé le service mais qui n'en seraient pas satisfaits par exemple. De plus, vu que les facteurs de recommandations sont calculés sur la base de requêtes et messages écrits par des utilisateurs (pouvant contenir des fautes d'orthographes, mots incomplets, etc.), les recommandations peuvent ne pas toujours répondre aux attentes et intérêts d'un demandeur de services. Ce dernier problème a été considéré dans l'approche de [Zhang 2007] qui utilise des concepts d'une ontologie comme termes pour le calcul des facteurs de recommandations.

Dans les approches avec recommandation explicite présentées par [Manikrao 2005, Mehta 2004], le critère principal utilisé pour la recommandation est **l'évaluation des utilisateurs passés**. Ce critère permet de recommander des services Web en tenant compte des **degrés de satisfaction** exprimés par les demandeurs de services passés et permet ainsi d'**éviter** les « **faux positifs** ». Toutefois, l'historique des utilisations reste un critère important et il a été combiné aux évaluations passées dans [Mehta 2004]. L'inconvénient majeur des techniques de recommandations explicites réside dans le fait qu'elles nécessitent une **intervention** de la part des utilisateurs de services alors que des recherches antérieures ont pu montrer que les utilisateurs sont très peu susceptibles de fournir des évaluations [Claypool 2001].

L'utilisation des techniques de recommandations pour la découverte de services Web permet de réaliser des découvertes de services ne nécessitant aucun effort de la part du demandeur de services qui n'a même pas besoin de formuler une requête [Blake 2007]. Ces techniques permettent aussi d'éviter les « *faux positifs* » et de découvrir le « bon » service en utilisant les évaluations d'autres utilisateurs [Manikrao 2005, Mehta 2004]. De plus, toutes les techniques proposées, à l'exception de [Manikrao 2005, Zhang 2007], permettent de remédier au problème de coût de découverte de services observé avec les approches de découverte de services conventionnelles (basées sur des techniques d'appariement sémantique par exemple). Cependant, les approches de découverte de services Web basées sur les techniques de recommandation présentent un **faible rappel et une faible précision**. En effet, en utilisant une technique de recommandation pour la sélection de services Web, le demandeur de services ne verra pas tous les services Web répondant

à ses besoins et pourra avoir un grand nombre de services ne répondant pas à ses besoins.

Dans notre travail, nous visons à tirer profit des avantages offerts par les techniques de recommandations pour la découverte de services Web tout en gardant un bon rappel et une bonne précision. En effet, nous utilisons une technique de recommandation que pour sélectionner un ou plusieurs annuaire(s) et ainsi limiter l'espace de recherche d'un demandeur de services. Une technique offrant un meilleur rappel et une meilleure précision, comme un appariement sémantique par exemple, est par la suite utilisée pour la sélection de services parmi ces annuaires.

2.5 Conclusion

Dans ce chapitre, nous avons d'abord présenté des approches de découverte de services Web dans un environnement distribué d'annuaires. Afin d'améliorer le processus de découverte de services, ces approches proposent d'organiser leurs annuaires. Cette étude nous a affirmé qu'une organisation d'un réseau d'annuaires permet d'améliorer le processus de découverte en limitant l'espace de recherche d'un demandeur de services. Nous avons aussi étudié les limites de ces approches selon leurs **critères d'organisation** et leurs **techniques d'organisation**. La majorité de ces approches structurent un réseau d'annuaires selon le **domaine métier** des annuaires ou de leurs services Web. Toutefois, la découverte d'un service est plutôt conduite par une fonctionnalité recherchée que par un domaine métier. Une organisation selon les **fonctionnalités offertes par les services Web des annuaires** est ainsi plus adéquate que celle basée sur le domaine métier. De plus, les approches proposées pour organiser ces réseaux d'annuaires **ne sont pas automatiques** et **nécessitent une intervention humaine** qui peut constituer un frein à leurs utilisations dans un scénario B2B. **Dans le chapitre 3, nous allons proposer une nouvelle approche automatique et implicite pour l'organisation d'un réseau d'annuaires en communautés selon les fonctionnalités offertes par leurs services Web.**

Une telle organisation devra être constamment mise à jour pour assurer sa consistance vu l'aspect dynamique dans lequel évoluent les services Web et leurs annuaires. Nous avons ainsi présenté dans la section 2.3 quelques travaux qui se sont intéressés à la gestion de communautés de catalogues et de services Web. Les mécanismes de gestion introduits dans ces travaux traitent des ressources (catalogues et services Web) organisées exclusivement en communautés et ne sont pas adaptés à notre contexte. **Dans le chapitre 4, nous allons proposer une approche de gestion de communautés d'annuaires traitant les différentes étapes du cycle de vie d'un annuaire au sein d'une communauté et d'une communauté d'annuaires.**

Enfin, nous avons étudié quelques approches de découverte de services Web utilisant des techniques de recommandation. L'utilisation des techniques de recommandation pour la découverte de services Web présente deux avantages majeurs :

(1) réaliser des découvertes de services en évitant les « *faux positifs* » grâce aux évaluations des autres utilisateurs et (2) surmonter le problème de coût de découverte observé avec les approches de découverte basées sur un appariement sémantique. Cependant, les approches utilisant un appariement sémantique restent plus efficaces en termes de rappel/précision qui est un critère aussi important que le coût pour un demandeur de services. **Dans le chapitre 5, nous allons présenter une nouvelle approche de découverte de services Web alliant techniques de recommandation et appariement sémantique tout en assurant un bon rapport *(rappel,précision)/(coût de découverte)***. Cette approche tiendra aussi compte des besoins non fonctionnels d'un demandeur de services lors du processus de découverte.

Chapitre 3

Communautés d'annuaires : La Construction

Sommaire

3.1 Introduction

Dans ce chapitre, nous présentons notre approche pour la construction de communautés d'annuaires. Rappelons que dans notre travail, nous nous intéressons à organiser un réseau d'annuaires de services Web afin d'améliorer le rapport (rappel,précision)/coût de découverte de services. Dans la section 2.2.1, nous avons présenté différentes approches de découverte de services Web dans des environnements distribués d'annuaires. Ces approches proposent d'organiser leur réseaux d'annuaires en groupes en fonction du domaine métier, des propriétés fonctionnelles ou non-fonctionnelles des services Web offerts. Des telles organisations d'annuaires

contribuent à améliorer le processus de découverte d'un service Web et en particulier le processus de découverte d'un annuaire de services Web. Dans notre travail, nous proposons d'organiser les annuaires de services Web sous forme de communautés en nous basant sur les fonctionnalités des services Web annoncés par ces annuaires.

Le reste de ce chapitre est organisé comme suit : Dans la section 3.2 nous introduisons les descriptions WSRD des annuaires. Nous présentons notre approche de création automatique de ces descriptions en utilisant une agrégation des descriptions de services d'un annuaire. Dans la section 3.3, nous présentons le concept de communauté que nous utilisons pour organiser un réseau d'annuaires. Nous spécifions trois modèles pour caractériser : un annuaire (section 3.3.3), une communauté d'annuaires (section 3.3.4) et un réseau de communautés d'annuaires (section 3.3.5). Dans la section 3.4, nous présentons notre approche de construction de communautés en utilisant les descriptions WSRD. Nous utilisons une technique de clustering floue qui permet, en utilisant uniquement les descriptions WSRD des annuaires, d'organiser automatiquement un réseau d'annuaires en communautés. Enfin, nous concluons dans la section 3.5.

3.2 Descriptions d'annuaires WSRD

Avant d'entamer ce chapitre, nous mettons à la disposition du lecteur quelques désignations utilisées pour la suite de ce mémoire :

- **Annuaire et service** : Nous désignons un « annuaire de services Web » par « annuaire » et un « service Web » par « service » .
- **Services d'un annuaire** : Le terme « services d'un annuaire » sert à désigner « l'ensemble des services Web dont les descriptions sont publiées dans cet annuaire » .
- **Fonctionnalités d'un annuaire** : Par « fonctionnalités d'un annuaire » nous désignons « les fonctionnalités offertes par les services de cet annuaire » .
- **Fonctionnalités d'une communauté** : Par « fonctionnalités d'une communauté » nous désignons « les fonctionnalités offertes par les annuaires de cette communauté » .

Dans notre travail, nous supposons que les fournisseurs de services utilisent le langage de description sémantique de services SAWSDL [Lausen 2007] pour décrire leurs services. Ce choix vient du fait que ce langage est largement utilisé pour décrire sémantiquement un service. Toutefois, nous avons tenu à proposer une approche adaptable avec des annuaires offrant des services décrits avec d'autres langages de description de services Web sémantiques tels que OWL-S [Martin 2004a], WSMO [De Bruijn 2005] ou YASA [Chabeb 2008].

Les fonctionnalités offertes par les services d'un annuaire, exprimées par leurs descriptions, représentent les fonctionnalités de l'annuaire. Par conséquent, une agrégation des descriptions de services d'un annuaire doit pouvoir refléter les fonctionnalités de l'annuaire. Nous caractérisons un annuaire par une *Web Service Registry Description* [Sellami 2010a], abrégée en WSRD et littéralement traduite de

l'anglais en « Description d'annuaire de services Web » . La description WSRD d'un annuaire résulte de l'agrégation des différentes descriptions de services qu'il annonce.

La description WSRD d'un annuaire est construite sur la base des descriptions de services qui y sont publiées. De ce fait, nous nous sommes basés sur le format de descriptions de services WSDL [Moreau 2007] pour définir une description WSRD avec laquelle nous visons à décrire un annuaire et non pas un service. Pour cela, nous utilisons seulement la partie abstraite d'une description WSDL [1] pour définir une description WSRD. La structure d'une description WSRD est présentée dans la Figure 3.1.

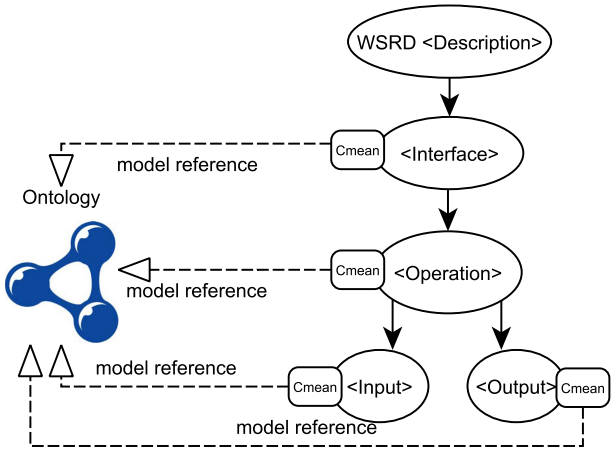


FIGURE 3.1 – Structure d'une description WSRD

Par analogie aux descriptions WSDL, une description WSRD caractérise un annuaire en utilisant les éléments WSDL suivants : `<interface>`, `<operation>`, `<input>` et `<output>`. Chacun de ces éléments est annoté par un (ou plusieurs) concept(s) sémantique(s) d'une ontologie en utilisant l'attribut `modelReference`. `modelReference` est un attribut introduit dans le langage SAWSDL et permettant de spécifier une association entre un élément WSDL et un concept dans un modèle sémantique. Les différents éléments d'une description WSRD résultent de l'agrégation par la moyenne des éléments des descriptions de services. Ainsi, ils offrent une description abstraite de la moyenne des fonctionnalités d'un annuaire.

La création d'une description WSRD d'un annuaire se fait en trois étapes :

– **étape 1** : Tout d'abord, on extrait les concepts annotant les descriptions de services publiés dans l'annuaire, ainsi que leur nombre d'occurrences (section 3.2.1). Nous appelons la description issue de cette étape la **WSRD « initiale »** .

1. WSDL est un langage de description de services Web basé sur XML. Il décrit la signature d'un service en utilisant des balises (aussi appelées éléments : <service>,<input>, etc.).

– **étape 2** : Ensuite, on calcule les nuages de concepts potentiels à l'annotation des éléments de notre description WSRD. Un nuage de concepts contient les concepts de l'ontologie utilisée pondérés par leur similarité à l'ensemble des concepts extraits à la première étape (section 3.2.2). Nous appelons la description résultante **WSRD « intermédiaire »** .
– **étape 3** : Finalement, on réduit les nuages créés pour ne garder que les concepts qui serviront à annoter notre description **WSRD « finale »** . Cette réduction est réalisée en sélectionnant à partir des différents nuages les concepts les plus représentatifs des concepts extraits à la première étape (section 3.2.3).

Dans la suite de cette section, nous détaillons et illustrons à travers un exemple les trois étapes de la construction d'une description WSRD. Dans ce qui suit, nous supposons que tous les fournisseurs de services utilisent les mêmes ressources sémantiques. Ceci est assuré par des ontologies communes ou par des mécanismes de médiations si différentes ontologies sont utilisées. De plus, et vu que nous traitons avec des annuaires privés, les descriptions des services annoncés par un annuaire sont homogènes. En effet, ces services ont le même domaine métier, utilisent une même sémantique et sont décrits en utilisant le même langage de description ce qui forme une condition essentielle pour la bonne exécution de notre approche.

3.2.1 Étape 1 : Extraction des concepts

Dans cette première étape, on extrait les concepts annotant les descriptions SAWSDL de services publiés dans l'annuaire pour lequel nous voulons construire la description WSRD. Les concepts annotant les éléments `<interface>`, `<operation>`, `<input>` et `<output>` des descriptions de services de l'annuaire ainsi que leur nombre d'occurrences forment la description WSRD « initiale » (voir Définition 3.1).

Dans la Figure 3.2, nous représentons cette première étape de la création d'une description WSRD. Nous illustrons la création d'une description WSRD « initiale » à partir de deux descriptions de services écrites en SAWSDL. Dans un souci de représentativité, nous prenons l'exemple d'un annuaire offrant seulement deux descriptions en SAWSDL et nous supposons que les concepts utilisés pour les annoter appartiennent à une petite ontologie de 7 concepts $\{C_1, C_2, \ldots, C_7\}$. En appliquant la première étape sur cet annuaire, les concepts annotant les différents éléments des deux descriptions de services seront extraits. Ces concepts et leur nombre d'occurrences seront conservés dans les ensembles correspondant de la WSRD « initiale » . Nous obtenons ainsi la description WSRD initiale (I, O, In, Out) telle que :

$$
\begin{aligned}
I &= \{(C_5, 1), (C_6, 1)\} \\
O &= \{(C_1, 1), (C_2, 3)\} \\
In &= \{(C_7, 4)\} \\
Out &= \{(C_3, 1), (C_4, 1), (C_5, 1), (C_6, 1)\}
\end{aligned}
$$

DÉFINITION 3.1 (WSRD « INITIALE »)
Une WSRD « initiale » est un quadruplet d'ensembles (I, O, In, Out). I (resp. O, In et Out) est l'ensemble contenant les concepts extraits relatifs à l'élément `<interface>` (resp. `<operation>`, `<input>` et `<output>`). Un élément de I (resp. O, In et Out) est un couple (C_i,nb_i) tels que :
- C_i représente un concept extrait de l'élément `<interface>` (resp. `<operation>`, `<input>` ou `<output>`) d'une description de service SAWSDL.
- $nb_i = f_I(C_i)$ (resp. $f_O(C_i)$, $f_{In}(C_i)$ et $f_{Out}(C_i)$) est un entier représentant le nombre de fois où le concept C_i a été identifié dans l'élément `<interface>` (resp. `<operation>`, `<input>` et `<output>`) parmi les différentes descriptions de services de l'annuaire.
- f_I (resp. f_O, f_{In} et f_{Out}) est la fonction permettant de récupérer le nombre d'occurrences nb_i d'un concept C_i dans l'ensemble I (resp. O, In et Out). Cette fonction est définie tel que $f_I(C_i)(resp.\ f_O(C_i),\ f_{In}(C_i)\ et\ f_{Out}(C_i)) = nb_i \Leftrightarrow (C_i, nb_i) \in I$ (resp. O, In et Out).
- $(C_1,nb_1) \in I$ (resp. O, In et Out) et $(C_2,nb_2) \in I$ (resp. O, In et Out) $\Rightarrow C_1 \neq C_2$

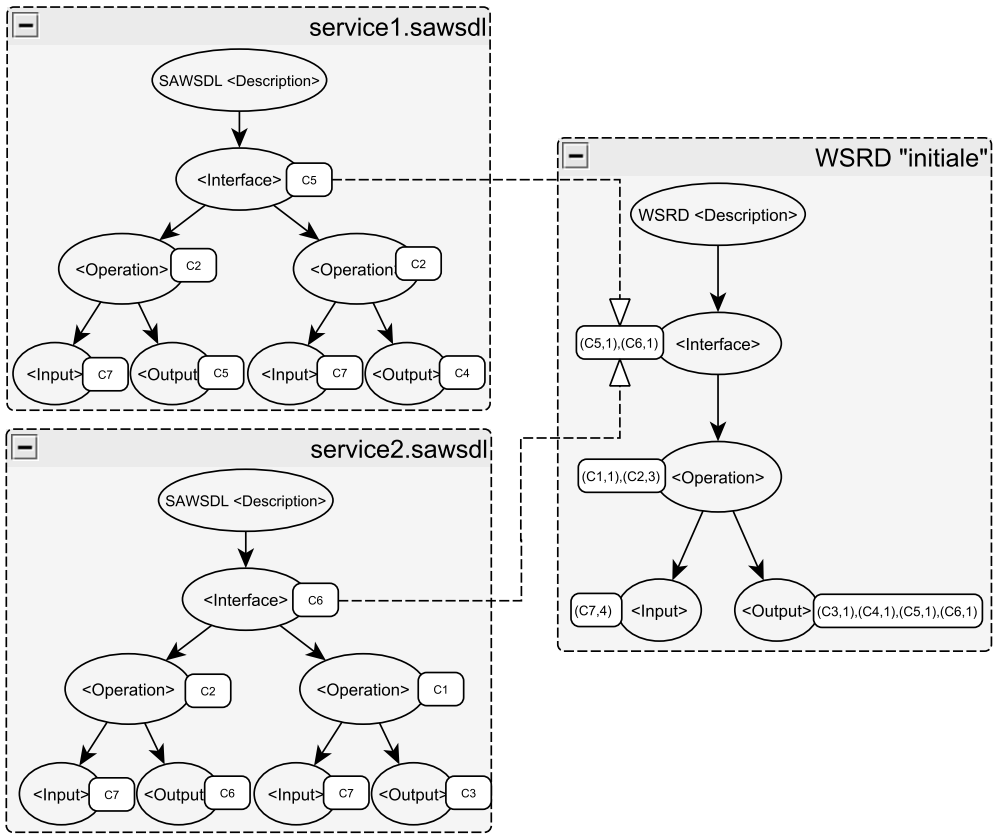


FIGURE 3.2 – Une description WSRD « initiale » résultant de deux descriptions SAWSDL

3.2.2 Étape 2 : Construction des nuages de concepts

Dans cette deuxième étape, nous calculons et associons un poids à chacun des concepts appartenant à l'ontologie utilisée pour annoter les descriptions de services. Ces poids seront utilisés dans la troisième étape pour sélectionner le ou les concepts qui serviront à annoter une description WSRD. Nous définissons ainsi pour chaque élément WSRD un ensemble de paires (*concept*,*poids*) où : *concept* représente un candidat pour annoter l'élément WSRD et *poids* la valeur permettant d'identifier le candidat le plus représentatif d'un élément WSRD. Nous appelons ces ensembles, formés par les paires (*concept*,*poids*), les nuages de concepts des éléments de la WSRD. Ces données nous permettent de créer la WSRD « intermédiaire » définie dans Définition 3.2.

Définition 3.2 (WSRD « intermédiaire »)
Une WSRD « intermédiaire » est un quadruplet d'ensembles $(H_I, H_O, H_{In}, H_{Out})$. Nous appelons H_I (resp. H_O, H_{In} et H_{Out}) le nuage de concepts relatifs à l'élément `<interface>` (resp. `<operation>`, `<input>` et `<output>`). Un nuage H_I (resp. H_O,H_{In} et H_{Out}) est un ensemble formé par des couples (C_i,s_i) tels que :

- C_i est un concept de l'ontologie utilisée pour annoter les descriptions de services.
- s_i est le poids associé à C_i. Il représente la moyenne pondérée des degrés de similarité entre C_i et les différents concepts associés à l'élément I (resp. O, In et Out) de la WSRD « initiale » . s_i, que nous dénotons aussi $f_{H_I}(C_i)$ (resp. $f_{H_O}(C_i)$, $f_{H_{In}}(C_i)$ et $f_{H_{Out}}(C_i)$), est calculé en utilisant la formule suivante :

$$f_{H_k}(C_i) = s_i = \frac{\sum_{j=1}^{t}(f_k(C_j) \times Similarity[C_i, C_j])}{\sum_{j=1}^{t} f_k(C_j)}, For\ i = 1 \dots t,\ k = I,\ O,\ In\ ou\ Out \tag{3.1}$$

  Où :
  - t : le nombre de concepts formant l'ontologie utilisée.
  - C_i, $i = 1 \dots t$: les concepts formant l'ontologie utilisée.
  - f_k : la fonction permettant de récupérer le nombre d'occurrences nb_j d'un concept C_j dans l'ensemble k créé de la première étape.
  - *Similarity* : une matrice contenant les degrés de similarité entre tous les concepts de l'ontologie utilisée.
- f_{H_k} est une fonction permettant de récupérer le poids associé à un concept d'un nuage H_k tel que $f_{H_k}(C_i) = s_i \Leftrightarrow (C_i, s_i) \in H_k$.
- (C_1,s_1) $\in H_k$ et (C_2,s_2) $\in H_k \Rightarrow C_1 \neq C_2$.

Dans le nuage de concepts d'un élément de cette WSRD « intermédiaire » , le *poids* associé à un concept indique son degré de représentativité de l'ensemble des concepts associés au même élément dans la WSRD « initiale » . Le *poids* d'un concept est la moyenne pondérée des degrés de similarité sémantique entre ce concept et chacun des concepts de l'élément correspondant dans la WSRD « initiale » (voir formule (3.1)).

Pour calculer la matrice *Similarity*, nous adoptons une mesure de calcul de simi-

larité largement utilisée qui a été proposée par Resnik [Resnik 1995]. Cette mesure calcule la similarité entre deux concepts en comptant le nombre d'arcs trouvés dans le chemin liant ces deux concepts dans l'ontologie. Cette matrice est construite une fois et sera par la suite réutilisée pour la création des différents nuages. Toutefois, notre approche est indépendante de la mesure de calcul de similarité utilisée pour la création de *Similarity* et d'autres mesures, telles que [Lin 1998, Jiang 1997] peuvent être utilisées.

Pour illustrer l'étape de construction des nuages, nous reprenons l'exemple de l'étape précédente (Figure 3.2). Pour cet exemple, nous disposons de la matrice de similarité *Similarity* suivante :

$$Similarity = \begin{pmatrix} 1 & 0.43 & 0.56 & 0.6 & 0.2 & 0.7 & 0.44 \\ 0.43 & 1 & 0.23 & 0.41 & 0.56 & 0.66 & 0.36 \\ 0.56 & 0.23 & 1 & 0.5 & 0.42 & 0.15 & 0.48 \\ 0.6 & 0.41 & 0.5 & 1 & 0.1 & 0.35 & 0.23 \\ 0.2 & 0.56 & 0.42 & 0.1 & 1 & 0.54 & 0.39 \\ 0.7 & 0.66 & 0.15 & 0.35 & 0.54 & 1 & 0.45 \\ 0.44 & 0.36 & 0.48 & 0.23 & 0.39 & 0.45 & 1 \end{pmatrix}$$

Similarity contient les degrés de similarité entre les sept concepts de notre ontologie. Par exemple, $Similarity[1,4] = Similarity[4,1]$ représente le degré de similarité entre les concepts C_1 et C_4 et qui est égale à 0.6.

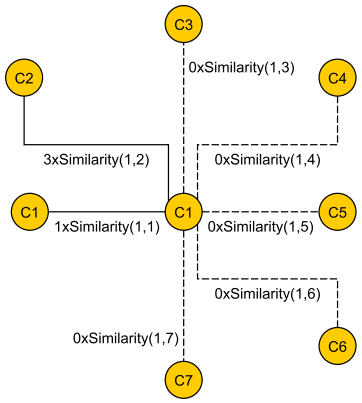
Pour construire la WSRD « intermédiaire » , on calcule les différents nuages associés aux éléments de la description WSRD. On commence par H_O, le nuage associé à l'élément `<operation>`. Les concepts associés à l'élément `<operation>` extraits à la première étape sont représentés par $O = \{< C_1, 1 >, < C_2, 3 >\}$ dans la WSRD « initiale » (voir section 3.2.1). La Figure 3.3 illustre l'étape de calcul de s_1, le poids associé au concept C_1 dans le nuage H_O en utilisant la formule (3.1). Dans O, le concept C_1 a été identifié une seule fois et C_2 trois fois. Le poids s_1 est alors égal à la moyenne des similarités entre les concepts ($C1$,$C1$) et (C_1,C_2) respectivement pondérées par 1 et 3.

Pour finir, on applique la même formule pour les autres concepts $C_i,\ i = 2 \cdots 7$ et on calcule les nuages liés aux éléments `<interface>`, `<input>` and `<output>` de la même façon. Nous obtenons la WSRD « intermédiaire » (H_I,H_O,H_{In},H_{Out}) où :

$$\begin{aligned} H_I &= \{(C_1, 0.45), (C_2, 0.61), (C_3, 0.29), (C_4, 0.23), (C_5, 0.77), (C_6, 0.77), (C_7, 0.42)\} \\ H_O &= \{(C_1, 0.57), (C_2, 0.86), (C_3, 0.31), (C_4, 0.46), (C_5, 0.47), (C_6, 0.67), (C_7, 0.38)\} \\ H_{In} &= \{(C_1, 0.44), (C_2, 0.36), (C_3, 0.48), (C_4, 0.23), (C_5, 0.39), (C_6, 0.45), (C_7, 1)\} \\ H_{Out} &= \{(C_1, 0.54), (C_2, 0.45), (C_3, 0.51), (C_4, 0.66), (C_5, 0.38), (C_6, 0.46), (C_7, 0.34)\} \end{aligned}$$

3.2.3 Étape 3 : Réduction des nuages de concepts

Chaque élément de la description WSRD « finale » d'un annuaire (c.-à-d. `<interface>`, `<operation>`, `<input>` ou `<output>`) est annoté en utilisant un ou plusieurs concepts. Nous appelons ces concepts les C_{mean}. Un C_{mean} annotant un élément



$$
\begin{aligned}
f_{H_O}(C_1) = s_1 &= \frac{\sum_{j=1}^{7}(f_O(C_j)\times Similarity[C_1,C_j])}{\sum_{j=1}^{7} f_O(C_j)} \\
&= \frac{f_O(C_1)\times Similarity[C_1,C_1]+f_O(C_2)\times Similarity[C_1,C_2]}{f_O(C_1)+f_O(C_2)} \\
&= \frac{1\times 1+3\times 0.43}{1+3} \\
&= 0.57
\end{aligned}
$$

FIGURE 3.3 – Construction du nuage H_O : calcul du poids associé au concept C_1

de la WSRD « finale » d'un annuaire résulte de l'agrégation par la moyenne des concepts annotant ce même élément dans la WSRD « initiale » . Ainsi, le ou les C_{mean} annotant(s) un élément d'une WSRD « finale » représente(nt) un concept « moyen » de ces concepts (c.-à-d. les concepts extraits à la première étape).

Dans cette troisième et dernière étape de la création d'une WSRD, nous visons à sélectionner parmi chacun des quatre nuages de concepts des éléments WSRD, le (ou les) C_{mean} qui soi(en)t le(s) plus représentatif(s) des concepts de l'élément correspondant (nous parlons ici des concepts contenus dans la WSRD « initiale »). Dans le cas où plus d'un C_{mean} est sélectionné, nous associons à chaque C_{mean} un poids indiquant sa valeur représentative par rapport aux autres.

Pour réduire un nuage de concepts, nous proposons deux méthodes : une méthode est dite forte et une méthode est dite faible. Dans la suite de cette section, nous présentons ces deux méthodes de réduction et nous les illustrons à travers un exemple.

3.2.3.1 Réduction forte de nuages

La méthode de réduction forte consiste à sélectionner un seul C_{mean} par nuage pour annoter l'élément WSRD qui lui est associé. Pour réaliser une réduction forte sur le nuage $H_e = \{< C_i, s_i >\}$ d'un élément WSRD e, nous sélectionnons tout simplement le concept C_i ayant la plus grande valeur s_i. Ce concept prendra le rôle de concept « moyen » (C_{mean}) de l'élément WSRD e. Ainsi, C_i est sélectionné si et seulement si : $\exists s_i \in [0,1], (C_i, s_i) \in H_e$ et $\nexists (C_j, s_j) \in H_e / s_j > s_i$. La WSRD ainsi obtenue est appelée WSRD « finale » *par réduction forte* et est définie dans Définition 3.3.

Pour illustrer cette étape, nous reprenons l'exemple de la section 3.2.2. A l'étape précédente, nous avons obtenu la WSRD « intermédiaire » (H_I, H_O, H_{In},H_{Out}). En utilisant une **réduction forte**, nous choisissons à partir de chaque nuage (H_I,H_O, H_{In} et H_{Out}) le concept auquel est associé la plus grande valeur s_i. Ce concept re-

DÉFINITION 3.3 (WSRD « FINALE » *par réduction forte*)
Une WSRD « finale » *par réduction forte* est un quadruplet de concepts $(C^{I}_{mean}, C^{O}_{mean}, C^{In}_{mean}, C^{Out}_{mean})$. C^{I}_{mean} (resp. C^{O}_{mean}, C^{In}_{mean} *et* C^{Out}_{mean}) est le C_{mean} annotant l'élément WSRD `<interface>` (resp. `<operation>`, `<input>` et `<output>`). Ce C_{mean} est sélectionné à partir du nuage H_I (resp. H_O, H_{In} *et* H_{Out}) si et seulement si : $\exists s_{mean} \in [0,1], (C_{mean}, s_{mean}) \in H_I$ (resp. H_O, H_{In} *et* H_{Out}) et $\nexists (C_j, s_j) \in H_I$ *(resp.* H_O, H_{In} *et* H_{Out}*)*$/s_j > s_{mean}$

présente le C_{mean} de l'élément correspondant au nuage. Ainsi, le C_{mean} de l'élément `interface` sera C_5 associé à la valeur 0.77, C_2 associé à 0.86 pour H_O, C_7 associé à 1 pour H_{In} et C_4 associé à 0.66 pour H_{Out}. Une représentation graphique de la description WSRD « finale » est fournie dans la Figure 3.4.

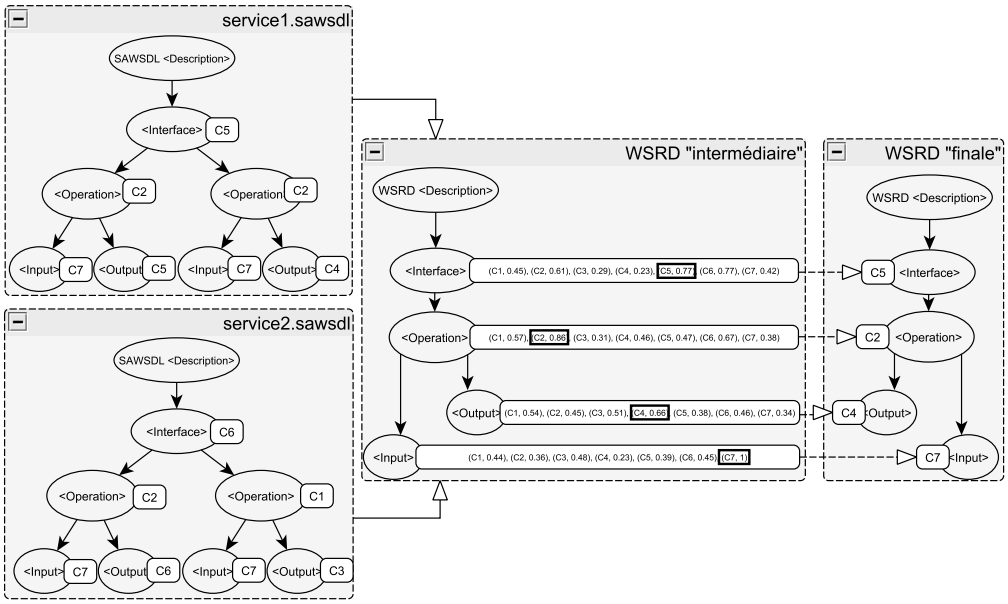


FIGURE 3.4 – Une description WSRD « finale » résultant d'une réduction forte

3.2.3.2 Réduction faible de nuages

Avec une réduction faible, et contrairement à une réduction forte, un élément WSRD est annoté en utilisant plusieurs C_{mean}. En employant une réduction forte, la description WSRD résultante reflétera la fonctionnalité *majeure* offerte par un annuaire et des fonctionnalités *mineures* risquent de ne pas être représentées. Avec une réduction faible, pour qu'une description WSRD puisse bien refléter les fonctionnalités offertes par un annuaire, nous devons répondre à ces deux questions : **combien** de concepts devront être sélectionnés et **quels** seront ces concepts ? Par

exemple, choisir les n premiers concepts (ayant les plus grandes valeurs s_i) peut bien représenter les fonctionnalités d'un annuaire lors de la réduction d'un nuage de C_{mean} mais peut être non-représentatif pour un autre [2].

Dans un nuage de concepts nous pouvons observer des ensembles de concepts très denses comme les `Ensembles a` et `b` dans la Figure 3.5 par exemple. En choisissant plusieurs concepts appartenant à un même ensemble dense comme C_{mean} nous risquons d'obtenir des C_{mean} redondants (c.-à-d. reflétant une même fonctionnalité ou des fonctionnalités proches). En même temps, si un ensemble contient un grand nombre de concepts (l'`Ensemble b` de la Figure 3.5 par exemple), en choisissant un seul concept comme C_{mean} à partir de cet ensemble, des fonctionnalités risquent de ne pas être représentées. Aussi, nous pouvons observer des concepts associés à des faibles poids dans un nuage de concepts (l'`Ensemble c` par exemple). Ces concepts reflètent certes une fonctionnalité minime dans l'annuaire mais doivent être considérés lors de la description des fonctionnalités d'un annuaire.

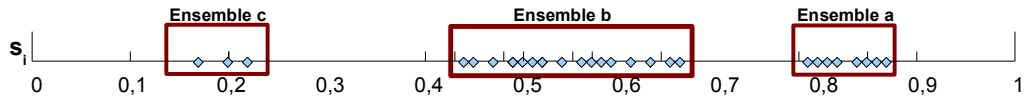


FIGURE 3.5 – Exemple d'un nuage de concepts

Nous proposons une méthode de réduction faible permettant de sélectionner d'une façon efficace plusieurs C_{mean} pour annoter une description WSRD. Pour éliminer les concepts redondants tout en minimisant la perte de connaissance, nous définissons une distance σ devant séparer deux C_{mean} sélectionnés. Nous utilisons l'écart type entre les différents concepts d'un nuage de concepts comme valeur pour σ vu qu'il représente un indicateur de la dispersion des concepts dans le nuage. Cette condition, c.-à-d. l'écart entre les valeurs de deux C_{mean} sélectionnés doit être plus grand que σ, nous évite de choisir deux concepts redondants et qui n'ont pas de valeur ajoutée à une description WSRD comme C_{mean}. Aussi, notre méthode de réduction faible tient compte de tous les concepts, même les moins représentatifs des fonctionnalités d'un annuaire, lors de la sélection des C_{mean}.

Notre méthode de réduction faible est décrite par l'Algorithme 1. Pour assurer la réduction faible d'un nuage H_e relatif à un élément WSRD e selon notre algorithme, nous commençons par trier H_e selon les poids associés à ses différents concepts (ligne 1). Le concept C_{start} associé à la plus grande valeur s_{start} est ajouté à l'ensemble de C_{mean} sélectionné (ligne 2-4). Nous calculons aussi la valeur de l'écart type σ entre les différentes valeurs s du nuage (ligne 5). Le prochain C_{mean} que nous sélectionnons, appelé dans l'algorithme $C_{current}$, devra être associé à une

2. Ceci peut être le cas d'un annuaire dont le plus grand nombre de services offrent une certaine fonctionnalité et qui seront représentés alors que la minorité des services ne le seront pas.

valeur $s_{current}$ qui soit inférieure ou égale à s (la valeur associée au C_{mean} précédemment choisi (ligne 6)) moins σ (ligne 10). Cette condition permet d'éviter les C_{mean} redondants. Ensuite, nous assignons à s la valeur associée au C_{mean} sélectionné et nous répétons les étapes précédentes (ligne 7-12) jusqu'à ce que $s - \sigma$ soit ≤ 0. La WSRD ainsi obtenue est appelée WSRD « finale » *par réduction faible* et est définie dans Définition 3.4.

Dans l'Algorithme 1, nous utilisons une fonction Tri dont la complexité est de $O(n \times log(n))$ où n exprime le nombre de concepts dans le nuage H_e. Ainsi, sa complexité de l'Algorithme 1 est $O(n \times log(n))$.

DÉFINITION 3.4 (WSRD « FINALE » *par réduction faible*)
Une WSRD « finale » *par réduction faible* est un quadruplet d'ensembles $(CM_{H_I}, CM_{H_O}, CM_{H_{In}}, CM_{H_{Out}})$. CM_{H_I} (resp. CM_{H_O}, $CM_{H_{In}}$ *et* $CM_{H_{Out}}$) contient les C_{mean} annotant l'élément WSRD `<interface>` (resp. `<operation>`, `<input>` et `<output>`) auxquels sont associés des valeurs $s_{mean} = f_{H_I}(C_{mean})$ (resp. $f_{H_O}(C_{mean})$, $f_{H_{In}}(C_{mean})$ et $f_{H_{Out}}(C_{mean})$) indiquant leurs représentativités. Les ensembles CM_{H_I}, CM_{H_O}, $CM_{H_{In}}$ *et* $CM_{H_{Out}}$ sont construits en utilisant l'Algorithme 1.

Algorithme 1 *Réduction faible*(H_e)

Entrée : H_e /* *Le nuage de concepts d'un élément e*/*
Sortie : CM /* *Un ensemble contenant les* C_{mean} *sélectionnés et leurs poids*/*

1: $H_e = Tri(H_e)$ /* *Tri est une fonction qui trie un ensemble*/*
2: $C_{start} = H_e.getElement()$
3: s_{start}=$f_{H_e}(C_{start})$;
4: CM.add((C_{start}, s_{start})) ;
5: σ =ÉcartType(s_i) ; /* *ÉcartType est une fonction qui calcule l'écart type*/*
6: $s = s_{start}$;
7: **tant que** $(s - \sigma \geq 0)$ **faire**
8: $C_{current} = H_e.getElement()$
9: $s_{current} = f_{H_e}(C_{current})$
10: **si** $(s_{current} \leq s - \sigma)$ **alors**
11: CM.add($(C_{current}, s_{current})$) ;
12: $s = s_{current}$;
13: **retour** CM

Pour illustrer le déroulement d'une réduction faible d'un nuage, nous reprenons la WSRD « intermédiaire » obtenue dans la section 3.2.2. Nous commençons par appliquer notre algorithme de réduction faible (Algorithme 1) sur le nuage H_{Out} de l'élément `<operation>`. Le premier C_{mean} sélectionné est C_4 associé à la valeur $s_i = 0.66$ la plus grande dans le nuage. L'écart type entre les différentes valeurs s_i du nuage H_{Out} est $\sigma = 0.11$. Le prochain C_{mean} qui est sélectionné à partir du nuage

est celui associé à la plus grande valeur s_i tel que $s_i \leq s_{start} - \sigma = 0.66 - 0.11 = 0.55$. Ainsi, le concept sélectionné sera C_1 avec une valeur s_1 égale à 0.54. En répétant la dernière étape jusqu'à ce que $s - \sigma$ soit ≤ 0, nous obtenons : $CM_{H_{Out}} = \{< C_4, 0.66 >, < C_1, 0.54 >, < C_5, 0.38 >\}$. La Figure 3.6 illustre une réduction faible appliquée au nuage de l'élément `<output>`.

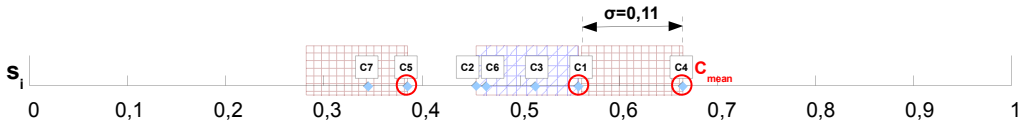


FIGURE 3.6 – Réduction faible du nuage H_{Out} de l'élément `<output>`

En réexécutant notre algorithme pour les autres nuages, nous obtenons la WSRD « finale » *par réduction faible* $(CM_{H_I}, CM_{H_O}, CM_{H_{In}}, CM_{H_{Out}})$ telle que :

$$
\begin{aligned}
CM_{H_I} &= \{(C_5, 0.77), (C_1, 0.45), (C_4, 0.23)\} \\
CM_{H_O} &= \{(C_2, 0.86), (C_6, 0.67), (C_5, 0.47)\} \\
CM_{H_{In}} &= \{(C_7, 1), (C_3, 0.48), (C_4, 0.23)\} \\
CM_{H_{Out}} &= \{(C_4, 0.66), (C_1, 0.54), (C_5, 0.38)\}
\end{aligned}
$$

Une représentation graphique de la description WSRD « finale » *par réduction faible* est fournie dans la Figure 3.7.

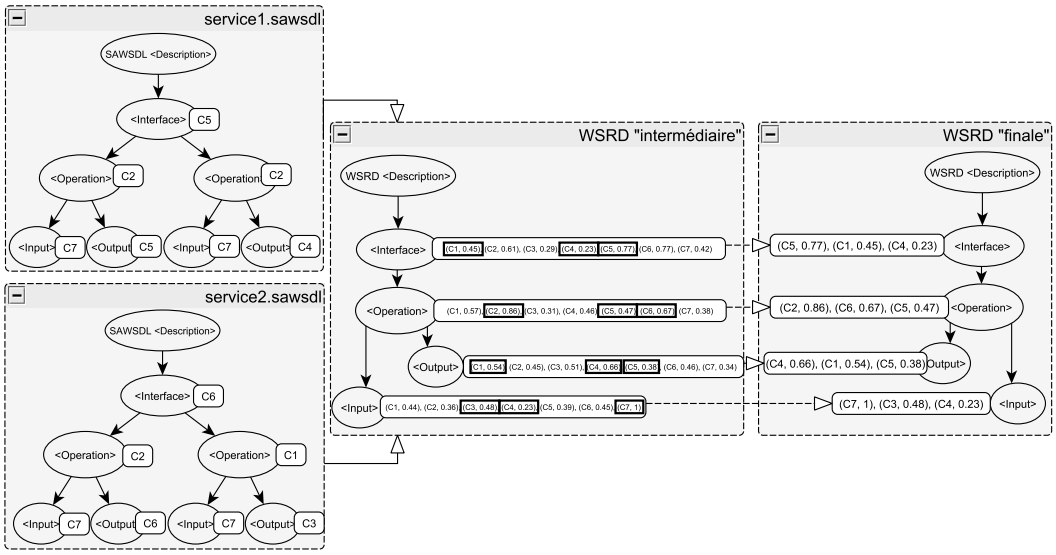


FIGURE 3.7 – Une description WSRD « finale » résultant d'une réduction faible

Une description WSRD, issue d'une **réduction forte**, reflète seulement la **fonctionnalité principale** d'un annuaire et ignore les autres fonctionnalités. Une description WSRD résultant d'une **réduction faible** reflète cependant la **fonctionnalité moyenne** d'un annuaire et permet d'avoir un aperçu global sur l'ensemble des fonctionnalités offertes. Par exemple, dans la description WSRD résultant d'une description forte de la Figure 3.4, l'élément `<output>` est seulement annoté par le concept C_4 alors que les descriptions des services offerts par l'annuaire sont annotés par C_3, $C4$, C_5 *et* C_6. En utilisant une réduction faible, la description WSRD résultante (voir Figure 3.7) représente mieux l'élément `<output>` en utilisant les concepts C_1, $C4$ *et* C_5 (C_3 et C_6 sont représentés par C_1 qui est assez proche de ces 2 concepts, voir matrice *Similarity*). Nous recommandons l'utilisation d'une réduction faible pour la création d'une WSRD vu que la charge de calcul est presque la même (voir Chapitre 6 pour plus de détails).

3.2.4 Synthèse

Dans cette section, nous avons présenté notre modèle sémantique de description d'annuaires (WSRD) et une approche pour sa construction. Cette approche a la spécificité d'être **automatique** et **implicite**. En effet, la description WSRD d'un annuaire est créée automatiquement en utilisant comme seule entrée ses descriptions de services. En associant une description WSRD à un annuaire, il ne sera plus considéré par un demandeur de services comme une « boîte noire » vu qu'il pourra avoir une idée sur les fonctionnalités de l'annuaire.

Dans ce travail, les descriptions WSRD nous permettent de décrire les fonctionnalités d'un annuaire et ainsi organiser un réseau d'annuaires selon ses fonctionnalités en groupes appelés communautés. Dans la section suivante, nous donnons notre définition d'une communauté d'annuaires et nous spécifions un modèle pour décrire une communauté, un annuaire et un réseau de communautés. Nous montrons dans la section 3.4 comment construire de telles communautés d'annuaires.

3.3 Communautés d'annuaires

Dans notre travail, nous nous basons sur les fonctionnalités d'un annuaire pour organiser un réseau d'annuaires en communautés. Dans cette section, nous commençons par définir une communauté d'annuaires dans la section 3.3.1. Par la suite, nous donnons dans la section 3.3.2 un aperçu sur quelques notions de la théorie des graphes que nous allons utiliser pour représenter une communauté. Enfin , nous présentons trois modèles pour définir : un annuaire (section 3.3.3), une communauté d'annuaires (section 3.3.4) et un réseau de communautés (section 3.3.5).

3.3.1 Définitions

Le dictionnaire Larousse définit une communauté comme « *un ensemble de personnes unies par des liens d'intérêts, des habitudes communes, des opinions ou des*

caractères communs » . Dans la communauté académique des services Web, Benatallah et al. définissent une communauté de services Web comme « *une collection de services Web offrant la même fonctionnalité avec des propriétés non-fonctionnelles différentes* » [Benatallah 2003]. Maamar et al. considèrent une communauté de services Web comme « *un moyen de fournir une description commune d'une fonctionnalité sans apporter une référence explicite et concrète à un service Web qui pourra implémenter cette fonctionnalité au moment de l'exécution* » [Maamar 2007].

Dans le même esprit, nous définissons une communauté d'annuaires comme un groupe d'annuaires offrant des services Web ayant des **fonctionnalités sémantiquement similaires** [Sellami 2011b]. Ainsi, un réseau d'annuaires pourra être organisé en communautés et chaque annuaire appartiendra à une ou plusieurs de ces communautés avec un certain degré. Dans chaque communauté, nous attribuons à un annuaire le rôle de *leader* et aux autres le rôle de *disciple* (voir Figure 3.8). L'annuaire *leader* est l'annuaire le plus représentatif des fonctionnalités d'une communauté. Il joue un rôle important dans la gestion d'une communauté et de ses *disciples*. Les annuaires *disciples* assisteront le *leader* pour répondre à une requête de découverte de services. Vu qu'un annuaire peut appartenir à plusieurs communautés, un annuaire *leader* pour une communauté c_1 pourra être un *disciple* pour une autre communauté c_2. La relation pondérée *leader-disciple* dans une communauté indique le degré de similarité entre les fonctionnalités offertes par les deux annuaires.

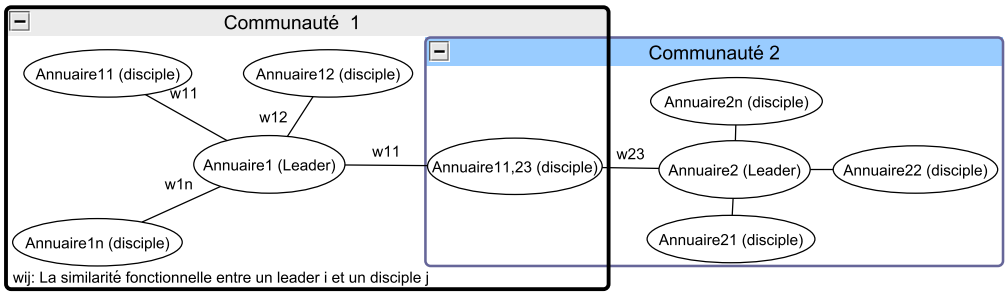


FIGURE 3.8 – Architecture de communautés d'annuaires

3.3.2 Théorie des graphes : notions de base

Les graphes permettent de modéliser une grande variété de problèmes en se ramenant à une étude de sommets et d'arêtes. Un problème donné est par la suite formalisé mathématiquement par un graphe G défini par un couple formé de deux ensembles : un ensemble $V = \{v_1, v_2, ..., v_n\}$ dont les éléments sont appelés sommets ou nœuds, et un ensemble $E = \{e_1, e_2, ..., e_m\}$ dont les éléments sont appelés des arêtes et représentent les connexions entre sommets. G est alors noté $G = (V, E)$.

Le nombre de sommets du graphe G est appelé son ordre. Il est donc le cardinal de V et noté $|V| = n$. Sa taille représente le nombre des arêtes et noté $|E| = m$.

Un graphe est dit orienté si chacune de ses arêtes est une paire ordonnée de sommets. Dans ce cas, les arêtes sont appelées arcs. Dans le cas contraire, nous parlons de graphe non-orienté. Les deux types de graphes peuvent être pondérés (valués) par une fonction $w : E \longrightarrow \mathbb{R}$ attribuant un poids à chaque arête. Un graphe pondéré est alors noté $G = (V, E, w)$. Aussi, un graphe est dit complet, si tous ses sommets sont reliés deux à deux par une arête.

3.3.2.1 Graphe et matrice associée

Un graphe orienté peut être représenté par une matrice appelée « matrice d'adjacence » . Une matrice d'adjacence est une matrice binaire carrée qui fait correspondre les sommets origine des arcs (placés en ligne dans la matrice) aux sommets destination (placés en colonne). L'existence d'un arc (v_i, v_j) se traduit par la présence d'un 1 à l'intersection de la ligne i et de la colonne j, l'absence d'arc par la présence d'un 0 [Lopez 2003].

Ainsi, tout graphe $G = (V, E)$ d'ordre n peut être représenté par sa matrice d'adjacence $A_G = (a_{i,j})$ de dimension $n \times n$ où :

$$a_{i,j} = \begin{cases} 1 & \text{si} \quad (i,j) \in E \\ 0 & \text{sinon} \end{cases}$$

Un exemple de la matrice d'adjacence d'un graphe orienté est donné dans la Table 3.1.

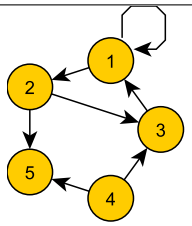
Graphe orienté	matrice d'adjacence
	$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$

TABLE 3.1 – Exemple de la matrice d'adjacence d'un graphe orienté

Si le graphe est non orienté, la matrice d'adjacence sera symétrique : l'existence d'une arête (v_i, v_j) se traduit par la présence d'un 1 à l'intersection de la ligne i et de la colonne j ainsi qu'à l'intersection de la ligne j et de la colonne i (voir exemple dans la Table 3.2).

De plus, si le graphe G=(V,E,w) est pondéré, la matrice d'adjacence peut être utilisée pour stocker les poids des arêtes. Par conséquent, les valeurs de la matrice d'adjacence d'un tel graphe sont définies comme suit [McConnell 2008](voir exemple dans la Table 3.3) :

$$a_{i,j} = \begin{cases} w_{i,j} & \text{si} \quad (i,j) \in E \\ 0 & \text{si} \quad i = j \\ \infty & \text{si} \quad (i,j) \notin E \end{cases}$$

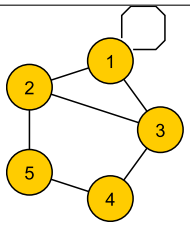
Graphe non orienté	matrice d'adjacence
	$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$

TABLE 3.2 – Exemple de la matrice d'adjacence d'un graphe non orienté

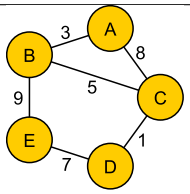
Graphe pondéré	matrice d'adjacence
	$\begin{pmatrix} 0 & 3 & 8 & \infty & \infty \\ 3 & 0 & 5 & \infty & 9 \\ 8 & 5 & 0 & 1 & \infty \\ \infty & \infty & 1 & 0 & 7 \\ \infty & 9 & \infty & 7 & 0 \end{pmatrix}$

TABLE 3.3 – Exemple de la matrice d'adjacence d'un graphe pondéré

3.3.2.2 Graphe particulier : graphe en étoile

Les graphes peuvent être classés selon l'organisation des sommets ainsi que les relations qui les lient. Nous nous intéressons dans notre travail à un type particulier appelé graphe en étoile (*star graph*) qui est un graphe biparti particulier.

Un graphe est dit biparti [Bondy 2007] si l'ensemble de ses sommets peut être partitionné en deux sous ensembles X et Y de sorte que deux sommets du même sous ensemble ne soient jamais adjacents. Un tel graphe peut être représenté par le tuple $G = (X, Y, E)$ où $X \cup Y$ est l'ensemble des sommets et E représente l'ensemble des arêtes et est noté $K_{n,m}$ où $n = |X|$ et $m = |Y|$.

Un graphe biparti est dit complet (Figure 3.9) si chaque sommet de X est lié à tous les sommets de Y. Dans le cas où $|X| = n = 1$ ou $|Y| = m = 1$, G est dit un graphe en étoile (Figure 3.10).

3.3.2.3 Opérations sur les graphes

Dans le Chapitre 4, nous spécifierons diverses opérations de gestion pour les communautés d'annuaires. Vu que nous représentons les communautés en utilisant des notions issues de la théorie des graphes, les opérations de gestion qui leur sont associées peuvent, dans certains cas, être assimilées à des manipulations sur les graphes. Dans ce qui suit, nous introduisons les notations de quelques opérations applicables sur des graphes.

Vu que les graphes sont définis par un ensemble de sommets et d'arêtes, nous utilisons la terminologie de la théorie des ensembles pour définir quelques opérations sur les graphes. Ces différentes opérations sont définies comme suit :

– **Ajout d'un sommet**

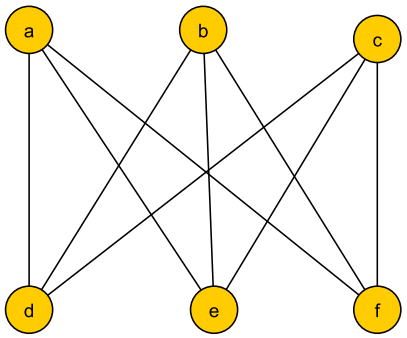


FIGURE 3.9 – Exemple d'un graphe biparti complet

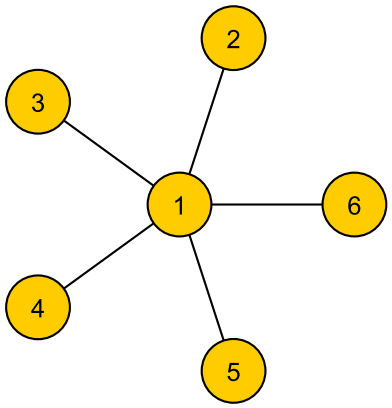


FIGURE 3.10 – Exemple d'un graphe en étoile

L'opération d'ajout d'un sommet v à un graphe $G = (V, E)$ tel que $v \notin V$ produit un nouveau graphe ayant $V \cup \{v\}$ comme ensemble de sommets et E comme ensemble d'arêtes. Ce nouveau graphe est noté $G \cup \{v\}$ [3]

– **Suppression d'un sommet**
Soit $v \in V$ un sommet de G. Lorsque nous supprimons v, toutes les arêtes ayant ce sommet comme extrémité sont également supprimées. Le graphe résultant est $(V \setminus \{v\}, E \setminus \{e \in E : e = (v, v_j) \vee e = (v_j, v), \forall v_j \in V\})$ que nous notons $G - \{v\}$.
– **Ajout d'une arête**
L'ajout d'une arête e à un graphe $G = (V, E)$ produit un graphe avec le même ensemble de sommets V et $E \cup \{e\}$ comme ensemble d'arêtes. Ce nouveau graphe est noté $G \cup \{e\}$.

3. Formellement défini par $G \cup \{v\} = (V \cup \{v\}, E)$

- **Suppression d'une arête**
  La suppression d'une arête e d'un graphe $G = (V, E)$ n'entraîne que la suppression de cette arête en gardant les extrémités (sommets) de e . Le graphe résultant est $(V, E \setminus \{e\})$ et noté : $G - \{e\}$.
- **Union de deux graphes**
  De l'union de deux graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$ résulte un nouveau graphe $G_3 = (V_3, E_3)$ tel que $V_3 = V_1 \cup V_2$ et $E_3 = E_1 \cup E_2$.
- **Intersection de deux graphes**
  L'intersection de deux graphes $G_1 = (V_1, E_1)$ et$G_2 = (V_2, E_2)$, notée $G_1 \cap G_2$, est le graphe $(V_1 \cap V_2, E_1 \cap E_2)$.
- **Complément d'un graphe** Le complément d'un graphe $G = (V, E)$ est un graphe $\overline{G}$ ayant les mêmes sommets que G. Dans $\overline{G}$ deux sommets sont adjacents, si et seulement si, ils ne le sont pas dans G. Ainsi, $\overline{G} = (V, V \times V \setminus E)$ et $G \cup \overline{G}$ est un graphe complet.

3.3.3 Modélisation d'un annuaire de services

Nous utilisons les descriptions WSRD (section 3.2) pour décrire les fonctionnalités des annuaires. Nous représentons la description WSRD d'un annuaire par un vecteur f. Les détails sur cette représentation sont fournis dans la section 3.4.2. Puisqu'une description WSRD d'un annuaire reflète plusieurs fonctionnalités, alors un annuaire peut appartenir à plusieurs communautés en même temps. Ainsi, nous associons à chaque annuaire un ensemble MEM contenant ses degrés d'appartenance à chacune des communautés existantes. Ainsi, un annuaire est défini comme suit :

Définition 3.5 (Annuaire)
Un annuaire est un tuple $r = (id, f, MEM)$ où :
- id est l'identifiant de l'annuaire.
- f est un représentation vectorielle de la WSRD d'un annuaire décrivant les fonctionnalités offertes par l'annuaire r.
- MEM représente les degrés d'appartenance de l'annuaire r aux différentes communautés. MEM est représentée par une relation binaire définie par $MEM = \{(c, d) | c \in C, d \in [0, 1]\}$ où :
  - C est l'ensemble des communautés.
  - d est le degré d'appartenance de l'annuaire r à une communauté c.

Nous représentons aussi le domaine et le codomaine des degrés d'appartenance $MEM \subseteq C \times [0, 1]$ comme :
- $dom(MEM) = \{c | (c, d) \in MEM \text{ et } d \in [0, 1]\}$
- $ran(MEM) = \{d | (c, d) \in MEM \text{ et } c \in C\}$

3.3.4 Modélisation d'une communauté d'annuaires

Dans notre approche, une communauté d'annuaires est principalement caractérisée par sa fonctionnalité moyenne f représentant la moyenne des fonctionnalités offertes par ses annuaires. Dans une communauté, nous distinguons deux types d'annuaires : *leader* et *disciple*. Par conséquent, l'ensemble des membres de la communauté (les nœuds) peut être divisé en un singleton $L = \{l\}$ représentant le *leader* et un ensemble $Fl = \{fl_i | i : 1..n\}$ où n est le nombre des *disciples* dans la communauté (voir Définition 3.6). Ainsi, les nœuds d'une communauté sont modélisés par un graphe en étoile G où les sommets représentent les annuaires et chaque arête représente la similarité fonctionnelle entre le *leader* et un *disciple*. La similarité entre les fonctionnalités offertes par un *leader* et un *disciple* est calculée en utilisant la mesure cosinus (des détails seront fournis dans la section 3.4.3).

Définition 3.6 (Communauté d'annuaires)
Une communauté d'annuaires est un tuple $c = (id, f, G)$ avec :
- id est l'identifiant de la communauté.
- f est un vecteur représentant la fonctionnalité moyenne de la communauté c.
- $G = (L, Fl, E, w)$ est un graphe en étoile non orienté et pondéré où :
  - L est le *leader* de la communauté : l'annuaire ayant le degré d'appartenance le plus grand à la communauté c.
  - Fl est l'ensemble des *disciples* de la communauté.
  - $E \subseteq L \times Fl \cup Fl \times L$ est l'ensemble des arêtes.
  - $w : E \longrightarrow [0, 1]$ est une fonction affectant un poids à chaque arête. Ce poids représente la similarité entre deux nœuds.

Pour chaque communauté c_j nous définissons une fonction U_j qui retourne le degré d'appartenance d'un annuaire r_i : $U_j(r_i) = d \Rightarrow (d, c_j) \in r_i.MEM$.

3.3.5 Modélisation d'un réseau de communautés

A ce niveau, notre environnement distribué d'annuaires composé d'un ensemble de communautés est modélisé par un ensemble de graphes en étoile. Vu que le nombre d'annuaires (nœuds) peut être très grand et qu'un annuaire peut appartenir à plusieurs communautés à la fois, la gestion des communautés est une tâche fastidieuse et coûteuse. Pour faire face à ce problème et pour avoir une vue globale du réseau, nous définissons un autre graphe CG, appelé *graphe des communautés*, dans lequel les nœuds représentent les communautés et les arêtes représentent les relations entre elles. Si deux communautés ont au moins un annuaire en commun, alors elles sont adjacentes et une arête les relie. Dans ce cas, nous calculons la distance entre les vecteurs f de ces deux communautés (Des détails sur la mesure servant à calculer cette distance sont fournis dans la section 3.4.3). Cette distance représente le poids de l'arête reliant ces deux communautés. Notre réseau de communautés est ainsi défini comme suit :

Définition 3.7 (Réseau de communautés d'annuaires)
Un réseau de communautés d'annuaires est représenté par un graphe non orienté et pondéré $CG = (C, E, w)$ avec :
- C est un ensemble fini de nœuds. Chaque nœud représente une communauté d'annuaires.
- $E \subseteq C \times C$ est un ensemble d'arêtes représentant les relations entre les communautés.
- $w : E \longrightarrow [0, 1]$ est une fonction attribuant un poids calculé en fonction de la similarité entre deux nœuds.

3.3.6 Synthèse

Dans ce travail, nous utilisons les communautés comme un moyen pour organiser un environnement distribué d'annuaires. Nous démontrerons par la suite qu'une telle organisation, c.-à-d. selon les fonctionnalités des annuaires, permet d'améliorer le processus de découverte d'un service. En effet, l'espace de recherche d'un demandeur de services pourra être réduit à la communauté d'annuaires contenant les services offrant les fonctionnalités recherchées. De plus, vu qu'un demandeur de services est essentiellement à la recherche d'une certaine fonctionnalité, une telle organisation est logique. Effectivement, la requête du demandeur de services exprime la fonctionnalité recherchée et à chaque communauté nous associons les fonctionnalités offertes. Cet homogénéité entre offre et demande permet de retrouver aisément la communauté adéquate à une requête par un simple appariement sémantique. Dans la section suivante, nous montrons comment les descriptions WSRD sont utilisées pour organiser un réseau d'annuaires en communautés.

3.4 Construction de communautés d'annuaires

Dans cette section, nous présentons notre approche de clustering floue pour l'organisation d'un réseau d'annuaires en communautés. L'approche que nous proposons est automatique et les communautés sont construites sans aucune intervention humaine avec pour seule entrée les descriptions WSRD des annuaires à organiser.

Nous commençons cette section en exposant les motivations et inspirations de notre approche de construction de communautés d'annuaires.

3.4.1 Motivations

Une communauté d'annuaires réunit dans un même « groupe » les annuaires offrant des fonctionnalités similaires. Vu qu'un annuaire offre généralement des services proposant des fonctionnalités différentes, il est difficile de prédire à l'avance les différentes classes (c.-à-d. les communautés d'annuaires) des fonctionnalités des différents annuaires. Pour organiser un réseau d'annuaires en communautés, nous

utilisons une technique de **clustering de données** [4], où les différentes communautés sont déduites des descriptions WSRD, plutôt qu'une technique de classification, où les différentes communautés doivent être définies à l'avance. En utilisant une technique de clustering dynamique, les différents clusters [5] (c.-à-d. les communautés d'annuaires) seront identifiés à partir de l'ensemble de données en entrée (c.-à-d. les descriptions WSRD qui décrivent les annuaires) et chaque élément (c.-à-d. annuaire) sera affecté à un cluster. De plus, l'un des objectifs d'une technique de clustering est de `minimiser` la similarité entre les clusters en la `maximisant` entre les différents éléments du même cluster. Ceci afin d'avoir des communautés offrant des fonctionnalités moyennes différentes, et en même temps, des fonctionnalités d'annuaires similaires au sein d'une même communauté.

Un annuaire peut offrir des fonctionnalités différentes et ainsi appartenir à plusieurs communautés. Pour cette raison, l'utilisation d'une technique de clustering exclusive est inadéquate pour la construction de communautés d'annuaires. En effet, en utilisant une technique de clustering exclusive, « *les données sont regroupées d'une façon exclusive, de sorte que si une donnée appartient à un cluster elle ne pourra pas appartenir à un autre* [6] ». Pour cette raison, nous proposons d'utiliser une technique de clustering floue [Zadeh 1965] pour organiser un réseau d'annuaires en communautés. Une telle technique permet de regrouper des données, dans notre cas des annuaires, d'une façon inclusive en permettant à chaque annuaire d'appartenir à un ou plusieurs clusters.

Pour mettre en place cette organisation inclusive, nous nous sommes inspirés de la méthode de clustering floue fuzzy C-means [Dunn 1973, Bezdek 1981]. Fuzzy C-means est une méthode de clustering non-supervisée souvent utilisée dans les domaines d'analyse de données et de reconnaissance de formes. Dans le domaine de la recherche d'information, plusieurs travaux [Jursic 2008, Saraçoglu 2007] utilisent aussi cette méthode pour organiser des documents en considérant qu'un document est un ensemble de termes. Chaque document est alors représenté par un vecteur dont les éléments sont les termes de ce document. En appliquant l'algorithme fuzzy C-means, chaque document est affecté à un ou plusieurs cluster(s). De même, dans notre contexte, nous assimilons les annuaires à organiser, représentés par leurs WSRD, à des documents et les concepts annotant ces WSRD aux termes de ces documents. Nous adaptons ainsi l'algorithme fuzzy C-means pour la création de notre réseau de communautés.

Notre approche de construction de communautés se déroule sur trois étapes :

1. Dans la première étape (section 3.4.2), nous représentons sous forme vectorielle les descriptions WSRD associées aux différents annuaires de notre réseau. Ces vecteurs forment les entrées nécessaires à notre méthode de clustering.
2. Dans la deuxième étape (section 3.4.3), nous définissons la mesure utilisée pour calculer la distance entre deux annuaires représentés par leurs vecteurs.

4. *Data clustering* en anglais.
5. Grappes en français.
6. `http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/`

Cette mesure est requise par l'algorithme fuzzy C-means que nous utilisons.

3. Dans la troisième étape (section 3.4.4), nous utilisons les résultats des étapes précédentes pour réaliser un clustering floue pour créer des communautés d'annuaires.

3.4.2 Définition d'un modèle vectoriel pour les descriptions WSRD

Afin d'appliquer la méthode fuzzy C-means à notre contexte, nous considérons chaque *annuaire comme un document* et l'ensemble des *concepts sémantiques qui l'annotent comme les termes de ce document*. Ces annuaires sont organisés en clusters représentant les communautés. Les données à organiser sont les descriptions WSRD qui représentent les annuaires r_i. Nous utilisons un modèle vectoriel pour représenter ces descriptions. Chaque description WSRD d'un annuaire r_i est représentée par un vecteur $r_i.f = [w_1, w_2, \ldots, w_t]$ où t est le nombre de concepts formant l'ontologie utilisée et w_t est le poids du concept dans le vecteur. Afin de représenter une description WSRD sous forme d'un vecteur, nous définissons l'Algorithme 2 permettant de calculer les différents poids w_i comme suit :

- Pour l'élément `<interface>` (resp. `<operation>`, `<input>` et `<output>`) d'une description WSRD, nous reprenons l'ensemble CM_{H_I} associé (resp. CM_{H_O}, $CM_{H_{In}}$ et $CM_{H_{Out}}$) (voir Définition 3.4). Nous extrayons les poids s_{mean} associés à chacun des concepts C_{mean} dans cet ensemble. Ces valeurs sont stockées dans un vecteur $v_I = [e_1, e_2, \ldots, e_t]$ (resp. v_O, v_{In} et v_{Out}). Ces vecteurs ont la même taille t que le vecteur $r_i.f$ représentant la description WSRD. Si un concept de l'ontologie utilisée ne se trouve pas dans l'ensemble CM_{H_I} (resp. CM_{H_O}, $CM_{H_{In}}$ et $CM_{H_{Out}}$), alors sa valeur dans v_I (resp. v_O, v_{In} et v_{Out}) est égale à 0 (voir Algorithme 2, ligne (1-6)).
- Les vecteurs calculés dans la première étape représentent les différents éléments de la description WSRD. Puisque chaque élément a une signification spécifique pour représenter les fonctionnalités d'un annuaire, un poids est attribué à chaque vecteur. Ainsi, nous définissons quatre poids : α pour v_I, β pour v_O, δ pour v_{In} et λ pour v_{Out} tel que $\alpha + \beta + \delta + \lambda = 1$.Ainsi, le vecteur $r_i.f$ de la description WSRD est calculé comme suit (ligne (7)) :

$$r_i.f = \alpha \times v_I + \beta \times v_O + \delta \times v_{In} + \lambda \times v_{Out}$$

La complexité de l'Algorithme 2 est $O(t)$ où t exprime le nombre de concepts formant l'ontologie utilisée. Ainsi la complexité de cet algorithme est linéaire.

Pour illustrer la représentation d'une description WSRD selon notre modèle vectoriel, nous reprenons l'exemple de la description WSRD de la Figure 3.7. Nous supposons aussi que l'ontologie est composée seulement par 7 concepts. Pour calculer les différents poids w_i du vecteur $r_i.f = [w_1, w_2, \ldots, w_7]$ représentant la description WSRD, nous exécutons les étapes détaillées ci-dessus.

1. Tout d'abord, les vecteurs décrivant les différents éléments de la description WSRD sont créés :

Algorithme 2 $WSRD2Vector(CM_{H_I}, CM_{H_O}, CM_{H_{In}}, CM_{H_{Out}})$

Entrée : $(CM_{H_I}, CM_{H_O}, CM_{H_{In}}, CM_{H_{Out}})$ /* *Une description WSRD « finale »*/

Sortie : f /* *Un vecteur de dimension t représentant la WSRD*/

1: **pour** k in $\{I,\ O,\ In,\ Out\}$ **faire**
2: **pour** $i = 1 \rightarrow t$ **faire**
3: **si** $f_{H_k}(C_i) == null$ **alors**
4: $v_k[i] = 0$
5: **sinon**
6: $v_k[i] = f_{H_k}(C_i)$
7: $f = \alpha \times v_I + \beta \times v_O + \delta \times v_{In} + \lambda \times v_{Out}$
8: **retour** f

$$
\begin{array}{lllllllll}
v_I & = [0.45 & 0 & 0 & 0.23 & 0.77 & 0 & 0] \\
v_O & = [0 & 0.86 & 0 & 0 & 0.47 & 0.67 & 0] \\
v_{In} & = [0 & 0 & 0.48 & 0.23 & 0 & 0 & 1] \\
v_{Out} & = [0.54 & 0 & 0 & 0.66 & 0.38 & 0 & 0]
\end{array}
$$

2. Pour spécifier les poids des différents éléments de la description WSRD, nous supposons que les poids des éléments `<interface>` et `<operation>` ont la même importance. Les éléments `<input>` et `<output>` sont moins importants que ces derniers : $\alpha = \beta = 0.4 > \delta = \lambda = 0.1$.
3. Finalement, le vecteur $r_i.f$, représentant la description WSRD, est la somme pondérée des vecteurs des différents éléments WSRD.

$$r_i.f = 0.4 \times v_{interface} + 0.4 \times v_{operation} + 0.1 \times v_{input} + 0.1 \times v_{output}$$

Les différents poids w_i du vecteur r_i sont calculés comme suit :

$$w_1 = 0.4 * 0.45 + 0.4 * 0 + 0.1 * 0 + 0.1 * 0.54 = 0.234$$

La même formule est appliquée aux différents éléments et nous obtenons le vecteur r_i suivant :

$$r_i = [0.234 \quad 0.344 \quad 0.048 \quad 0.181 \quad 0.534 \quad 0.268 \quad 0.1]$$

Pour cet exemple, nous avons donné des valeurs aux différents poids $\alpha,\ \beta,\ \delta\ et\ \lambda$. Toutefois, ces valeurs devront être ajustées moyennant des expérimentations répétitives selon le contexte et les spécificités des fournisseurs d'annuaires.

3.4.3 Définition de la métrique de distance utilisée

Nous utilisons une métrique de distance afin d'établir les degrés d'appartenance de chaque annuaire aux différentes communautés. Dans la première étape, nous avons associé à chaque annuaire un vecteur représentant sa description WSRD. Vu son adéquation aux données de grandes dimensions [Qian 2004], nous utilisons la mesure cosinus [Salton 1987] pour établir la similarité entre deux vecteurs. La fonction cosinus retourne des valeurs comprises entre 0 et 1 : **0** aucune similarité et **1** vecteurs identiques.

Soit deux vecteurs $r_1.f$ et $r_2.f$ non nuls[7] représentant les fonctionnalités de deux annuaires, la similarité (cosinus) est obtenue en divisant le produit scalaire de ces vecteurs par leurs normes (voir formule (3.2)) :

$$cosine(r_1.f, r_2.f) = \frac{r_1.f \times r_2.f}{\|r_1.f\| \times \|r_2.f\|} \tag{3.2}$$

La fonction cosinus permet de calculer la similarité entre deux vecteurs et non pas la distance comme requis par l'algorithme de la méthode fuzzy C-means. Nous utilisons la formule (3.3) pour déduire la distance entre deux vecteurs à partir de la fonction de similarité cosinus :

$$distance(r_1.f, r_2.f) = 1 - cosine(r_1.f, r_2.f) \tag{3.3}$$

3.4.4 Construction des communautés

Pour construire les communautés d'annuaires, nous utilisons la méthode fuzzy C-means. Nous procédons en deux phases répétitives : (i) calcul des centres des communautés, aussi appelés centroides et (ii) affectation des différents annuaires aux communautés. Le centre d'une communauté représente sa fonctionnalité moyenne f (voir Définition 3.6). Chaque annuaire est affecté à une communauté avec un degré d'appartenance.

Le clustering des annuaires est basé sur la minimisation de la fonction objectif (3.4) suivante :

$$J = \sum_{j=1}^{|C|} \sum_{i=1}^{N} [U_j(r_i)]^m distance(r_i.f, c_j.f)^2 \tag{3.4}$$

où $\mid C \mid$ est un nombre prédéfini exprimant le nombre de communautés, N le nombre d'annuaires à organiser, r_i est le $i^{ème}$ annuaire de notre ensemble d'annuaires à organiser, $U_j(r_i)$ est la fonction d'appartenance de r_i à la communauté j (voir Définition 3.6), m le coefficient de pondération qui détermine le degré de flou (un réel supérieur à 1) et *distance* est la métrique de distance que nous avons définie dans la formule (3.3).

Le clustering est basé sur un schéma de minimisation alterné de la fonction J, avec, dans un premier temps, la détermination des centres des communautés $c_j.f$ (formule (3.5)) , et, dans un second temps, l'allocation des annuaires aux différentes communautés en mettant à jour leurs vecteurs d'appartenance $U_j(r_i)$ (formule (3.6)). Les $\mid C \mid$ vecteurs représentant les centroides sont initialement générés au hasard. La mise à jour des vecteurs d'appartenance et des centroides est faite en utilisant respectivement la formule (3.5) et (3.6) :

$$c_j.f = \frac{\sum_{i=1}^{N} U_j(r_i)^m \cdot r_i}{\sum_{i=1}^{N} U_j(r_i)^m} \tag{3.5}$$

7. Les vecteurs ne sont pas nuls vu que nous supposons que tous les fournisseurs de services utilisent des descriptions de services annotés sémantiquement en SAWSDL (voir section 3.2)

$$U_j(r_i) = \frac{1}{\sum_{k=1}^{|C|} (\frac{distance(r_i.f, c_j.f)}{distance(r_i.f, c_k.f)})^{\frac{2}{m-1}}} \tag{3.6}$$

A l'issue de cette étape, nous obtenons $\mid C \mid$ communautés représentées par leurs centroides. Ces centroides représentent la fonctionnalité moyenne f de ces communautés conformément à la Définition 3.6. De plus, à chaque communauté c_j sont associés N vecteurs d'appartenance $U_j(r_i), i = 1 \ldots N$ indiquant les degrés d'appartenance, compris entre 0 et 1, des N annuaires à la communauté. Ces données nous permettent de déduire les degrés d'appartenance MEM (voir Définition 3.5) des N annuaires à organiser en communautés. Aussi, dans chaque communuauté, le rôle de *Leader* est attribué à l'annuaire possèdant le degré d'appartenance le plus important.

Toutefois, en utilisant cette technique, le degré d'appartenance de certains annuaires à certaines communautés peut être très faible. Pour cela, nous définissons un seuil `th` pour les degrés d'appartenance. Si le degré d'appartenance d'un annuaire à une communauté est inférieur à ce seuil, il ne sera pas considéré comme membre. Ce seuil est nécessaire pour préserver la « réputation » d'une communauté et pour s'assurer que les fonctionnalités de ces membres ne soient pas trop différent que ceux annoncées par la communauté (exprimées par leurs fonctionnalités moyennes). Un scénario complet de construction de communautés d'annuaires selon notre approche est fourni dans le chapitre de mise en œuvre.

La complexité de la méthode fuzzy C-means utilisée pour la construction des communautés est $O(N \times t \times \mid C \mid^2 \times i)$ [Rao 2010] où N exprime le nombre d'annuaires à organiser, t le nombre de concepts formant l'ontologie utilisée, $\mid C \mid$ le nombre de communautés et i le nombre d'itérations nécessaires pour minimiser la fonction objective J.

3.4.5 Synthèse

Dans cette section, nous avons présenté notre approche de constructions de communautés. Cette approche permet d'organiser un réseau d'annuaires en communautés en utilisant comme **seule entrée** les descriptions WSRD des annuaires et aucune intervention humaine n'est nécessaire. Cette organisation a été assurée en utilisant une technique de clustering floue afin de répondre aux besoins de notre contexte (voir section 3.4.1). Cette technique nous permet d'avoir les centroides des communautés (représentant leurs fonctionnalités moyennes) et les vecteurs d'appartenance des annuaires aux différentes communautés. Aussi, nous avons défini un seuil `th` pour les degrés d'appartenance afin de préserver la « réputation » d'une communauté. Les données obtenues permettent de mettre en place une organisation des annuaires en communautés. La complexité de notre méthode de construction de communautés est $O(N \times t + N \times t \times \mid C \mid^2 \times i)$ où N exprime le nombre d'annuaires à organiser, t le nombre de concepts formant l'ontologie utilisée, $\mid C \mid$ le nombre de communautés et i le nombre d'itérations nécessaires pour minimiser la fonction objective J.

3.5 Conclusion

Ce chapitre nous a permis d'atteindre le premier objectif de notre thèse : proposer une « **approche implicite pour l'organisation d'un ensemble d'annuaires de services en groupes.** ». L'approche d'organisation que nous avons proposée utilise les fonctionnalités d'un annuaire comme **critère** d'organisation. De plus la **méthode** utilisée est implicite et automatique et ne nécessite aucune information ou donnée de la part d'un fournisseur d'annuaires. Aussi, notre approche peut être adaptée et utilisée avec des annuaires offrant des services décrits dans différents langages de description de services Web sémantiques.

Pour ce faire, nous avons défini un modèle sémantique (descriptions WSRD) pour la description des fonctionnalités d'un annuaire. Ces descriptions WSRD sont construites d'une façon automatique et implicite en réalisant une agrégation par la moyenne des éléments des descriptions de services d'un annuaire.

Nous avons par la suite défini le concept de communautés d'annuaires. Nous avons utilisé les communautés comme moyen pour organiser automatiquement un environnement distribué d'annuaires selon leurs descriptions WSRD. En effet, ces descriptions reflètent les fonctionnalités des annuaires et permettent ainsi de regrouper des annuaires selon leurs fonctionnalités en communautés.

Pour assurer une mise en place automatique de notre réseau de communautés, nous avons utilisé une approche de clustering floue. La première étape de cette approche consiste à représenter les descriptions WSRD des annuaires sous forme vectorielle. Cette représentation nous a permis d'appliquer l'algorithme fuzzy C-means pour organiser notre réseau d'annuaires. Les travaux présentés dans ce chapitre ont été validés par quatre publications [Sellami 2010a, Sellami 2010b, Sellami 2011b, Sellami 2011a]. Nous proposons dans le chapitre suivant une approche permettant de tenir dynamiquement cette organisation à jour après les différents changements qui peuvent se produire.

Chapitre 4

Communautés d'annuaires : La Gestion

Sommaire

4.1 Introduction

Dans ce chapitre, nous présentons notre approche de gestion de communautés d'annuaires. Cette approche a pour but de garantir la consistance d'une organisation d'un réseau d'annuaires en communautés. En effet, les communautés ainsi que leurs membres (c.-à-d. les annuaires) opèrent dans un environnement très dynamique dont les changements sont principalement initiés par les fournisseurs de services et les fournisseurs d'annuaires. Un fournisseur de services peut publier/-supprimer une description d'un service. De même, un fournisseur d'annuaires peut rendre son annuaire disponible ou indisponible à n'importe quel moment. Dans cet environnement dynamique, la similarité inter et intra communautés doit être vérifiée pour assurer l'un des objectifs de notre approche de *clustering* : **minimiser** la similarité entre les communautés en la **maximisant** entre les différents membres de chaque communauté.

Pour maintenir l'organisation d'un réseau de communautés et garantir cet objectif, nous ne pouvons pas nous permettre de relancer notre approche d'organisation d'annuaires (section 3.4) à chaque fois qu'un changement survient, principalement

pour des raisons de coût. Cette approche est seulement utilisée comme un « **démarreur à froid** » pour l'organisation d'un réseau d'annuaires. Ainsi, nous devons définir des mécanismes de gestion pour assurer l'intégrité de cette organisation. Ces mécanismes de gestion garantiront la persistance de notre organisation même après les différents changements qui pourront se produire vu la dynamicité de l'environnement.

Ce chapitre est structuré comme suit. Tout d'abord, la section 4.2 introduit un exemple d'une organisation d'un réseau d'annuaires en communautés. Cet exemple est utilisé tout au long du chapitre pour illustrer notre approche de gestion de communautés. Ensuite, la section 4.3 spécifie le cycle de vie d'un annuaire au sein d'une communauté (rejoindre une communauté, mise à jour des fonctionnalités d'un annuaire, etc.) ainsi que les opérations de gestion nécessaires à chacune des étapes identifiées. Ensuite, la section 4.4 décrit les étapes du cycle de vie d'une communauté dans un réseau de communautés (création, fusion, division, etc.) et expose les opérations de gestion nécessaires à chacune de ces étapes. Enfin, la section 4.5 conclut ce chapitre.

4.2 Exemple support

Pour illustrer notre approche de gestion de communautés, nous considérons un exemple d'un réseau d'annuaires organisé en communautés selon notre approche de construction de communautés de la section 3.4. Ce réseau est formé par sept annuaires organisés en quatre communautés comme le montre la Figure 4.1.

Nous reprenons les définitions 3.5 et 3.6 pour représenter les annuaires et les communautés formant notre exemple. Les annuaires utilisés sont représentés dans la Table 4.1 et les communautés dans la Table 4.2.

	id	f	MEM
$\mathbf{r_1}$	r_1	[0.314 0.816 0.112 0.268 0.653]	$\{(c_1, 0^1), (c_2, 0.998), (c_3, 0), (c_4, 0)\}$
$\mathbf{r_2}$	r_2	[0.345 0.870 0.000 0.321 0.697]	$\{(c_1, 0), (c_2, 0.999), (c_3, 0), (c_4, 0)\}$
$\mathbf{r_3}$	r_3	[0.277 0.679 0.442 0.045 0.536]	$\{(c_1, 0.996), (c_2, 0.002), (c_3, 0), (c_4, 0)\}$
$\mathbf{r_4}$	r_4	[0.148 0.643 0.179 0.538 0.304]	$\{(c_1, 0), (c_2, 0), (c_3, 1), (c_4, 0)\}$
$\mathbf{r_5}$	r_5	[0.272 0.741 0.083 0.595 0.378]	$\{(c_1, 0), (c_2, 0), (c_3, 0.001), (c_4, 0.998)\}$
$\mathbf{r_6}$	r_6	[0.300 0.781 0.024 0.652 0.420]	$\{(c_1, 0), (c_2, 0), (c_3, 0), (c_4, 0.999)\}$
$\mathbf{r_7}$	r_7	[0.261 0.641 0.323 0.224 0.497]	$\{(c_1, 0.972), (c_2, 0.019), (c_3, 0.004), (c_4, 0.003)\}$

[1] Nous utilisons 0 pour représenter toutes valeurs$<th$.

TABLE 4.1 – Les annuaires utilisés pour notre exemple

A travers les éléments $G.L$ et $G.Fl$ de la Table 4.2, nous pouvons déduire les différents membres de chaque communauté.

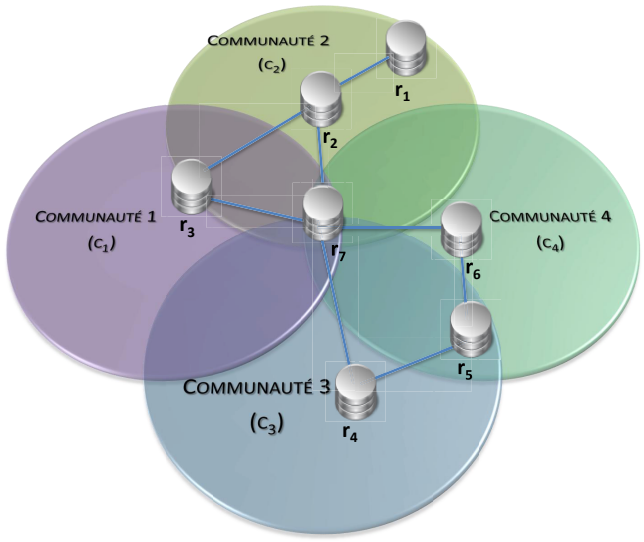


FIGURE 4.1 – Exemple d'une organisation en communautés

4.3 Gestion d'un annuaire

La Figure 4.2 présente le processus de gestion d'un annuaire et d'une communauté. Le cycle de vie d'un annuaire débute quand un fournisseur d'annuaires décide de **publier son annuaire** dans le réseau (voir opération (1) de la Figure 4.2). Cet annuaire **rejoindra le réseau** (opération (2)) et **adhérera aux communautés** (opération (3)) décrivant des fonctionnalités similaires aux fonctionnalités qu'il offre. Vu qu'un fournisseur de services peut **publier** (opération (4)) ou **supprimer** (opération (5)) des descriptions de **services** dans cet annuaire, la description WSRD de ce dernier peut changer et **une mise à jour des fonctionnalités de l'annuaire** (opération (6)) est nécessaire. Dans ce cas, il convient de vérifier la

Communauté	**id**	**f**	$\mathbf{G} = (\mathbf{L}, \mathbf{Fl}, \mathbf{E}^2, \mathbf{w}^3)$
$\mathbf{c_1}$	c_1	[0.269 0.660 0.383 0.132 0.516]	$(r_3, \{r_7\}, E, w)$
$\mathbf{c_2}$	c_2	[0.329 0.842 0.056 0.294 0.674]	$(r_2, \{r_1, r_3, r_7\}, E, w)$
$\mathbf{c_3}$	c_3	[0.148 0.643 0.179 0.537 0.304]	$(r_4, \{r_5, r_7\}, E, w)$
$\mathbf{c_4}$	c_4	[0.286 0.761 0.053 0.623 0.399]	$(r_6, \{r_5, r_7\}, E, w)$

[3] Nous ne spécifions pas les valeurs de E et w vu qu'elles ne sont pas utilisées dans nos exemples.

TABLE 4.2 – Les communautés utilisées pour notre exemple

pertinence de cet annuaire dans les communautés auxquelles il appartient : si le degré d'appartenance de l'annuaire est inférieur à un certain seuil `th`, il **quitte la communauté** (opération (7)) et il est invité à en rejoindre une autre (opération (3)). Finalement, l'annuaire **quitte le réseau tout entier** (opération (8)) si son fournisseur décide de le **démonter** (opération (9)). Dans la suite de cette section, nous détaillons les opérations (2), (3) et (6) du cycle de vie d'un annuaire et nous définissons les opérations de gestion qui leur sont associées. Nous ne détaillons pas les autres étapes vu qu'elles ne représentent pas de déclencheurs directs de changements dans l'organisation des communautés et aucune opération de gestion ne leur est associée.

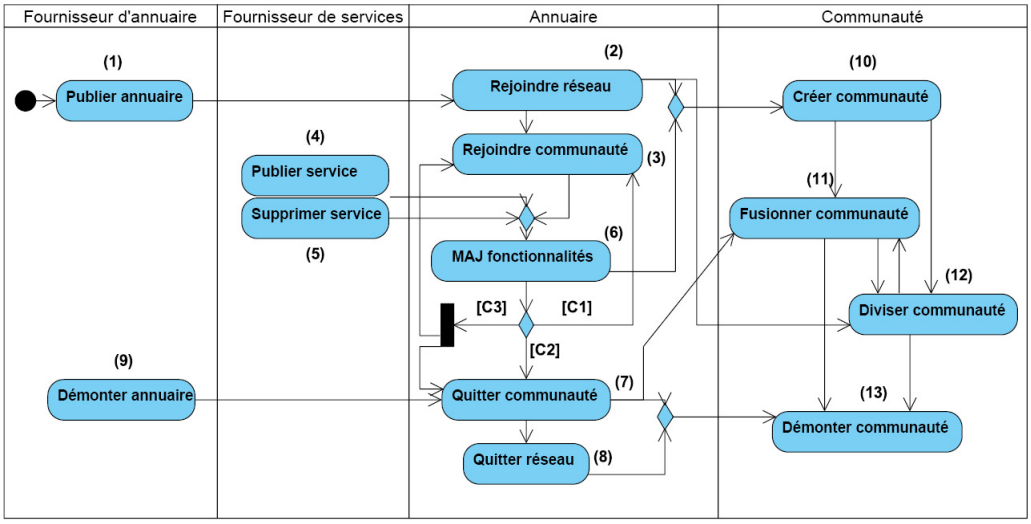


FIGURE 4.2 – Processus de gestion des communautés

4.3.1 Rejoindre le réseau

Quand un nouvel annuaire r rejoint notre réseau d'annuaires, sa description WSRD est calculée automatiquement en utilisant notre approche introduite dans la section 3.2. La sélection des communautés pour ce nouvel annuaire r se déroule comme indiquée dans l'Algorithme 3. Cet algorithme nécessite comme entrée les fonctionnalités $r.f$ de l'annuaire décrites par sa WSRD. $r.f$ est utilisée pour calculer les degrés d'appartenance MEM de r aux différentes communautés dans le réseau. Le degré d'appartenance de r à une communauté c_i est égale à l'inverse de la distance entre la fonctionnalité moyenne $c_i.f$ de la communauté et les fonctionnalités de l'annuaire $r.f$ (Algorithme 3, ligne 2). Tenant compte du fait que le degré d'appartenance doit être supérieur au seuil `th` (ligne 4), l'annuaire est guidé vers les communautés appropriées (c.-à-d. r sera ajouté à l'ensemble des sommets du graphe $c.G$ représentant la communauté (ligne 5)). Pour ces communautés, r peut prendre le rôle de *leader* ou de *disciple* selon ses degrés d'appartenance (voir sec-

tion 4.3.2). Si tous les degrés d'appartenance de r sont inférieurs à `th`, une nouvelle communauté sera créée (voir section 4.4.1)

Algorithme 3 Sélection des communautés

Entrée : $r.f$ */* le vecteur f représentant les fonctionnalités d'un annuaire r */*

1: **pour tout** Communauté $c_i \in C$ **faire**
2: $m \longleftarrow U_i(r)$[1]
3: $r.MEM \longleftarrow r.MEM \cup \{(c_i, m)\}$
4: **si** $m >$ `th` **alors**
5: V[2] $(c_i.G) \longleftarrow V(c_i.G) \cup \{r\}$

La complexité de l'Algorithme 3 est quadratique. En effet, elle est $O(\mid C \mid^2)$ où $\mid C \mid$ exprime le nombre de communautés.

Exemple

Pour illustrer cette étape, nous reprenons l'exemple introduit dans la section 4.2. Quand un nouvel annuaire r_8 rejoint ce réseau, sa description WSRD, servant à calculer ses fonctionnalités offertes, est automatiquement calculée. La sélection des communautés qu'il devra rejoindre est faite selon les étapes introduites dans l'Algorithme 3. L'algorithme calcule le degré d'appartenance de r_8 aux différentes communautés en fonction de ses fonctionnalités $r_8.f$ et l'annuaire rejoindra toutes les communautés pour lesquelles il présente un degré d'appartenance supérieur au seuil th.

Pour la communauté c_1 par exemple, le degré d'appartenance m de r_8 est calculé comme suit :

$$\begin{aligned} \text{m} &= U_1(r_8) \\ &= \frac{1}{\sum_{k=1}^{4} (\frac{distance(r_8.f, c_1.f)}{distance(r_8.f, c_k.f)})^2} \\ &= 0.162 \end{aligned}$$

Cette étape est répétée pour les trois autres communautés et on obtient ainsi les degrés d'appartenance MEM suivants :

	id	f	MEM
r_8	r_8	[0.234 0.314 0.048 0.181 0.534]	$\{(c_1, 0.162), (c_2, 0.698), (c_3, 0.054), (c_4, 0.085)\}$

Les degrés d'appartenance de r_8 sont supérieurs au seuil th pour les quatre communautés et le nouvel annuaire est ainsi membre des quatre communautés. L'algorithme met alors à jour les listes des membres des 4 communautés (l'ensemble des nœuds des graphes représentatifs de ces communautés) en y ajoutant r_8 :

- $V(c_1.G) = V(c_1.G) \cup \{r_8\}$
- $V(c_2.G) = V(c_2.G) \cup \{r_8\}$

1. $U_i(r)$ permet de calculer le degré d'appartenance d'un annuaire r à une communauté c_i (voir Equation 3.5)

2. V désigne l'ensemble des nœuds du graphe représentant une communauté. V est donc égal à $L \cup Fl$

– $V(c_3.G) = V(c_3.G) \cup \{r_8\}$
– $V(c_4.G) = V(c_4.G) \cup \{r_8\}$

L'annuaire pourra avoir le rôle de *leader* ou de *disciple* dans ces communautés selon ses degrés d'appartenance. Ce point sera traité dans la section suivante.

4.3.2 Rejoindre une communauté

Lorsqu'un nouvel annuaire est publié, ou après la mise à jour des fonctionnalités d'un annuaire existant, l'annuaire pourra rejoindre de nouvelles communautés. Toutefois, quand un annuaire rejoint une communauté, il peut avoir un degré d'appartenance qui soit plus grand que celui du *leader* de cette communauté. Pour gérer cette situation, nous utilisons l'Algorithme 4 pour resélectionner le *leader* de la communauté si nécessaire. Cet algorithme vérifie si un annuaire r (qui vient de rejoindre le réseau ou dont les fonctionnalités ont été mise à jour) prendra ou pas le rôle de *leader* dans l'une des communautés à laquelle il appartient. Pour cela, notre algorithme compare le degré d'appartenance d_r de l'annuaire à une communauté c au degré d'appartenance d_l du *leader* l de cette même communauté (ligne 5). Si le degré d'appartenance du *leader* est supérieur à celui du nouvel annuaire, ce dernier prendra le rôle d'un *disciple* dans cette communauté (ligne 6-7).

Dans le cas contraire ($d_r > d_l$), tous les liens entre le *leader* l et les *disciples* de la communauté seront supprimés (ligne 9-11), l anciennement leader de c deviendra un *disciple* (ligne 12), l'annuaire r prendra la place du *leader* (ligne 13) et les différents *disciples* sont joints au nouveau *leader* (ligne 14-16).

Algorithme 4 Resélection d'un *leader*

Entrée : r /* *Un annuaire**/

1: **pour tout** Communauté $c \in dom(r.MEM)$ **faire**
2: $\{l\} \longleftarrow c.G.L$
3: $d_r \in [0,1]$ tel que $(c, d_r) \in r.MEM$
4: $d_l \in [0,1]$ tel que $(c, d_l) \in l.MEM$
5: **si** $d_l > d_r$ **alors**
6: $c.G.Fl \longleftarrow c.G.Fl \cup \{r\}$
7: $E(c.G) \longleftarrow E(c.G) \cup \{(r, l)\}$
8: **sinon**
9: **pour tout** $fl \in c.G.Fl$ **faire**
10: $E(c.G) \longleftarrow E(c.G) - \{(l, fl)\}$
11: $c.G.Fl \longleftarrow c.G.Fl \cup \{l\}$
12: $c.G.L \longleftarrow \{r\}$
13: **pour tout** $fl \in c.G.Fl$ **faire**
14: $E(c.G) \longleftarrow E(c.G) \cup \{(r, fl)\}$

La complexité de l'Algorithme 4 est $O(\mid Fl \mid \times \mid C \mid)$ où $\mid Fl \mid$ exprime le nombre de *disciples* dans la communauté et $\mid C \mid$ le nombre de communautés.

Exemple

Pour tester l'algorithme de resélection d'un *leader*, nous reprenons notre exemple de réseau de communautés après l'ajout de l'annuaire r_8 (voir section 4.3.1). L'Algorithme 4 compare le degré d'appartenance d_{r_8} de r_8 aux quatre communautés du réseau avec les degrés d'appartenance d_l des *leader* de ces communautés.

Pour la communauté c_1 par exemple, notre algorithme commence par extraire les données suivantes :

- $\{l_{c_1}\} = c_1.G.L = r_3$ le leader de la communauté c_1 (ligne 2).
- $d_{r_8} = 0.162$ le degré d'appartenance de r_8 à la communauté c_1 (ligne 3).
- $d_{l_{c_1}} = d_{r_3} = 0.996$ le degré d'appartenance de r_3 à la communauté c_1 (le *leader* de c_1) (ligne 4).

L'algorithme compare alors les degrés d'appartenance d_{r_8} et d_{r_3} des annuaires r_8 et r_3 à la communauté c_1 (ligne 5). Dans cet exemple, d_{r_3} est supérieur à d_{r_8} donc r_3 restera le leader de c_1 et r_8 sera un *disciple*. L'algorithme met alors à jour la listes des *disciples* de c_1 et ajoute la nouvelle arête au graphe représentatif de la communauté c_1 :

- $c_1.G.Fl = c_1.G.Fl \cup \{r_8\}$
- $E(c_1.G) = E(c_1.G) \cup \{(r_8, r_3)\}$

Nous répétons ces étapes pour les trois autres communautés. r_8 est aussi un *disciple* pour c_2, c_3 et c_4 et les communautés sont mises à jour comme suit :

	id	**f**	$\mathbf{G} = (\mathbf{L}, \mathbf{Fl}, \mathbf{E}, \mathbf{w})$
$\mathbf{c_1}$	c_1	[0.269 0.660 0.383 0.132 0.516]	$(r_3, \{r_7, r_8\}, E \cup \{(r_8, r_3)\}, w)$
$\mathbf{c_2}$	c_2	[0.329 0.842 0.056 0.294 0.674]	$(r_2, \{r_1, r_3, r_7, r_8\}, E \cup \{(r_8, r_2)\}, w)$
$\mathbf{c_3}$	c_3	[0.148 0.643 0.179 0.537 0.304]	$(r_4, \{r_5, r_7, r_8\}, E \cup \{(r_8, r_4)\}, w)$
$\mathbf{c_4}$	c_4	[0.286 0.761 0.053 0.623 0.399]	$(r_6, \{r_5, r_7, r_8\}, E \cup \{(r_8, r_6)\}, w)$

4.3.3 Mise à jour des fonctionnalités d'un annuaire

Nous rappelons que les services publiés dans un annuaire changent fréquemment (de nouveaux services arrivent et d'autres partent). Par conséquent, les fonctionnalités d'un annuaire doivent être régulièrement mises à jour. Après une mise à jour de ses fonctionnalités, un annuaire peut rester dans la même communauté ou peut être retiré. Pour cela, son degré d'appartenance à la communauté devra tout d'abord être mis à jour. Si ce degré est toujours supérieur au seuil d'acceptation `th`, l'annuaire reste dans la même communauté, sinon il est retiré.

Après une mise à jour des fonctionnalités d'un ou de plusieurs annuaires d'une communauté, quatre événements peuvent se produire :

- $E = E'$, c.-à-d. aucun changement ne se produit dans l'ensemble des membres d'une communauté, où E et E' représentent deux ensembles de membres d'une communauté c respectivement avant et après la mise à jour des fonctionnalités de certains annuaires appartenant à c.
- $E \subset E'$, c.-à-d. des nouveaux annuaires ont rejoint la communauté. (Figure 4.2.[C1])
- $E' \subset E$, c.-à-d. certains annuaires ont quitté la communauté. (Figure 4.2.[C2])

– $E \nsubseteq E' \wedge E' \nsubseteq E$, c.-à-d. des nouveaux annuaires ont rejoint la communauté et d'autres l'ont quitté. (Figure 4.2.[C3])

Après avoir définie les opérations de gestion associées aux opérations (2), (3) et (6) du cycle de vie d'un annuaire (voir Figure 4.2), nous présentons dans la section suivante les opérations de gestion associées aux différentes étapes du cycle de vie d'une communauté.

4.4 Gestion d'une communauté

Les différentes étapes du cycle de vie d'une communauté sont principalement la création, le démontage, la fusion et la division d'une communauté. Quand les degrés d'appartenance d'un annuaire à toutes les communautés existantes deviennent inférieurs au seuil `th`, une **nouvelle communauté est créée** (Figure 4.2 opération (10)). Aussi, une **communauté est démontée** (opération (13)) si elle devient vide. Afin d'assurer que notre réseau de communautés n'offre que des communautés différentes avec des annuaires offrant des fonctionnalités similaires, une communauté pourra être **fusionnée** (opération (11)) à une autre ou **divisée** (opération (12)).

Dans cette section, nous détaillons ces différentes étapes. Nous identifions aussi leurs déclencheurs, c.-à-d. l'action ou l'événement déclencheur de l'une de ces étapes, les préconditions nécessaires et les effets résultant de l'exécution de l'opération de gestion associée à chacune de ces étapes.

4.4.1 Création d'une communauté

Lorsque le degré d'appartenance d'un annuaire r aux différentes communautés d'un réseau est inférieur au seuil `th`, cet annuaire offre forcément une nouvelle fonctionnalité qui n'était pas disponible dans le réseau. Cette situation peut se produire lorsqu'un nouvel annuaire rejoint le réseau (voir section 4.3.2) ou bien après la mise à jour des fonctionnalités d'un annuaire existant (voir section 4.3.3).

Dans ce cas, une nouvelle communauté $c_{new} = (id, f, G)$ est établie automatiquement. L'annuaire r ayant déclenché la création de c_{new} prendra le rôle de *leader* dans cette communauté. La fonctionnalité moyenne de la communauté, $c_{new}.f$, sera la même que la fonctionnalité $r.f$ proposée par l'annuaire. Par la suite, des *disciples* sont recrutés pour la nouvelle communauté. Pour cela, les degrés d'appartenance des annuaires existants à la communauté c_{new} sont calculés. Tout annuaire possédant un degré d'appartenance supérieur au seuil `th` sera membre de c_{new}.

La création d'une communauté, ainsi que les ***précondition*** et ***effet*** sont illustrés dans l'Algorithme 5. Sa complexité est $O(\mid V(c.G) \mid \times \mid C \mid^2)$ où $\mid V(c.G) \mid$ exprime le nombre d'annuaires dans la communauté c et $\mid C \mid$ le nombre de communautés.

Exemple

Pour tester l'algorithme de création d'une communauté, nous reprenons notre exemple de réseau de communautés après l'ajout de l'annuaire r_8 (voir section 4.3.1). Pour cet exemple, nous augmentons la valeur du seuil `th`. Les degrés d'appartenance

Algorithme 5 Création d'une communauté

Entrée : r /* *l'annuaire ayant déclenché la création**/
Sortie : c_{new} /* *la nouvelle communauté**/

1: ***précondition*** : $\forall d \in ran(r.MEM), d < \texttt{th}$
2: Communauté c_{new} ;
3: List disciples ;
4: $c_{new}.f = r.f$
5: **pour tout** communautés $c \in C$ **faire**
6: **pour tout** annuaires $r_c \in V(c.G)$ **faire**
7: $m \longleftarrow U_{new}(r_c)$
8: $r_c.MEM \longleftarrow r_c.MEM \cup (c_{new}, m)$
9: **si** $m > \texttt{th}$ **alors**
10: $disciples \longleftarrow disciples \cup \{r_c\}$
11: $E(c_{new}.G) \longleftarrow E(c_{new}.G) \cup \{(l, r_c)\}$
12: ***effet*** : $c_{new} \in V(CG) \wedge c_{new}.G.L = \{r\} \wedge c_{new}.f = r.f \wedge c_{new}.G.Fl = disciples$
13: **retour** c_{new}

$r_8.MEM$ de l'annuaire r_8 aux quatre communautés sont ainsi inférieurs au nouveau seuil ce qui représente notre ***précondition*** pour la création d'une nouvelle communauté (ligne 1) et déclenche l'Algorithme 5. Nous appelons cette communauté c_5 et sa fonctionnalité moyenne sera égale aux fonctionnalités offertes par r_8. L'algorithme « essaye » alors de recruter des disciples pour la nouvelle communauté c_5 à partir des quatre communautés de notre réseau.

Pour la communauté c_1 par exemple, notre algorithme commence par récupérer l'ensemble de ces membres (*leader* et *disciples*) : $V(c_1.G) = \{r_3, r_7\}$. L'algorithme teste par la suite si ces deux annuaires peuvent être ou pas des *disciples* pour c_5 selon les étapes décrites dans les lignes 7-11. Nous appliquons ici ces étapes pour l'annuaire r_3 :

- **Ligne 7** : le degré d'appartenance de r_3 à c_5 est : $m = U_5(r_3) = 0.001$.
- **Ligne 8** : $r_3.MEM = r_3.MEM \cup (c_5, m) = \{(c_1, 0.996), (c_2, 0.002), (c_3, 0), (c_4, 0), (c_5, 0.001)\}$.
- **Ligne 9-11** : $m < \texttt{th}$ donc r_3 n'est pas un disciple pour c_5.

En effectuant ces mêmes étapes pour r_7 et pour les membres des communautés c_2, c_3 et c_4 nous obtenons des résultats similaires, c.-à-d. des annuaires présentant des degrés d'appartenance à c_5 inférieurs à `th`, et aucun *disciple* n'est recruté pour c_5.

A l'issue de l'exécution de notre algorithme, la communauté c_5 est ajoutée au graphe représentant notre réseau de communautés. c_5 est représentée comme suit :

	id	**f**	**G** = (**L**, **Fl**, **E**, **w**)
$\mathbf{c_5}$	c_5	[0.234 0.314 0.048 0.181 0.534]	(r_8, ∅, ∅, w)

4.4.2 Démontage d'une communauté

Une communauté c est automatiquement démontée, quand elle devient vide $|V(c.G)| = 0$ (tous ses membres la quittent ou n'existent plus). Ceci est la seule condition qui déclenche la disparition d'une communauté. Cette ***précondition*** est spécifiée comme suit :$c \in V(CG) \wedge |V(c.G)| = 0$.

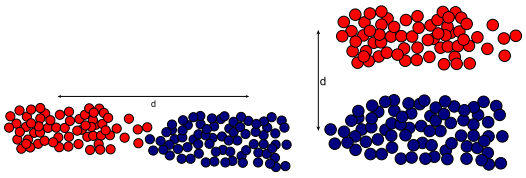
Après la suppression d'une communauté c, il faut vérifier qu'elle n'est l'extrémité d'aucune arête dans le graphe modélisant le réseau de communautés CG. Cet ***effet*** est spécifié comme suit : $c \notin V(CG) \wedge \forall c_1 \in V(CG), (c, c_1) \notin E(CG)$.

4.4.3 Fusion de communautés

Quand il s'agit de fusionner deux communautés, la première condition qui vient naturellement à l'esprit est la proximité [1]. Deux communautés sont considérées comme proches lorsque leur centre [2] le sont. Cette condition est exprimée comme suit : deux communautés c_1 et c_2 telles que $(c_1, c_2) \in E(CG)$, où CG est le graphe modélisant notre réseau de communautés, peuvent être fusionnées si $w(c_1, c_2) < \xi$, où :

- $w(c_1, c_2)$ désigne le poids de l'arête d'extrémités (c_1, c_2) et représente la distance séparant les centres de ces deux communautés (voir Définition 3.7).
- $\xi \in [0, 1]$ est le seuil à partir duquel deux communautés peuvent être fusionnées.

Cependant, cette condition de proximité est calculée en utilisant une distance géométrique sans prise en compte de la dispersion des annuaires. Ainsi, une exception à cette condition peut se produire pour deux communautés lorsque leur centre sont proches mais avec une faible densité entre eux. Ce cas peut être observé lorsque le nombre d'annuaires dans l'intersection de ces deux communautés est faible (Figure 4.3(a)) ou lorsque les deux communautés sont complètement séparées (Figure 4.3(b)). Ainsi, la proximité entre centres de deux communautés est une condition nécessaire mais pas suffisante pour vérifier la similarité entre ces deux communautés.



(a) (b)

FIGURE 4.3 – Distance entre les centres des clusters

Ainsi, nous rajoutons une deuxième condition pour la fusion de deux communautés à celle de la proximité définie ci-dessus. Cette condition sert à vérifier si l'une

1. *Closeness* en anglais
2. Le centre d'une communauté représente sa fonctionnalité moyenne f (voir Définition 3.6)

des deux communautés est complètement incluse dans l'autre garantissant ainsi une fusion sans qu'aucune fonctionnalité ne soit méconnue.

Quand ces deux conditions sont satisfaites pour deux communautés c_1 et c_2, elles seront fusionnées en produisant une nouvelle communauté c_{merg}. Le centre de c_{merg} sera représenté par la moyenne pondérée des centres $c_1.f$ et $c_2.f$ des deux communautés :

$$c_{merg}.f = \frac{c_1.f \times nb_1 + c_2.f \times nb_2}{nb_1 + nb_2} \tag{4.1}$$

Où :

- $nb_1 = \#\{r|(c_1, d_1) \in r.MEM \wedge (c_2, d_2) \in r.MEM \wedge d_1 \geq d_2\}$, le nombre d'annuaires dans l'intersection de c_1 et c_2 dont le degré d'appartenance c_1 est plus élevé.
- $nb_2 = \#\{r|(c_1, d_1) \in r.MEM \wedge (c_2, d_2) \in r.MEM \wedge d_2 \geq d_1\}$, le nombre d'annuaires dans l'intersection de c_1 et c_2 dont le degré d'appartenance c_2 est plus élevé.

L'union des membres des communautés c_1 et c_2 forme l'ensemble des membres de la nouvelle communauté c_{merg}. Parmi ces annuaires, nous désignons le *leader* comme étant celui avec le plus grand degré d'appartenance à c_{merg} et les autres annuaires prendront le rôle de *disciple*. Comme conséquence à la fusion de deux communautés, la communauté c_{merg} est ajoutée au graphe CG et les deux communautés c_1 and c_2 sont supprimées. Ainsi, toute arête ayant une extrémité dans l'une de ces deux communautés est supprimée.

La Fusion de deux communautés, ainsi que la ***précondition*** nécessaire et l'***effet*** de la fusion sont illustrés dans l'Algorithme 6.

La complexité de l'Algorithme 6 est $O(\mid V(c.G) \mid \times \mid C \mid)$ où $\mid V(c.G) \mid$ exprime le nombre d'annuaires dans la communauté c et $\mid C \mid$ le nombre de communautés.

Exemple

Pour tester l'algorithme de fusion de deux communautés, nous reprenons notre exemple support de réseau de communautés (voir section 4.2) et nous fixons le seuil de proximité ξ à 0.2. Nous supprimons l'annuaire r_4 le *leader* de c_3. L'algorithme de resélection de *leader* est alors exécuté (Algorithme 4) et r_7 est désigné comme nouveau *leader* pour c_3. Les communautés sont mises à jour comme suit :

	id	**f**	$\mathbf{G = (L, Fl, E, w)}$
$\mathbf{c_1}$	c_1	[0.269 0.660 0.383 0.132 0.516]	(r_3, {r_7}, E, w)
$\mathbf{c_2}$	c_2	[0.329 0.842 0.056 0.294 0.674]	(r_2, {r_1, r_3, r_7}, E, w)
$\mathbf{c_3}$	c_3	[0.148 0.643 0.179 0.537 0.304]	(r_7, {r_5}, E, w)
$\mathbf{c_4}$	c_4	[0.286 0.761 0.053 0.623 0.399]	(r_6, {r_5, r_7}, E, w)

La communauté c_3 est alors composée de r_7 et r_5 et est totalement incluse dans c_4. De plus, la distance entre c_3 et c_4 est inférieure au seuil ξ : $w(c_3, c_4) = distance(c_3.f, c_4.f) = 0.016$. La ***précondition*** pour la fusion des deux communautés c_3 et c_4 est ainsi validée (ligne 1) et l'Algorithme 6 est déclenché.

Algorithme 6 Fusion de deux communautés

Entrée : c_1, c_2 /* *les deux communautés ayant déclenchées la fusion**/

Sortie : c_{merg} /* *la nouvelle communauté**/

1: ***précondition*** : $c_2 \in V(CG), \exists c_1 | w(c_1, c_2) < \xi \wedge V(c_1.G) \subset V(c_2.G)$
2: Communauté c_{merg} ;
3: List membres ;
4: $m_{max} = 0$;
5: $nb_1 = \#\{r | (c_1, d_1) \in r.MEM \wedge (c_2, d_2) \in r.MEM \wedge d_1 \geq d_2\}$
6: $nb_2 = \#\{r | (c_1, d_1) \in r.MEM \wedge (c_2, d_2) \in r.MEM \wedge d_2 \geq d_1\}$
7: $c_{merg}.f = \frac{c_1.f \times nb_1 + c_2.f \times nb_2}{nb_1 + nb_2}$
8: membres=$c_1.G.L \cup c_2.G.L \cup c_1.G.FL \cup c_2.G.FL$
9: **pour tout** annuaires $r \in membres$ **faire**
10: $m \longleftarrow U_{merg}(r)$
11: $r.MEM \longleftarrow r.MEM \cup (c_{merg}, m)$
12: **si** $m > m_{max}$ **alors**
13: leader=r
14: $c_{merg}.G.L = leader$
15: $c_{merg}.G.Fl = membres \setminus \{leader\}$
16: ***effet*** : $V(CG) = (V(CG) - \{c_1, c_2\}) \cup \{c_{merg}\}$
17: **retour** c_{merg}

L'algorithme commence par calculer les fonctionnalités moyennes de la communauté c_5 résultant de la fusion de c_3 et c_4 en utilisant l'équation 4.1 comme suit :

$$c_5.f = \frac{c_3.f \times nb_3 + c_4.f \times nb_4}{nb_3 + nb_4} \text{ (ligne7)}$$

Selon la Table 4.1, le degré d'appartenance de r_7 à c_3 (0.004) est plus élevé que celui à c_4 (0.003) et celui de r_5 à c_4 (0.998) est plus élevé que celui à c_3 (0.001). nb_3 et nb_4 sont ainsi égaux à 1 et :

$$c_5.f = \frac{c_3.f + c_4.f}{2} = [0.217\ 0.702\ 0.116\ 0.580\ 0.3515]$$

L'ensemble des *membres* de la nouvelle communauté c_5 est alors composé par ceux des deux communautés c_3 et c_4 :

$$membres = \{r_5, r_6, r_7\} \text{ (ligne 8)}$$

L'algorithme calcule alors les degrés d'appartenance de ces annuaires à la nouvelle communauté afin d'élire le *leader* (ligne 9-10). Nous appliquons ici ces étapes pour l'annuaire r_5 :

– **Ligne 9** : le degré d'appartenance de r_5 à c_5 est : $m = U_5(r_5) = 0.999$.
– **Ligne 10** : $r_5.MEM = r_5.MEM \cup (c_5, m) = \{(c_1, 0), (c_2, 0), (c_5, 0.999)\}$.

En effectuant ces mêmes étapes pour r_6 et r_7 nous obtenons les degrés d'appartenance à c_5 suivants : $U_5(r_6) = 0.992$ et $U_5(r_7) = 0.004$. Ainsi, l'algorithme

attribue à r_5 le rôle de *leader* dans la communauté c_5 vu qu'il possède le degré d'appartenance le plus élevé (ligne 12-13). Les membres de la nouvelle communauté sont alors définis par notre algorithme comme suit :

- **Ligne 14** : le *leader* $c_5.G.L = r_5$
- **Ligne 15** : les disciples $c_5.G.Fl = membres \setminus \{r_5\} = \{r_6, r_7\}$

Comme effet à l'exécution de cet algorithme, c_5 est ajouté au graphe représentant le réseau de communauté et les communautés c_3 et c_4 en sont soustraites (ligne 16) : $V(CG) = (V(CG) - \{c_3, c_4\}) \cup \{c_5\}$ et notre réseau de communautés est mis à jour comme suit :

	id	**f**	**G = (L, Fl, E, w)**
$\mathbf{c_1}$	c_1	[0.269 0.660 0.383 0.132 0.516]	$(r_3, \{r_7\}, E, w)$
$\mathbf{c_2}$	c_2	[0.329 0.842 0.056 0.294 0.674]	$(r_2, \{r_1, r_3, r_7\}, E, w)$
$\mathbf{c_5}$	c_3	[0.217 0.702 0.116 0.580 0.351]	$(r_5, \{r_6, r_7\}, E, w)$

4.4.4 Division d'une communauté

Une communauté doit être divisée si elle devient clairsemée[4] autour de son centre. La sparsité (Eng. sparsity) d'une communauté est décrite par une densité d'annuaires faible aux alentours du centre de la communauté et une dispersion entre ses membres (Figure 4.4).

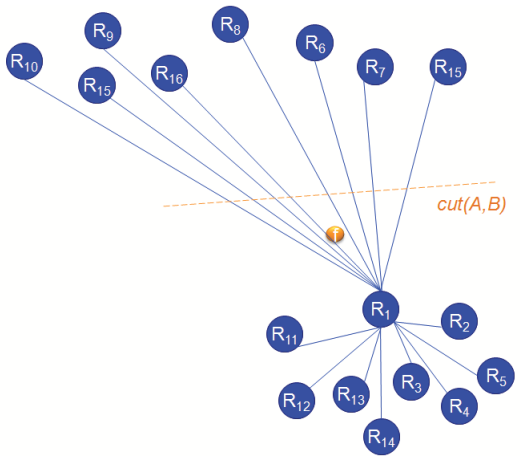


FIGURE 4.4 – Précondition de la division d'une communauté

Quand cette condition est satisfaite, la division d'une communauté peut être vue comme un problème de partitionnement de graphe. Pour diviser une communauté, nous nous inspirons de l'algorithme *min-max cut* [Ding 2001, Nie 2010]. Cet algorithme permet de partitionner un graphe en minimisant la similarité entre les deux sous-graphes résultants tout en maximisant la similarité à l'intérieur de chacun de ses sous-graphes.

4. *Sparse* en anglais

Le *min-max cut* algorithme prend un graphe pondéré G, représenté par sa matrice d'adjacence (voir section 3.3.2.1), en entrée et permet de le diviser en deux graphes G_1 et G_2. Cet algorithme est basé sur la minimisation de la fonction objectif suivante :

$$Mcut = \frac{W(G_1, G_2)}{W(G_1, G_1)} + \frac{W(G_1, G_2)}{W(G_2, G_2)}$$

Où :

- $W(G_1, G_2) = \sum_{u \in G_1, v \in G_2} w(u, v)$ est une fonction permettant de calculer la somme des poids des arrêtes joignant les nœuds des deux graphes G_1 et G_2 ($w(u, v)$ est une fonction retournant le poids de l'arête joignant les deux sommets u et v (voir Définition 3.6)).
- $W(G_1, G_1)$ (respectivement $W(G_2, G_2)$) permet de calculer la somme des poids des arrêtes joignant les nœuds du graphe G_1 (respectivement G_2).

Une communauté est représentée par un graphe en étoile et les différentes pondérations dans ce graphe sont associées aux arêtes reliant *leader* et *disciples*. Toutefois, et afin que l'algorithme *min-max cut* puisse diviser le graphe en deux sous graphes, nous devons aussi considérer les poids entre les différents *disciples*. Pour cela, ces poids doivent être fournis avec le graphe de la communauté en entrée pour l'algorithme.

Pour réaliser la division d'une communauté, nous utilisons l'algorithme *min-max cut* avec comme entrée la matrice d'adjacence $A_{G \cup \overline{G}}$ du graphe résultant de l'union du graphe G d'une communauté c avec son complément $\overline{G}$ (voir section 3.3.2.3). Les arêtes du graphe $\overline{G}$ sont pondérées avec les degrés de similarité entre les différents *disciples*. Les valeurs $a_{i,j}$ de la matrice d'adjacence $A_{G \cup \overline{G}}$ sont définies comme suit :

$$a_{i,j} = \begin{cases} 0 & \text{si} \quad i = j \\ w(r_i, r_j) = cosine(r_i.f, r_j.f) & \text{sinon} \end{cases}$$

Où $w(r_i, r_j)$ représente la similarité entre deux nœuds r_i et r_j (c.-à-d. annuaires) du graphe $G \cup \overline{G}$. Cette similarité est obtenue en utilisant la mesure cosinus (voir formule (3.2)).

Ainsi, la matrice d'adjacence du graphe complet $c.G \cup \overline{c.G}$ contiendra les degrés de similarités entre tous les membres de la communauté c et l'application de l'algorithme *min-max cut* (en utilisant cette matrice) permettra de diviser ce graphe en deux sous graphes. Ces étapes sont décrites dans le lignes 4-15 de l'Algorithme 7. La matrice d'adjacence $A_{G \cup \overline{G}}$ est tout d'abord construite (lignes 1-14) comme décrit plus haut. Par la suite, cette matrice est passée comme paramètre d'entrée à la fonction *Cut* implémentant l'algorithme *min-max cut*.

Par la suite (lignes 16-17), nous utilisons l'Algorithme 8 pour créer deux communautés (c_1 et c_2) à partir des sous graphes (G_1 et G_2) résultants. L'Algorithme 8 permet de :

1. calculer la fonctionnalité moyenne $c.f$ de la nouvelle communauté. Nous avons choisi d'utiliser la moyenne des fonctionnalités des nœuds (c.-à-d. annuaires r) du sous graphe pour représenter la fonctionnalité moyenne de la communauté (lignes 2-5).

Algorithme 7 Division d'une communauté

Entrée : c /* *la communauté ayant déclenchée la division* */
Sortie : c_1, c_2 /* *les nouvelles communautés* */

précondition : c est clairsemée
Communauté c_1,c_2 ;
Graphe G_1,G_2 ;
pour tout annuaires $r_i \in V(c.G)$ **faire**
  pour tout annuaires $r_j \in V(c.G)$ **faire**
    si $r_i \in c.G.L$ **ou** $r_j \in c.G.L$ **alors**
      $A_{G \cup \overline{G}}[i][j] = c.G.w(r_i, r_j)$
      $A_{G \cup \overline{G}}[j][i] = A_{G \cup \overline{G}}[i][j]$
    sinon
      si $r_i = r_j$ **alors**
        $A_{G \cup \overline{G}}[i][j] = 0$
      sinon
        $A_{G \cup \overline{G}}[i][j] = cosine(r_i.f, r_j.f)$
        $A_{G \cup \overline{G}}[j][i] = A_{G \cup \overline{G}}[i][j]$
$Cut(A_{G \cup \overline{G}}, G1, G2)$ /* *une fonction implémentant l'algorithme* min-max cut */
/* *cette fonction crée les deux graphes* G_1 *et* G_2 */
$c_1 = Algorithme\ 8(G_1)$
$c_2 = Algorithme\ 8(G_2)$
effet : $V(CG) = (V(CG) - \{c\}) \cup \{c_1, c_2\}$
retour c_1, c_2

2. sélectionner le *leader* $c.G.L$ de la nouvelle communauté. Le *leader* sera l'annuaire possédant le plus grand degré de similarité avec la communauté c (lignes 7-11).
3. définir l'ensemble des *disciples* $c.G.Fl$ de la nouvelle communauté. Cet ensemble est représenté par l'ensemble des nœuds (c.-à-d. annuaires) du sous graphe duquel on soustrait le leader (ligne 12).
4. définir l'ensemble des arêtes $c.G.E$ de la nouvelle communauté. En fait, le sous graphe que nous utilisons est un graphe complet puisqu'il résulte de division d'un graphe complet. Vu que nous représentons une communauté par un graphe en étoile, nous supprimons toutes les arêtes du sous graphe et nous définissons comme ensemble $c.G.E$ les arêtes reliant le *leader* aux différents *disciples* (lignes 13-15).
5. mettre à jour les degrés d'appartenance MEM des membres de c. Pour cela, nous ajoutons pour chaque annuaire r de c son degré d'appartenance à c en utilisant la formule (3.5) (lignes 16-18).

Comme conséquence à la division de la communauté c, la communauté c est supprimée du graphe CG et les deux communautés c_1 et c_2 sont ajoutées (Algorithme 7 ligne 18).

Algorithme 8 Création d'une nouvelle communauté suite à une division

Entrée : $V(G)$ /* *l'ensemble des nœuds d'un graphe**/
Sortie : c /* *une nouvelle communauté**/
1: Communauté c ;
2: $somme = null$ /* *un vecteur initialisé à null**/
3: **pour tout** annuaires $r \in V(G)$ **faire**
4: $somme = somme + r.f$
5: $c.f = \frac{somme}{|V(G)|}$
6: $max = 0$
7: **pour tout** annuaires $r \in V(G)$ **faire**
8: **si** $cosine(r.f, c.f) > max$ **alors**
9: $max = cosine(r.f, c.f)$
10: $leader = r$
11: $c.G.L = leader$
12: $c.G.Fl = V(G) \setminus \{leader\}$
13: $c.G.E = \{\}$
14: **pour tout** annuaires $r \in V(c.G.Fl)$ **faire**
15: $c.G.E = E \cup \{(r, c.G.L)\}$
16: **pour tout** annuaires $r \in V(c.G)$ **faire**
17: $m \longleftarrow U_c(r)$
18: $r.MEM \longleftarrow r.MEM \cup (c, m)$
19: **retour** c

La complexité de l'Algorithme 7 est quadratique. En effet, la complexité de

Cut (ligne 15 de l'Algorithme 7) est de $O(\mid E(c.G) \mid + \mid V(c.G) \mid)$ [Ding 2001] où $\mid E(c.G) \mid$ exprime le nombre d'arrêtes dans le graphe G représentant la communauté c. Vu que G est un graphe complet, $\mid E(c.G) \mid = \frac{|V(c.G)| \times (|V(c.G)|-1)}{2}$. Ainsi la complexité de *Cut* est $O(\mid V(c.G) \mid^2)$. Pour l'Algorithme 8 (ligne 16-17 de l'Algorithme 7), sa complexité est $O(\mid V(c.G) \mid \times \mid C \mid)$. En considérant que le nombre de communautés est généralement inférieur au nombre d'annuaires ($\mid C \mid < \mid V(c.G) \mid$), la complexité de l'Algorithme 7 est $O(\mid V(c.G) \mid^2)$ (ligne 4-14 et) où $\mid V(c.G) \mid$ exprime le nombre d'annuaires dans la communauté c.

4.5 Conclusion

Dans ce chapitre, nous avons identifié les différentes étapes des cycles de vie d'un annuaire au sein d'une communauté et celui d'une communauté d'annuaires. Par la suite, nous avons spécifié les opérations de gestion nécessaires pour quelques-unes de ces étapes. Les étapes nécessitant des mécanismes de gestion sont essentiellement celles pouvant impliquer (directement ou indirectement) un changement dans l'organisation de notre réseau. Ceci nous a permis d'assurer la consistance de notre organisation en communautés face aux différents changements qui peuvent arriver vu la dynamicité de notre environnement. Ainsi, nous assurons le deuxième objectif de notre thèse : « **proposer une approche pour la gestion d'un ensemble d'annuaires organisés en groupes** ».

Parmi les étapes du cycle de vie d'un annuaire, nous avons défini les actions qui devront être enclenchées pour gérer : l'arrivée d'un nouvel annuaire, l'adhésion d'un annuaire à une communauté et la mise à jour des fonctionnalités d'un annuaire.

Pour le cycle de vie d'une communauté, les étapes qui nécessitent des mécanismes de gestion sont : la création d'une communauté, le démontage d'une communauté, la fusion de communautés et la division de communautés. Pour chacune de ces étapes, nous avons défini les conditions nécessaires pour le déclenchement d'une opération de gestion, l'opération de gestion adéquate et les conséquences de l'opération de gestion.

Notre approche de gestion des communautés permet de préserver la consistance d'une organisation en communautés conçue selon notre approche de création de communautés (voir section 3.4) face aux différents changements qui peuvent se produire. La validation de ce travail a été faite à travers deux publications [Bouchaalaa 2011a, Bouchaalaa 2011b] et la simulation d'un gestionnaire de communautés que nous présentons dans le chapitre 6. Dans le chapitre 6, nous évaluerons aussi notre approche de gestion en comparant les résultats obtenus aux résultats d'une réexécution de notre approche de construction de communautés présentée dans la section 3.4. Dans le chapitre suivant, nous nous intéressons au processus de découverte de services dans un réseau d'annuaires organisés en communautés.

Chapitre 5

Découverte d'annuaires dans un réseau de communautés

Sommaire

5.1 Introduction

A travers les deux chapitres précédents, nous avons assuré les deux premiers objectifs de notre thèse : proposer « une approche implicite pour organiser un ensemble d'annuaires » et « une approche pour la gestion d'un ensemble d'annuaires organisés en groupes ». Dans ce chapitre, notre objectif est de répondre au troisième et dernier objectif de notre thèse en proposant « une approche pour la découverte d'annuaires dans un réseau d'annuaires organisés en communautés ».

Dans un réseau d'annuaires organisé en communautés, nous considérons la découverte de services comme un processus séquentiel en deux étapes : la première étape est la découverte d'annuaires répondant le mieux à la requête d'un demandeur de services. La deuxième étape consiste à la sélection de services dans le(s) annuaire(s) choisi(s) dans la première étape. Dans ce chapitre nous nous intéressons uniquement à la première étape. La deuxième étape, a été étudiée au sein de notre équipe [Chabeb 2010, Nguyen 2010, Nguyen 2011].

L'organisation d'un réseau d'annuaires en communautés permet de limiter l'espace de recherche d'un demandeur de services en réduisant le nombre de communautés et ainsi celui des annuaires à parcourir. En effet, elle permet d'appliquer un premier filtre dans le processus de découverte d'annuaires en sélectionnant la ou les communauté(s) d'annuaires adéquate(s) aux **besoins fonctionnels** d'un demandeur de services.

Vu que le nombre d'annuaires dans une communauté peut être très important, la découverte d'un annuaire au sein d'une communauté est en soi un problème à résoudre. Pour faire face à ce problème et afin d'améliorer le processus de découverte, nous proposons d'utiliser des techniques de recommandation afin de limiter l'espace de recherche d'un demandeur de services. Nous adoptons une approche de filtrage collaboratif basée sur les résultats des processus de **découvertes précédentes** et les **besoins non fonctionnels** des demandeurs de services pour recommander le (ou les) annuaire(s) adéquat(s) à une requête de découverte de services.

Le reste de ce chapitre est organisé comme suit. Dans la section 5.2, nous présentons le format utilisé dans notre approche pour représenter une requête de découverte de services. Cette requête permet de représenter la fonctionnalité recherchée par un demandeur de services ainsi que sa caractérisation (ses besoins non fonctionnels et sa trace d'exécution). Dans la section 5.3, nous présentons les deux étapes, la sélection de communauté et d'annuaires, du processus de découverte d'annuaires de notre approche. Nous présentons enfin dans la section 5.4 la technique de recommandation que nous proposons pour assurer l'étape de sélection d'annuaires. Enfin, nous concluons dans la section 5.5.

5.2 Requête de découverte de services Web

Avant d'introduire notre approche de découverte, nous définissons dans cette section la requête de découverte de services utilisée. Cette requête, doit exprimer les besoins fonctionnels et non fonctionnels d'un demandeur de services. Un demandeur de services est à la recherche d'une fonctionnalité particulière offerte par un service qui est généralement représentée par une description de service. Ainsi, nous utilisons le même format que celui de la description du service pour décrire les besoins fonctionnels d'un demandeur de services. De plus, la requête de découverte de services doit refléter les besoins non fonctionnels du demandeur de services ainsi que sa trace d'exécution. Ces données sont utilisées par une technique de recommandation employée pour recommander un annuaire à un demandeur de services (les détails seront fournis dans la section 5.3.2).

5.2.1 Définition d'une requête

Nous décrivons une requête de découverte de services Q par un couple $< SD, RC >$ où :

SD représente le besoin fonctionnel du demandeur de services par une **description abstraite d'un service**. SD est exprimé en utilisant le langage SAWSDL.

Nous utilisons SAWSDL puisque c'est un langage largement utilisé par les fournisseurs de services pour décrire leurs services et par notre approche pour construire les communautés (voir section 3.2).

RC abréviation de l'anglais ***R**equester **C**haracterization*. Représente la **caractérisation** d'un demandeur de services. Nous définissons une caractérisation comme étant une structure de données contenant la trace des services déjà utilisés par le demandeur de services et ses besoins non fonctionnels. La caractérisation d'un demandeur de services, ou caractérisation utilisateur tout court, est définie dans Définition 5.1.

Définition 5.1 (Caractérisation utilisateur)
Nous définissons une caractérisation utilisateur RC par le couple (T,B_{nf}) où :
- $T = \{t_1, t_2, \ldots, t_k\}$ est la *trace* de la caractérisation. Elle représente la liste des services invoqués par un consommateur de services. Dans cette trace, un service invoqué est représenté par le concept sémantique t annotant l'opération utilisée. Nous rappelons que tous les fournisseurs de services utilisent les mêmes ressources sémantiques, ce qui assure la cohérence d'une trace. La trace joue un rôle important dans la recommandation d'un annuaire vu que des demandeurs de services possédant des traces similaires, posséderaient les mêmes besoins en termes de services recherchés.
- B_{nf} est l'ensemble des besoins non fonctionnels d'un demandeur de services. Ces besoins sont représentés par des couples (b_i, v_i) où b_i est un concept sémantique, pris d'une ontologie de propriétés non fonctionnelles, représentant un besoin non fonctionnel et v_i est le poids souhaité pour cette propriété.

Notre approche de découverte d'annuaires adopte une technique de recommandation pour sélectionner les annuaires adéquats aux besoins d'un demandeur de services. Cette technique utilise la caractérisation du demandeur de services ainsi que les caractérisations des demandeurs de services précédents. Les caractérisations des demandeurs de services précédents désignent les caractérisations appartenant à des demandeurs de services ayant déjà découvert un service. Les détails sur notre technique de recommandation seront fournis dans les section 5.3.2 et 5.4.

5.2.2 Représentation d'une requête

Le langage SAWSDL, basé sur WSDL, permet de décrire les fonctionnalités d'un service, appelé partie *abstraite*, ainsi que les protocoles de communications et la localisation du service, appelé partie *concrète*. Le besoin fonctionnel SD d'un demandeur de services est décrit en utilisant uniquement la partie abstraite d'une description SAWSDL. De cette façon, une SD est décrite en utilisant les éléments WSDL suivants : `<interface>`, `<operation>`, `<input>` et `<output>`. Chacun de ces éléments est annoté par un (ou plusieurs) concept sémantique d'une ontologie en utilisant l'attribut `modelReference`. La structure d'une description SD est illustrée dans la Figure 5.1.

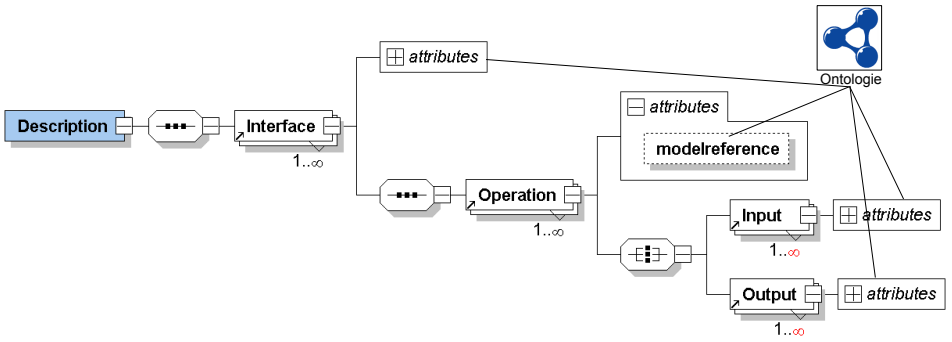


FIGURE 5.1 – Structure d'une SD

Nous représentons une caractérisation utilisateur RC aussi par un document XML. Nous décrivons la trace d'un demandeur de services (T) à travers l'élément `<trace>` en utilisant des éléments `<operation>` pour chaque opération (t_k) utilisée. Nous définissons aussi un élément `<non_functional_requirements>` représentant les besoins non fonctionnels (`<requirement>`) d'un demandeur de services (B_{nf}) sous forme de paires (besoin (b_i) (`<label>`),valeur (v_i) (`<value>`)). La structure d'une RC est illustrée dans la Figure 5.2.

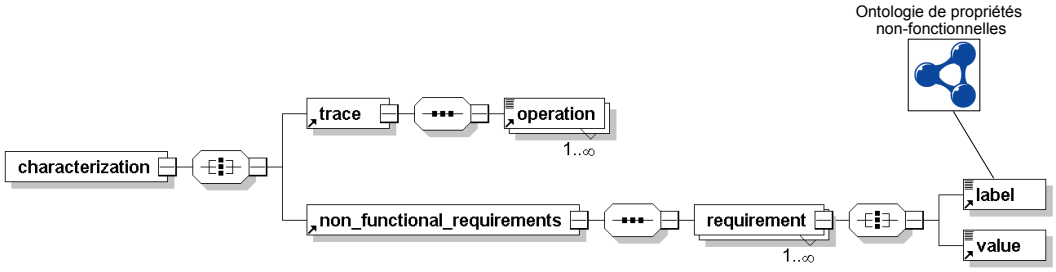


FIGURE 5.2 – Structure d'une RC

Nous donnons un exemple d'une SD et d'une RC d'une requête utilisateur respectivement dans les annexes A.3 et A.4.

5.3 Découverte d'annuaire(s)

Afin d'être sûr de découvrir le service le plus approprié à ses besoins, un demandeur de services devra parcourir tous les annuaires disponibles. Évidemment, un tel processus n'est pas concevable dans la réalité, surtout quand le nombre d'an-

nuaires disponibles et les services qu'ils offrent est très important. Dans ce travail, nous proposons un système de découverte de services permettant de réduire l'espace de recherche d'un demandeur de services dans un environnement distribué d'annuaires et organisés en communautés. La réduction de l'espace de recherche se fait tout d'abord en sélectionnant la communauté d'annuaires la plus adéquate au demandeur de services. Par la suite, notre système recommande un ou plusieurs annuaire(s) de cette communauté répondant le mieux aux besoins du demandeur de services en fonction de sa caractérisation. Dans la suite de cette section, nous détaillons ces deux étapes.

5.3.1 Sélection de communauté

L'étape de sélection d'une communauté est la première dans notre processus de découverte d'un annuaire. Cette étape agit comme un premier filtre pour réduire l'espace de recherche d'un demandeur de services. Afin d'améliorer le processus de découverte d'un service, il est plus logique d'orienter la recherche sur la communauté offrant des fonctionnalités similaires à son besoin fonctionnel SD.

Notre méthode de sélection d'une communauté pour une requête d'un demandeur de services est décrite dans l'Algorithme 9. La sélection d'une communauté est basée sur le calcul la similarité entre le besoin fonctionnel SD exprimé dans la requête Q d'un demandeur de services et les différentes fonctionnalités moyennes f des communautés c. Pour calculer cette similarité, nous utilisons la fonction cosinus (formule (3.2)) vu que nous l'avons utilisé pour construire notre réseau de communautés (voir section 3.4). Nous rappelons que cette fonction permet de calculer le degré de similarité entre deux vecteurs. Pour cela, nous transformons SD, la description du besoin fonctionnel du demandeur de services écrite en SAWSDL, sous un format vectoriel. Nous utilisons le même modèle vectoriel introduit dans la section 3.4.2 et qui a été utilisé pour créer les représentations vectorielles des descriptions WSRD. Finalement, la communauté possédant le degré de similarité le plus élevé sera sélectionnée pour la requête du demandeur de services. La complexité de l'Algorithme 9 est de $O(t+ \mid V(c.G) \mid)$ où t exprime le nombre de concepts formant l'ontologie utilisée et $\mid V(c.G) \mid$ le nombre d'annuaires dans la communauté c.

5.3.2 Sélection d'annuaires

Après la sélection d'une communauté, le demandeur de services pourrait être à nouveau confronté au grand nombre de services offerts par les annuaires de cette communauté. L'exécution d'un appariement sémantique entre la requête de l'utilisateur et tous les services disponibles pour découvrir le service le plus adéquat nécessitera beaucoup de ressources. Pour remédier à ce problème, nous proposons de limiter l'espace de recherche d'un demandeur de services à l'intérieur d'une communauté en utilisant une technique de recommandation qui guide sa requête vers l'annuaire le plus approprié à ses besoins. Pour l'étape de sélection d'annuaires, nous

Algorithme 9 *Sélection de communautés*

Entrée : $Q =< SD, RC >$ /* *Requête d'un demandeur de services*/
Sortie : id /* *Identifiant de la communauté d'annuaires sélectionnée*/

1: Sim_{max}=0 ;
2: $SD_v = RepresentationVectorielle(SD)$; /* *la représentation vectorielle de SD*/
3: **pour tout** Communauté $c \in C$ **faire**
4: **si** $Sim_{max} < cosine(SD_v, c.f)$ **alors**
5: $Sim_{max} = cosine(SD_v, c.f)$;
6: $id_{max} = c.id$;
7: **retour** id_{max} ;

mettons l'accent sur les historiques des utilisations et les besoins non-fonctionnels des demandeurs de services passées alors que l'étape de sélection de communauté était centrée sur les fonctionnalités offertes par les annuaires.

La technique de recommandation que nous proposons est réalisée sur la base de la caractérisation utilisateur *RC* et la liste des caractérisations des demandeurs de services précédents. Ce choix est motivé dans la section 5.4.1. Nous associons à chaque annuaire dans notre réseau de communautés une liste de caractérisations des demandeurs de services précédents ayant utilisé l'un des services de cet annuaire. Ainsi, l'annuaire possédant des caractérisations d'utilisateurs passés similaires à la caractérisation du demandeur de services est recommandé au demandeur de services.

Dans notre travail, nous proposons de fusionner les caractérisations d'utilisateurs passés d'un annuaire en une seule caractérisation globale appelée *GRC* (abréviation de l'anglais ***G**lobal **R**egistry **C**haracterization*). La *GRC* d'un annuaire est définie dans la Définition 5.2.

Définition 5.2 (Caractérisation globale d'un annuaire)
Nous définissons une caractérisation globale d'un annuaire *GRC* par le couple (T,B_{nf}) où :
- $T = \{(t_1, n_1), (t_2, n_2), \ldots, (t_k, n_k)\}$ représente la *trace globale* d'une caractérisation globale par une liste des concepts sémantiques t annotant les opérations présentes dans les traces des caractérisations des utilisateurs passés et le nombre d'occurrences n de ces concepts dans ces différentes traces.
- B_{nf} est l'ensemble des besoins non fonctionnels globaux. Ils sont représentés par des couples (b_i, v_i) où b_i est extrait à partir des besoins non fonctionnels des caractérisations des utilisateurs passés d'un annuaire et v_i est la moyenne de toutes les valeurs correspondant à ce concept dans ces caractérisations.

De même qu'une *RC*, une *GRC* est représentée par un document XML. Nous décrivons sa trace globale (T) à travers l'élément `<global_trace>` en utilisant des éléments `<operation>` pour chaque opération utilisée. Chaque `<operation>` est re-

présentée sous forme de paires (opération (t_k) (`<name>`),nombre d'occurrences (n_k) (`<number>`)). Nous définissons aussi un élément `<global_non_functional_req>` représentant les besoins non fonctionnels (`<requirement>`) d'un demandeur de services (B_{nf}) sous forme de paires (besoin (b_i) (`<label>`),valeur (v_i) (`<value>`)). La structure d'une *GRC* est illustrée dans la Figure 5.3.

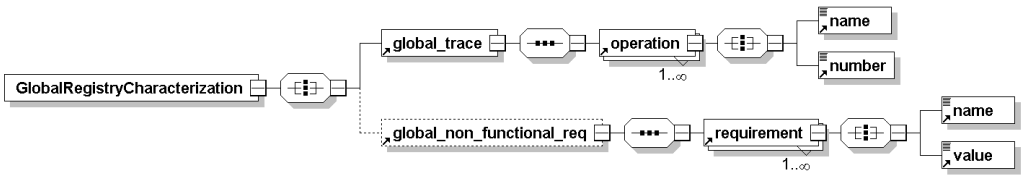


FIGURE 5.3 – Structure d'une *GRC*

Dans la Figure 5.4 nous donnons un exemple de fusion de deux caractérisations utilisateurs $RC1$ et $RC2$ en une seule caractérisation globale *GRC*. Un exemple complet d'une *GRC* est donné dans l'annexe A.5.

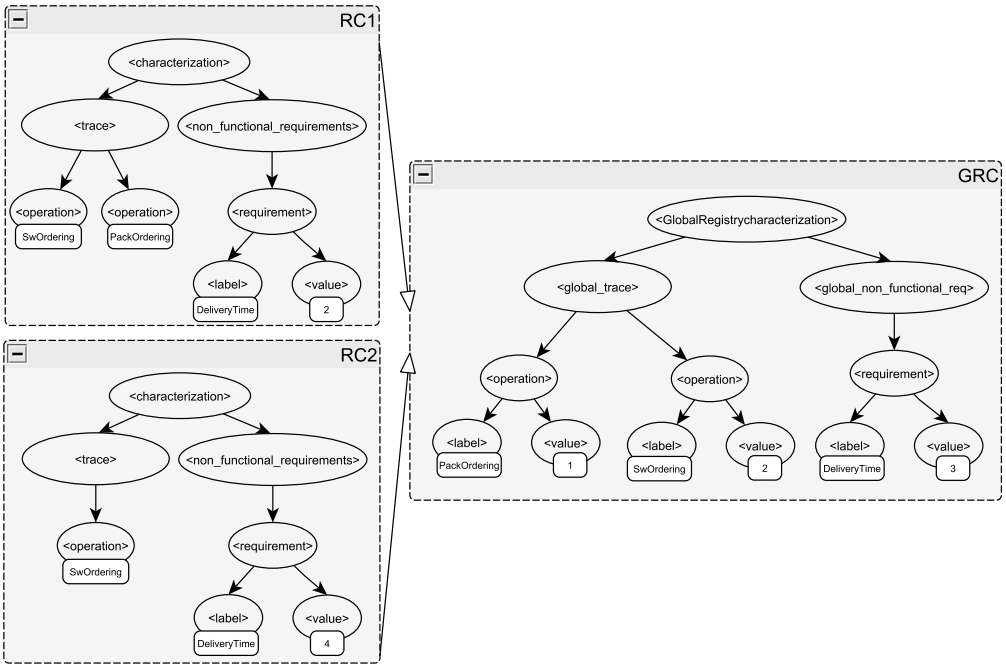


FIGURE 5.4 – Exemple du calcul d'une caractérisation globale *GRC*

Notre méthode de sélection d'annuaires à partir d'une communauté est décrite dans l'Algorithme 10. En utilisant l'identifiant de la communauté sélectionnée pour la requête (voir section 5.3.1) et la caractérisation *RC* du demandeur de services, cet algorithme permet de recommander des annuaires à un demandeur de services. Pour chaque annuaire r appartenant à la communauté sélectionnée (ligne (3) dans

l'Algorithme 10), on calcule le facteur de recommandation associé en utilisant la RC du demandeur de services et la GRC_r de l'annuaire r. Ce facteur est calculé en utilisant une fonction RF (des détails sur notre technique de recommandation sont fournis dans la section 5.4). Les paires *annuaires* (représentés par leur identifiant $r.id$) et *facteur de recommandation associé* (calculé en utilisant $RF(RC, GRC_r)$) sont sauvegardés dans un ensemble L (ligne (4)). Cet ensemble est par la suite trié selon les facteurs de recommandations associés aux annuaires (ligne (5)) et les N-annuaires possédant les facteurs de recommandations les plus élevés seront recommandés au demandeur de services (ligne (6-8)).

Algorithme 10 *Recommandation d'annuaires*

Entrée : $id_{selectionné}$ /* *Identifiant de la communauté sélectionnée*/*
Entrée : $Q =< SD, RC >$ /* *Requête d'un demandeur de services*/*
Sortie : R /* *Ensemble des identifiants des N annuaires recommandés*/*

1: $L = \{\}$
2: $R = \{\}$
3: **pour tout** $r \in V(c.G)$ *tel que* $(c.id = c.id_{selectionné})$ **faire**
4: $L = L \cup (r.id, RF(RC, GRC_r))$ /* *RF est une fonction qui calcule le facteur de recommandation associé à un annuaire*/*
5: L=Tri(L) ; /* *Tri est une fonction qui trie un ensemble L sur la base du facteur de recommandation*/*
6: **pour** $i = 1$ to N **faire**
7: $R = R \cup getId(L.getElement(i))$
8: **retour** R

5.3.3 Synthèse

Les deux étapes précédentes, la sélection de communauté (section 5.3.1) et la sélection d'annuaires (section 5.3.2), permettent de réduire considérablement l'espace de recherche d'un demandeur de services. Il est probable que cette réduction entraîne une perte de services pouvant répondre à la requête du demandeur de services. Toutefois, cette perte est maîtrisée puisqu'on élimine les communautés qui répondent mal au besoin fonctionnel SD d'un demandeur de services et les annuaires qui ne satisfont pas sa caractérisation RC.

Nous rappelons que le but de notre approche de découverte d'annuaires n'est pas d'avoir la complétude dans les réponses à une requête de découverte de services mais plutôt d'obtenir rapidement des réponses pertinentes. Ainsi, en réduisant l'espace de recherche du demandeur de services à quelques annuaires, un appariement sémantique entre la requête du demandeur de services, précisément SD, et les descriptions des services offerts par les annuaires sélectionnés est possible dans un intervalle de temps raisonnable. Dans la section suivante, nous exposons la technique de recommandation utilisée pour l'étape de sélection d'annuaires.

5.4 Recommandation d'annuaires basée sur les caractérisations

5.4.1 Choix de techniques de recommandation

Les systèmes de recommandation représentent une forme spécifique de filtrage d'information. Ils visent à aider les utilisateurs en leur proposant des produits et des services intéressants dans des situations où le nombre et la complexité de l'offre dépassent la capacité de l'utilisateur de les examiner et prendre une décision [Resnick 1997, Felfernig 2007]. Ces systèmes sont utilisés dans divers domaines tels que la recherche documentaire, le commerce électronique, etc. L'objectif de ces systèmes est de donner des conseils à un utilisateur sur les produits ou services qui pourraient l'intéresser en lui laissant le choix du produit (ou service). Ainsi, quand un utilisateur est amené à choisir un produit (ou service) parmi une large liste d'éléments, un système de recommandation pourra l'aider [Werthner 2007]. Actuellement, plusieurs applications commerciales (l'entreprise de commerce électronique Amazon [1], le service de recommandation musicale Last.fm [2]) et universitaires (le système Phoaks [Terveen 1997]) utilisent les systèmes de recommandation. Plus de détails sur les systèmes de recommandation en termes de concepts, définitions et techniques sont disponibles dans [Adomavicius 2005, Perugini 2004]. Sur la base de la technique de recommandation utilisée (basés sur le contenu ou à filtrage collaboratif), les systèmes de recommandation peuvent être classés en deux catégories.

En utilisant une technique de filtrage basée sur le contenu, la recommandation est assurée en analysant la similarité entre des produits (ou services) et les intérêts d'un utilisateur (représentés par son profil). Les produits (ou services) les plus similaires au profil de l'utilisateur lui sont suggérés. La recommandation basée sur le contenu est similaire à un processus de classification. Dans une classification, des produits (ou services) similaires sont regroupés en fonction de leurs attributs communs, et un profil est créé pour représenter chaque groupe (classe). Par la suite, ces profils sont utilisés pour classer les nouveaux produits (ou services). Similairement, lors d'une recommandation basée sur le contenu, les profils des utilisateurs sont considérés comme des classes. Ils sont utilisés pour recommander un produit (ou service) et pour classer de nouveaux produits (ou services). La technique utilisée pour recommander un produit dépend essentiellement du type (textuel, numérique, etc.) et de la structure (structuré ou pas) de son contenu. Dans le cas de documents textes par exemple, le contenu d'un document est assimilé aux différents termes (mots) qui le forment.

Avec un système de recommandation utilisant une technique de filtrage collaboratif, les utilisateurs devront évaluer les produits (ou services) qu'ils connaissent ou qu'ils ont utilisés en leur donnant une note. Toutefois, un utilisateur n'est pas toujours motivé pour évaluer un produit. Pour cela, d'autres alternatives, qui se basent sur le suivi de l'utilisation du système de recommandation par les utilisa-

1. http ://www.amazon.com/
2. http ://www.lastfm.fr/

teurs, ont été proposées pour évaluer les produits d'un système de recommandation. Les produits les mieux évalués seront recommandés à des utilisateurs ayant des profils similaires aux profils des évaluateurs. L'idée à l'origine est que les utilisateurs ayant des profils similaires ont tendance à avoir les mêmes intérêts. La principale limitation d'une recommandation basée sur le filtrage collaboratif est que les utilisateurs ne recevront pas de recommandation tant qu'aucune évaluation n'est faite par un autre utilisateur. Plus de détails sur les approches de filtrage collaboratif peuvent être obtenus dans [Herlocker 2001, Herlocker 2004].

Dans notre travail, nous appliquons une approche de filtrage collaboratif pour améliorer le processus de découverte d'un service et plus précisément la découverte d'annuaires. L'objectif d'une approche de filtrage collaboratif est de trouver pour un *item* donné les « voisins » les plus proches en se basant sur des critères tels que les préférences utilisateurs, les scores, les données d'utilisation, ou les points d'intérêts. Le filtrage collaboratif *item-to-item* [Sarwar 2001] est une variante des approches de filtrage collaboratif utilisée efficacement pour trouver les *items* les plus pertinents pour un *item* donné. Supposons qu'un utilisateur est à l'écoute d'une chanson à partir d'une application intégrant un système de recommandation. Après avoir écouté une chanson, l'utilisateur peut être intéressé par une autre chanson ayant le même rythme. L'application, via son système de recommandation, peut lui suggérer des chansons ayant des rythmes similaires après l'analyse de l'onde sonore de la chanson en cours. Dans ce cas, un algorithme de filtrage collaboratif *item-to-item* peut être appliqué et le critère de recommandation dans ce cas est la similarité entre les ondes sonores. De nombreux algorithmes sont utilisés pour trouver les similitudes *item-to-item* tel celui utilisé efficacement par Amazon [Linden 2001].

De plus, nous adoptons une technique de recommandation implicite ne nécessitant aucune intervention de la part des utilisateurs de services afin de contourner les problèmes pouvant être observés avec les techniques explicites (voir Chapitre 2).

5.4.2 Recommandation d'annuaires dans une communauté

Nous proposons dans cette section, une technique de recommandation implicite pour mettre en œuvre la deuxième étape de notre approche de découverte d'annuaire (la sélection d'annuaires, voir section 5.3.2). Cette technique utilise une approche de filtrage collaboratif *item-to-item* pour recommander un ou plusieurs annuaire(s) à un demandeur de services. Nous utilisons la caractérisation d'un utilisateur et les caractérisations globales des annuaires comme critères pour cette recommandation. Avant d'introduire la formule que nous utilisons pour calculer le facteur de recommandation d'un annuaire (représenté par sa GRC) pour un demandeur de services (représenté par sa RC), nous donnons les définitions des termes utilisés dans cette formule :

- *Facteur de Recommandation* : nous définissons RF comme la fonction calculant le facteur de recommandation entre deux caractérisations. La recommandation est réalisée en fonctions de différents *facteurs de similarité*.
- *Facteur de Similarité* : nous définissons SF comme la fonction calculant le

facteur de similarité entre deux *ensembles de caractérisations*.
- *Ensemble de caractérisations* : représente les différentes caractéristiques extraites à partir d'une caractérisation. Nous avons identifié deux types : l'ensemble des *traces* désigné par T et l'ensemble des *besoins non fonctionnels* désigné par B_{nf} (voir Définition 5.2).
- *Le poids de SF* : nous associons à chaque fonction de calcul du *facteur de similarité SF* un poids reflétant son importance lors du calcul du facteur de recommandation. Nous définissons α comme poids associé à $SF(T_{GRC}, T_{RC})$ et β comme poids associé à $SF(B_{nf\,GRC}, B_{nf\,RC})$.

En utilisant ces différentes définitions, nous définissons le facteur de recommandation RF de la caractérisation globale GRC_i d'un annuaire i et la caractérisation RC d'un demandeur de services par :

$$RF(GRC_i/RC) = \alpha \times SF(T_i, T_{RC}) + \beta \times SF(B_{nf\,i}, B_{nf\,RC}) \tag{5.1}$$

Cette fonction de recommandation est utilisée pour calculer des facteurs de recommandations pour tous les annuaires d'une communauté (voir Algorithme 10). L'annuaire possédant le plus grand facteur de recommandation aura la « plus grande » probabilité de satisfaire la requête d'un demandeur de services. En effet, ce facteur est calculé sur la base des similarités respectives entre les *traces* et les *besoins non fonctionnels* extraits à partir de la RC d'un utilisateur et de la GRC d'un annuaire. Vu que la GRC d'un annuaire représente la moyenne des caractérisations d'utilisateurs ayant utilisé un service fourni par cet annuaire, une grande similarité entre RC et GRC indique une ressemblance dans les besoins d'un demandeur de services et ceux ayant déjà utilisé les services de cet annuaire.

Dans la suite de cette section, nous proposons deux fonctions pour le calcul des facteurs de similarité entre les *traces* et les *besoins non fonctionnels* d'une caractérisation utilisateur RC et une caractérisation globale d'un annuaire GRC (section 5.4.2.1 et 5.4.2.2). Nous présentons dans la section 5.4.3 un scénario illustrant le déroulement de notre technique de recommandations d'annuaires basée sur les caractérisations.

5.4.2.1 Facteur de similarité entre les ensembles de *trace*

Le modèle vectoriel VSM (de l'anglais *Vector Space Model*) introduit par Salton et al. [Salton 1975] est un modèle algébrique permettant la représentation d'objets sous forme vectorielle afin de pouvoir calculer la similarité entre eux. La similarité entre deux objets est alors déduite à partir de l'angle entre les vecteurs qui les représentent. Tous les objets sont représentés par des vecteurs dans le même espace k-dimensionnel et chaque composante d'un vecteur représente le poids de ce vecteur dans l'une des k dimensions. Une métrique de pondération largement utilisée pour le calcul de ces poids dans VSM est TF-IDF [Manning 2008] (de l'anglais *Term Frequency-Inverse Document Frequency*). Le principe de TF-IDF est de refléter l'importance d'un mot dans une collection de documents et en pratique, il est utilisé

pour évaluer l'importance d'une composante d'un vecteur dans l'une des dimensions d'un espace k-dimensionnel.

En recherche d'information, VSM est utilisé pour trouver les similitudes entre des documents dans un corpus. Chaque document est alors représenté comme un vecteur et à chaque mot du document est associé un poids. Ce poids représente l'importance du mot dans le corpus et il est généralement calculé en utilisant la pondération TF-IDF. Les données d'un corpus peuvent être représentées par une matrice d'occurrence $m \times n$ (en anglais *term-document matrix*) où m est le nombre de termes (ou mots) et n est le nombre de documents dans le corpus. La similarité entre deux documents est alors inférée par le cosinus de l'angle entre leurs deux vecteurs respectifs.

Nous nous sommes inspirés de cette méthode de calcul de similarité entre documents pour calculer la similarité entre les ensembles de *traces* extraites à partir d'une caractérisation RC et d'une caractérisation globale d'un annuaire GRC. Nous considérons un ensemble de traces T_j comme étant un « document » et chaque concept t_i annotant une opération dans un ensemble de traces comme étant un « mot » dans le document. Ainsi, les données de notre corpus peuvent être représentées par une matrice d'occurrence $A_{m \times n}$, où m est le nombre de concepts et n est le nombre des ensembles de traces T_i. Chaque entrée $A[i, j]$ dans cette matrice représente le nombre de fois que le concept t_i a été identifié dans l'ensemble de traces T_j.

Ainsi, nous utilisons VSM pour représenter les ensembles de traces par des vecteurs et nous appliquons TF-IDF pour calculer le poids de chaque concept t_i dans une trace T_j. La similarité entre les deux vecteurs est par la suite inférée en utilisant la fonction cosinus (voir formule (3.2)).

Le vecteur $WT_{grc_i} = \{w_{i,1}, w_{i,2}, ..., w_{i,n}\}$, représentant les poids de chaque concept dans l'ensemble de traces d'une caractérisation globale d'un annuaire GRC_i, est calculé en utilisant la formule (5.2) :

$$w_{i,j} = TF_{i,n} \times IDF_j = \frac{|T_{i,j}|}{|T_i|} \times \log \frac{|GRC|}{|t_j|} \tag{5.2}$$

Où :

- $T_i = \{(t_1, n_1), (t_2, n_2), \ldots, (t_k, n_k)\}$ est la *trace globale* d'une caractérisation globale GRC_i d'un annuaire r_i (voir Définition 5.2).
- $|T_{i,j}|$ est le nombre de fois où un concept t_j apparaît dans T_i.
- $|T_i|$ est le nombre d'occurrences totales des concepts t dans la *trace globale* T_i de la GRC_i.
- $|GRC|$ est le nombre total de caractérisations globales GRC (c.-à-d. le nombre total de *trace globale*).
- $|t_j|$ est le nombre de fois où le concept t_j a été identifié dans les ensembles T_i parmi toutes les caractérisations globales GRC_i.

Nous utilisons aussi la pondération TF-IDF (formule (5.2)) pour calculer le vecteur représentant la *trace* d'une caractérisation utilisateur RC. Nous rappelons que cette formule est basée sur les poids associés aux différents concepts d'une trace. Toutefois, la *trace* d'un demandeur de services est représentée uniquement par la

liste des concepts annotant les opérations utilisées. Vu que cette liste n'est pas pondérée (c.-à-d. aucune valeur n'est associée aux concepts comme c'est le cas pour les *traces globales*), nous utilisons le nombre d'occurrences d'un concept t dans les *traces globales* de toutes les GRC comme poids pour ce concept. Ainsi, la *trace* d'une caractérisation utilisateur RC est représenté par $T_i = \{(t_1, n_1), (t_2, n_2), ..., (t_k, n_k)\}$ où :

- t_i est un concept sémantique représentant une opération dans la *trace*.
- n_i le nombre d'occurrences d'un concept t dans les *traces globales* de toutes les GRC.

En utilisant les vecteurs WT_{rc} et WT_{grc} représentant respectivement la *trace* d'une caractérisation utilisateur RC et la *trace globale* d'une caractérisation globale d'un annuaire GRC, le facteur de similarité entre les ensembles de *trace* de RC et de GRC (formule (5.3)) est calculé en utilisant la fonction cosinus :

$$SF(T_{rc}, T_{grc}) = cosine(WT_{rc}, WT_{grc}) = \frac{WT_{rc} \times WT_{grc}}{\|WT_{rc}\| \times \|WT_{grc}\|} \tag{5.3}$$

5.4.2.2 Facteur de similarité entre les ensembles de *besoins non fonctionnels*

Les valeurs associées aux besoins non fonctionnels d'une RC ou GRC peuvent avoir des significations différentes (par exemple : temps en ms, un prix, etc.). A cause de l'hétérogénéité des poids des différents concepts b_i d'un ensemble de *besoins non fonctionnels* B_{nf} (voir Définition 5.1 et 5.2), nous ne pouvons pas utiliser la notion d'occurrence de concepts pour calculer la pondération TF-IDF. Ainsi, l'utilisation du modèle vectoriel VSM avec une pondération TF-IDF n'est pas une solution adéquate pour représenter les ensembles de *besoins non fonctionnels* sous forme vectorielle.

Nous proposons donc une fonction différente pour le calcul du facteur de similarité entre deux ensembles de *besoins non fonctionnels* d'une RC et d'une GRC. Avant d'introduire cette fonction, nous donnons les définitions des termes utilisés dans cette fonction :

- *L'ensemble des besoins non fonctionnels* : un ensemble de couples (b_i, v_i) dénoté par B_{nf} (voir Définition 5.1 et 5.2).
- *L'ensemble commun de besoins non fonctionnels* : Cet ensemble est défini par C=$\{(c_1, v_{rc1}, v_{grc_1}), (c_2, v_{rc2}, v_{grc_2}), ..., (c_n, v_{rcn}, v_{grc_n})\}$ où :
  - c_i est un concept commun entre $B_{nf_{rc}}$ et $B_{nf_{grc}}$.
  - v_{rci} (resp. v_{grc_i}) est la valeur associée au concept c_i dans $B_{nf_{rc}}$ (resp. $B_{nf_{grc}}$).

Pour calculer le facteur de similarité SF, nous supposons que les différents besoins non-fonctionnels peuvent avoir des importances différentes pour un demandeur de services (par exemple, pour un certain demandeur de services le délai de livraison peut être plus important que le prix du service). Nous définissons alors des poids p_i pour pondérer et différencier ces différents besoins et nous utilisons la formule (5.4) pour calculer SF :

$$SF = \sum_{i=1}^{n} p_i \times \frac{\min(v_{rci}, v_{grc_i})}{\sup(v_{rci}, v_{grc_i})} \tag{5.4}$$

Où p_i est le poids associé à un besoin non-fonctionnel c_i et $\sum_{i=1}^{n} p_i = 1$.

5.4.3 Exemple

Nous illustrons maintenant le déroulement du processus de recommandation d'un annuaire à travers un scénario pratique. Nous prenons l'exemple simplifié d'une entreprise d'assemblage d'ordinateurs. Afin de garantir sa chaîne de production, l'entreprise doit prendre contact avec d'autres entreprises pour se procurer les composants nécessaires. Pour cet exemple, nous supposons que l'entreprise a besoin d'un processeur, d'un système d'exploitation et d'un carton d'emballage pour finir l'assemblage d'un ordinateur commandé.

Nous supposons que cette entreprise a déjà utilisé deux services pour l'achat d'un processeur et d'un système d'exploitation pour l'ordinateur qu'elle est en train d'assembler. Les deux opérations utilisées, à savoir `buyHardComponent` et `buySoftComponent` (référencées respectivement par les concepts *HwOrdering* et *SwOrdering*), sont ajoutées à la *trace* de la caractérisation *RC* de l'entreprise. La caractérisation de cette entreprise est illustrée dans la Figure 5.5.

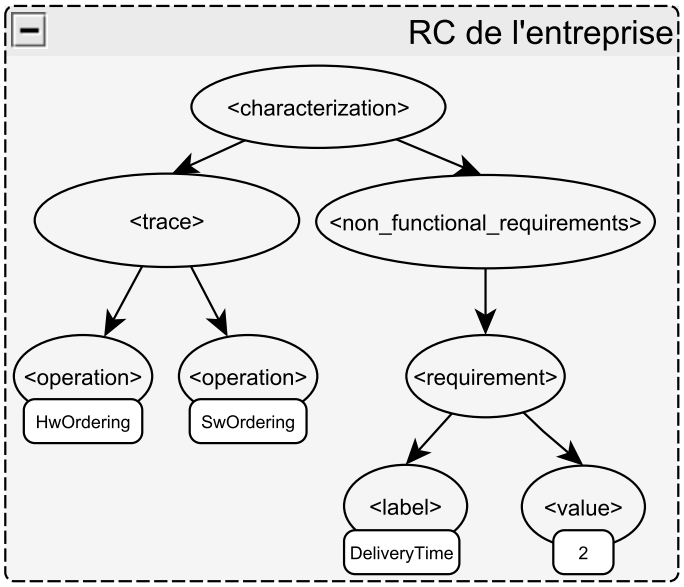


FIGURE 5.5 – La caractérisation *RC* de l'entreprise

A ce stade, l'entreprise souhaite trouver un service lui permettant d'acheter le carton d'emballage. La première étape dans notre approche de découverte d'an-

nuaires est la sélection de la communauté la plus adéquate aux besoins de l'entreprise. En utilisant notre méthode de sélection de communauté introduite dans la section 5.3.1, la requête de l'entreprise est dirigée vers une communauté d'annuaires appelée *Packaging community*. Nous supposons que cette communauté est formée de 100 annuaires. Afin de réduire l'espace de recherche de la requête de l'entreprise, nous utilisons notre technique de recommandation pour sélectionner l'annuaire le plus adéquat à la requête. Cette recommandation est basée sur la caractérisation RC de l'entreprise et des 100 caractérisations globales GRC des annuaires de la communauté sélectionnée.

Pour simplifier la présentation de l'exemple, nous appliquons notre technique de recommandation introduite dans la section 5.4 sur deux annuaires seulement. Leurs caractérisations globales sont représentées dans la Figure 5.6. Pour calculer les facteurs de recommandation entre la caractérisation RC de l'entreprise et les caractérisations globales GRC_1 et GRC_2, nous commençons par calculer le facteur de similarité entre les *traces* de RC et GRC_1. A partir de ces deux caractérisations nous extrayons :

– la *trace globale* de la caractérisation globale GRC_1 :

$$T_{grc_1} = \{(SwOrdering, 150), (PackOrdering, 10)\}$$

– la *trace* de la caractérisation RC en associant chaque concept de la trace à son nombre d'occurrences dans les *traces globales* des 100 GRC de la communauté :

$$T_{rc} = \{(HwOrdering, 430), (SwOrdering, 540), (PackOrdering, 330)\}$$

Parmi les 100 GRC de la *Packaging community*, nous supposons que l'opération *HwOrdering* apparaît dans la *trace* de 30 d'entre eux, *SwOrdering* dans 20 et *PackOrdering* dans 15. En appliquant la formule (5.2), nous obtenons WT_{grc_1} et WT_{rc} les vecteurs pondérés de l'ensemble de trace de GRC_1 et RC :

$$WT_{grc_1} = (0, 0.65, 0.05) \text{ et } WT_{rc} = (0.17, 0.29, 0.20)$$

En appliquant la formule (5.3), nous obtenons le facteur de similarité entre la *trace* de GRC_1 et RC :

$$\begin{aligned} SF(T_{rc}, T_{grc_1}) &= cosine(WT_{rc}, WT_{grc_1}) \\ &= \frac{WT_{rc} \times WT_{grc_1}}{\|WT_{rc}\| \times \|WT_{grc_1}\|} \\ &\approx 0.78 \end{aligned}$$

Pour établir le facteur de similarité entre les ensembles des besoins non fonctionnels de GRC_1 et RC nous utilisons la formule (5.4) :

$$SF(B_{nf_{rc}}, B_{nf_{grc1}}) = \frac{\frac{2}{4}}{1} = 0.5$$

De la même façon, nous obtenons les facteurs de similarité entre les ensembles des caractérisations de GRC_2 et RC suivants :

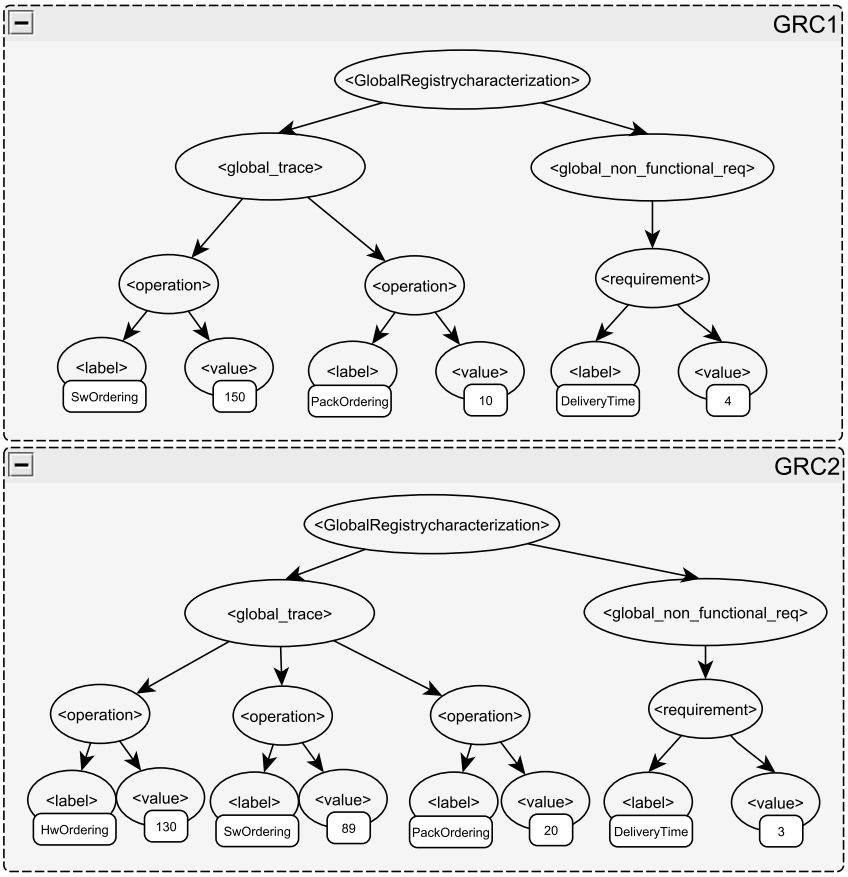


FIGURE 5.6 – Les caractérisations globales GRC_1 et GRC_2

$$SF(T_{rc}, T_{grc_2}) \approx 0.91 \text{ et } SF(B_{nf_{rc}}, B_{nf_{grc2}}) \approx 0.66$$

Selon cet exemple et en supposant que $(\alpha = 0.8 > \beta = 0.2)$ [3], nous obtenons les facteurs de recommandation suivants en utilisant la formule (5.1) :

$$RF(GRC_1/RC) = 0.8 \times 0.78 + 0.2 \times 0.5 \approx 0.724$$
$$RF(GRC_2/RC) = 0.8 \times 0.91 + 0.2 \times 0.66 \approx 0.86$$

Sur la base de ces résultats, l'annuaire possédant la caractérisation globale GRC_2 est recommandé à l'entreprise avant celui possédant la caractérisation globale GRC_1.

3. Nous considérons qu'une similarité entre les traces est plus importante qu'une similarité entre les besoins non fonctionnels.

5.4.4 Synthèse

Dans cette section, nous avons présenté notre approche de recommandation utilisée pour la découverte d'annuaires. La recommandation que nous fournissons est basée sur la caractérisation d'un demandeur de services et les caractérisations globales des annuaires. La caractérisation d'un demandeur de services représente la liste des services qu'il a déjà utilisés et ses besoins non-fonctionnels. De même, la caractérisation globale d'un annuaire représente la « moyenne » des traces et des besoins non fonctionnels des utilisateurs ayant utilisé l'un des services de cet annuaire. Ainsi, notre technique tire **implicitement** profit des expériences d'utilisateurs passées sans interagir avec les demandeurs de services et sans demander explicitement des informations d'évaluation.

5.5 Conclusion

Dans ce chapitre, nous avons assuré le troisième objectif de notre thèse en proposant une approche pour la découverte d'annuaires dans un réseau d'annuaires organisés en communautés. L'approche que nous avons proposée est formée de deux étapes : la sélection de communauté et la sélection d'annuaires.

L'approche de sélection de communauté profite de l'organisation des annuaires en communautés selon leurs fonctionnalités. En effet, nous avons utilisé le besoin fonctionnel SD d'un demandeur de services exprimé dans sa requête de découverte pour trouver la communauté adéquate. Vu l'homogénéité entre offre (ensemble de communautés décrites par leurs fonctionnalités) et demande (requête décrivant un besoin fonctionnel), la sélection de la communauté adéquate à une requête a simplement été réalisée en calculant la similarité entre la requête et les fonctionnalités des communautés.

Pour la sélection d'un (ou plusieurs) annuaire(s) à partir d'une communauté sélectionnée, nous avons utilisé une technique de recommandation. L'approche de recommandation que nous avons proposée est basée sur la caractérisation d'un demandeur de services et les caractérisations globales des annuaires.

Cette technique a permis de réaliser une sélection d'annuaires personnalisée (c.-à-d. selon la caractérisation du demandeur de services) et implicite (c.-à-d. ne nécessitant pas d'efforts supplémentaires de la part du demandeur de services). C'est vrai que, contrairement à la réalisation d'un appariement sémantique entre la requête d'un demandeur de services et toutes les descriptions de services offertes par les annuaires d'une communautés, en utilisant une technique de recommandation certains annuaires peuvent être écartés du processus de découverte. Toutefois, comme nous l'avons précisé au début de ce mémoire : « Dans un environnement multi-annuaires, nous sommes presque sûrs d'obtenir des réponses pour une requête de découverte de services mais ce que nous souhaitons c'est de les obtenir rapidement ». Les résultats obtenus dans ce chapitre ont été validés par trois publications [Sellami 2009, Sellami 2010c, Wives 2010].

L'approche que nous avons proposée dans ce chapitre permet d'assurer la pre-

mière étape de la découverte de services dans un contexte multi-annuaires. La deuxième étape, à savoir la sélection de services, n'a pas été abordée dans cette thèse mais a été étudiée au sein de notre équipe [Chabeb 2010, Nguyen 2010, Nguyen 2011]. Pour sélectionner un service parmi le ou les annuaire(s) recommandé(s), un appariement sémantique entre la requête de découverte de services SD et l'ensemble des descriptions de services annoncées par les annuaires peut être réalisé [Chabeb 2010]. Nous pouvons aussi considérer la technique de recommandation à filtrage collaboratif (basée sur le comportement d'un demandeur de services) introduite dans [Nguyen 2010, Nguyen 2011] pour la sélection d'un service parmi le ou les annuaires recommandés.

Chapitre 6

Mise en œuvre

Sommaire

6.1 Introduction

Dans le Chapitre 5, nous avons présenté notre approche pour la découverte de services dans un environnement d'annuaires organisés en communautés. Dans ce chapitre, nous nous intéressons à la mise en œuvre des différentes contributions présentées dans les chapitres précédents. Pour cela, nous commençons par mettre en place un « banc d'essai » formé par un réseau d'annuaires organisés en communautés selon notre approche introduite dans le section 3.4. Ce banc d'essai est utilisé pour expérimenter un système de découverte de services implémentant notre approche. Nous proposons aussi un gestionnaire de communautés d'annuaires mettant en œuvre les mécanismes de gestion que nous avons proposés dans le chapitre 4.

Le reste de ce chapitre est organisé comme suit. Dans la section 6.2, nous exposons les trois étapes réalisées pour la préparation de notre « banc d'essai ». Nous commençons par la génération d'une collection de descriptions de services et sa publication sur plusieurs annuaires. Nous présentons par la suite l'outil `WSRDCreator` utilisé pour générer les descriptions WSRD d'annuaires. Les descriptions WSRD

sont finalement passées comme paramètres d'entrée à l'outil `CommunityBuilder` permettant d'organiser nos annuaires en communautés. Dans la section 6.3, nous exposons le travail effectué pour mettre en œuvre notre approche de découverte d'annuaires introduite dans le chapitre 5. Pour cela, nous commençons par présenter une architecture de système de découverte d'annuaires. Nous déployons par la suite notre « banc d'essai » au-dessus d'un réseau P2P et nous présentons les expérimentations effectuées sur un système de découverte mis en œuvre selon notre architecture proposée. Enfin, nous présentons dans la section 6.4 l'outil `CommunityManager` permettant de simuler graphiquement un réseau de communautés d'annuaires. De plus, cet outil implémente notre approche de gestion de communautés.

6.2 Préparation du « banc d'essai »

Le « banc d'essai » pour nos expérimentations est formé d'un réseau d'annuaires organisés en communautés (voir Figure 6.1). Pour le préparer, nous commençons par générer une collection de descriptions de services et nous les publions dans plusieurs annuaires (étape (1) dans Figure 6.1). Par la suite, nous calculons pour chacun de ces annuaires sa description WSRD (étape (2)). Ces descriptions WSRD sont par la suite utilisées pour organiser cet ensemble ou réseau d'annuaires en communautés (étape (3)). Dans la suite de cette section, nous détaillons les différentes étapes de la préparation de notre « banc d'essai ».

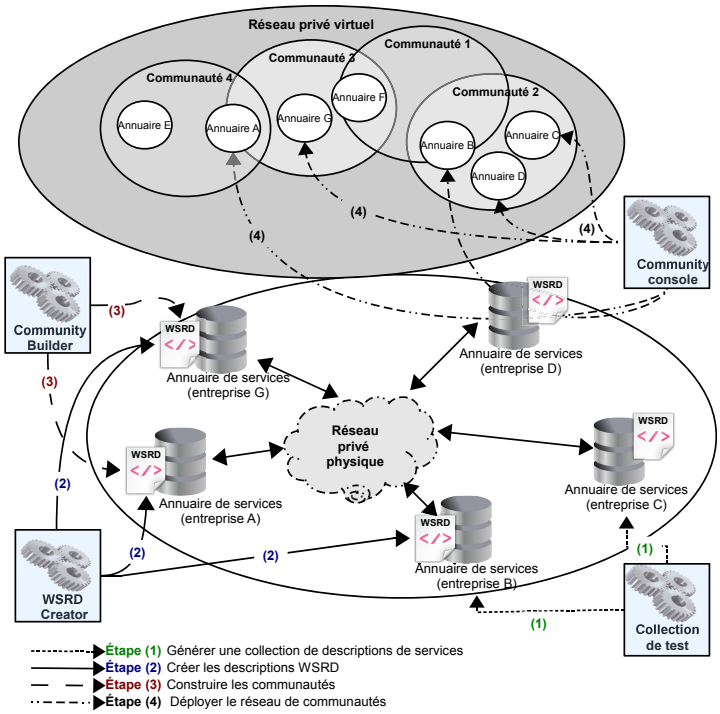


FIGURE 6.1 – Aperçu sur notre « banc d'essai »

6.2.1 Génération d'une collection de descriptions de services

Afin de générer la collection de descriptions de services que nous allons utiliser pour nos expérimentations, nous adoptons un générateur de descriptions de services développé au sein de notre équipe et introduit dans [Chabeb 2010]. Ce générateur permet la production de collections de descriptions de services sémantiques YASA [Chabeb 2008]. Afin de fournir un corpus de descriptions qui soit le plus « réaliste » possible, les auteurs du générateur se sont basés sur un autre générateur de descriptions de services introduit dans [Oh 2006]. Ce dernier a la particularité de générer des descriptions de services en se basant sur une collection de services réels.

Nous avons donc repris le générateur de [Chabeb 2010] et nous l'avons adapté pour générer une collection de descriptions de services sémantiques en SAWSDL au lieu des descriptions en YASA générées initialement par l'outil. L'arborescence de la collection générée est illustrée dans la Figure 6.2. Ces descriptions sont annotées en utilisant une ontologie de domaine décrite en OWL. Pour nos expérimentations, nous avons généré une collection de 1400 descriptions en SAWSDL. Nous avons aussi généré sept descriptions abstraites de services $SD_i, i = 1 \dots 7$, également en SAWSDL qui représentent le besoin fonctionnel de la requête Q d'un demandeur de services telle que nous l'avons définie dans la section 5.2. Ces descriptions abstraites sont sauvegardées dans le dossier `queries` (voir Figure 6.2). Parmi les 1400 descriptions produites par notre générateur, 700 sont réparties en sept ensembles de 100 descriptions (sauvegardées dans sept dossier `SDi`, $i = 1 \dots 7$) où chaque ensemble regroupe les descriptions pertinentes à une requête de découverte de services SD_i. Ces ensembles représentent les réponses aux requêtes SD_i et sont appelés « *relevance sets* ». Les autres 700 descriptions, qui forment notre collection, sont produites aléatoirement (c.-à-d. générées indépendamment des requêtes et n'appartiennent à aucun des *relevance sets*) et sauvegardées dans le dossier `services`.

Par la suite, nous partageons les 1400 descriptions SAWSDL générées en sept groupes et nous les publions dans sept annuaires $R_i, i = 1 \dots 7$ [1] selon l'organisation présentée dans la Figure 6.3. Chaque annuaire R_i contiendra les 100 descriptions du dossier `SDi` (le *relevance set* de la requête SD_i) et 100 descriptions sélectionnées au hasard à partir du dossier `services` (la collection de services générés aléatoirement). Les annuaires sont déployés en utilisant *jUDDI*[2], l'implémentation *open source* d'UDDI.

6.2.2 Création des descriptions WSRD

Afin d'organiser les annuaires que nous avons construits en communautés, nous calculons pour chaque annuaire sa description WSRD selon notre approche introduite dans la section 3.2. Pour cela, nous avons développé l'API `WSRDCreator` [3] pour la création de descriptions WSRD à partir d'un ensemble de descriptions de services

1. Dans cet exemple nous utilisons uniquement sept annuaires dans un soucis de lisibilité.
2. http ://ws.apache.org/juddi/
3. http ://www-inf.it-sudparis.eu/SIMBAD/tools/WSRDGen/

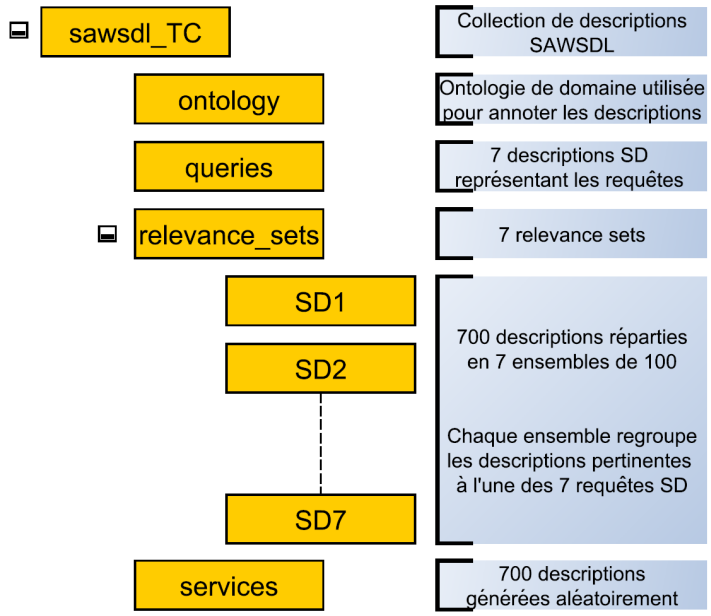


FIGURE 6.2 – Arborescence de la collection de descriptions de services générée

SAWSDL. `WSRDCreator` implémente les trois étapes de construction de WSRD et permet de générer des descriptions WSRD en utilisant nos deux méthodes de réduction (faible ou forte).

La Figure 6.4(a) est une capture d'écran de notre outil lors de la création de la description WSRD de l'annuaire R_1 en utilisant une réduction faible. Notre outil permet aussi de générer la représentation vectorielle (voir Figure 6.4(c)) de la description crée (voir Figure 6.4(b)) selon le modèle vectoriel que nous avons défini dans la section 3.4.2. Nous rappelons que nous transformons les descriptions WSRD en vecteurs qui serviront d'*input* pour notre approche de construction de communautés d'annuaires. Ce vecteur représente les fonctionnalités $R_1.f$ offertes par l'annuaire R_1 comme défini dans la section 3.3.3. Nous créons de la même façon les descriptions WSRD pour les autres annuaires et nous récupérons les représentations vectorielles de leurs descriptions WSRD [4].

Lors de nos expérimentations, nous nous sommes aussi intéressés au temps d'exécution du processus de création d'une description WSRD. Pour cela, nous avons créé une collection plus grande de descriptions SAWSDL (3000 descriptions). Nous avons créé des descriptions WSRD en utilisant nos deux techniques de réduction et nous avons progressivement augmenté la taille de la collection de 100 descriptions à 3000. Pour chaque description WSRD générée, nous considérons le temps d'exécution comme indiqué dans la Figure 6.5. Nous remarquons que la courbe représentant le temps d'exécution en utilisant une réduction faible est très proche de celle repré-

4. `WSRDCreator` sauvegarde les descriptions WSRD ainsi que leurs représentations vectorielles sous forme d'un fichier XML.

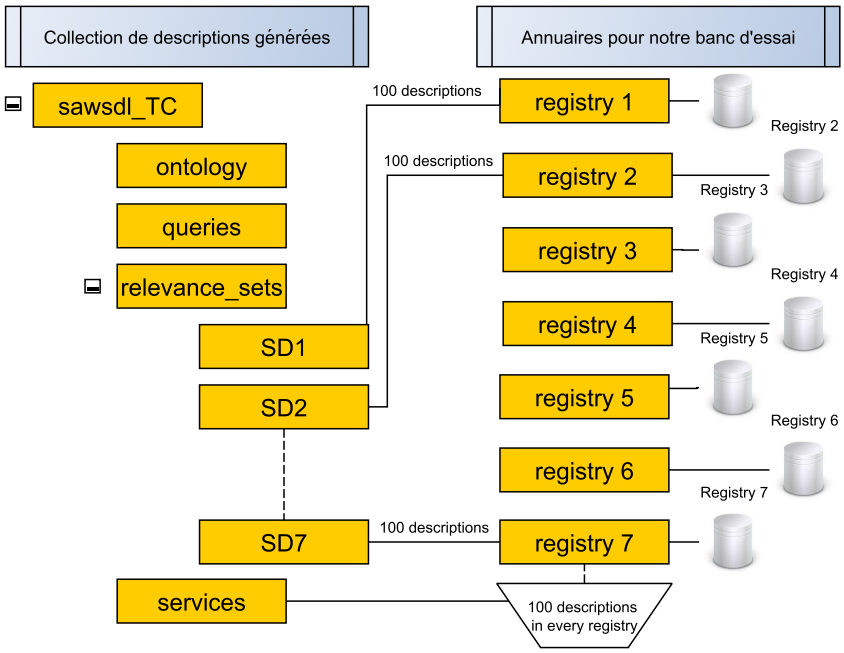


FIGURE 6.3 – Organisation de la collection générée en sept annuaires

sentant le temps d'exécution en utilisant une réduction forte. Vu que la charge de calcul est presque la même, nous recommandons l'utilisation d'une réduction faible pour la création d'une WSRD. De plus, nos expérimentations montrent que notre approche de création des descriptions WSRD est utilisable dans des situations réalistes. Par exemple, la création de la description WSRD d'un annuaire contenant 1400 descriptions de services se fait en 650 ms.

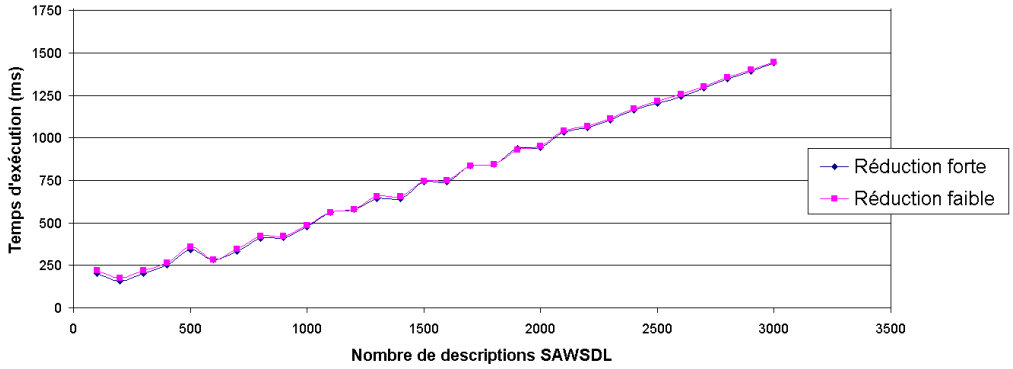
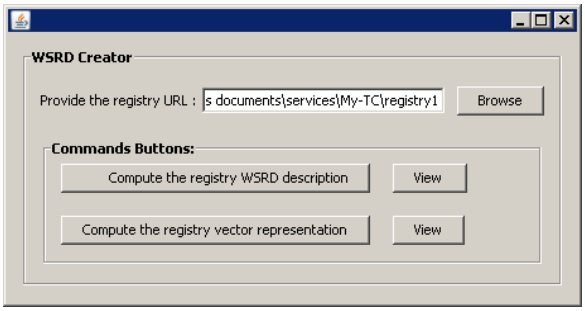
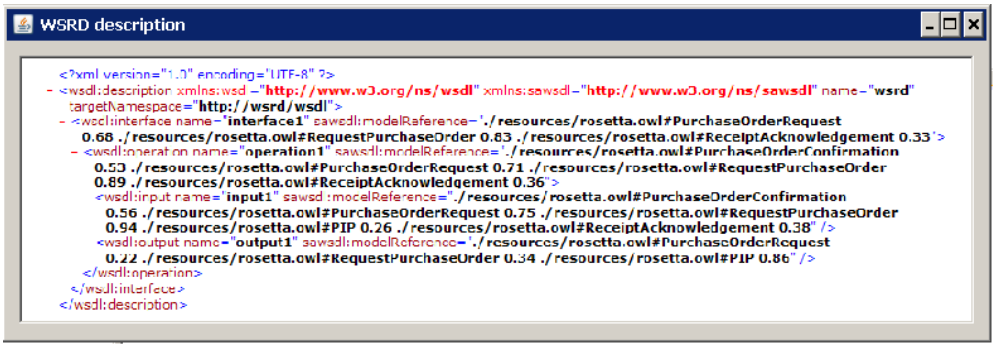


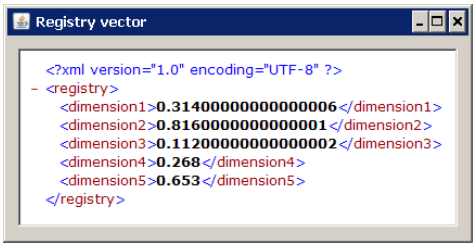
FIGURE 6.5 – Temps d'exécution pour le calcul des descriptions WSRD



(a) Création de la description WSRD de l'annuaire R_1



(b) Description WSRD de l'annuaire R_1



(c) Représentation vectorielle de la description WSRD

FIGURE 6.4 – Interface graphique de `WSRDCreator`

6.2.3 Construction des communautés

Dans cette dernière étape de la préparation de notre « banc d'essai », nous utilisons les descriptions WSRD générées à l'étape précédente pour organiser notre réseau d'annuaires en communautés. Pour cela, nous utilisons `CommunityBuilder` ; l'outil que nous avons développé pour la construction de communautés d'annuaires selon notre approche de clustering floue présentée dans la section 3.4. En utilisant `CommunityBuilder` (voir Figure 6.6), nous avons construit quatre communautés pour les sept annuaires que nous avons créés.

Les entrées de notre outil sont les représentations vectorielles f des différents annuaires à organiser et le nombre de communautés souhaité. Les centres de ces communautés $c.f$, représentant la fonctionnalité moyenne f de chaque commu-

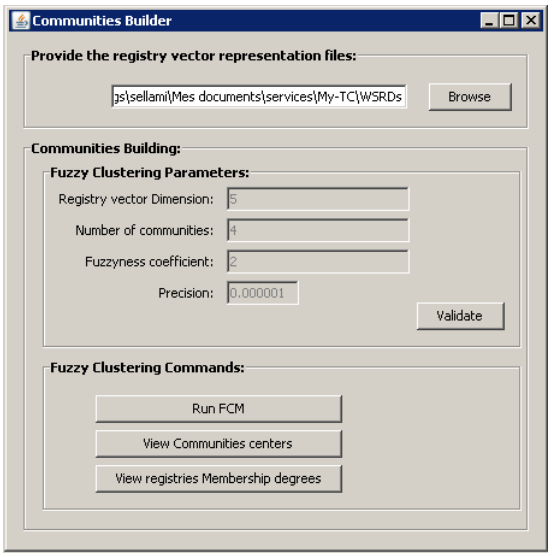
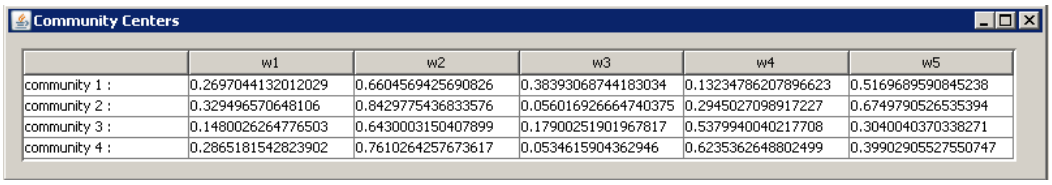


FIGURE 6.6 – Interface graphique de `CommunityBuilder`

nauté, sont représentés dans la Figure 6.7(a). Ils sont représentés par des vecteurs 5-dimensionnel où chaque dimension correspond à un concept dans notre ontologie. Dans la Figure 6.7(b), nous présentons les degrés d'appartenance MEM de chacun des annuaires R_m aux quatre communautés construites. De plus, `CommunityBuilder` génère quatre descripteurs de communauté et sept descripteurs d'annuaire sous forme de fichiers XML. Un descripteur de communauté contient l'identifiant de la communauté et sa fonctionnalité moyenne. Un descripteur d'annuaire contient l'identifiant de l'annuaire, sa fonctionnalité moyenne et l'ensemble de ses degrés d'appartenance aux différentes communautés. Nous fournissons des exemples d'un descripteur de communauté et d'un descripteur d'annuaire respectivement dans les Annexe A.1 et A.2.

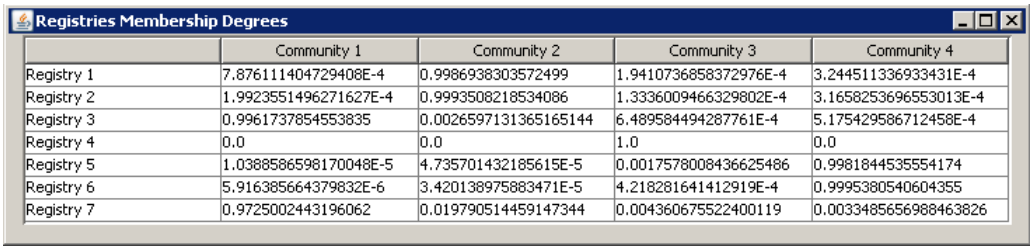
Dans l'exemple précédent, nous avons utilisé uniquement sept annuaires dans un soucis de lisibilité. Pour tester la faisabilité notre approche de construction de communautés, nous avons poussé plus loin nos expériences en testant notre `CommunityBuilder` avec un plus grand nombre d'annuaires. Nous avons utilisé un ensemble de 100 descriptions WSRD et nous les avons organisées en cinq communautés. L'organisation résultant de cette expérimentation est illustrée dans la Figure 6.8. Pour cette expérimentation, les vecteurs représentant les différentes descriptions WSRD sont 6-dimensionnels vu que nous avons utilisé une ontologie avec 6 concepts. Puisque que nous ne pouvons visualiser les résultats que sur des plans à 2 ou 3 dimensions, nos résultats sont illustrés par deux plans 3-dimensionnel : la Figure 6.8(a) illustre l'organisation des 100 annuaires en utilisant les 3 premières dimensions et la Figure 6.8(b) en utilisant les 3 dimensions suivantes.



Community Centers

	w1	w2	w3	w4	w5
community 1 :	0.2697044132012029	0.6604569425690826	0.38393068744183034	0.13234786207896623	0.5169689590845238
community 2 :	0.329496570648106	0.8429775436833576	0.056016926664740375	0.2945027098917227	0.6749790526535394
community 3 :	0.1480026264776503	0.6430003150407899	0.17900251901967817	0.5379940040217708	0.3040040370338271
community 4 :	0.2865181542823902	0.761026425767361 7	0.0534615904362946	0.6235362648802499	0.39902905527550747

(a) Représentation vectorielle des centres des quatre communautés construites



Registries Membership Degrees

	Community 1	Community 2	Community 3	Community 4
Registry 1	7.876111404729408E-4	0.9986938303572499	1.9410736858372976E-4	3.244511336933431E-4
Registry 2	1.9923551496271627E-4	0.9993508218534086	1.3336009466329802E-4	3.1658253696553013E-4
Registry 3	0.9961737854553835	0.0026597131365165144	6.489584494287761E-4	5.175429586712458E-4
Registry 4	0.0	0.0	1.0	0.0
Registry 5	1.0388586598170048E-5	4.735701432185615E-5	0.0017578008436625486	0.9981844535554174
Registry 6	5.916385664379832E-6	3.420138975883471E-5	4.218281641412919E-4	0.9995380540604355
Registry 7	0.9725002443196062	0.019790514459147344	0.004360675522400119	0.0033485656988463826

(b) Les degrés d'appartenance des annuaires aux quatre communautés construites

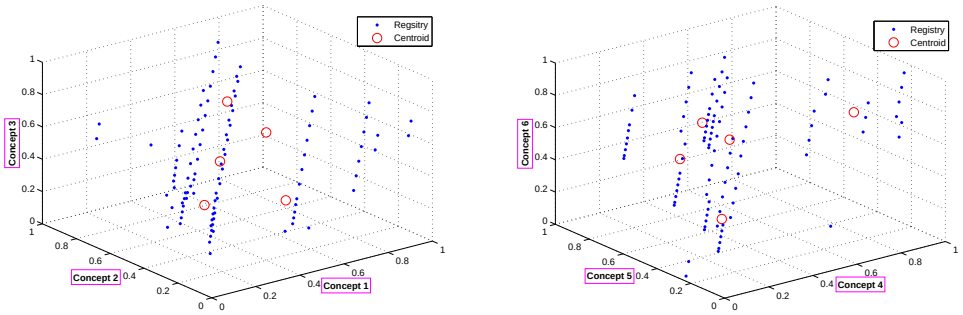
FIGURE 6.7 – Résultats de `CommunityBuilder`

6.2.4 Synthèse

Dans cette section nous avons préparé les différentes entrées nécessaires au déploiement d'un réseau d'annuaires organisés en communautés. Tout d'abord, nous avons généré une collection de 1400 descriptions de services que nous avons publiées dans sept annuaires UDDI implémentés en utilisant *jUDDI*. Par la suite, nous avons développé un outil, `WSRDCreator`, pour calculer les descriptions WSRD de ces sept annuaires. Cette expérimentation a montré que notre approche pour la création de descriptions WSRD est utilisable dans des situations réalistes. Enfin, nous avons développé `CommunityBuilder` pour organiser notre réseau d'annuaires en communautés. Cet outil a été implémenté conformément à notre approche de création de communautés introduite dans la section 3.4 et fournit comme sortie des descripteurs de communautés et d'annuaires décrivant l'organisation en communautés du réseau d'annuaires.

6.3 Découverte d'annuaires dans un réseau de communautés

Dans cette section, nous présentons le travail effectué pour mettre en œuvre notre approche de découverte d'annuaires introduite dans le chapitre 5. Pour cela, nous commençons par définir une architecture de système de découverte d'annuaires dans la section 6.3.1. Par la suite, nous présentons un système de découverte mis en œuvre selon cette architecture dans la section 6.3.2 et des métriques d'évaluation dans la section 6.3.3.



(a) Les axes XYZ représentent les poids w_1,w_2 and w_3 du vecteur d'une description WSRD

(b) Les axes XYZ représentent les poids w_4,w_5 and w_6 du vecteur d'une description WSRD

FIGURE 6.8 – Organisation de 100 annuaires en 5 communautés.

6.3.1 Architecture du système de découverte

En nous basant sur notre approche de découverte de services dans un environnement de communautés d'annuaires, nous avons conçu une architecture pour la mise en place d'un système de découverte d'annuaires. En raison de la dynamicité de notre réseau d'annuaires (un annuaire peut rejoindre/quitter une communauté à tout moment), un réseau P2P est bien adapté pour le mettre en place. De plus, un réseau P2P garantit le passage à l'échelle de notre système.

L'architecture proposée est formée par deux couches : la *Presentation & Trading layer* et la *Service Providing layer* (voir Figure 6.9). Dans la suite, nous présentons ces deux couches et nous montrons comment une requête de découverte de services est traitée.

6.3.1.1 La *Presentation & Trading layer*

La *Presentation & Trading layer* représente le point d'accès pour les demandeurs de services. Elle aide les demandeurs de services à exprimer leurs requêtes et besoins. Cette couche est constituée de plusieurs pairs *Presentation & Trading (P&T)*, fonctionnellement identiques, afin d'assurer la disponibilité du système et éviter un point unique de défaillance.

Quand un demandeur de services se connecte au système de découverte, la découverte est assistée par un de ces pairs *P&T*. Pour limiter l'espace de recherche, le pair *P&T* recommande à un demandeur de services l'annuaire le plus « apte » à répondre à sa requête.

Cela se fait en deux étapes :

1. Tout d'abord, le pair *P&T* route la requête d'un demandeur de services vers la communauté d'annuaires la plus adéquate sur la base de ses besoins fonctionnels (voir section 5.3.1 : Sélection de communauté). Cette étape est assurée par le composant *Community Selection* du pair *P&T*.

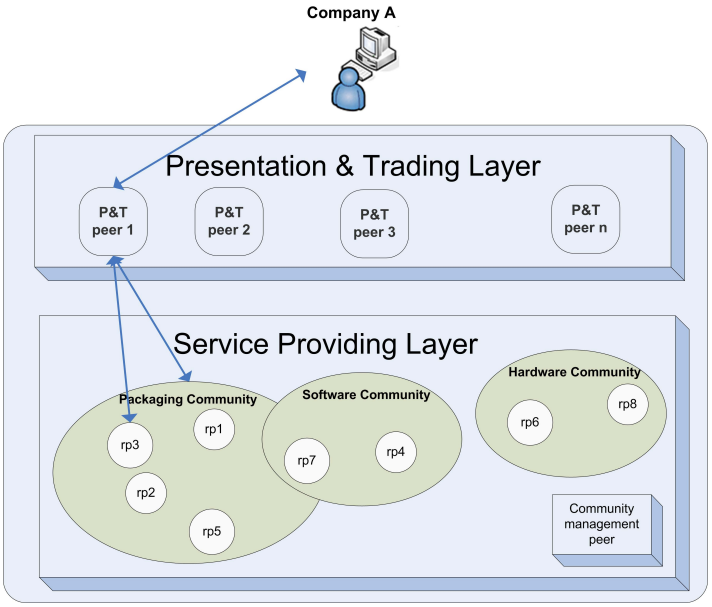


FIGURE 6.9 – Architecture du système de découverte de services

2. Par la suite, le pair *P&T* recommande au demandeur de services un ou plusieurs annuaire(s) qui ont la plus grande « probabilité » de satisfaire sa requête (voir section 5.3.2 : Sélection d'annuaires). Cette étape est assurée par le composant *Recommender System* d'un pair *P&T* qui implémente notre technique de recommandation proposée dans la section 5.4.

En résumé, les différents composants constituant un pair *P&T* (voir Figure 6.10) sont :

- *Community selection.* Identifie la communauté la plus adéquate pour la requête du demandeur de services. Ceci est fait sur la base des similarités entre le besoin fonctionnel SD, exprimé dans la requête Q d'un demandeur de services, et les différentes fonctionnalités moyennes f des communautés existantes. La communauté possédant le degré de similarité le plus élevé sera sélectionnée.
- *Recommender system.* Guide le demandeur de services vers l'annuaire le plus approprié à ses besoins. La recommandation est réalisée sur la base de la caractérisation utilisateur RC et la liste des caractérisations des demandeurs de services précédents.
- *Data.* Contient des copies des ontologies utilisées et une liste des communautés existantes.
- *GUI.* L'interface graphique proposée aux demandeurs de services pour utiliser et interagir avec le système de découverte de services.
- *Watchdog.* Surveille les interactions du demandeur de services avec le système de découverte et met à jour sa caractérisation.

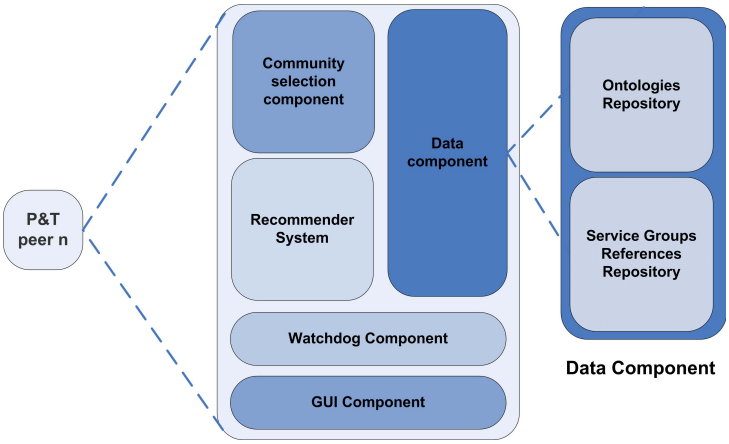


FIGURE 6.10 – Structure d'un *Presentation & Trading* pair

6.3.1.2 La *Service Providing layer*

Cette couche contient un réseau P2P de *registry peers*. Chaque *registry peer* représente un proxy (ou mandataire) d'un annuaire. Ces *registry peers* suivent la même organisation en communautés des annuaires qu'ils mandatent. Cette organisation est gérée et mise à jour selon les mécanismes de gestion que nous avons définis dans le chapitre 4 par le composant *community management* de la *Service Providing layer*.

6.3.2 Mise en œuvre de l'architecture proposée

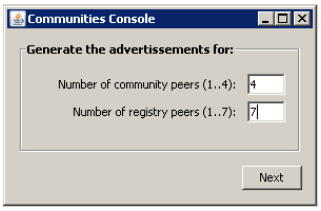
Nous présentons dans cette section la mise en œuvre de l'architecture proposée. Nous commençons par mettre en place la *service providing layer* en déployant notre « banc d'essai » en utilisant la plate-forme JXTA (section 6.3.2.1). Nous implémentons par la suite un *trading peer* de la *Presentation & Trading layer* qui servira de point d'accès aux demandeurs de services à notre système de découverte de services (section 6.3.2.2).

6.3.2.1 Déploiement de notre « banc d'essai »

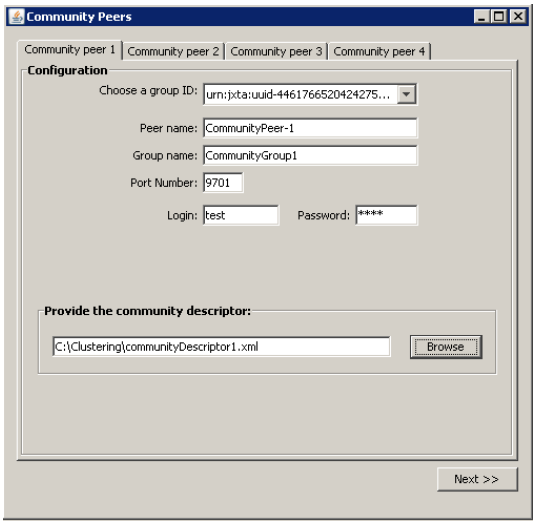
La plate-forme JXTA de Sun Microsystems's [Traversat 2003] est adaptée pour déployer notre « banc d'essai ». En effet, une communauté d'annuaires peut alors être considérée comme un *Peer Group*[5], où chaque annuaire est mandaté par un pair JXTA. Pour simuler un réseau d'annuaires organisés en communautés (la *service providing layer*), nous implémentons plusieurs *Peer Groups*, c.-à-d. des communautés d'annuaires, formées par plusieurs pairs JXTA, c.-à-d. des *registry peers*. Ces pairs JXTA formeront un réseau virtuel de communautés d'annuaires au-dessus du

5. *Peer Group* est l'un des concepts offerts par JXTA et est défini comme étant un ensemble de pairs offrant un service spécifique

réseau physique formé par les annuaires que nous avons mis en place lors de la section 6.2. Dans un réseau JXTA, un identifiant unique appelé *URN* est associé à chaque ressource (par exemple : *Peer*, *Peer Group*, *Pipe*[6], etc.). De plus, à toute ressource est associé un document XML, appelé *Advertisement*, qui est publié sur un réseau JXTA pour le décrire. Dans ce qui suit nous illustrons le déploiement de notre « banc d'essai » sur la plate-forme JXTA en utilisant la `CommunityConsole` que nous avons développée (voir Figure 6.11).



(a) Interface pour la spécification de la taille du réseau



(b) Interface pour la configuration des différents *registry peers* et *communautés*

(c) Interface de la console du réseau de communautés

FIGURE 6.11 – Interface graphique de `CommunityConsole`.

Notre `CommunityConsole` permet de créer automatiquement les données de configuration (c.-à-d. les *URN* et les *Advertisements*) pour les *Peers* (représentant les *registry peers*), les *Peer Groups* (représentant les *communautés d'annuaires*) et les *Pipes* afin de déployer un réseau de communautés organisé comme décrit par nos descripteurs [7] de communautés et d'annuaires. En utilisant notre `CommunityConsole`, nous commençons par fournir le nombre d'annuaires à déployer ainsi que le nombre

6. *Pipe* est un concept JXTA permettant d'établir des canaux de communications entre différents pairs dans un réseau JXTA.

7. Nous rappelons qu'à l'issue des trois étapes de la préparation de notre « banc d'essai » (section 6.2), nous avons créé quatre descripteurs de communautés et sept descripteurs d'annuaires.

de communautés (voir Figure 6.11(a)). Par la suite, une autre interface graphique nous permet de spécifier les descripteurs pour chacun de ces annuaires et communautés (voir Figure 6.11(b)).

Une fois les descripteurs spécifiés, la `CommunityConsole` génère automatiquement le réseau de communautés d'annuaires. `CommunityConsole` fournit aussi une console (voir Figure 6.11(c)) pour déployer le réseau ainsi créé. Nous rappelons que chaque *registry peer* déployé servira de mandataire à l'un des annuaires UDDI implémenté en utilisant jUDDI dans la section 6.2.1. Nous utilisons l'API UDDI4J [8] pour assurer la communication des *registry peers* avec les annuaires qu'ils mandatent. Nous rappelons que dans la section 3.4, nous avons conditionné l'appartenance d'un annuaire à une communauté par son degré d'appartenance. Selon cette condition, un annuaire n'est membre d'une communauté que si son degré d'appartenance est supérieur à un seuil `th`. Sur la base des degrés d'appartenance des annuaires aux différentes communautés (voir Figure 6.7(b)) nous obtenons l'organisation suivante (voir Table 6.1) :

Communauté	***Leader***	***Disciples***
Community 1	Registry 3	Registry 7
Community 2	Registry 2	Registry 1, Registry 3, Registry 7
Community 3	Registry 4	Registry 5, Registry 7
Community 4	Registry 6	Registry 5, Registry 7

TABLE 6.1 – Organisation des annuaires en communautés

Le « banc d'essai » ainsi déployé sera utilisé dans la section suivante pour expérimenter notre approche de découverte de services.

6.3.2.2 Mise en place d'un système de découverte de services Web

Dans cette section nous détaillons les expérimentations que nous avons conduites en utilisant un système de découverte de services développé selon l'architecture proposée dans la section 6.3.1. Pour mettre en place ce système, nous avons développé un *trading peer* selon le modèle proposé dans la Figure 6.10. Ce pair offre une interface graphique au demandeur de services lui permettant de formuler ses requêtes et implémente notre approche de découverte de services. Dans la suite, nous présentons étape par étape le déroulement du processus de découverte de services en utilisant notre système illustré par la Figure 6.12.

Dans la section 5.2 nous avons défini une requête de découverte de services Q par la paire $< SD, RC >$ où SD est une description abstraite d'un service qui exprime le besoin fonctionnel du demandeur de services et RC est la caractérisation d'un demandeur de services. Notre système de découverte de services, à travers le composant *GUI* du *trading peer*, assiste un demandeur de services dans la formulation de sa requête et permet de générer semi-automatiquement la requête Q. La

8. http ://uddi4j.sourceforge.net/

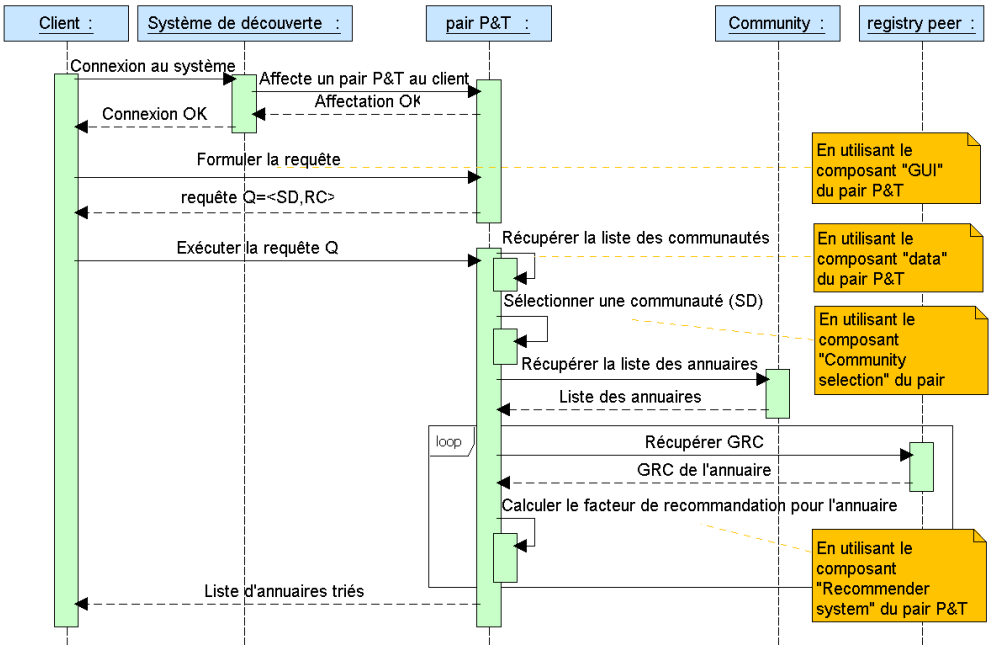


FIGURE 6.12 – Diagramme de séquence du système de découverte

Figure 6.13 présente l'interface graphique de notre système d'aide à la formulation d'une requête. Dans cet exemple, formuler une requête revient tout simplement à remplir les champs du formulaire proposé. La requête générée est donnée dans l'Annexe A.3 (SD) et l'Annexe A.4 (RC). Dans cet exemple, la partie fonctionnelle SD de notre requête a été formulée de façon à correspondre à la description abstraite SD_1 générée dans notre collection de test et qui représente le besoin fonctionnel d'une requête (voir section 6.2.1).

En exécutant la requête, le composant *Community selection* du *trading peer* sélectionne la communauté la plus adéquate à la requête sur la base des besoins fonctionnels SD exprimés. Dans notre exemple, c'est la communauté 2 qui est sélectionnée (voir Figure 6.14). Cette communauté contient bien l'annuaire R_1, comme l'indique la Figure 6.7(b), proposant les services du *relevance set* SD_1. Ceci montre la pertinence de notre approche de sélection de communauté pour cet exemple.

Par la suite, le système de découverte met en œuvre notre technique de recommandation pour recommander l'un des annuaires de la communauté 2 au demandeur de services. Cette recommandation est faite sur la base de la caractérisation RC du demandeur de services et des caractérisations globales GRC des annuaires. Vu que les GRC résultent des caractérisations des utilisateurs passés, nous avons exécuté plusieurs requêtes de découverte sur notre système afin que des GRC soient crées pour les annuaires. Dans notre exemple, le *trading peer* contacte la communauté sélectionnée (la communauté 2) pour récupérer la liste de ses annuaires membres. Par la suite, le *Recommender system* de notre *trading peer* communiquera avec les diffé-

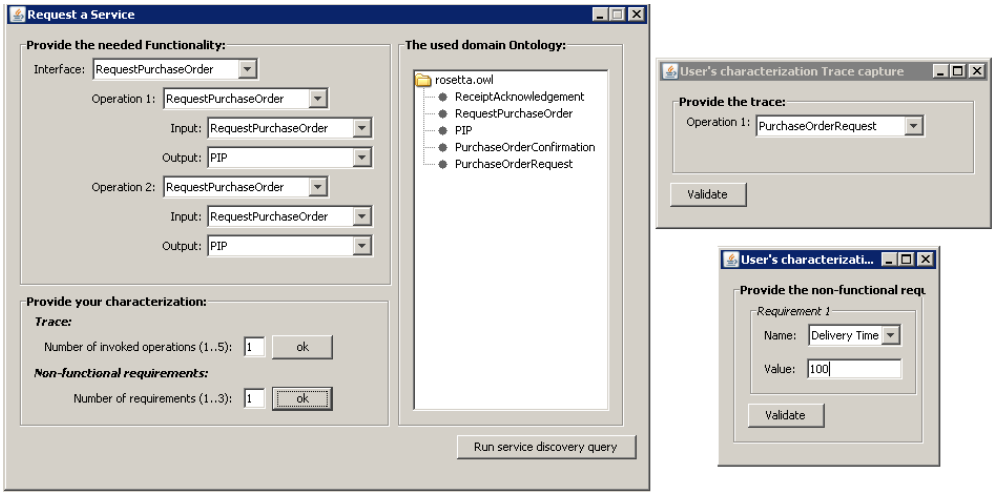


FIGURE 6.13 – Interfaces graphiques offertes par le *trading peer* pour la création de la requête

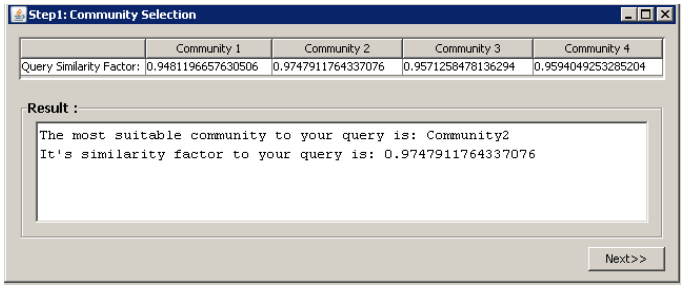


FIGURE 6.14 – Sélection de communauté

rents *registry peers* de la communauté sélectionnée, deux dans notre exemple, pour recommander l'un d'eux. Il récupère la caractérisation globale du premier annuaire dans la liste et calcule les différents facteurs de similarité vis-à-vis de la caractérisation du demandeur de services. Ces facteurs de similarité permettent de calculer le facteur de recommandation. Ces différents calculs sont effectués en utilisant les formules proposées dans la section 5.4. Ces mêmes étapes sont reconduites pour calculer le facteur de recommandation du deuxième annuaire sur la liste. Enfin, le *trading peer* recommandera l'annuaire R_2 (voir Figure 6.15).

6.3.3 Evaluation de l'approche de sélection de communauté

Nous évaluons dans cette section les performances de notre approche de sélection de communauté en termes de rappel et de précision. Nous reprenons la collection de descriptions de services générée dans la section 6.2.1 (voir Figure 6.2) et nous la distribuons sur sept annuaires comme indiqué dans la Table 6.2.

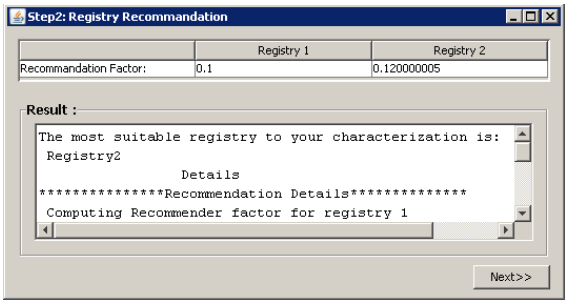


FIGURE 6.15 – Sélection d'annuaires

	SD1	SD2	SD3	SD4	SD5	SD6	SD7	services
Registry 1	34	0	25	0	0	32	18	100
Registry 2	0	57	0	4	8	21	0	100
Registry 3	6	0	42	0	0	0	48	100
Registry 4	14	0	0	46	20	0	0	100
Registry 5	0	0	0	50	50	0	0	100
Registry 6	24	43	12	0	22	33	12	100
Registry 7	22	0	21	0	0	14	22	100

TABLE 6.2 – Organisation de la collection générée en sept annuaires

Nous utilisons par la suite `WSRDCreator` pour générer les descriptions WSRD associées à ces annuaires et nous utilisons `CommunityBuilder` pour organiser les annuaires en deux communautés. L'organisation résultante est illustrée à travers la Table 6.3.

Communauté	***Leader***	***Disciples***
Community 1	Registry 3	Registry 1, Registry 2, Registry 6, Registry 7
Community 2	Registry 5	Registry 4

TABLE 6.3 – Organisation des annuaires en communautés

Afin de calculer la précision et le rappel de notre approche de sélection de communauté, nous exécutons sept requêtes de découverte de services en utilisant notre système de découverte de services. Nous formulons les requêtes de façon à ce que la partie fonctionnelle SD de chaque requête Q_i corresponde à la description abstraite SD_i générée dans notre collection de test et qui représente le besoin fonctionnel d'une requête (voir section 6.2.1). Les résultats des exécutions des différentes requêtes sont donnés dans la Table 6.4 où : la première colonne contient le nom de la requête exécutée, la deuxième le nom de la communauté sélectionnée, la troisième le nombre d'annuaires contenant des réponses à la requête et qui appartiennent à

cette communauté, la quatrième le nombre total d'annuaires appartenant à cette communauté et la cinquième le nombre total d'annuaires contenant des réponses à la requête parmi toutes les communautés.

Requête	Communauté sélectionnée	Annuaires pertinents récupérés	Annuaires récupérés	Annuaires pertinents
Q1	Community1	4	5	5
Q2	Community1	2	5	2
Q3	Community1	4	5	4
Q4	Community2	2	2	3
Q5	Community2	2	2	4
Q6	Community1	4	5	4
Q7	Community1	4	5	4

TABLE 6.4 – Résultats de l'expérimentation

En utilisant les résultats de la Table 6.4, nous évaluons les performances de notre approche de sélection de communauté en terme de rappel et de précision. Le rappel indique le taux d'annuaires découverts et offrants des réponses à une requête (formule (6.1a)) et la précision le taux d'annuaires offrant des réponses à une requête parmi tous ceux découverts (formule (6.1b)) :

$$Rappel = \frac{Nombre\ d'annuaires\ découverts\ contenant\ des\ réponses\ à\ la\ requête}{Nombre\ d'annuaires\ contenant\ des\ réponses\ à\ la\ requête} \tag{6.1a}$$

$$Précision = \frac{Nombre\ d'annuaires\ découverts\ contenant\ des\ réponses\ à\ la\ requête}{Nombre\ d'annuaires\ découverts} \tag{6.1b}$$

Une approche de sélection de communauté parfaite offre des réponses dont le rappel et la précision sont égaux à 100% (c.-à-d. une approche qui sélectionne **toutes** les communautés dont les annuaires offrent des services répondant à la requête **sans** sélectionner d'autres communautés). Notre approche offre un rappel de 85% avec une précision de 80% (voir Table 6.5). Ici, il faut bien rappeler que nous évaluons uniquement la première étape de notre approche de découverte d'annuaires (la sélection de communauté) et non pas la deuxième (la sélection d'annuaires).

6.3.4 Synthèse

Dans cette section, nous avons proposé une architecture P2P pour le développement d'un système de découverte de services dans un réseau d'annuaires organisé en communautés. Dans cette architecture, le pair *P&T* permet de limiter l'espace de recherche dans un processus de découverte de services en sélectionnant la communauté adéquate à une requête de découverte (le composant *Community selection*) et en recommandant un ou plusieurs annuaires aux demandeurs de services (le composant *Recommender System*).

Requête	Rappel	Précision
Q1	80%	80%
Q2	100%	40%
Q3	100%	80%
Q4	66%	100%
Q5	50%	100%
Q6	100%	80%
Q7	100%	80%
Moyenne	85%	80%

TABLE 6.5 – Rappel et précision

Par la suite, nous avons développé un système de découverte de services selon cette architecture. Les composants *Community selection* et *Recommender System* implémentent respectivement nos approches de sélection de communauté (voir section 5.3.1) et de recommandation d'annuaires (voir sections 5.3.2 et 5.4). Pour expérimenter notre système de découverte de services, nous avons déployé notre « banc d'essai » sur un réseau P2P basé sur JXTA.

6.4 Simulation des mécanismes de gestion de communautés

Nous nous intéressons dans cette section à la validation des mécanismes de gestion de communautés que nous avons définis dans la chapitre 4. Dans ce but, nous avons développé `CommunityManager` permettant de :

- simuler des changements dans le réseau de communautés (ajout d'un annuaire, suppression d'un annuaire, etc.) afin de tester les mécanismes de gestion définis, et
- visualiser graphiquement un réseau de communautés d'annuaires en utilisant les API JGrapht [9] et JGraph [10].

Dans la suite de cette section, nous présentons `CommunityManager` et les fonctionnalités qu'il propose. Puis, nous expérimentons ses mécanismes de gestion à travers trois scénarios : ajout d'un annuaire, suppression d'un annuaire et mise à jour des fonctionnalités d'un annuaire. Finalement, nous évaluons nos mécanismes de gestion en comparant les résultats obtenus aux résultats d'une réexécution de notre approche de construction de communautés.

9. http ://www.jgrapht.org/
10. http ://www.jgraph.com/

6.4.1 L'outil *Community Manager*

L'outil *Community Manager* a été conçu et développé pour tester la faisabilité et valider les algorithmes et les opérations de gestion proposés. Pour illustrer la faisabilité de notre approche, nous simulons un réseau de communautés d'annuaires en reprenant la même organisation de notre « banc d'essai ». Ainsi, nous utilisons les mêmes descripteurs de communautés et d'annuaires obtenus dans la section 6.2.3 et le même seuil `th` défini dans la section 6.3.2.1 comme entrée d'initialisation pour notre *Community Manager*. La Figure 6.16 représente le résultat obtenu avec cette configuration. Le cadre 1 de la Figure 6.16, offre une représentation des quatre graphes $c.G$ ($c \in C$) (voir Définition 3.6) correspondant aux quatre communautés. Le *leader* de chaque communauté est représenté avec un rectangle foncé. Le graphe CG représentant notre réseau de communautés est présenté dans le Cadre 2. Dans ce graphe, une arête reliant deux nœuds (c.-à-d. communautés) désigne deux communautés adjacentes ayant des annuaires en commun (voir Définition 3.7).

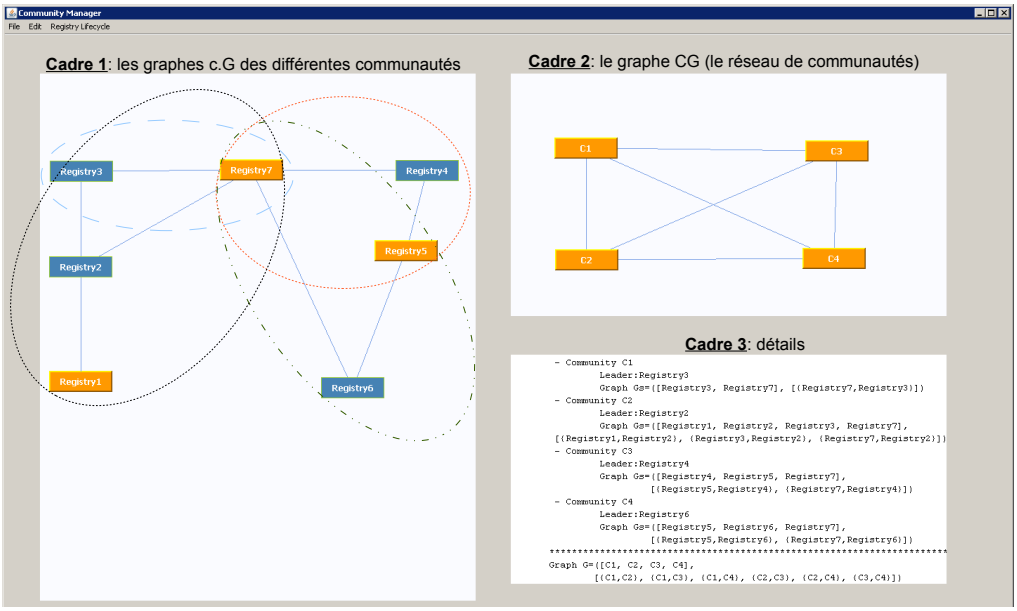


FIGURE 6.16 – Community Manager

Community Manager implémente les différents mécanismes de gestion définis dans le Chapitre 4. Dans ce qui suit, nous présentons trois scénarios d'utilisation déclenchant des changements dans l'organisation du réseau de communautés. Ces changements susciteront les mécanismes de gestion appropriés pour conserver la consistance de l'organisation.

6.4.1.1 Scénario 1 : Ajout d'un annuaire

Pour tester les algorithmes de sélection de communauté (Algorithme 3) et de resélection d'un *leader* (Algorithme 4), nous ajoutons un nouvel annuaire dans notre réseau d'annuaires. Cet annuaire, nommé r_8, est décrit par un descripteur d'annuaire reflétant une fonctionnalité moyenne $r_8.f$ égale à [0.234 0.314 0.048 0.181 0.534]. Conformément à l'Algorithme 3, notre *Community Manager* commence par calculer les degrés d'appartenance m de r_8 aux communautés $c_i, i = 1 \ldots 4$ du réseau et teste pour chaque communauté si r_8 en sera un membre (c.-à-d. si $m > th$). Les degrés d'appartenance obtenus sont $r_8.MEM = \{(c_1, 0.162), (c_2, 0.698), (c_3, 0.054), (c_4, 0.085)\}$ et le nouvel annuaire appartient ainsi aux deux communautés c_1 et c_2. L'algorithme de resélection d'un *leader* assignera à r_8 le rôle de disciple dans ces deux communautés (Figure 6.17).

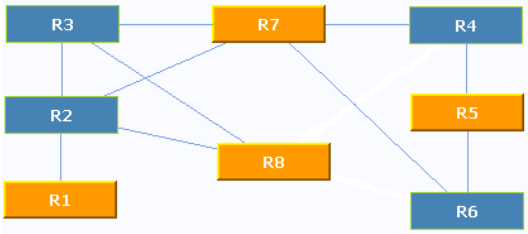
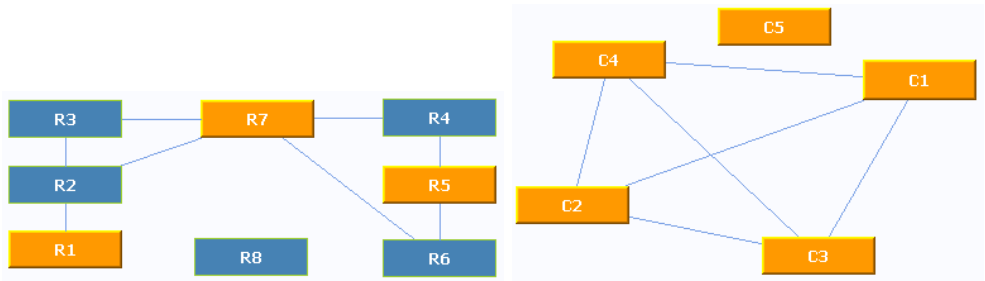


FIGURE 6.17 – Ajout de l'annuaire r_8

Nous reconduisons maintenant la même expérience en augmentant la valeur du seuil `th`. Dans ce cas, tous les degrés d'appartenance de r_8 sont inférieurs à `th` et ainsi r_8 n'appartient à aucune des communautés. Ceci représente la **précondition** à la création d'une communauté comme définie dans la section 4.4.1. Conformément à l'algorithme de création de communauté (Algorithme 5), une nouvelle communauté est établie automatiquement (Figure 6.18(b)) avec r_8 comme *leader* et aucun *disciple* n'a été recruté (Figure 6.18(a)).



(a) Les graphes $c.G$ des différentes communautés

(b) Le réseau de communautés CG

FIGURE 6.18 – Ajout de l'annuaire r_8

6.4.1.2 Scénario 2 : Suppression d'un annuaire

Dans ce deuxième scénario, nous simulons la suppression de l'annuaire r_3. r_3 représente le *leader* de la communauté c_1 formée par les deux annuaires r_3 et r_7 (voir Figure 6.16). Cette suppression implique l'exécution de l'algorithme de resélection d'un *leader* (Algorithme 4) qui assignera r_7 comme nouveau *leader* pour c_1 (Figure 6.19).

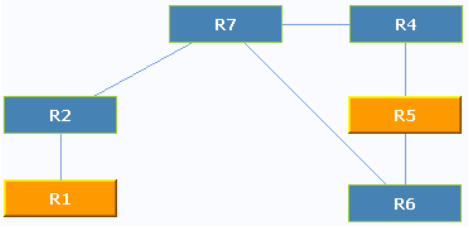
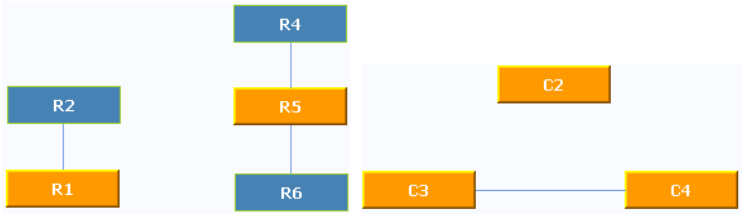


FIGURE 6.19 – Suppression de l'annuaire r_3

Pour tester le démontage d'une communauté, nous supprimons aussi l'annuaire r_7 (Figure 6.20(a)). La communauté c_1 devient vide, la **précondition** de démontage d'une communauté définie dans la section 4.4.2 est satisfaite, et la communauté est automatiquement supprimée (Figure 6.20(b)). Par conséquent, le graphe de communautés CG est mis à jour en supprimant c_1 et toutes les arêtes dont c_1 était l'une des extrémités.



(a) Les graphes $c.G$ des différentes communautés (b) Le réseau de communautés CG

FIGURE 6.20 – Suppression de l'annuaire r_7

6.4.1.3 Scénario 3 : Mise à jour des fonctionnalités d'un annuaire

A travers ce troisième scénario, nous testons l'algorithme de fusion (Algorithme 6) de deux communautés. Sur la base de notre réseau de communautés initial présenté dans la Figure 6.16, nous simulons une mise à jour des fonctionnalités de l'annuaire r_4. Nous changeons le vecteur représentant la fonctionnalité moyenne f de r_4 en [0.4 0.8 0.1 0.1 0.6]. r_4 quitte ainsi la communauté c_3 pour rejoindre c_1 et r_7 est désigné comme *leader* de c_3.

La communauté c_3 est alors composée de r_5 et r_7 et est totalement incluse dans c_4 formée par r_5, r_6 et r_7. Ceci peut engendrer la **fusion de la communauté** c_3

avec c_4. Cependant, selon la précondition définie dans l'Algorithme 6, il faut que la distance entre c_3 et c_4 soit inférieure à un seuil ξ que nous avons fixé à 0.2 (voir section 4.4). Ces deux conditions étant vérifiées, les communautés c_2 et c_3 sont ainsi supprimées du graphe G et c_{merg}, qui est ici c_5, est y ajoutée (Figure 6.21).

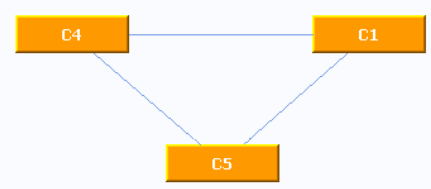


FIGURE 6.21 – Le réseau de communautés CG résultant

Ces scénarios montrent bien la faisabilité de notre approche de gestion pour gérer les différents événements pouvant déclencher des changements organisationnels dans un réseau de communautés. Dans la section suivante, nous évaluons les mécanismes de gestion que nous avons défini pour conserver une organisation en communautés.

6.4.2 Evaluation

Dans cette section, nous testons la consistance de l'organisation de nos **mécanismes de gestion** par rapport à une **réexécution de notre approche de construction de communautés** suite à des changements dans notre réseau de communautés. Nous rappelons que l'objectif des mécanismes de gestion que nous proposons est de préserver la consistance d'un réseau de communautés. Certes, une réexécution de notre approche de construction de communautés garantit, avec la plus grande précision, la consistance de l'organisation des annuaires mais reste cependant inconcevable pour des raisons de coûts.

Réorganisation en utilisant les mécanismes de gestion

Pour effectuer ce test, nous utilisons l'organisation initiale décrite par les mêmes descripteurs de communautés et d'annuaires obtenus dans la section 6.2.3. Nous introduisons successivement les changements suivants :

1. Nous introduisons dans le réseau deux nouveaux annuaires r_8 et r_9.
2. Nous supprimons par la suite les trois annuaires r_1, r_4 et r_5.

Le réseau d'annuaires ainsi obtenu est formé de six annuaires $(r_2, r_3, r_6, r_7, r_8, r_9)$ répartis en quatre communautés. Les fonctionnalités $r.f$ de ces annuaires, représentées par leurs WSRD, sont présentées dans la Table 6.6.

Après ces changements, le graphe des annuaires montrant les appartenances de ces annuaires aux différentes communautés C_1, C_2, C_3 et C_4 est illustré dans la Figure 6.22(a).

Réorganisation en utilisant `CommunityBuilder`

	w1	**w2**	**w3**	**w4**	**w5**
$r_2.f$	0.345	0.87	0	0.321	0.697
$r_3.f$	0.278	0.679	0.442	0.046	0.536
$r_6.f$	0.3	0.781	0.024	0.652	0.42
$r_7.f$	0.261	0.637	0.282	0.269	0.353
$r_8.f$	0.234	0.314	0.048	0.181	0.534
$r_9.f$	0.334	0.426	0.099	0.456	0.689

TABLE 6.6 – La base des descriptions WSRD résultantes après introduction des changements

Nous testons maintenant la réorganisation, après les mêmes changements, d'un réseau de communautés en réexécutant notre approche de construction de communautés. Nous reprenons la base des descriptions WSRD résultantes (voir Table 6.6) et nous la fournissons comme entrée à notre `CommunityBuilder` (voir section 6.2.3). Les annuaires sont organisés en quatre communautés.

Les centres de ces communautés $c.f$ sont représentées dans la Table 6.7. Dans la Table 6.8, nous présentons les degrés d'appartenance MEM des six annuaires r_m aux quatre communautés construites.

	w1	**w2**	**w3**	**w4**	**w5**
$C_1.f$	0.3001	0.7684	0.0315	0.6386	0.4258
$C_2.f$	0.3431	0.8516	0.0082	0.3242	0.6906
$C_3.f$	0.2747	0.6658	0.4001	0.1033	0.4949
$C_4.f$	0.2587	0.3507	0.0674	0.2493	0.5655

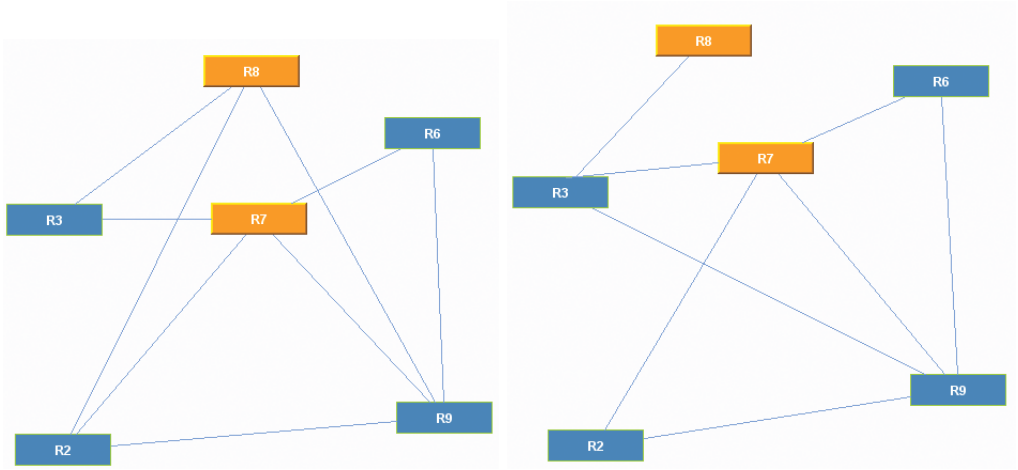
TABLE 6.7 – Les centres des communautés

	C_1	C_2	C_3	C_4
r_2	0.0024	0.9945	0.0016	0.0015
r_3	0.0121	0.0202	0.9452	0.0225
r_6	0.9957	0.0023	0.0009	0.0011
r_7	0.1476	0.1348	0.5275	0.1901
r_8	0.0172	0.0214	0.0290	0.9323
r_9	0.1662	0.1812	0.1195	0.5331

TABLE 6.8 – Les degrés d'appartenance des annuaires aux différentes communautés

Nous relançons le *CommunityManager* avec comme entrée les descripteurs de communauté et d'annuaire générés par le `CommunityBuilder` afin d'observer la nouvelle réorganisation. Le graphe des annuaires résultant est illustré dans la Figure 6.22(b).

Discussion



(a) En utilisant les mécanismes de gestion (b) En utilisant `CommunityBuilder`

FIGURE 6.22 – Les graphes d'annuaires résultants après les réorganisations

En comparant les deux graphes résultants, nous constatons que les *leaders* des communautés sont les mêmes pour les deux graphes. Ceci représente un bon indicateur sur la consistance de l'organisation de notre approche de gestion de communautés vu que le *leader* d'une communauté reflète bien sa fonctionnalité moyenne.

De plus, nous remarquons que la répartition des annuaires sur les quatre communautés est similaire, mais pas exactement identique. Les seules différences résident au niveau des arêtes (R_3, R_9), (R_8, R_2) et (R_8, R_9). Dans l'organisation résultante après une réexécution de notre approche de construction de communautés (voir Figure 6.22(b)), r_8 n'appartient pas aux communautés dont r_2 et r_9 sont *leaders* et r_3 n'appartient pas à la communauté dont r_9 est *leader*. Par contre, dans l'organisation résultante en utilisant les mécanismes de gestion (voir Figure 6.22(a)) r_8 appartient aux deux communautés dont r_2 et r_9 sont *leader* et r_3 n'appartient pas à celle dont r_9 est *leader*.

Cette différence se traduit par des degrés d'appartenance MEM différents. En effet, en utilisant notre approche de gestion, les centres des communautés ne sont pas mis à jour à chaque changement. Tandis que, avec notre approche de construction de communautés, les centres sont mis à jours à chaque itération de notre algorithme de clustering (voir section 3.4.4, equation 3.6). Cette différence au niveau des centres des communautés engendre une différence lors du calcul de la distance séparant un annuaire d'une communauté et cause des différences dans les degrés d'appartenance MEM (voir equation 3.4 et 3.5).

6.4.3 Synthèse

Dans cette section, nous avons développé un outil (*CommunityManager*) permettant de simuler graphiquement un réseau de communautés. *CommunityManager* permet également de simuler l'ajout/suppression d'un annuaire dans un réseau de

communautés ou la mise à jour de ses fonctionnalités. Les différents mécanismes de gestion de communautés d'annuaires définies dans le chapitre 4 sont aussi implémentés par *CommunityManager*.

Lors de nos expérimentations, nous avons reproduit trois scénarios suscitant des changements organisationnels sur un réseau de communautés simulé par notre *CommunityManager*. Ces tests ont bien montré que notre approche de gestion permet de gérer les changements qui peuvent avoir lieu dans un réseau de communautés.

Nous avons aussi testé la consistance de l'organisation de nos mécanismes de gestion par rapport à une réexécution de notre approche de construction de communautés. Ces tests ont montré que les résultats sont pratiquement les mêmes. Vu qu'une réexécution de notre approche de construction de communautés est très coûteuse en termes de ressources, notre approche de gestion de communauté permet de garder la consistance d'une organisation en communautés à un moindre coût.

6.5 Conclusion

Ce chapitre a été consacré à la mise en œuvre et la validation de nos approches de construction de communautés d'annuaires, de découverte de services et de gestion de communautés. Pour réaliser nos expérimentations, nous avons commencé par mettre en place un « banc d'essai » formé par un réseau d'annuaires organisés en communautés. Pour cela, nous avons généré une collection de descriptions de services sémantiques que nous avons publiée sur différents annuaires. Ensuite, nous avons développé l'outil `WSRDCreator` permettant de créer les descriptions WSRD des annuaires selon notre approche de construction de descriptions WSRD. Puis, nous avons utilisé `CommunityBuilder` pour organiser les annuaires en communautés selon leurs fonctionnalités décrites par leurs WSRD. Enfin, nous avons utilisé `CommunityConsole` pour déployer le réseau d'annuaires au-dessus d'une infrastructure P2P conformément à l'organisation en communautés établie. Ces différentes expérimentations montrent bien la **faisabilité de nos approches** de création de descriptions WSRD et de construction de communautés.

Nous avons aussi proposé une architecture pour le développement de systèmes de découverte de services dans un réseau d'annuaires organisé en communautés. Un système de découverte de services a été ainsi développé selon cette architecture. Le scénario de découverte de services mis en œuvre indique bien la **consistance de l'organisation** de notre approche de construction de communautés.

Enfin, nous avons introduit l'outil `CommunityManager` permettant de : simuler graphiquement un réseau d'annuaires, simuler l'ajout/suppression d'un annuaire dans un réseau de communautés ou la mise à jour de ses fonctionnalités et mettre en œuvre notre approche de gestion de communautés. Les expérimentations réalisées à travers `CommunityManager` nous ont permis de **tester notre approche de gestion** et la consistance de l'organisation de nos mécanismes de gestion par rapport à une réexécution de notre approche de construction de communautés.

Chapitre 7

Conclusions et perspectives

7.1 Conclusions

L'objectif principal de ce travail de thèse était de fournir une approche de découverte d'annuaires de services Web dans un environnement distribué d'annuaires. Pour assurer cet objectif, nous avons proposé : une approche pour organiser un réseau d'annuaires en communautés, une approche pour conserver la consistance de cette organisation et une approche de découverte tirant profit de cette organisation.

L'approche d'organisation que nous proposons se distingue des approches actuelles par le fait qu'elle est **implicite**, **automatique** et basée sur les **fonctionnalités** des services Web publiés. Nous utilisons les descriptions de services annoncés par les annuaires pour créer implicitement, sans aucune intervention de la part d'un fournisseur d'annuaires, des descriptions WSRD reflétant les fonctionnalités de ces annuaires. Ces descriptions sont utilisées pour organiser des annuaires selon leurs fonctionnalités en groupes que nous appelons communautés. Vu qu'un demandeur de services est à la recherche d'une certaine fonctionnalité, une telle organisation est plus logique que celles basées sur le domaine métier utilisées actuellement par plusieurs approches [Verma 2005, Pilioura 2004, Sivashanmugam 2004, Pilioura 2009, Du 2005, Ayorak 2007]. Pour assurer une organisation automatique d'un réseau d'annuaires en communautés, nous utilisons une approche de clustering floue.

Vu la dynamicité d'un environnement orienté services, des mécanismes de gestion sont nécessaires pour maintenir la consistance d'une organisation d'un réseau d'annuaires en communautés et assurer son intégrité. Nous avons donc identifié les différentes étapes des cycles de vie d'un annuaire au sein d'une communauté et celui d'une communauté d'annuaires. Nous avons, par la suite, spécifié les opérations de gestion nécessaires pour quelques-unes de ces étapes. Les étapes nécessitant des mécanismes de gestion sont essentiellement celles pouvant impliquer (directement ou indirectement) un changement dans l'organisation de notre réseau. Pour chaque mécanisme de gestion défini, nous spécifions des pré-conditions permettant ainsi une gestion de communautés **dynamique** sans **aucune intervention humaine**.

L'organisation d'un réseau d'annuaires en communautés permet de limiter l'espace de recherche d'un demandeur de services en réduisant le nombre de communautés et ainsi celui des annuaires à parcourir. Ainsi, l'approche de découverte d'annuaires proposée tire profit de cette organisation en sélectionnant la ou les communauté(s) d'annuaires adéquate(s) aux besoins fonctionnels d'un demandeur de services. Vu que le nombre d'annuaires dans une communauté peut être très important, nous utilisons des techniques de recommandation afin de limiter l'espace de

recherche d'un demandeur de services. Nous adoptons une approche de filtrage collaboratif implicite basée sur les résultats des processus de découvertes précédentes et les besoins non fonctionnels des demandeurs de services pour recommander le (ou les) annuaire(s) adéquat(s) à une requête de découverte de services. L'approche de découverte d'annuaires proposée prend ainsi en considération les besoins **fonctionnels**, **non-fonctionnels** et le **comportement** (représenté par la trace d'exécution) du demandeur de services pour la sélection d'annuaires.

Nous avons aussi validé et mis en œuvre nos propositions à travers le développement et l'utilisation d'un certain nombre d'outils. Nous avons développé : l'outil `WSRDCreator` pour la création de descriptions WSRD d'un ensemble d'annuaires, l'outil `CommunityBuilder` pour organiser des annuaires en communautés selon leurs description WSRD et l'outil `CommunityConsole` pour déployer un réseau d'annuaires, organisé en communautés, au-dessus d'une infrastructure P2P. Les expérimentations menées avec ces outils montrent bien la faisabilité de nos approches de création de descriptions WSRD et de construction de communautés. Nous avons aussi introduit l'outil `CommunityManager` permettant de simuler graphiquement un réseau d'annuaires et de mettre en œuvre notre approche de gestion de communautés. Les expérimentations réalisées à travers cet outil nous ont permis de tester notre approche de gestion et la consistance de notre organisation. Finalement, nous avons proposé une architecture pour le développement de systèmes de découverte de services dans un réseau d'annuaires organisé en communautés. Un système de découverte de services a été ainsi développé selon cette architecture et les expérimentations réalisées indiquent bien la consistance de l'organisation en communautés de notre approche de construction de communautés.

A travers les différentes contributions proposées, nous avons résolu la problématique de recherche exprimée au début de notre thèse. En effet, nous avons proposé une approche de découverte multi-annuaires permettant de réduire l'espace de recherche d'un demandeur de services, en organisant les annuaires en communautés et en utilisant des techniques de recommandations, tout en garantissant des résultats de découverte pertinents.

7.2 Perspectives

Dans cette section, nous présentons quelques directions de travaux futurs. Nous présentons d'abord les perspectives liées aux trois principales contributions de notre thèse : l'organisation d'un réseau d'annuaires en communautés, sa gestion et la découverte d'annuaires dans un réseau de communautés d'annuaires. Nous présentons ensuite un travail en cours illustrant l'adaptation du travail de notre thèse à un autre contexte.

1. Perspectives liées à l'organisation d'un réseau d'annuaires en communautés

Lors de la construction de la description WSRD d'un annuaire, nous perdons une partie de la sémantique structurelle des descriptions SAWSDL d'un annuaire

en utilisant une réduction faible. En effet, les liens entre les concepts annotant les différents éléments d'une même description de services ne sont pas représentés. Nous projetons d'étudier cet aspect pour éviter cette perte de sémantique.

Aussi, et vu que les descriptions de services contenues dans un annuaire changent fréquemment (publication/retrait de descriptions de services), nous envisageons de proposer une nouvelle méthode incrémentale pour la mise à jour de descriptions WSRD. Suite à la publication (resp. retrait) d'une description de service, la nouvelle WSRD sera calculée en fonction de l'ancienne WSRD et la description du service publiée (resp. retirée). Une telle méthode évite la réexécution de notre approche de création de WSRD (section 3.2) à chaque fois que la description d'un service soit publiée/retirée.

Enfin, nous envisageons de généraliser notre approche pour qu'elle puisse être utilisée avec des annuaires de services offrant des services décrits avec d'autres langages de description de services Web sémantiques tels que OWL-S [Martin 2004a], WSMO [De Bruijn 2005] ou YASA [Chabeb 2008].

2. Perspectives liées à la gestion d'un réseau de communautés d'annuaires

Dans le chapitre 6, nous avons pu constater que la structure d'une réorganisation d'un réseau d'annuaires moyennant notre approche de gestion est similaire, mais pas identique, à celle obtenue par une réexécution de notre approche de construction de communautés. Ainsi, à court terme l'utilisation de notre approche de gestion n'aura pas d'impact négatif sur le processus de découverte d'annuaires.

Nous envisageons d'étudier l'impact de notre approche de gestion d'un réseau de communautés d'annuaires sur le rappel et la précision de notre approche de découverte à long terme (c.-à-d. après l'arrivée et le départ d'un nombre important d'annuaires). Selon ses résultats, nous pourrions décider de la nécessité ou pas de réexécuter notre approche de construction de communautés pour maintenir l'organisation d'un réseau d'annuaires après plusieurs changements. Pour savoir à partir de quand une réexécution de notre approche de construction de communautés devient nécessaire, nous pourrons utiliser une fréquence de changements incrémentale afin de détecter le moment où l'impact sur le rappel et la précision devient inacceptable.

3. Perspectives liées à la découverte dans un réseau de communautés d'annuaires

Dans le chapitre 6, nous avons évalué la première étape de notre approche de découverte d'annuaires, c.-à-d. la sélection de communautés. Vu que nous utilisons une technique de recommandation pour la deuxième étape, c.-à-d. la sélection d'annuaires, nous envisageons de demander à des utilisateurs différents de tester notre système de découverte afin de pouvoir collecter leurs différentes caractérisations et ainsi pouvoir évaluer cette deuxième étape. A plus long terme, nous envisageons de proposer une approche de découverte de services Web en intégrant à notre approche une technique de sélection de services développée dans notre équipe comme troisième étape. Cette sélection de services se base sur une technique

d'appariement sémantique [Chabeb 2010] ou sur une technique de recommandation [Nguyen 2010, Nguyen 2011].

4. Application de nos travaux à d'autres domaines

Le projet ANR PAIRSE[1] traite des problèmes liés au besoin de partage de données dans des environnements P2P (hétérogénéité, traitement de requête, etc.) en utilisant une approche à base de *data providing (DP) services*. Un *DP service* n'offre pas une fonctionnalité mais permet uniquement d'accéder à une source de données.

Dans le cadre de ce projet, nous envisageons d'adopter les résultats de notre travail de thèse afin d'organiser les *DP services* en clusters et gérer le réseau de cluster résultant. Vu que ces services sont représentés par des vues RDF paramétrées, nous adaptons notre approche de clustering pour concorder avec ce format de descriptions de services. Aussi, nous envisageons de considérer les relations sémantiques offertes par les vues RDF lors de l'organisation d'un ensemble de *DP services*. Des résultats préliminaires ont été validés par une publication [Zhou 2011a] et plus de détails peuvent être trouvés dans [Zhou 2011b].

1. https://picoforge.int-evry.fr/cgi-bin/twiki/view/Pairse/Web/WebHome

Annexe A

Fichiers sources

A.1 Exemple d'un descripteur de communauté

```
<?xml version="1.0" encoding="UTF-8"?>
<Community>
  <id>C1</id>
  <mean_functionality>
    <concept>
      <name>ReceiptAcknowledgement</name>
      <value>0.2697044132012029</value>
    </concept>
    <concept>
      <name>RequestPurchaseOrder</name>
      <value>0.6604569425690826</value>
    </concept>
    <concept>
      <name>PIP</name>
      <value>0.38393068744183034</value>
    </concept>
    <concept>
      <name>PurchaseOrderConfirmation</name>
      <value>0.13234786207896623</value>
    </concept>
    <concept>
      <name>PurchaseOrderRequest</name>
      <value>0.5169689590845238</value>
    </concept>
  </mean_functionality>
</Community>
```

A.2 Exemple d'un descripteur d'annuaire

```
<?xml version="1.0" encoding="UTF-8"?>
<Registry>
  <id>Registry1</id>
  <Mean_functionality>
    <concept>
      <name>ReceiptAcknowledgement</name>
      <value>0.31400000000000006</value>
    </concept>
    <concept>
      <name>RequestPurchaseOrder</name>
      <value>0.8160000000000001</value>
    </concept>
    <concept>
      <name>PIP</name>
      <value>0.11200000000000002</value>
    </concept>
```

```
    <concept>
      <name>PurchaseOrderConfirmation</name>
      <value>0.268</value>
    </concept>
    <concept>
      <name>PurchaseOrderRequest</name>
      <value>0.653</value>
    </concept>
  </Mean_functionality>
  <Memberships>
    <Community>
      <id>Community1</id>
      <m>7.876111404729408E-4</m>
    </Community>
    <Community>
      <id>Community2</id>
      <m>0.9986938303572499</m>
    </Community>
    <Community>
      <id>Community3</id>
      <m>1.9410736858372976E-4</m>
    </Community>
    <Community>
      <id>Community4</id>
      <m>3.244511336933431E-4</m>
    </Community>
  </Memberships>
</Registry>
```

A.3 La description SD générée lors de nos expérimentations

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:description xmlns:wsdl="http://www.w3.org/ns/wsdl" xmlns:sawsdl="http:
    //www.w3.org/ns/sawsdl" name="SemanticQuery" targetNamespace="http://
    query/sawsdl">
  <wsdl:interface name="interface1" sawsdl:modelReference="./resources/
      rosetta.owl#RequestPurchaseOrder">
    <wsdl:operation name="operation1" sawsdl:modelReference="./resources/
        rosetta.owl#RequestPurchaseOrder">
      <wsdl:input name="input1" sawsdl:modelReference="./resources/rosetta.
          owl#RequestPurchaseOrder" />
      <wsdl:output name="output1" sawsdl:modelReference="./resources/rosetta.
          owl#PIP" />
    </wsdl:operation>
    <wsdl:operation name="operation2" sawsdl:modelReference="./resources/
        rosetta.owl#RequestPurchaseOrder">
      <wsdl:input name="input2" sawsdl:modelReference="./resources/rosetta.
          owl#RequestPurchaseOrder" />
      <wsdl:output name="output2" sawsdl:modelReference="./resources/rosetta.
          owl#PIP" />
    </wsdl:operation>
  </wsdl:interface>
</wsdl:description>
```

A.4 La caractérisation RC générée lors de nos expérimentations

```
<?xml version="1.0" encoding="UTF-8"?>
<characterization name="Requester characterization" targetNamespace="http://
    characterization/xml">
  <trace>
    <operation>./resources/rosetta.owl#PurchaseOrderRequest</operation>
  </trace>
  <non_func_req>
    <requirement>
      <label>./resources/nonFuncReq.owl#DeliveryTime</label>
      <value>100</value>
    </requirement>
  </non_func_req>
</characterization>
```

A.5 Exemple d'une caractérisation GRC

```
<?xml version="1.0" encoding="UTF-8"?>
<global_characterization name="Global Registry characterization"
  targetNamespace="http://characterization/xml">
  <global_trace>
    <operation>
      <name>
        ./resources/rosetta.owl#ReceiptAcknowledgement
      </name>
      <number>
        20
      </number>
    </operation>
    <operation>
      <name>
        ./resources/rosetta.owl#RequestPurchaseOrder
      </name>
      <number>
        10
      </number>
    </operation>
    <operation>
      <name>
        ./resources/rosetta.owl#PurchaseOrderConfirmation
      </name>
      <number>
        40
      </number>
    </operation>
  </global_trace>

  <global_non_func_req>
    <requirement>
      <name>
        ./resources/nonFuncReq.owl#DeliveryTime
      </name>
      <value>
        150
      </value>
```

```
    </requirement>
  </global_non_func_req>
</global_characterization>
```

Bibliographie

[Adomavicius 2005] Gediminas Adomavicius and Alexander Tuzhilin. *Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions.* IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pages 734–749, 2005. (Cité en page 83.)

[Ayorak 2007] Evren Ayorak and Ayse Basar Bener. *Super Peer Web Service Discovery Architecture.* In Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, April 15-20, 2007, Istanbul, Turkey, pages 1360–1364. IEEE, 2007. (Cité en pages 8, 12, 13, 16, 17 et 119.)

[Bellwood 2002] Tom Bellwood, Luc Clï¿½ment, David Ehnebuske, Andrew Hately, Maryann Hondo, Yin Leng Husband, Karsten Januszewski, Sam Lee, Barbara McKee, Joel Munter and Claus von Riegen. *UDDI Version 3.0.* Rapport technique, 2002. (Cité en page 1.)

[Benatallah 2003] Boualem Benatallah, Quan Z. Sheng and Marlon Dumas. *The Self-Serv Environment for Web Services Composition.* IEEE Internet Computing, vol. 7, pages 40–48, 2003. (Cité en page 44.)

[Bentahar 2007] Jamal Bentahar, Zakaria Maamar, Djamal Benslimane and Philippe Thiran. *An Argumentation Framework for Communities of Web Services.* IEEE Intelligent Systems, vol. 22, no. 6, pages 75–83, 2007. (Cité en pages 7 et 22.)

[Bentahar 2008] Jamal Bentahar, Zakaria Maamar, Wei Wan, Djamal Benslimane, Philippe Thiran and Sattanathan Subramanian. *Agent-based communities of web services : an argumentation-driven approach.* Service Oriented Computing and Applications, vol. 2, no. 4, pages 219–238, 2008. (Cité en page 22.)

[Bezdek 1981] James C. Bezdek. Pattern recognition with fuzzy objective function algorithms. Kluwer Academic Publishers, 1981. (Cité en page 51.)

[Birukou 2007] Aliaksandr Birukou, Enrico Blanzieri, Vincenzo D'Andrea, Paolo Giorgini and Natallia Kokash. *Improving Web Service Discovery with Usage Data.* IEEE Software, vol. 24, no. 6, pages 47–54, 2007. (Cité en pages 25 et 27.)

[Blake 2007] M. Brian Blake and Michael F. Nowlan. *A Web Service Recommender System Using Enhanced Syntactical Matching.* In 2007 IEEE International Conference on Web Services (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA, pages 575–582, 2007. (Cité en pages 25 et 27.)

[Blake 2008] M. Brian Blake and Michael F. Nowlan. *Taming Web Services from the Wild.* IEEE Internet Computing, vol. 12, no. 5, pages 62–69, 2008. (Cité en page 25.)

[Bondy 2007] J. A. Bondy and U. S. R. Murty. Graph theory. Springer London, 2007. (Cité en page 46.)

[Bouchaalaa 2011a] Olfa Bouchaalaa, Mohamed Sellami, Walid Gaaloul, Samir Tata and Mohamed Jmaiel. *Graph-based management of communities of Web service registries.* In WEBIST 2011, Proceedings of the 7th International Conference on Web Information Systems and Technologies, Noordwijkerhout, The Netherlands, May 6-9, 2011. INSTICC Press, 2011. (Cité en page 73.)

[Bouchaalaa 2011b] Olfa Bouchaalaa, Mohamed Sellami, Walid Gaaloul, Samir Tata and Mohamed Jmaiel. *Modeling and Managing communities of Web Services registries.* In Web Information Systems and Technologies - 7th International Conference, WEBIST 20011, The Netherlands, May 6-9, 2011, Revised Selected Papers, Lecture Notes in Business Information Processing. Springer, 2011. (Cité en page 73.)

[Breininger 2001] Kathryn Breininger, Lisa Carnahan, Joseph M. Chiusano, Suresh Damodaran, Mike DeNicola Fujitsu, Anne Fischer, Sally Fuger, Jong Kim, Kyu-Chul Lee, Joel Munter, Farrukh Najmi, Joel Neu, Sanjay Patil, Neal Smith, Nikola Stojanovic, Prasad Yendluri and Yutaka Yoshida. *OASIS/ebXML Registry Information Model v2.0.* Rapport technique, 2001. (Cité en page 1.)

[Chabeb 2008] Yassin Chabeb and Samir Tata. *Yet Another Semantic Annotation for WSDL (YASA4WSDL).* In IADIS WWW/Internet 2008 Conference, pages 462–467, October 2008. (Cité en pages 32, 95 et 121.)

[Chabeb 2010] Yassin Chabeb, Samir Tata and Alain Ozanne. *YASA-M : A Semantic Web Service Matchmaker.* In The IEEE 24rd International Conference on Advanced Information Networking and Applications, AINA 2010, April 20-23, Perth, Australia, 2010. (Cité en pages 2, 75, 92, 95 et 122.)

[Claypool 2001] Mark Claypool, Phong Le, Makoto Waseda and David Brown. *Implicit interest indicators.* In IUI, pages 33–40, 2001. (Cité en page 27.)

[De Bruijn 2005] Jos De Bruijn*and al. Web Service Modeling Ontology (WSMO).* Rapport technique, 2005. (Cité en pages 32 et 121.)

[Ding 2001] Chris H. Q. Ding, Xiaofeng He, Hongyuan Zha, Ming Gu and Horst D. Simon. *A Min-max Cut Algorithm for Graph Partitioning and Data Clustering.* In ICDM '01 : Proceedings of the 2001 IEEE International Conference on Data Mining, pages 107–114, Washington, DC, USA, 2001. IEEE Computer Society. (Cité en pages 69 et 73.)

[Du 2005] Zongxia Du, Jinpeng Huai and Yunhao Liu. *Ad-UDDI : An Active and Distributed Service Registry.* In Technologies for E-Services, 6th International Workshop, TES, Trondheim, Norway, September 2-3, 2005, Revised Selected Papers, pages 58–71, 2005. (Cité en pages 8, 10, 12, 15, 16, 17 et 119.)

[Dunn 1973] J.C. Dunn. *A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters.* Journal of Cybernetics, vol. 3, pages 32–57, 1973. (Cité en page 51.)

[Felfernig 2007] Alexander Felfernig, Gerhard Friedrich and Lars Schmidt-Thieme. *Guest Editors' Introduction : Recommender Systems.* IEEE Intelligent Systems, vol. 22, no. 3, pages 18–21, 2007. (Cité en page 83.)

[Herlocker 2001] J.L. Herlocker and J.A. Konstan. *Content-independent task-focused recommendation.* Internet Computing, IEEE, vol. 5, no. 6, pages 40 –47, 2001. (Cité en page 84.)

[Herlocker 2004] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen and John T. Riedl. *Evaluating collaborative filtering recommender systems.* ACM Trans. Inf. Syst., vol. 22, pages 5–53, January 2004. (Cité en page 84.)

[Jiang 1997] Jay J. Jiang and David W. Conrath. *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy.* The Computing Research Repository, CoRR, vol. cmp-lg/9709008, 1997. (Cité en page 37.)

[Jursic 2008] Matjaz Jursic and Nada Lavrac. *Fuzzy clustering of documents.* In Conference on Data Mining and Data Warehouses, SiKDD 2008, Ljubljana, Slovenia, 2008. (Cité en page 51.)

[Lausen 2007] Holger Lausen and Joel Farrell. *Semantic Annotations for WSDL and XML Schema.* W3C recommendation, W3C, Août 2007. http ://www.w3.org/TR/2007/REC-sawsdl-20070828/. (Cité en pages 8 et 32.)

[Lin 1998] Dekang Lin. *An Information-Theoretic Definition of Similarity.* In The Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconson, USA, July 24-27, 1998, pages 296–304, 1998. (Cité en page 37.)

[Linden 2001] Gregory D. Linden, Jennifer A. Jacobi and Eric A. Benson. *Collaborative recommendations using item-to-item similarity mappings*, July 2001. (Cité en page 84.)

[Lopez 2003] Pierre Lopez. *Cours de graphes.* Rapport technique, LAAS-CNRS, Avril 2003. (Cité en page 45.)

[Maamar 2007] Z. Maamar, M. Lahkim, D. Benslimane, P. Thiran and S. Sattanathan. *Web Services Communities - Concepts & Operations* -. In The 3rd International Conference on Web Information Systems and Technologies (WEBIST'2007), 2007. (Cité en page 44.)

[Maamar 2009] Zakaria Maamar, Subramanian Sattanathan, Philippe Thiran, Djamal Benslimane and Jamal Bentahar. *An Approach to Engineer Communities of Web Services - concepts, architecture, operation, and deployment.* International Journal of E-Business Research (IJEBR), vol. 9, no. 4, Décembre 2009. (Cité en pages 18, 21, 22 et 23.)

[Manikrao 2005] Umardand Shripad Manikrao and T. V. Prabhakar. *Dynamic Selection of Web Services with Recommendation System.* In NWESP '05 : Proceedings of the International Conference on Next Generation Web Services Practices, page 117, Washington, DC, USA, 2005. IEEE Computer Society. (Cité en pages 24 et 27.)

[Manning 2008] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. Introduction to information retrieval. Cambridge University Press, Cambridge, UK, 2008. (Cité en page 85.)

[Martin 2004a] David Martin*and al. OWL-S : Semantic Markup for Web Services.* Rapport technique, 2004. (Cité en pages 12, 32 et 121.)

[Martin 2004b] David Martin, Mark Burstein, Erry Hobbs, Ora Lassila, Drew Mcdermott, Sheila Mcilraith, Srini Narayanan, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan and Katia Sycara. *OWL-S : Semantic Markup for Web Services.* Rapport technique, Novembre 2004. (Cité en page 8.)

[Maximilien 2004] E. Michael Maximilien and Munindar P. Singh. *A Framework and Ontology for Dynamic Web Services Selection.* IEEE Internet Computing, vol. 8, no. 5, pages 84–93, 2004. (Cité en page 8.)

[McConnell 2008] Jeffrey J. McConnell. Analysis of algorithms : an active learning approach. Jones and Bartlett publishers, 2008. (Cité en page 45.)

[McGuinness 2004] Deborah L. McGuinness and Frank van Harmelen. *OWL Web Ontology Language Overview.* W3C recommendation, W3C, Février 2004. http ://www.w3.org/TR/2004/REC-owl-features-20040210/. (Cité en page 12.)

[Medjahed 2005] Brahim Medjahed and Athman Bouguettaya. *A Dynamic Foundational Architecture for Semantic Web Services.* Distributed and Parallel Databases, vol. 17, no. 2, pages 179–206, 2005. (Cité en pages 18, 20, 22 et 23.)

[Mehta 2004] Bhaskar Mehta, Claudia Niederï¿½e, Avare Stewart, Claudio Muscogiuri and Erich J. Neuhold. *An Architecture for Recommendation Based Service Mediation.* In Semantics for Grid Databases, First International IFIP Conference on Semantics of a Networked World : ICSNW 2004, Paris, France, June 17-19, 2004. Revised Selected Papers, pages 250–262, 2004. (Cité en pages 24 et 27.)

[Mitra 2007] Nilo Mitra and Yves Lafon. *SOAP Version 1.2 Part 0 : Primer (Second Edition).* Rapport technique, W3C, Avril 2007. http ://www.w3.org/TR/2007/REC-soap12-part0-20070427/. (Cité en page 1.)

[Moreau 2007] Jean-Jacques Moreau, Sanjiva Weerawarana, Roberto Chinnici and Arthur Ryman. *Web Services Description Language (WSDL) Version 2.0 Part 1 : Core Language.* W3C recommendation, W3C, Juin 2007. http ://www.w3.org/TR/2007/REC-wsdl20-20070626. (Cité en pages 1 et 33.)

[Nguyen 2010] Ngoc Chan Nguyen, Walid Gaaloul and Samir Tata. *Collaborative Filtering Technique for Web Service Recommendation Based on User-Operation Combination.* In On the Move to Meaningful Internet Systems : OTM 2010 - Confederated International Conferences : CoopIS, IS, DOA and

ODBASE, Hersonissos, Crete, Greece, October 25-29, 2010, Proceedings, Part I, pages 222–239, 2010. (Cité en pages 2, 75, 92 et 122.)

[Nguyen 2011] Ngoc Chan Nguyen, Walid Gaaloul and Samir Tata. *A Web Service Recommender System Using Vector Space Model and Latent Semantic Indexing*. In 25th IEEE International Conference on Advanced Information Networking and Applications, AINA 2011, Biopolis, Singapore, March 22-25, 2011, pages 602–609, 2011. (Cité en pages 2, 75, 92 et 122.)

[Nie 2010] Feiping Nie, Chris Ding, Dijun Luo and Heng Huang. *Improved Min-Max Cut Graph Clustering with Nonnegative Relaxation*. In Josï¿½ L. Balcázar, Francesco Bonchi, Aristides Gionis and Michï¿½le Sebag, editeurs, ECML/PKDD (2), volume 6322 of *Lecture Notes in Computer Science*, pages 451–466. Springer, 2010. (Cité en page 69.)

[Oh 2006] Seog-Chan Oh, Hyunyoung Kil, Dongwon Lee and Soundar R. T. Kumara. *WSBen : A Web Services Discovery and Composition Benchmark*. In 2006 IEEE International Conference on Web Services (ICWS 2006), 18-22 September 2006, Chicago, Illinois, USA, 2006. (Cité en page 95.)

[Paik 2002] Hye-Young Paik, Boualem Benatallah and Rachid Hamadi. *Dynamic Restructuring of E-Catalog Communities Based on User Interaction Patterns*. World Wide Web, vol. 5, no. 4, pages 325–366, 2002. (Cité en pages 19, 22 et 23.)

[Paik 2004] Hye-Young Paik, Boualem Benatallah and Farouk Toumani. *WS-CatalogNet : Building Peer-to-Peer e-Catalog*. In Henning Christiansen, Mohand-Said Hacid, Troels Andreasen and Henrik Legind Larsen, editeurs, FQAS, volume 3055 of *Lecture Notes in Computer Science*, pages 256–269. Springer, 2004. (Cité en page 18.)

[Paik 2005] Hye-Young Paik, Boualem Benatallah and Farouk Toumani. *Toward self-organizing service communities*. IEEE Transactions on Systems, Man, and Cybernetics, Part A, vol. 35, no. 3, pages 408–419, 2005. (Cité en pages 18 et 22.)

[Papazoglou 2003] Mike P. Papazoglou, Bernd J. Krämer and Jian Yang. *Leveraging Web-Services and Peer-to-Peer Networks*. In Advanced Information Systems Engineering, 15th International Conference, CAiSE 2003, Klagenfurt, Austria, June 16-18, 2003, Proceedings, Lecture Notes in Computer Science, pages 485–501. Springer, 2003. (Cité en page 17.)

[Perugini 2004] Saverio Perugini, Marcos A. Gonzalves and Edward A. Fox. *Recommender Systems Research : A Connection-Centric Survey*. Journal of Intelligent Information Systems, vol. 23, no. 2, pages 107–143, September 2004. (Cité en page 83.)

[Pilioura 2004] Thomi Pilioura, Georgios-Dimitrios Kapos and Aphrodite Tsalgatidou. *PYRAMID-S : A Scalable Infrastructure for Semantic Web Service Publication and Discovery*. In 14th International Workshop on Research

Issues in Data Engineering (RIDE-WS-ECEG 2004), Web Services for E-Commerce and E-Government Applications, 28-29 March 2004, Boston, MA, USA, pages 15–22, 2004. (Cité en pages 8, 9, 15, 18 et 119.)

[Pilioura 2009] Thomi Pilioura and Aphrodite Tsalgatidou. *Unified publication and discovery of semantic Web services.* ACM Transactions on the Web (TWEB), vol. 3, no. 3, 2009. (Cité en pages 8, 9, 11, 17 et 119.)

[Qian 2004] Gang Qian, Shamik Sural, Yuelong Gu and Sakti Pramanik. *Similarity between Euclidean and cosine angle distance for nearest neighbor queries.* In Proceedings of the 2004 ACM symposium on Applied computing, 2004. (Cité en page 53.)

[Ran 2003] Shuping Ran. *A model for web services discovery with QoS.* SIGecom Exch., vol. 4, no. 1, pages 1–10, 2003. (Cité en page 8.)

[Rao 2010] Vuda Sreenivasa Rao and Dr. S Vidyavathi. *COMPARATIVE INVESTIGATIONS AND PERFORMANCE ANALYSIS OF FCM AND MFPCM ALGORITHMS ON IRIS DATA.* Indian Journal of Computer Science and Engineering, vol. 1, no. 2, pages 145–151, 2010. (Cité en page 55.)

[Ratnasamy 2001] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard M. Karp and Scott Shenker. *A scalable content-addressable network.* In SIGCOMM, pages 161–172, 2001. (Cité en page 16.)

[Resnick 1997] Paul Resnick and Hal R. Varian. *Recommender systems.* Commun. ACM, vol. 40, no. 3, pages 56–58, March 1997. (Cité en page 83.)

[Resnik 1995] Philip Resnik. *Using Information Content to Evaluate Semantic Similarity in a Taxonomy.* In The 1995 International Joint Conference on AI, IJCAI-95, 20-25 August 1995,Montreal/Quebec, Canada, pages 448–453, 1995. (Cité en page 37.)

[Roman 2005] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Cristoph Bussler and Dieter Fensel. *Web Service Modeling Ontology.* Appl. Ontol., vol. 1, pages 77–106, January 2005. (Cité en page 8.)

[Salton 1975] G. Salton, A. Wong and C. S. Yang. *A vector space model for automatic indexing.* Commun. ACM, vol. 18, no. 11, pages 613–620, 1975. (Cité en pages 25 et 85.)

[Salton 1987] Gerard Salton and Chris Buckley. *Term Weighting Approaches in Automatic Text Retrieval.* Rapport technique, 1987. (Cité en page 53.)

[Saraçoglu 2007] RIdvan Saraçoglu, Kemal Tutuncu and Novruz Allahverdi. *A fuzzy clustering approach for finding similar documents using a novel similarity measure.* Expert Systems with Applications, vol. 33, no. 3, pages 600 – 605, 2007. (Cité en page 51.)

[Sarwar 2001] Badrul Sarwar, George Karypis, Joseph Konstan and John Reidl. *Item-based collaborative filtering recommendation algorithms.* In WWW '01 : Proceedings of the 10th international conference on World Wide Web, pages 285–295, New York, NY, USA, 2001. ACM. (Cité en page 84.)

[Seco 2004] Nuno Seco, Tony Veale and Jer Hayes. *An Intrinsic Information Content Metric for Semantic Similarity in WordNet.* In Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004, 2004. (Cité en page 25.)

[Sellami 2009] Mohamed Sellami, Samir Tata, Zakaria Maamar and Bruno Defude. *A Recommender System for Web Services Discovery in a Distributed Registry Environment.* In Fourth International Conference on Internet and Web Applications and Services, ICIW 2009, 24-28 May, Venice, Italy, pages 418–423, 2009. (Cité en page 91.)

[Sellami 2010a] Mohamed Sellami, Olfa Bouchaala, Walid Gaaloul and Samir Tata. *WSRD : A Web Services Registry Description.* In The 10th international conference on New Technologies of Distributed Systems, NOTERE 2010, May 31- June 2, Tozeur, Tunisia, pages 89–96, 2010. (Cité en pages 32 et 56.)

[Sellami 2010b] Mohamed Sellami, Walid Gaaloul and Samir Tata. *Functionality-Driven Clustering of Web Service Registries.* In IEEE International Conference on Services Computing, SCC 2010, Miami, Florida, USA, 2010. (Cité en page 56.)

[Sellami 2010c] Mohamed Sellami, Walid Gaaloul, Samir Tata and Mohamed Jmaiel. *Using Recommendation to Limit Search Space in Web Services Discovery.* In 24th IEEE International Conference on Advanced Information Networking and Applications, AINA 2010, Perth, Australia, 20-13 April 2010, pages 974–981, 2010. (Cité en page 91.)

[Sellami 2011a] Mohamed Sellami, Walid Gaaloul and Samir Tata. *An Implicit Approach for Building Communities of Web Service Registries.* In iiWAS'2011 - The 13th International Conference on Information Integration and Web-based Applications and Services, 5-7 December, 2011, Ho Chi Minh City, Vietnam. ACM, 2011. (Cité en page 56.)

[Sellami 2011b] Mohamed Sellami, Walid Gaaloul and Samir Tata. *Implementation of Communities of Web Service Registries.* In IEEE International Conference on Web Services, ICWS 2011, Washington DC, USA, July 4-9, 2011. IEEE Computer Society, 2011. (Cité en pages 44 et 56.)

[Sivashanmugam 2003] Kaarthik Sivashanmugam, Kunal Verma, Amit P. Sheth and John A. Miller. *Adding Semantics to Web Services Standards.* In Proceedings of the International Conference on Web Services, ICWS '03, June 23 - 26, 2003, Las Vegas, Nevada, USA, pages 395–401, 2003. (Cité en page 15.)

[Sivashanmugam 2004] Kaarthik Sivashanmugam, Kunal Verma and Amit Sheth. *Discovery of Web Services in a Federated Registry Environment.* In ICWS '04 : Proceedings of the IEEE International Conference on Web Services, page 270, Washington, DC, USA, 2004. IEEE Computer Society. (Cité en pages 8, 9, 16, 17 et 119.)

[Subramanian 2007] Sattanathan Subramanian, Philippe Thiran, Zakaria Maamar and Djamal Benslimane. *Engineering Communities of Web Services*. In Gabriele Kotsis, David Taniar, Eric Pardede and Ismail Khalil Ibrahim, editeurs, iiWAS, volume 229 of *books@ocg.at*, pages 57–66. Austrian Computer Society, 2007. (Cité en page 23.)

[Terveen 1997] Loren Terveen, Will Hill, Brian Amento, David Mcdonald and Josh Creter. *PHOAKS : a system for sharing recommendations*. Commun. ACM, vol. 40, no. 3, pages 59–62, 1997. (Cité en page 83.)

[The Mckinsey Quarterly 2007] The Mckinsey Quarterly. *How businesses are using Web 2.0 : A McKinsey Global Survey*, 2007. (Cité en pages 1 et 7.)

[Traversat 2003] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J.-C. Hugly, E. Pouyoul and B. Yeager. *Project JXTA 2.0 Super-Peer Virtual Network*. Rapport technique, Sun Microsystems, 2003. (Cité en page 103.)

[van Eemeren 1996] Frans H. van Eemeren, Rob F. Grootendorst and Francisca S. Henkemans. Fundamentals of Argumentation Theory : A Handbook of Historical Backgrounds and Contemporary Applications. Lawrence Erlbaum Associates, 1996. (Cité en page 22.)

[Verma 2005] Kunal Verma, Kaarthik Sivashanmugam, Amit Sheth, Abhijit Patil, Swapna Oundhakar and John Miller. *METEOR-S WSDI : A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services*. Inf. Technol. and Management, vol. 6, no. 1, pages 17–39, 2005. (Cité en pages 8, 9, 10, 15, 16, 17 et 119.)

[Werthner 2007] H. Werthner, H.R. Hansen and F. Ricci. *Recommender Systems*. In System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on, page 167, 2007. (Cité en page 83.)

[Wives 2010] Leandro Krug Wives, Josï¿½ Palazzo Moreira de Oliveira, Zakaria Maamar, Samir Tata and Mohamed Sellami. *Web Services & Recommender Systems - A Research Roadmap*. In WEBIST 2010, Proceedings of the 6th International Conference on Web Information Systems and Technologies, Volume 1, Valencia, Spain, April 7-10, 2010, pages 119–124, 2010. (Cité en page 91.)

[Xu 2007a] Bin Xu and Dewei Chen. *Semantic Web Services Discovery in P2P Environment*. In 2007 International Conference on Parallel Processing Workshops (ICPP Workshops 2007), 10-14 September 2007, Xi-An, China, page 60, 2007. (Cité en pages 8, 13, 14, 15, 16 et 17.)

[Xu 2007b] Ziqiang Xu, Patrick Martin, Wendy Powley and Farhana H. Zulkernine. *Reputation-Enhanced QoS-based Web Services Discovery*. In IEEE International Conference on Web Services, July 9-13, 2007, Salt Lake City, Utah, USA, pages 249–256, 2007. (Cité en page 8.)

[Zadeh 1965] L.A. Zadeh. *Fuzzy Sets*. Information Control, vol. 8, pages 338–353, 1965. (Cité en page 51.)

[Zhang 2007] Cheng Zhang and Yanbo Han. *Service Recommendation with Adaptive User Interests Modeling*. In Distributed Computing and Internet Technology, 4th International Conference, ICDCIT 2007, Bangalore, India, December 17-20, pages 265–270, 2007. (Cité en pages 26 et 27.)

[Zhou 2011a] Zhangbing Zhou, Mohamed Sellami, Walid Gaaloul and Bruno Defude. *Clustering and Managing Data Providing Services Using Machine Learning Technique*. In International Conference on Semantics, Knowledge and Grid (SKG 2011), 24-26 October 2011, Bejing, China. IEEE Computer Society, 2011. (Cité en page 122.)

[Zhou 2011b] Zhangbing Zhou, Mohamed Sellami, Walid Gaaloul and Bruno Defude. *Data Providing Services Clustering and Management for Facilitating Service Discovery and Replacement*. Rapport technique, CNRS SIMBAD, Telecom SudParis, 2011. http ://www-inf.it-sudparis.eu/ sellami/TR.pdf. (Cité en page 122.)