الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

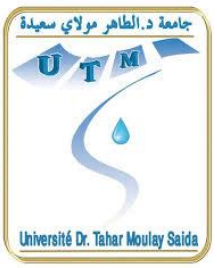
وزارة التعليم العالي و البحث العلمي

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Université Dr. Tahar Moulay SAIDA **جامعة د. الطاهر مولاي سعيدة**

Faculté : Technologie **كلية : التكنولوجيا**

Département : Informatique **قسم : الإعلام الآلي**



MEMOIRE DE MASTER

Option : RISR

THEME

Implémentation d'un système d'intégration de source hétérogène par service web

Présenté par :

Mokhtari Ikhlas

Kada Fethia

Encadré par :

Mr :Benyahia Kadda

2015 – 2016

Remerciement

Mon remerciement va en premier lieu à ALLAH le tout puissant de
M'avoir donné la foi et de m'avoir permis d'en arriver là.
Je tiens à remercier particulièrement mon encadreur
« MR Kadda Benyahiya » *pour son encadrement et pour*
L'intérêt qu'il a manifesté à mon travail.

Je remercierais très sincèrement les membres de jury d'avoir bien
Voulu d'accepter de faire partie de la commission d'examinateur
J'adresse également mes remerciements, à tous mes enseignants,
Qu'ils ont consentis pour nous permettre de suivre mes études dans les
Meilleures conditions possibles et n'avoir jamais cessez
De m'encourager tout au long de mes années d'étude.

Je tiens également à remercier tous mes collègues de promotion
Que j'ai eu le plaisir de les côtoyer pendant cette
période de formation.

Je remercierais tous ceux qui ont contribué de près ou de lion à
La réalisation de ce travail.

Dédicace

Du profond de mon cœur, je dédie ce travail à tous ceux qui me sont chers

Je dédie cet ouvrage qui m'est très précieux à mes très chers parents qui ont fait tout leur possible pour m'assister, me soutenir durant toute ma formation et m'avoir ainsi permis d'arriver à la réussite. Que Dieu me les gardent.

A mes chers soeurs: Imane,hiba hanaa .
A mes chers frères :houssem eddine,Yacine.

A mes tantes , mes oncles, mes cousins et cousines.

A mon encadreur Mr Kadda Benyahya
Que Dieu exauce ses voeux les plus chers.

A ma chère binôme Kada Fethia , je la remercie infiniment.

Ainsi mes chers amies :fatima et aicha et asma.

Ikhlas

Dédicace

Je dédie ce modeste mémoire :

A mon père , qui m'a toujours poussé et motivé dans mes études., et qui est prêt à leur sacrifier pour le bonheur de leurs enfants, témoignage d'affection et de grande reconnaissance, de m'avoir permis d'être ce que je suis, , d'entreprendre et de réussir, grâce à son soutien constant et son affection permanente. Que Dieu le garde et le protéger pour moi.

A mes chers soeurs: Houria,Hasnia et Zahira .
A mes chers frères :Abdelhak ,Fethi.

A mes tantes , mes oncles, mes cousins et cousines.
A mon encadreur Mr Kadda Benyahya
Que Dieu exauce ses voeux les plus chers.

A ma chère binôme Mikhtari Ikhlas , je la remercie infiniment.

Fethia

Résumé

La diversité des sources d'information distribuées et leur hétérogénéité est une des principales caractéristiques du Web aujourd'hui .Dans les environnements faiblement couplés (par exemple eHealth, eGov… etc.) l'accès à un nombre important de sources de données se fait par des services web qui permettent d'accéder et de rechercher des données élémentaires tenues par des sources hétérogènes autonomes indépendamment des systèmes et des plates-formes utilisées.

Ce type services est appelé Services Web DaaS (Data-as-a-Service).

Mots-clés : Services Web DaaS, Services Web d'accès aux données, composition des services Web, intégration des donnés.

Abstract

The diversity of sources of information distributed and heterogeneity is a leading Web features today .In loosely coupled environments (eg eHealth , eGov ... etc.) access to a large number of data sources is by web services that provide access and search data items held by autonomous heterogeneous sources independently of systems and platforms used.

This type is called web services DaaS (Data as a Service).

Keywords: DaaS Services Web, Web services data access, Web services composition data integration.

ملخص

نتحدث من خلال هده المذكرة على نوع من خدمات الويب التي تهتم بالبحث عن المعطيات الموجودة في قواعد بيانات مختلفة و موزعة . ان الاتصال و طلب الخدمة من الويب سيرفس ينتج عنه طلب معلومات من مصادر البيانات ، و هو ما يحتاج في الغالب الى تركيب و ادماج مجموعة من خدمات(الويب سيرفس).

عملنا في هذه المذكرة يتمثل في البحت عن معطيات متفرقة ، و تركيب خدمات الويب وفق انماط تسمح بالاستجابة لحاجة المستخدمين .هدا النوع من الخدمات يسمي خدمات الويب داس (البيانات كخدمة)

كلمات البحث خدمات الويب ، تركيب خدمات الويب ، خدمات الويب الوصول الى البيانات ، ادماج البيانات.

s

Sommaire

Introduction générale :

Les dernières décennies ont été marquées par le développement rapide des systèmes d'information distribués, et tout particulièrement par la démocratisation de l'accès à Internet. Cette évolution du monde informatique a entraîné le développement de nouveaux paradigmes d'interaction entre applications. Notamment, l'architecture orientée service (Service Oriented Architecture, ou SOA) a été mise en avant afin de permettre des interactions entre applications distantes. Cette architecture est construite autour de la notion de service, qui est matérialisé par un composant logiciel assurant une fonctionnalité particulière et accessible via son interface. Un service est toujours accompagné d'une description fournissant aux applications les informations nécessaires à son utilisation.

Les services sont implantés par les fournisseurs, qui mettent à disposition les descriptions de services sous forme de fichiers. Ces descriptions sont centralisées et stockées dans des annuaires. La notion d'annuaire est comparable aux annuaires téléphoniques que nous utilisons pour accéder à des personnes ou à des services. Les applications clientes envoient des requêtes aux annuaires pour sélectionner les services, de la même manière que nous recherchons un numéro dans un annuaire téléphonique. Elles téléchargent ensuite les descriptions des services sélectionnés et invoquent directement.

L'objectif de la notion de service est de promouvoir un accès simple et rapide aux fonctionnalités mises à disposition par les organisations. Aussi, la vision des services en tant que composants logiciels indépendants met en exergue les possibilités de coordination, ou composition, de plusieurs services pour fournir des fonctionnalités avancées. Afin de faciliter l'utilisation et la composition des services, les applications doivent réaliser des interactions faiblement couplées, c'est-à-dire qu'elles doivent être capables de fournir et utiliser des services tout en restant indépendantes les unes des autres, et sans divulguer leur fonctionnement interne. Cette exigence est satisfaite par l'adoption de protocoles et langages standardisés qui fournissent un es uniforme aux services et à leurs descriptions.

Dans des travaux de recherche, les services DaaS sont appelés : IaaS Services (Information-as-a-Service) Services, DaaS (Data-as- a-Service) Services, Data-Providing Services, Stateless Services, Information-Providing Services, ou simplement Data Services.

Les DaaS sont différents des Services web traditionnels du fait leurs invocation retourne seulement des données seulement et ils n'ont pas des effets qui peuvent changer l'état du monde. Beaucoup de domaines d'application comme la bioinformatique et les soins de santé ont largement utilisés des services web DaaS pour le partage de données.

Problématique :

La diversité des sources d'information distribuées et leur hétérogénéité est une des principales caractéristiques du Web d'aujourd'hui. Les Services Web peuvent résoudre le problème de cette hétérogénéité.

L'invocation d'un Service web DaaS a comme conséquence l'exécution d'une requête au-dessus des sources de données. Dans la plupart des cas, les requêtes des utilisateurs exigent la composition de plusieurs services.

Contribution

Notre travail vise la consultation de sources de données hétérogènes en utilisant les services web DaaS.

Nous avons créé un ensemble de sources de données hétérogènes, puis nous les avons consultées en utilisant les services web d'accès aux données. Les bases de données construites sont : Microsoft SQL Server, Microsoft Accès et MySQL server.

Ce mémoire est organisé en trois chapitres :

- **Chapitre 01 :** est consacré aux notions fondamentales des Services Web ou nous avons défini les standards de description, de publication et d'invocation de Services Web, la description de l'architecture orienté services SOA et de la technologie des services web ainsi les langages qui présentent les différentes couches de l'architecture de référence des services web seront décrits, l'architecture étendue services web qui a pour rôle d'enrichir les services web pour supporter les fonctionnalités liés à la sécurité.

- **Chapitre 02 : nous** allons présenter un Etat de l'art sur les services Web DaaS (Data-as-a-Service), Ensuite nous discutons sur l'intégration de données hétérogènes, aussi nous allons présenter quelques travaux relatifs aux services web DaaS.

- **Chapitre 03 :** Conception et Implémentation.

- Nous conclurons ce mémoire en présentant un certain nombre de perspectives de recherches que nous jugeons utiles à entreprendre.

Chapitre I :

Les Services web

1.1. Introduction :

Les Services Web prennent leur origine dans l'informatique distribuée et dans l'avènement du Web. Ils poursuivent un vieux rêve de l'informatique distribuée où les applications pourraient inter opérer à travers le réseau, indépendamment de leur plateforme et de leur langage d'implémentation. Les standards conçus pour la programmation distribuée les plus connus sont : CORBA (Common Object Request Broker Architecture) [1], RMI (RemoteMethod Invocation) et COM (Component Object Model) [2]. Ces modèles sont complexes, peu compatibles, et difficilement interopérables entre eux. Ils restent donc souvent utilisés en Intranet [3].

Les Services Web regroupent tout un ensemble de technologies bâties sur des standards. Ils permettent de créer des composants logiciels distribués, de décrire leur interface et de les utiliser indépendamment de la plateforme sur laquelle ils sont implémentés.

1.2. Définitions :

Les Services Web sont une instance de l'architecture orientée service (SOA) [4], sur le Web. Ils ont été proposés initialement par IBM et Microsoft [5], puis en partie standardisés sous l'égide du W3C.

Selon IBM [6] : « *Les Services Web sont auto contenus, applications modulaires, accessibles via le Web à travers des langages standards et ouverts, qui fournissent un ensemble de fonctionnalités pour les entreprises ou les individus* ». Cette définition met L'accent sur deux points. Le premier point est qu'un Service Web est vu comme une application accessible pour les autres applications à travers le Web. Le second point, les Services Web sont ouverts, ce qui signifie que les services publient des interfaces qui peuvent être invoquées par le biais de messages standards.

Une autre définition est celle donnée par le W3C [7] : « *Un Service Web est un système logiciel identifié par un URI , dont les interfaces publiques et leurs liaisons sont décrites en utilisant XML [8]. Sa définition peut être découverte par les autres systèmes logiciels.*

Ces systèmes peuvent alors interagir avec le Service Web de la manière décrite dans sa définition, en utilisant des messages basés sur XML et transmis par des protocoles Internet ».

1.3. Caractéristiques des Services Web :

Les Services Web possèdent les caractéristiques suivantes qui leur permettent une meilleure intégration dans les environnements hétérogènes :

- **Basé sur XML :** Les données dans les protocoles et les technologies des Services Web sont représentées en utilisant XML, ces technologies peuvent être interopérables. Comme un transport de données, XML élimine toute dépendance de gestion de réseau, du système d'exploitation, ou de la plateforme liée à un protocole.

- **Faiblement couplés :** Dans le développement de logiciels, le couplage se rapporte typiquement au degré de dépendance entre les composants/modules logiciels. Contrairement aux composants fortement couplés (tels que CORBA ou COM), les Services Web sont autonomes et peuvent fonctionner indépendamment les uns des autres. Il n'est pas nécessaire de connaitre la machine, le langage ou le système d'exploitation.

- **Auto-descriptif :** Les Services Web ont la capacité de se décrire d'une manière qui peut être facilement reconnu. Ainsi, l'interface, les informations de localisation et l'accès au Service Web est identifié par n'importe quelle application externe.

- **Modulaire :** Les Services Web fonctionnent de manière modulaire. Cela signifie qu'au lieu d'intégrer dans une seule application globale toutes les fonctionnalités, on crée plusieurs applications spécifiques et on les fait inter-opérer entre elles, et qui définissent chacune, une de ses fonctionnalités.

- **Réutilisable :** Une fonctionnalité, développée sous forme de Service Web, peut être réutilisée et combinée à d'autres fonctionnalités afin de composer de nouveaux services.

1.4. Architecture des Services Web :

1.4.1. Architecture de référence :

Pour promouvoir l'interopérabilité et l'extensibilité du paradigme des Services Web, une architecture de référence est nécessaire afin de préserver les objectifs initiaux visés par les Services Web.

L'Architecture SOA [4] est un modèle abstrait qui consiste à diviser un logiciel répondant à un problème, en un ensemble d'entités proposant des services. Chacune de ces entités peut utiliser les services proposés par d'autres entités. On obtient ainsi un réseau de services interagissant entre eux [9].

L'architecture SOA vise trois objectifs importants [10] :

- Identification des composants fonctionnels.
- Définition des relations entre ces composants.
- Etablissement d'un ensemble de contraintes sur chaque composant de manière à garantir les propriétés globales de l'architecture.

L'architecture de référence s'articule autour des trois rôles suivants [11] [12] :

- **Le fournisseur du service :** Correspond à la personne ou à l'organisation propriétaire du service. D'un point de vue technique, il est constitué par la plateforme d'accueil du service.

- **Le client :** Correspond au demandeur du service. D'un point de vue technique, il est constitué par l'application d'invocation du service. L'application client peut être elle-même un Service Web.

- **L'annuaire des services :** Il désigne l'entité logicielle qui joue le rôle de l'intermédiaire entre les clients et les fournisseurs de services. Il correspond à un registre de descriptions de services offrant des facilités de publication de services à l'intention des fournisseurs ainsi que des facilités de recherche de services à l'intention des clients.

Les interactions de base entre ces trois rôles incluent les opérations de publication, de découverte et d'invocation [12] [13]. Ce scénario est illustré par la Figure 1.1.

- **Publication :** Le fournisseur de service définit la description de son service et la publie dans un annuaire de service en vue d'être localisé par des clients.
- **Découverte :** Le client utilise les facilités de recherche disponibles au niveau de l'annuaire pour retrouver et sélectionner un service.
- **Invocation :** Le client examine ensuite la description du service sélectionné pour récupérer les informations lui permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré.

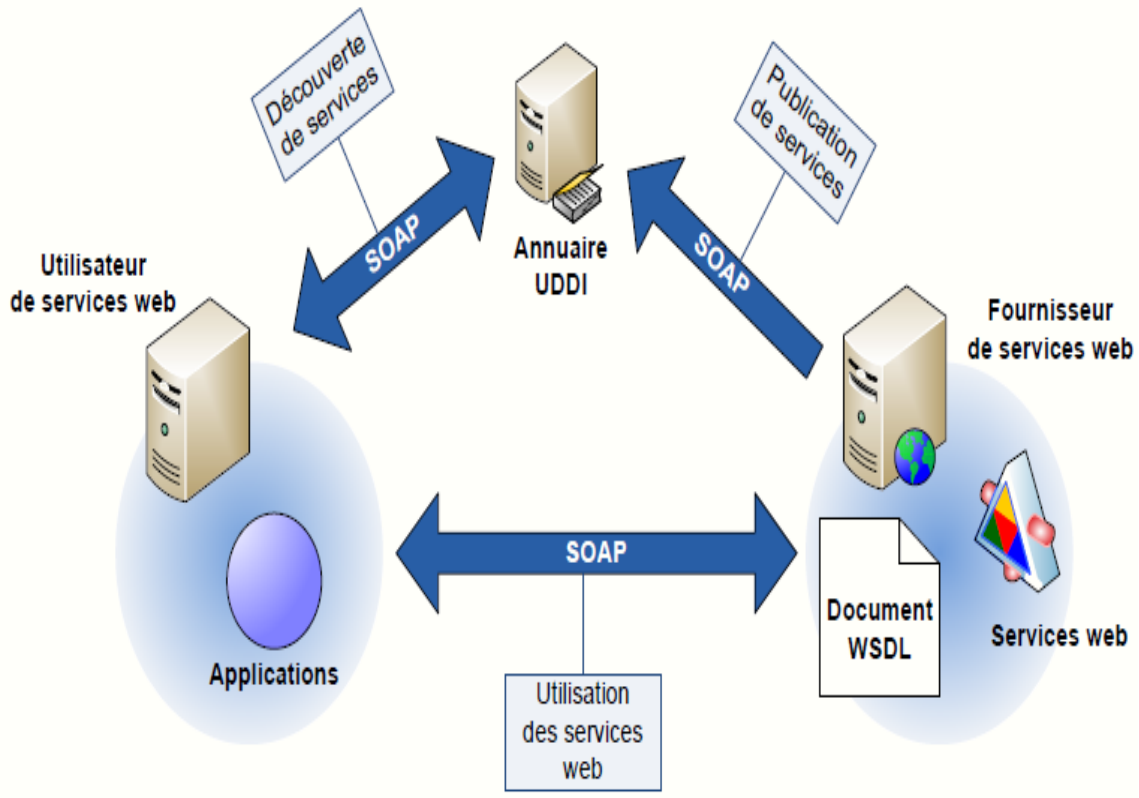


Fig 1.1 : Architecture de référence [14].

Pour garantir l'interopérabilité des trois opérations précédentes, des propositions de standards ont été élaborées pour chaque type d'interaction. Nous citons, notamment les standards suivants :

- **SOAP :** Un protocole permettant d'invoquer à distance les opérations offertes par un Service Web en utilisant des messages XML.

- **WSDL :** Un standard permettant de décrire l'interface d'un Service Web sous la forme d'un fichier de description en XML.
- **UDDI :** Un protocole d'annuaire permettant à la fois de publier et de retrouver un Service Web.

Cependant, cette architecture n'est pas suffisante pour permettre une utilisation effective des Services Web. Il existe de nombreux nouveaux standards émergents (tels que les standards de sécurité, d'administration et du Web sémantique) dans le domaine des Services Web que cette architecture ne peut pas aisément prendre en compte. Il s'avère donc nécessaire d'étendre l'architecture de base de Services Web.

1.4.2. Architecture Étendue :

Cette architecture étendue est aussi appelée *pile des Services Web*, du fait qu'elle est constituée de plusieurs couches se superposant les unes aux autres [12]. Elle utilise les couches standards de la première architecture en a joutant au-dessus d'autres couches spécifiques.

La Figure 1.2 est une proposition de vue globale simplifiée de la pile des Services Web[12].
Chaque couche de la pile répond à des préoccupations fonctionnelles différentes telles que la sécurité, les transactions, etc.

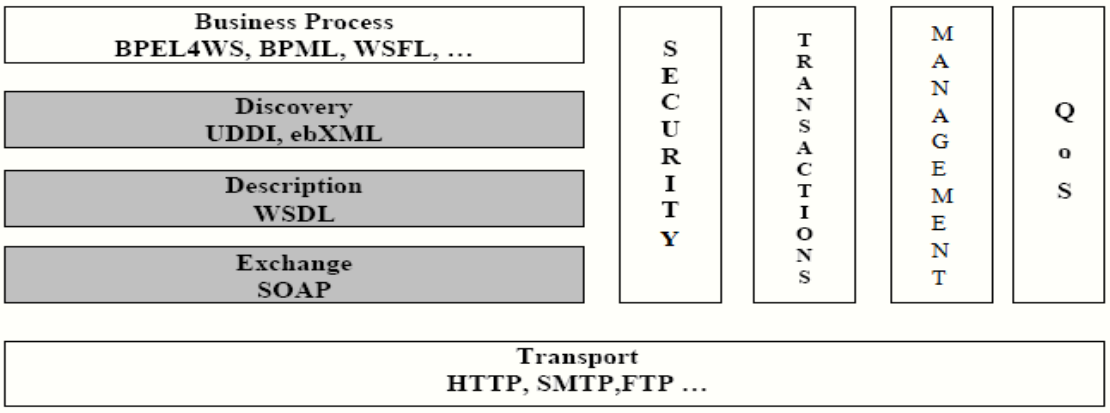


Fig 1.2 : Architecture en pile (étendue) [12].

La pile est constituée de plusieurs couches, chaque couche s'appuie sur un standard particulier. On retrouve, au-dessus de la couche de transport (HTTP, SMTP), les trois couches formant l'infrastructure de base décrite précédemment.

D'après cette architecture peut être décomposée en trois types de couches [12] :

- **L'infrastructure de base :** Elle est constituée de trois couches, ces couches s'appuient sur les standards des Services Web (SOAP, WSDL, UDDI). Elle définit le fondement technique de l'architecture de référence.
- **Les couches transversales [15] :** (sécurité, administration,..) Ce sont les couches qui rendent fiable l'utilisation effective des Services Web dans le monde industriel et dont on trouve toute les notions associées aux problématiques de sécurité et celles en rapport avec la QoS. La partie administration est un peu particulière, car il s'agit de mettre en place des enchainements de services et par conséquent de créer des Services Web composites. Des travaux tentent d'intégrer le Web sémantique dans ces couches transversales en ajoutant une couche verticale représentant le Web sémantique et étant utilisable par les quatre couches horizontales représentant les standards.
- **Business Process :** Cette couche permet l'intégration de Services Web, elle établit la représentation d'un Business Process comme un ensemble de Services Web. De plus, la description de l'utilisation de différents services composants ce service est disponible par l'intermédiaire de cette couche.

1.5. Les technologies des Services Web :

Services Web sont basés sur des technologies qui ont émergé comme standards Internet pour assurer l'interaction entre les opérations de recherche, de lien et de publication des Services Web. Un ensemble de spécifications considérées comme des standards ont été définies par le consortium W3C. Dans cette section, nous décrivons ces standards.

1.5.1. Le langage XML :

XML est un langage permettant la représentation de données ainsi que de documents structurés sans l'utilisation de balises prédéfinies, contrairement au langage HTML [8]. XML permet de créer ses propres balises selon le besoin. Ces balises n'ont

pas de signification pour le langage XML, mais elles ont un sens pour les applications qui les utilisent. Ainsi, ce langage peut être étendu de façon à y a jouter des balises spécialisées afin de décrire au mieux chaque type de données. Cette flexibilité confère à XML la popularité dont il jouit.

Historiquement, XML a été développé par le W3C en 1996, a profité des meilleurs aspects de SGML (Standard Generalized Markup Language) [16], et depuis 1998, une recommandation du W3C. Il est indépendant des plates-formes informatiques. Il est lisible par l'humain mais est destiné être lu par la machine.

1.5.2. SOAP :

Le protocole SOAP (*Simple Object Access Protocole*) est un standard proposé par le W3C pour l'échange des données [17]. Le but de SOAP est d'assurer l'interconnexion des Services Web en transportant les paquets de données encapsulés sous forme de texte structuré. Il repose sur deux standards HTTP et XML respectivement pour la structure des messages et pour le transport. Mais, il n'exclut pas l'utilisation d'autres protocoles de transport (SMTP, FTP, etc). Le protocole SOAP ne passe pas par des ports précis mais utilise le protocole HTTP qui lui, permet de traverser les proxys et les pare-feux (firewalls) contrairement à d'autres modes de communications telles que les communications via les sockets et RMI. Une application envoie une requête SOAP à un Service Web, et le Service Web retourne la réponse dans ce qu'on appelle une réponse SOAP.

- **Structure d'un message SOAP :**

Un message SOAP [18] présente une structure normalisée. Il est toujours constitué d'un élément racine, à savoir l'enveloppe (SOAP-ENV : *enveloppe*), qui contient un élément en-tête (SOAP-ENV : header) optionnel et un élément corps (SOAP-ENV : body) obligatoire, suivis d'éventuels éléments applicatifs spécifiques. La Figure 1.3 illustre les trois principaux éléments de SOAP.

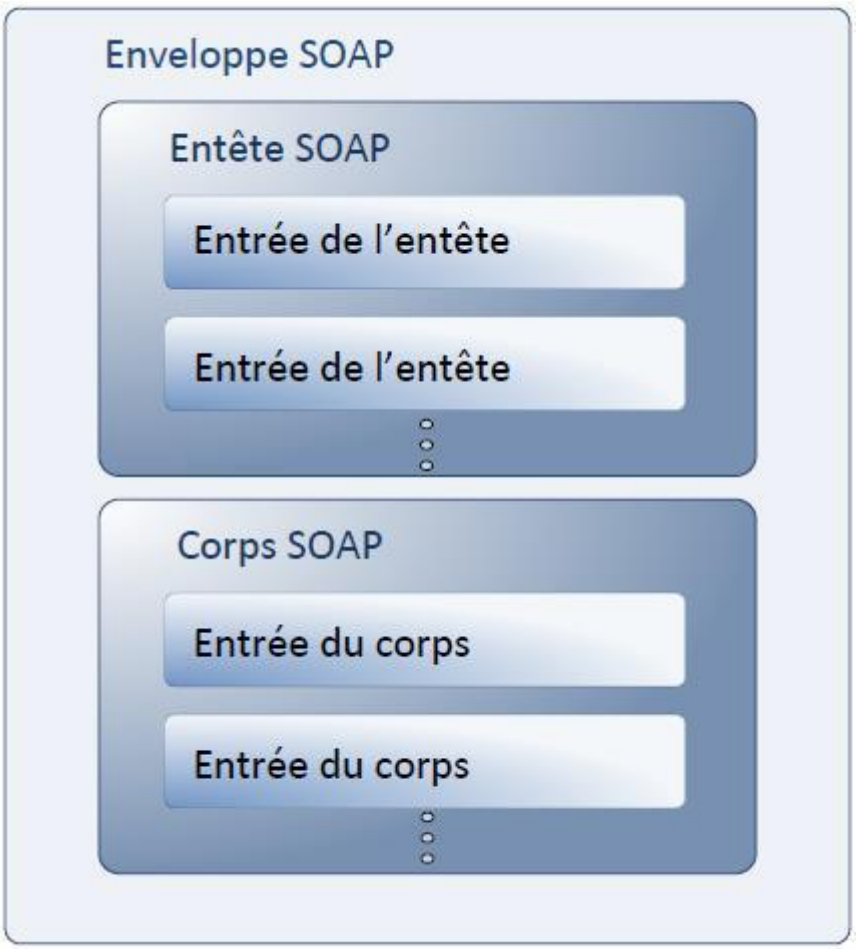


Fig 1.3 : Format d'un message SOAP [18].

1.5.3. WSDL :

Le langage WSDL [19] (*Services Web Description Language*) a été proposé initialement par IBM et Microsoft en 2001, WSDL est un langage de description syntaxique de Service Web sous format XML. Il est définit de façon indépendante du langage de développement :

- L'ensemble des opérations et des messages qui peuvent être transmis vers et depuis un Service Web donné.
- Les protocoles de communication et de transport utilisés.
- Les points d'accès au service.

Un document WSDL est constitué d'une suite d'éléments décrivant un Service Web sous forme d'un ensemble d'opérations exploitables depuis l'extérieur. Il est composé de deux niveaux :

- **Niveau abstrait :** constitué d'éléments qui définissent l'interface des Services Web tel que les types de données, les messages, les types de ports et cela de façon indépendante des protocoles utilisés.
- **Niveau concret :** qui permet de définir les protocoles utilisés par le Service Web et sa localisation.

- **Structure d'un document WSDL :**

  Une description WSDL d'un Service Web est représentée par la Figure 1.4.

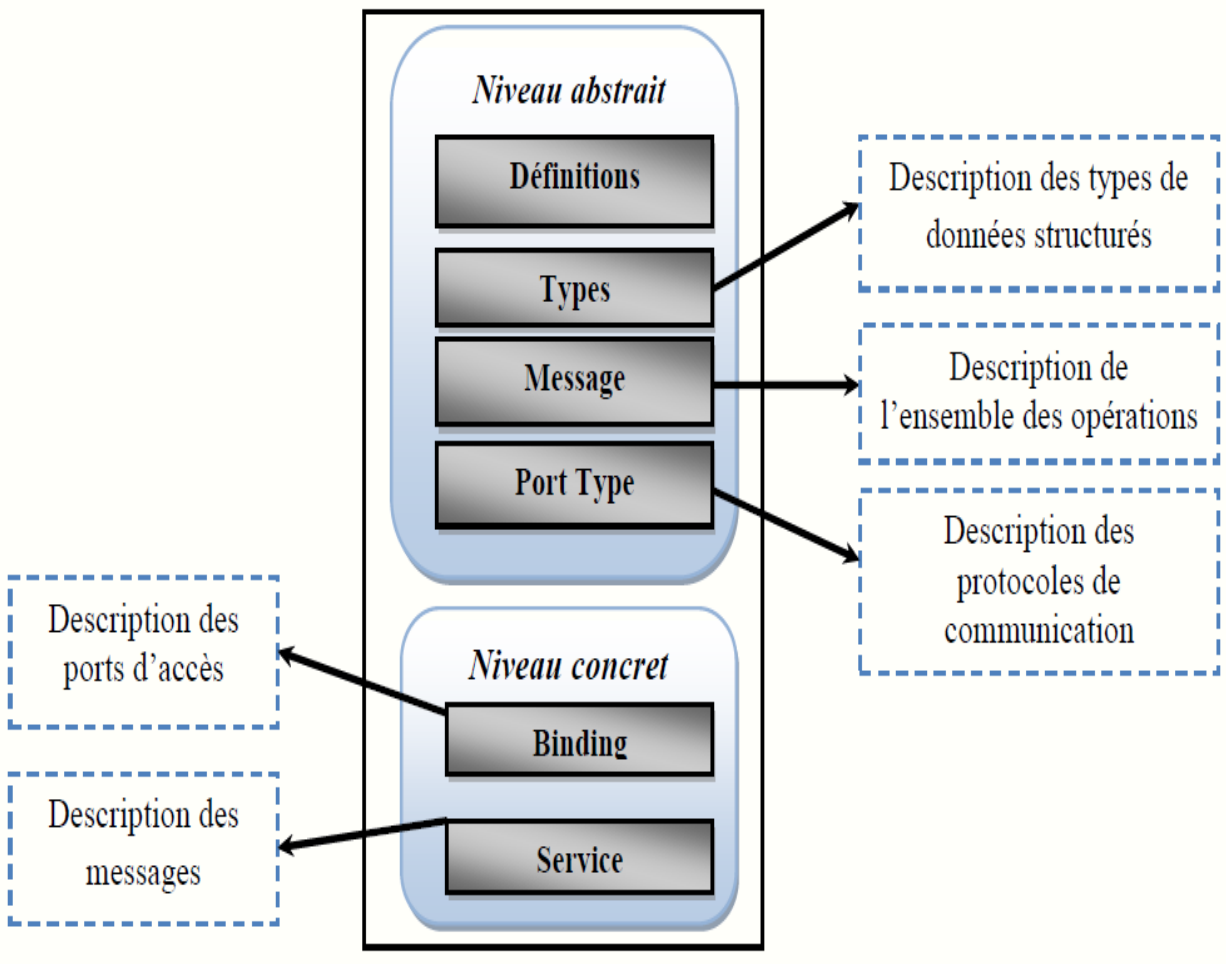


Fig 1.4 : Structure générale d'un document WSDL [19].

1.5.4. UDDI :

UDDI est introduit en 2000 par Ariba, Microsft et IBM [20]. Il a été créé pour faciliter la découverte et la recherche de Services Web en plus de leurs publications. Les organisations publient les informations décrivant leurs Services Web dans l'annuaire, et l'application client ayant besoin d'un certain service, consulte cet annuaire pour la recherche des informations concernant le Service Web qui fournit le service désiré pour une éventuelle interaction.

UDDI est subdivisé en deux parties principales : partie publication ou inscription, et partie découverte. La partie publication regroupe l'ensemble des informations relatives aux Entreprises et à leurs services. Ces informations sont introduites via une API d'enregistrement. La partie découverte facilite la recherche d'information contenue dans UDDI grâce à l'API SOAP.

- **Publication de Services Web avec UDDI :**

Les différents composants de la publication faite par UDDI sont des documents XML manipulant de l'information à propos du fournisseur (*Business Entity*), le service lui-même (*Business Service*), les accès au service (*Binding Template*), le type de service (*tModel*) et des relations entre deux parties (*Publisher Assertion*) voir Figure 1.5.

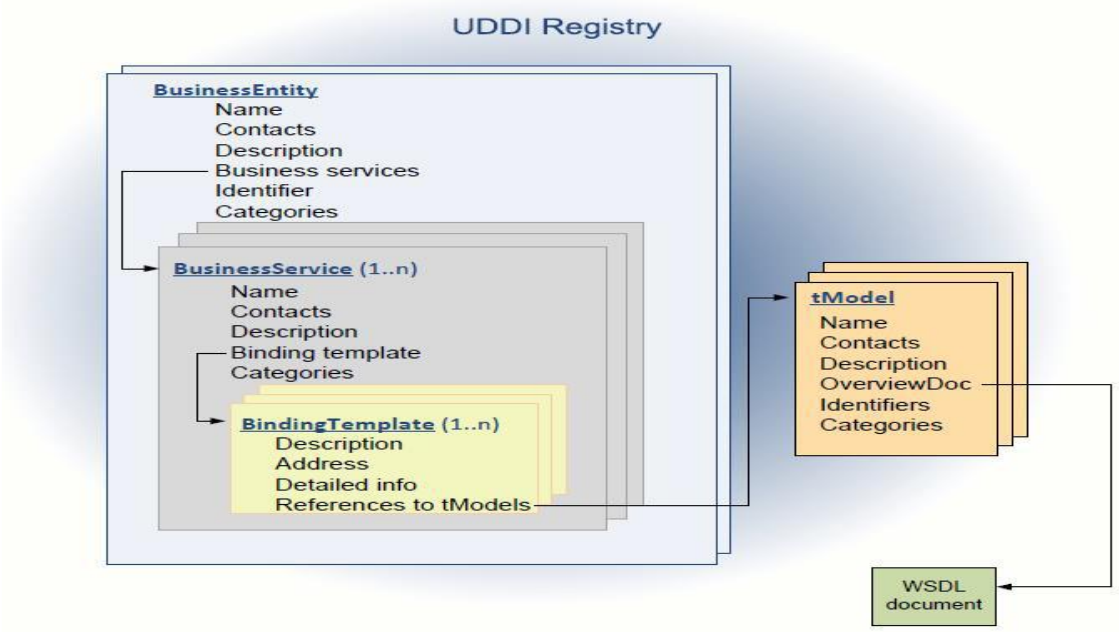


Fig 1.5 : Entités composant un annuaire UDDI [21].

- **Le fournisseur (Business Entity) :**Les informations concernant l'organisation hébergeant le service, le fournisseur et celles nécessaires à l'identification de l'entreprise sont répertoriées dans ce document XML. Ce document contient de l'information descriptive sur l'entreprise ou le fournisseur et sur les services proposés (lien vers l'entité *Business Service*).

- **Le service Business Service :** Ce composant représente les services proposés par l'organisation. La description des services contenue dans l'entité Business Service est de haut niveau (aucune information technique n'est décrite ici). Les informations à propos du nom du service et de son objectif sont représentées dans ce composant. Le fournisseur peut rassembler dans cette entité un ensemble de services répondant aux mêmes objectifs dans une même catégorie. Par exemple, une catégorie tourisme peut contenir un service météorologique et un service localisant les sites touristiques. Les catégories de service (contenant un ou plusieurs services) sont liées (selon le nombre de services) à un ou plusieurs points d'accès (*Binding Template*).

- **Les accès au service (*Binding Template*) :** Ce module décrit les points d'accès aux Services Web (URL) et le moyen d'accéder (les différents protocoles à utiliser) afin d'invoquer les services.

- **Le type de service (*tModel*) :** Le tModel permet d'associer un service à sa description WSDL. Le client potentiel peut ainsi avoir connaissance des conventions d'utilisation du service. La liaison entre les entités *Binding Template* et *tModel* est nécessaire pour l'invocation du service.

- ## Recherche de Services Web avec UDDI :

  La recherche et la sélection dans UDDI reposent sur la publication préalablement décrite du service et de son fournisseur. Le futur client peut connaitre par l'intermédiaire de l'UDDI : les fournisseurs d'un service, les services proposés par un fournisseur donné, les moyens d'invoquer un service. Pour apporter aux clients la réponse à ces questions, UDDI organise l'ensemble

des informations qu'il contient en trois parties, spécifiées en XML. Chacune d'elles peut être utilisée pour établir une recherche via UDDI. Ces parties sont les suivantes :

- **Les pages blanches (*White paper*) :** Ce composant permet de connaître les informations à propos de l'organisation proposant le service. Cette description contient toutes les informations jugées pertinentes pour identifier l'organisation (telles que son nom, son adresse physique). Le futur client du service retrouve dans les pages blanches les informations que le fournisseur a renseigné dans l'élément *Business Entity*lors de la publication.

- **Les pages jaunes (Yellowpaper) :** Les pages jaunes d'UDDI détaillent la description de l'organisation faite dans les pages blanches en répertoriant les services proposés. Dans cette section, sont décrits : la catégorie de l'entreprise, le secteur d'activité dans lequel exerce l'entreprise, les services offerts par cette organisation, le type de services et les conventions d'utilisation- prix, qualité de service, etc. La description des services contenue dans les pages jaunes est non technique et est renseignée par les fournisseurs eux-mêmes.

- **Les pages vertes *(Green paper*) :** Les pages vertes comportent les informations techniques liées aux Services Web et basées sur leur description WSDL.

1.6. Scénario général de fonctionnement des Services Web :

La dynamique de l'architecture des Services Web se décompose comme suit :

- Le fournisseur de service définit la description de son service et la publie dans un annuaire de service (dans le registre) en vue d'être localisé par des clients (étape 1).
- Le client utilise la facilité de recherche disponible au niveau de l'annuaire pour retrouver et sélectionner un service donné (étapes 2 et 3).
- Le client examine ensuite la description du service sélectionné (extrait sa description du registre) pour récupérer les informations nécessaires lui permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré (étapes 4 et 5).

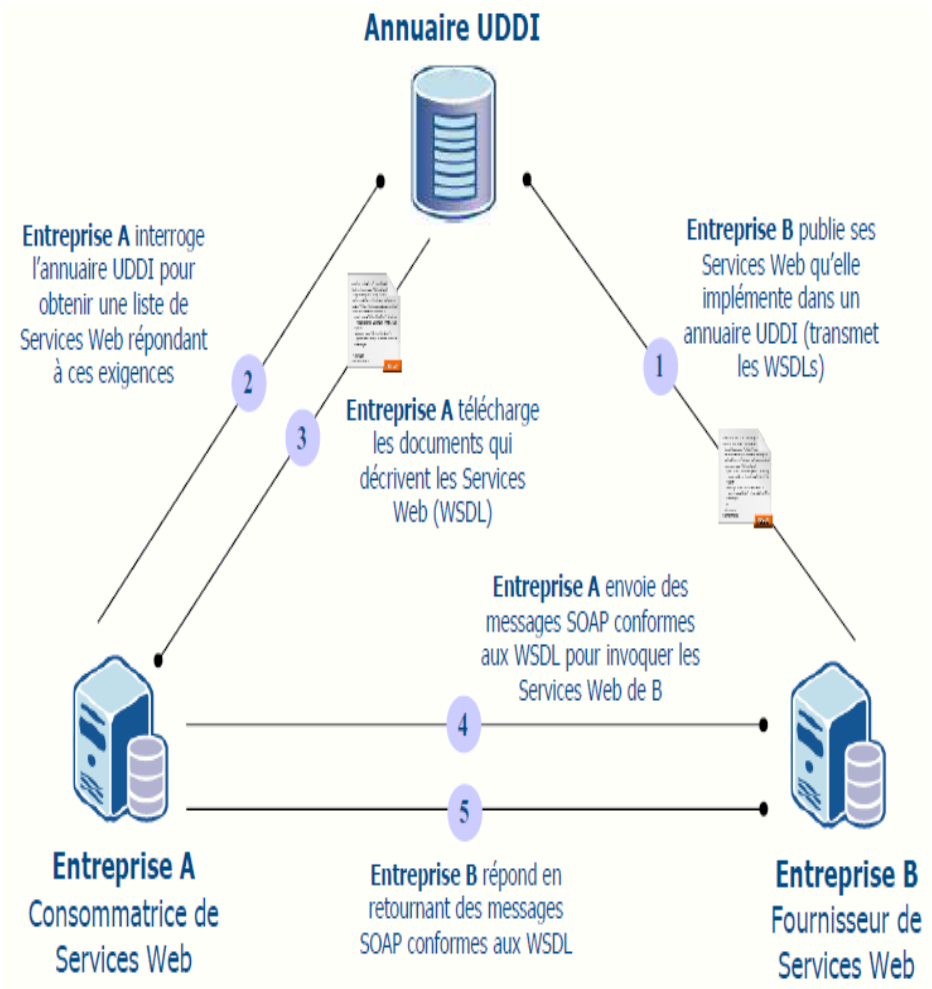


Fig 1.6 : Modèle de fonctionnement des Services Web.

1.7. Composition des Services Web :

Les Services Web, tels qu'ils sont présentés, sont conceptuellement limités à des fonctionnalités relativement simples qui sont modélisées par une collection d'opérations. Toutefois, pour certains types d'applications, il est nécessaire de combiner un ensemble de Services Web (Services Web simples) en Services Web plus complexes (Services Web agrégés ou composites) afin de répondre à des exigences plus complexes.

L'objectif de la composition de service est de créer de nouvelles fonctionnalités en combinant des fonctionnalités offertes par d'autres services existants, composés ou non en vue d'apporter une valeur ajoutée.

Un Service Web est dit composite lorsque son exécution implique des interactions avec d'autres Services Web, et des échanges des messages entre eux afin de faire appel à leurs fonctionnalités. La composition de Services Web spécifie quels services ont besoin d'être invoqués, dans quel ordre et comment gérer les conditions d'interaction.

1.7.1. Classification des Services Web selon leurs modèles d'interaction

Les Services Web sont classés selon leur modèle d'interaction comme suit [22] :

- **Modèle de Service Web atomique :**

Un Service Web atomique est décrit comme une boite noire, c'est-à-dire qu'il est spécifié en terme d'entrées/sorties ainsi que des pré-conditions et effets sans prendre en considération le fonctionnement du service.

- **Modèle de Service Web comportemental :**

Les Services Web basés sur un modèle comportemental sont souvent connus sous le nom de boite grise, ils sont décrits par l'ordre d'exécution de leurs opérations.

1.7.2. L'invocation de services dans une composition :

L'invocation des services dans une composition sont dans un ordre bien précis. On distingue les types d'exécution suivant :

- **Exécution séquentielle :**

Dans une exécution séquentielle, un service est invoqué une fois que tous les Services Web précédents ont été exécutés (Figure 1.7).

- **Exécution en parallèle :**

Dans ce cas, les Services Web s'exécutent en parallèle. Elle peut être représentée par l'opérateur AND (Figure 1.7), le Service Web s'exécute en parallèle avec le Service Web.

➢ **Exécution conditionnelle :**

Un chemin est choisi parmi plusieurs, ce choix est fait à l'aide d'une décision prise au moment de l'exécution. Elle peut être représentée par l'opérateur OR (Figure 1.7).

➢ **Exécution en boucle :**

Un Service Web peut être invoqué plusieurs fois (Figure 1.7).

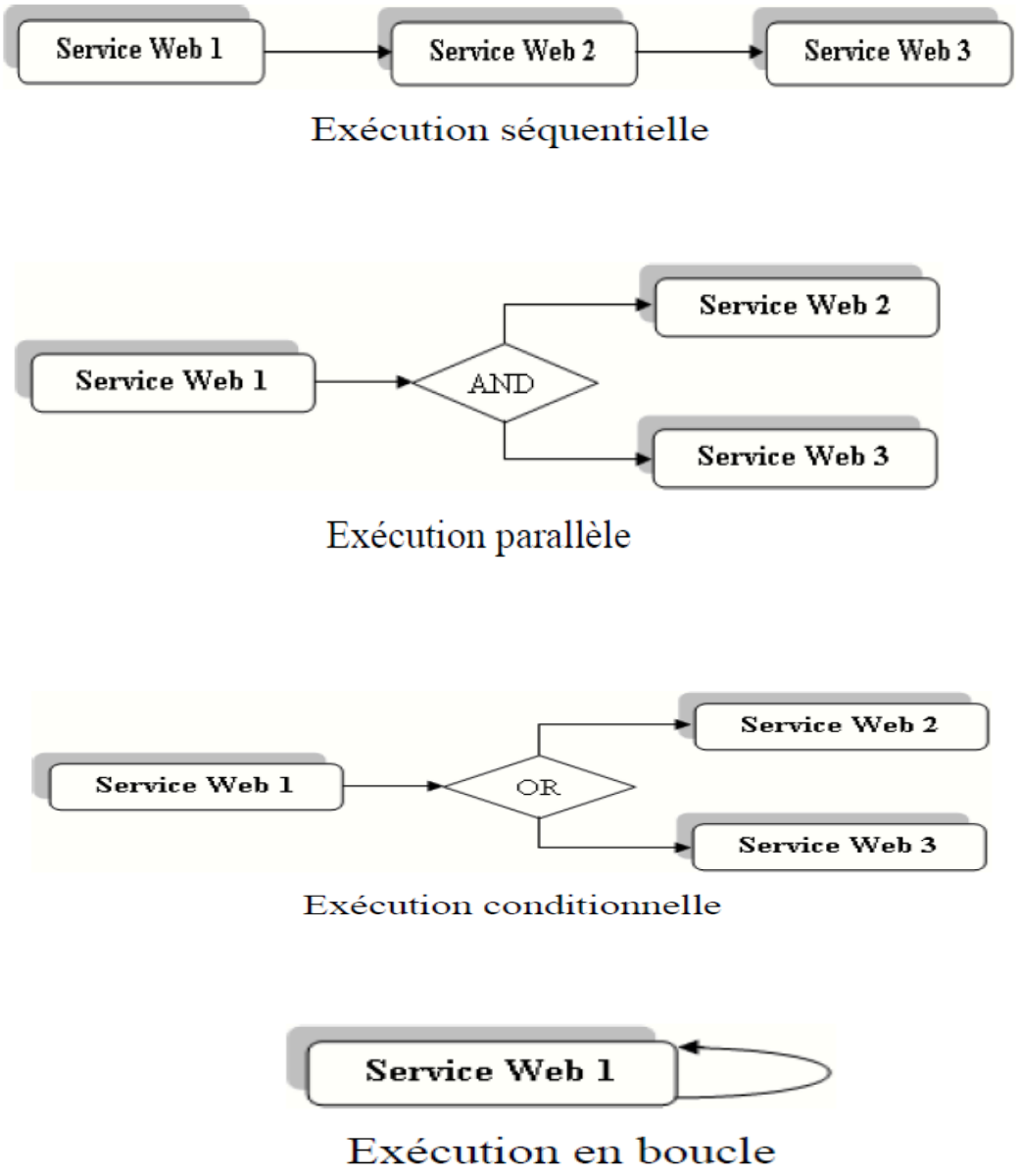


Fig 1.7 : Invocation des Services Web [22].

1.8. Conclusion :

Les services web est devenue une technologie de référence pour le développement des applications distribuées sur le web, l'utilisations des protocoles basées sur XML et standardisés par le W3C pour la description des services (WSDL) et l'échange des données et l'invocation des objets à distance (SOAP) formulé avec des messages XML et véhiculées au-dessus de protocole de web (http, SMTP,..) permet de surmonter les problèmes d'hétérogénéités et d'interopérabilités.

Les Services Web apportent des avantages indéniables dans les technologies d'intégration et en termes d'ouverture des entreprises vers ses partenaires en offrant des services directement issus de leur système d'information.

Nous présenterons dans le chapitre suivant un Etat de l'art sur les services web DaaS et l'intégration de données hétérogènes.

Chapitre II :

Etat de l'art

Les Services Web DaaS (Data as a Service)

Introduction:

Au cours de la dernière décennie, les Services Web ont été largement perçus comme un moyen standardisé pour l'intégration d'applications sur le Web. Ils reposent sur une architecture orientée service permettant ainsi aux applications de différents fournisseurs d'être encapsulées comme des services, puis publiées, localisées, invoquées, composées et coordonnées de manière à couplage faible. Récemment, les Services Web ont commencé être un support populaire pour la publication et le partage des données sur le Web. Les entreprises modernes évoluent vers une architecture orientée service de partage de données sur le Web en mettant leurs bases de données derrière les Services Web, fournissant ainsi la méthode interopérable d'interagir avec leurs données. En outre, les données qui ne sont pas stockées dans des bases de données traditionnelles sont également mises à disposition via des Services Web. Nous appelons ce type de Services Web en tant que Services Web DaaS.

Dans ce chapitre nous allons présenter les services web DaaS, dans la section suivante nous allons présenter L'intégration de sources d'information et les systèmes médiateurs, puis le rôle des services web dans les systèmes de médiation avec le détaille de deux projets : Picsel et Active XML. Ensuite nous allons présenter les différents Tavaux de recherche dans le domaine des services web DaaS.

2.2. Définition:

« *Un Service Web DaaS fournit une vue simplifiée, intégrée en temps réel, une information de haute qualité sur une entité commerciale spécifique comme un client ou un produit. Il peut être fourni par le middleware ou emballé comme un composant logiciel individuel. Les informations qu'il fournit proviennent d'un ensemble diversifié de ressources d'informations* » [23].

Une autre définition de composite software : *« Les Services Web DaaS sont une forme de Services Web optimisée pour les demandes d'intégration*

de données en temps réel. Ils visualisent les données pour découpler les emplacements physiques et logiques et donc, d'éviter la réplication des données inutiles. Les Services Web DaaS résument des structures de données complexes et de la syntaxe. Ils fédèrent des données disparates en composites utiles et supportent l'intégration de données à travers les applications de l'architecture orientée service » [24].

Les Services Web DaaS permettent l'accès aux sources de données des organisations. L'invocation d'un service DaaS résulte dans l'exécution d'une requête sur le schéma de la source de données. Lorsqu'un tel service est exécuté, il accepte d'un utilisateur une donnée d'entrée d'un format spécifié et il lui retourne des informations comme une sortie.

Les Services Web DaaS sont maintenant utilisés dans de nombreux domaines d'application comme un moyen standard pour la publication et le partage des données. Exemples de domaines d'application comprennent, entre autres, le partage des données scientifiques (par exemple, la bioinformatique, traitement et partage de données géo spatiales, etc), le partage des données médicales (eHealth), les entreprises d'intégration de données, le partage des données entre les organismes gouvernementaux (eGovernment), etc).

2.3. L'intégration de données hétérogènes :

2.3.1. Problématique de L'intégration de données :

Du fait du développement important de l'Internet, la recherche d'informations issues des sources de données réparties sur le réseau devient de plus en plus difficile. En effet, grâce à la révolution de nouvelles technologies de l'information, les entreprises aussi bien que les individus disposent d'une grande quantité de données. Ces données sont stockées dans des sources hétérogènes et autonomes.

Chaque source de données est décrite par sa localisation, le type de données qu'elle gère, ses possibilités d'interrogation et le format des résultats.

- La ***localisation*** d'une source de données englobe tout aussi bien le référencement du site sur lequel se situe la source(URL, adresse IP + port), que le protocole de communication utilisé (ex: TCP/IP), les moyens d'accès à la base (ODBC, JDBC) ainsi que le support (pages Web, SGBD). *Le type de données* géré par une source peut être structuré (base de données relationnelles), semi structuré (sources XML,OEM) ou non structuré (images, texte libre).

- Les ***possibilités d'interrogation*** définissent les langages de requêtes évolués et standardisé (SQL, OQL).

- Enfin, ***les formats des résultats*** qui peuvent être définis suivant divers modèles standards (XML, HTML, relationnel).

Les sources de données auxquelles nous avons accès dans un contexte interconnecté, via le Web, ont choisis leur propre représentation d'informations, de sorte que nous pouvons présent parler des sources de données Hétérogènes [25].

L'hétérogénéité se présente dans deux catégories :

- ***Structurelle*** : la manière dont sont représentées les données (exemple: l'adresse peut être représentée sur un seul champ, ou plusieurs : Rue, Code postal, Ville).

- ***Sémantique***: en rapport avec la signification des données (synonymes, homonymes, etc.)

➢Systèmes d'Intégration de données :

Les systèmes d'Intégration de données offrent des architectures d'interopérabilité sur une fédération de sources *distribuées*, *autonomes* et *hétérogènes*. Les entrepôts de données, les systèmes de médiation et les architectures P2P sont des exemples d'infrastructures permettant l'Intégration

de données, c'est-à-dire l'accès à des données produites par des sources autonomes. A travers des schémas virtuels, des métadonnées et des correspondances sémantiques, ils permettent d'accéder à ces sources de données de façon uniforme et transparente, en transformant par réécriture les requêtes d'un utilisateur en sous requêtes envoyées aux sources de données les plus appropriées.

➢ Définition & Composante

« Un système d'intégration de données fournit une vue unifiée de données provenant de sources multiples et hétérogènes. Il permet d'accéder à ces données à travers d'une interface uniforme, sans se soucier de leur structure ni de leur localisation » [26].

Formellement, un système d'intégration de données est un triplé I: <G, S, M>, où: G représente le schéma global modélisant le schéma intégré , S est l'ensemble des schémas des sources décrivant la structure des sources participantes au processus d'intégration, M est une correspondance entre G et S qui établit la connexion entre les éléments du schéma global et ceux des sources.

Un système d'intégration se compose de deux parties [27] (voir figure 2.1):

- Une partie (1) externe correspond aux utilisateurs du système intégré ou autres systèmes.
- Une partie (2) interne et comprend des sources et une interface uniforme qui permet à la partie externe d'interroger d'une manière transparente les sources de données,comme s'il n'y avait qu'une seule source.

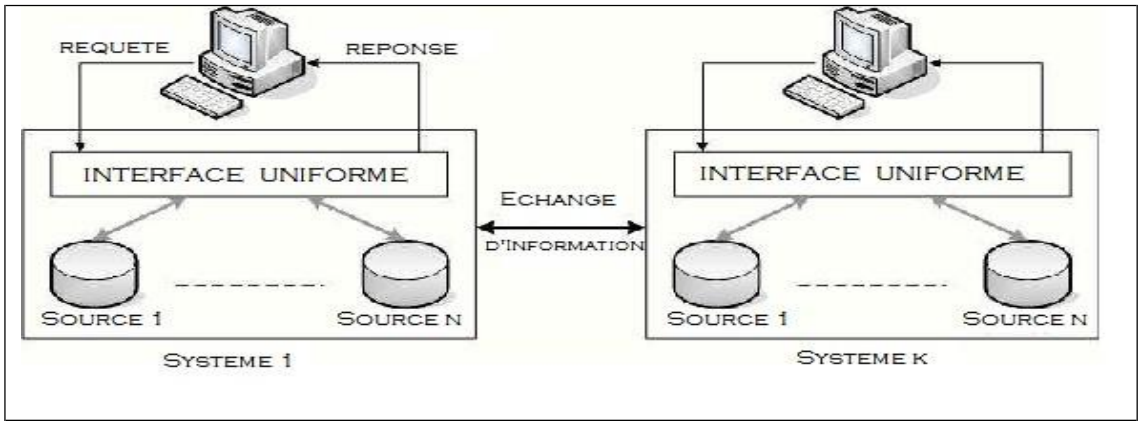


Fig 2.1 : Système d'intégration d'information [27].

- **Processus d'intégration :**

  Etant donné un ensemble de sources hétérogènes {S1, S2,… Sn}, le problème d'intégration consiste à construire un schéma intégré (ou schéma global) qui sera utilisé comme interface d'accès aux sources de données. La construction du schéma global à partir des schémas locaux est une tâche difficile. Cette difficulté est liée au fait que les sources stockent des différents types de données, en différents formats, ayant différentes significations et associées aux différents noms[28].

  Il convient d'abord d'indiquer qu'il existe plusieurs méthodologies permettant l'intégration des bases de données classiques. Nous présentons le processus dans le paragraphe suivant.

Le processus d'intégration est décomposé en trois phases distinctes :

- ***La pré-intégration*** : qui vise préparer l'intégration des schémas en les rendant plus homogènes. Elle consiste à traduire les schémas initiaux dans un modèle de données commun (réduction de l'hétérogénéité syntaxique).

- ***L'identification des correspondances*** : durant cette phase, les correspondances entre les éléments des schémas source sont détectées et formalisées.

- ***L'intégration*** : cette phase finale produit le schéma intégré et fournit les règles de traduction permettant de passer des schémas source au schéma intégré et inversement « mapping».

- **Les tâches d'un système d'Intégration :**

On peut distinguer quatre tâches principales d'un système d'Intégration. Les deux premières concernent la traduction des données provenant des sources différentes en résolvant le problème de l'hétérogénété physique/logique des sources en fournissant une interface d'accès uniforme. Les deux dernières résolvent le problème de l'hétérogénéité sémantique en reliant chaque source au schéma global. Ces quatre tâches sont décrites ci-après :

- ***Transformation de données*** (par exemple: la transformation de données relationnelles en XML): Les problèmes devant être résolus à ce niveau sont la perte d'information, la taille des données générées et la performance des traitements sur ces données.

- ***Traduction de requêtes:*** La traduction des requêtes d'un langage (exp. XQuer y) en un autre langage (exp.SQL) est liée au problème de transformation de données. Elle doit également prendre en compte la puissance d'expression du langage cible et nécessite souvent des extensions spécifique afin d'obtenir la puissance d'expression du langage source

- ***Réécriture de requêtes*****:** Cette tâche est différente de la tâche précédente et généralement plus complexe car elle doit prendre en compte l' hétérogénété structurelle et sémantique entre les schémas et joue un rôle important dans l'intégration de données sur le web.

➢ ***Fusion de données*****:** essaye de répondre au problème de la représentation multiple d'une même information dans différentes sources. Elle fait partie de la tâche de réécriture de requête.

2.3.2. Taxonomie de conflits d'intégration :

Plusieurs types de conflits dus à l'hétérogénéité peuvent être considérés dans l'établissement des correspondances entre schémas lors de l'intégration de données. De nombreuses taxonomies de conflits sont abordées par les chercheurs dont six types sont définis par *Parent* [29], qui sont décrit dans le tableau suivant :

Conflits	Description	Exemple
Conflits de Classification	Les types de correspondances décrivent des ensembles différents. mais liés sémantiquement.	Deux sources Médicales contient chacune une relation *Médecin* : la première, détermine tous les *Médecins* et la seconde uniquement les *Médecins spécialistes.*
Conflits de Description	S'il y a une différence entre les propriétés des types de correspondance (les types d'objets peuvent différer selon leurs: noms, clés, attributs,..etc)	Différence selon le nom : est l'utilisation de termes synonymes dans la désignation d'un même type de relation dans deux schéma différents : *Thésard dans l'un et Doctorant dans l'autre.*
Conflits Structurel	Si les éléments de correspondance sont décrits par des concepts de niveaux de représentation différents ou soumis a des contraintes différentes.	Une adresse est un attribut d'une table relationnelle dans un schéma A peut correspondre à une table dans un schéma B.
Conflits d'Hétérogénéité	Dans l'intégration, les schémas qui ne sont pas exprimés dans le même modèle doivent être traduits lors d'une phase de prétraitement.	
Conflits de Métadonnées	Si une donnée dans une base est en correspondance avec une métadonnée (le nom d'un attribut) dans le schéma d'une autre base.	Deux schémas à base de données praticiens avec praticien1 (id, neurologue,..) et praticien2 (id, fonction,..), certaines valeurs de l'attribut fonction de praticien2 peuvent correspondre au nom de l'attribut neurologue de praticien1 .
Conflits de Données	Si des occurrences des correspondances ont des valeurs en conflit pour des attribut en correspondance. Les conflits de données sont détectés lors de l'exécution d'une requête de l'utilisateur.	

Tab.1 : Taxonomie de conflits [29].

Goh et ses collègues [30] proposent une autre classification des conflits qui peuvent apparaitre lors de la mise en correspondance entre schémas. Ce sont détaillés dans le tableau suivant :

Type de conflits	Description	Exemple
Conflits de nommage	Les différents schémas utilisent des noms différents pour représenter le même concept.	Les cas de présence de synonymes et d'homonymes.
Conflits de graduation	Les concepts on la même signification dans deux schéma mais sont différents à cause de leur contexte.	On peut citer en exemple la mesure de température exemples : - Degrés Celsius ou Fahrenheit. - Dollar $ou l'Euro €.
Conflits de confusion	Les concepts paraissent avoir la même signification mais différent en réalité.	Ce type de confusion peut être causé par des contextes temporels différents. Par exemple le poids d'une personne dépend de la date ou elle s'est pesée.
Conflits de représentation	Deux schémas sources décrivent même concept de manière différente.	Dans une source, l'adresse peut être désignée par une chaine de caractères tandis que dans une autre, par une structure composée du numéro et du nom de la rue, du code postal et de la ville.

Tab. 2 : Taxonomie de conflits Description [30].

La diversification des sources de données, conduit à l'idée d'offrir à 1'utilisateur un système permettant d'avoir une vue centralisée uniforme des données. Ainsi, les utilisateurs vont se focaliser de spécifier ce qu'ils veulent et non pas perdre du temps en réfléchissant comment obtenir la réponse, en interrogeant chaque source et en combinant les différents résultats obtenus. A cet

événement, les chercheurs se sont investis dans l'intégration des sources de données qui est devenue un axe de recherche important.

2.4. Classification des systèmes d'intégration :

Plusieurs approches et systèmes d'intégration ont été proposés dans la littérature, souvent classifiés [31]. Il existe une classification des systèmes d' intégration en se basant sur trois critères:

- *Localisation de données intégrées*: qu'elles restent dans les sources originales ou qu'elles migrent vers le système global.
- *Nature de correspondance (mapping):* entre le schéma global et les schémas locaux.
- *L'automaticité du processus d'intégration*: permet de produire le schéma global, le mécanisme de médiation du schéma global et les schémas locaux, c'est -à- dire le système d'intégration.

2.4.1. Localisation de données intégrées :

Ce critère spécifie si les données des sources locales sont dupliquées au niveau du système intégré ou pas. Les données du système intégré peuvent être ***matérialisées*****:** l'architecture d'un entrepôt de données (les données issues des différentes sources sont dupliquées au sein du système

Ou ***virtuelles*****:** l'architecture de médiateur (le système intégré fournit alors une application chargée de jouer le rôle d'interface entre les bases de données locales et les applications d'utilisateurs comme dans le projet TSIMMIS [32].

2.4.1.1. Entrepôts de données (Architecture matérialisée):

Un entrepôt de données (Data Warehouse) se définit comme « une collection de données intégrées, orientées sujet, non volatiles, historisées, résumées et disponibles pour l'interrogation et l'analyse» [33]. Les entrepôts de données sont conçus dans un but particulier : rassembler

l'ensemble des informations d'une entreprise dans une base unique, pour faciliter l'analyse et la prise de décision rapide.

Une illustration de l'architecture de ces Systèmes est présentée dans Figure 2.1 Les données stockées dans l'entrepôt proviennent des sources multiples souvent hétérogènes. Après leur extraction et leur transformation, elles sont stockées et organisées dans l'entrepôt par sujet (clients, produits,). Il existe donc une phase d'intégration lors de la conception d'entrepôts mais cette intégration est matérialisée.

Cela signifie qu'il y a une duplication des données et qu'il n'est plus nécessaire d'accéder aux sources initiales pour répondre à une requête.

Il existe également un schéma global dans un entrepôt de données qui est en effet dynamique et de nouvelles sources sont susceptibles d' être intégrées fréquemment, des données stockées peuvent être réorganisées (agrégées, ajoutées, supprimées,), etc.

Dans un entrepôt de données on définit une approche LAV (Local-as-View: les sources sont des vues sur le schéma global) pour l'intégration de données dans un entrepôt car toutes les informations présentées dans les différentes sources ne sont pas nécessaires. Il est donc préférable de définir d'abord le schéma global de l'entrepôt qui reflète l'information nécessaire et puis établir la correspondance avec les sources (approche descendante), plutôt que de se concentrer sur les sources, avant de produire le schéma global (approcheascdante).

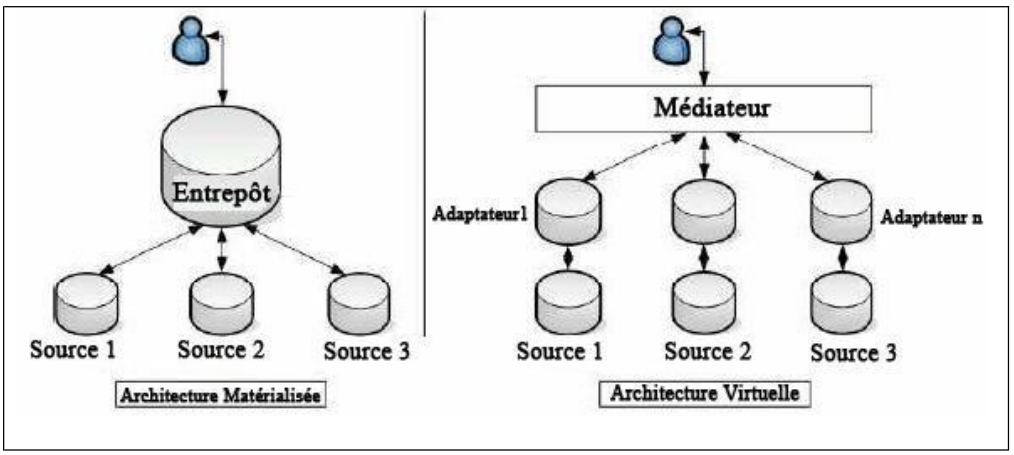


Fig 2.2 : Architecture matérialisée vers architecture virtuelle [33].

2.4.1.2. Systèmes de médiation (Architecture virtuelle) :

L'approche d'intégration par médiation constitue aujourd'hui la solution la plus courante pour relier différentes sources qui ne correspondent pas nécessairement à des bases de données. La notion de médiateur a été initialement proposée par *Wiederhold* [34] . Il définit un médiateur comme suit: *« Un* ***médiateur*** *doit être vu comme une couche logicielle permettant d'accéder de manière transparente pour l'utilisateur à différentes ressources (Bases de données, fichiers) réparties et hétérogènes ».* Pour ce faire, le *médiateur* exploite des connaissances (*métadonnées*) qui sont utiles aux différents services (interrogation, localisation des ressources).

L'approche par médiation est fondée sur la définition de vues. Les données ne sont pas stockées dans le système de médiation mais résident dans leur source d'origine. L'utilisateur à une vision unifiée des données sources: l'interrogation se fait par l'intermédiaire d'un *schéma global*. Il n'a pas de connaissance des *schémas locaux*.

L'architecture générale d'un système de médiation est présentée en Figure 2.1. Une requête globale est posée via le schéma global, celle-ci est ensuite décomposée en sous requêtes, traduites pour être exécutées sur les différentes sources concernées.

Le *médiateur* est chargé de localiser les données pertinentes pour répondre à la requête (en utilisant les métadonnées). L'interrogation effective des sources se fait par des adaptateurs qui constituent une interface d'accès aux différentes sources. Ces *adaptateurs* traduisent les sous requêtes exprimées dans le langage de requête spécifique de chaque source. Les résultats sont ensuite renvoyés au médiateur qui se charge de les intégrer avant de les présenter à l'utilisateur.

2.4.1.3. Médiateurs / Entrepôt de données (Architecture Mixte/Hybride) :

Avec le développement du Web, d'autres approches d'intégration tels que les systèmes hybrides (approche mixte) ont été proposés. Ces Systèmes combinent à la fois l'approche médiateur et l'approche entrepôt. Il s'agit, par exemple, d'un médiateur qui intègre plusieurs sources de données externes et qui exploite un entrepôt de données contenant des données conformes au schéma global du médiateur. Xylème en est un exemple de ce type d'architecture.

2.4.1.4. Systèmes P2P (Pair to Pair) :

L'émergence des systèmes de partage de fichiers pair à pair (Peer -to-Peer) a conduit les chercheurs pour considérer l'architecture P2P dans le contexte de l'intégration et le partage de données [35] [36] [37]. Ces systèmes P2P suivent une approche décentralisée pour l'intégration des pairs autonomes et distribués contenant des données qui peuvent être partagées. L'objectif principal de tels systèmes est de fournir une interopérabilité sémantique entre plusieurs sources en l'absence de schéma global.

Chaque pair est interconnecté avec un certain nombre de pairs du réseau (appelés voisins) à l'aide de formules de coordination. Edutella [35], SomeWhere [36], PIAZZA [37] sont des exemples de travaux récents sur les systèmes P2P.

2.4.2. Ontologies et intégration d'informations :

Généralement, les sources de données sont conçues indépendamment l'une de l'autre par des concepteurs différents. En conséquence, des données relatives à un même sujet peuvent être représentées différemment dans ces différentes sources. C'est le problème de l'hétérogénéité des données [30]. A identifié trois principales causes à l'hétérogénéité sémantique des données.

- ***Les conflits de nom*** ont lieu lorsque des noms différents sont utilisés pour décrire le même concept (synonyme) ou lorsque le même nom est utilisé pour des concepts différents (homonyme).

- ***Les conflits de mesure de valeur*** ont lieu lorsque différents systèmes de référence sont utilisés pour évaluer une valeur. C'est le cas, par exemple, lorsque différentes unités de mesure sont utilisées par les différentes sources de données.

- ***Les conflits de contexte*** ont lieu lorsque des concepts semblent avoir la même signification mais diffèrent en réalité dû à différents contextes de définition ou d'évaluation.

Parce qu'une ontologie peut servir de pivot pour définir la sémantique des données des différentes sources, leur utilisation est une solution pour résoudre les problèmes d'hétérogénéité des données. Dans le contexte de l'intégration d'information, et plus particulièrement des serveurs d'information de type médiateurs, les ontologies ont un rôle double bien spécifique. Elles interviennent au sein d'interfaces d'interrogation. Par ailleurs, il s'agit de schémas sur lesquels repose l'intégration des sources de données hétérogènes accessibles.

Plusieurs approches d'intégration à base ontologique ont été développées [38]. Ces dernières peuvent être divisées en trois catégories : approches avec une seule ontologie, approches avec ontologies multiples et approches hybrides. Dans l'approche avec une seule ontologie, chaque source référence la même ontologie globale de domaine. Les systèmes d'intégration SIMS [39] et COIN [40] sont des exemples de cette approche. En conséquence, une nouvelle source ne peut ajouter aucun nouveau concept sans exiger le changement de l'ontologie globale. Dans l'approche à multiples ontologies (exemple du projet OBSERVER [41], chaque source a sa propre ontologie développée indépendamment des autres sources.

Dans ce cas, les correspondances inter-ontologies sont difficiles à mettre en œuvre. L'intégration des ontologies est donc faite d'une façon manuelle ou semi-automatique [41]. Pour surmonter l'inconvénient des approches simples ou multiples d'ontologies, l'approche hybride a été proposée. Dans cette dernière,

chaque source a sa propre ontologie mais toutes les ontologies utilisent un vocabulaire partagé commun (exemple du projet KRAFT [42]).

- **La correspondance entre schéma global et schéma local :**

Les systèmes d'intégration de données peuvent être classifiés suivant la relation entre les schémas des sources locales par rapport au schéma global. Ce critère n'a que peu d'intérêt dans une approche matérialisée. L'interrogation des données dans une telle approche ne se fait en effet que sur l'entrepôt de données. La nécessité de garder une définition en permanence de la mise en correspondance entre l'entrepôt et les sources de données est ainsi inutile [43]. En revanche, il est très important de considérer ce critère dans une approche virtuelle. Les requêtes étant posées au médiateur, celui-ci doit être en mesure de transformer ces requêtes en sous-requêtes sur les sources de données. Il est donc nécessaire de garder une mise en correspondance permanente entre le médiateur et les sources de données. La correspondance des schémas a été largement étudiée par la communauté des bases de données.

Classiquement, on trouve dans la littérature deux approches pour construire le schéma médiateur d'un système d'intégration. L'approche GaV (*Global as View*) [44] définit le schéma médiateur comme une collection de vues sur les sources. A l'inverse, dans l'approche LaV (*Local as View*), le schéma médiateur est construit indépendamment des sources. Les sources sont ensuite définies comme des vues sur le schéma médiateur.

D'autres approches ont apparues par la suite, notamment, l'approche GlaV (*Generalized Local as View*) [45] qui est une variante de LaV mais qui peut être considérée comme une approche à part entière compte tenu de ses caractéristiques.

- **L'approche GaV (Global as View):**

L'approche GaV est la première à être proposée, provient du monde des bases de données fédérées. Elle consiste à définir à la main (ou de façon semi-automatique) le schéma global en fonction des schémas des sources de données à intégrer (schémas locaux). Chaque schéma local est défini comme étant un ensemble

de relations (ou de prédicats) et les relations globales sont définies comme étant des vues sur les relations des schémas de s sources à intégrer. Comme les requêtes d'un utilisateur s'expriment en termes de prédicats du schéma global, on obtient facilement une requête en termes de schémas des sources de données intégrées, en remplaçant les prédicats du schéma global par leurs définitions.

Parmi les systèmes utilisant GaV, on peut citer TSIMMIS [44] et MOMIS [46] .

Exemple **:** la définition des vues pour cette intégration est montrée dans la figure 2.3.

Soit la requête suivante qui est posée au médiateur : *Quelles sont les cours donnés*

aux étudiants de l'"UNIV-MASCARA" ?

SELECT*RG.Course*

FROM *RG*

WHERE *RG.Organisation = " UNIV-MASCARA "*

Cette requête sera reformulée en sous-requêtes sur les deux sources comme suit:

SELECT *R1a.Course*

FROM *R1a, R1b*

WHERE *R1a.Student = R1b.Student* ***AND*** *R1b.Organisation ="UNIV-MASCARA "*

UNION

SELECT *R2a.Course*

FROM *R2a*

WHERE *R2a.Organisation = " UNIV-MASCARA "*

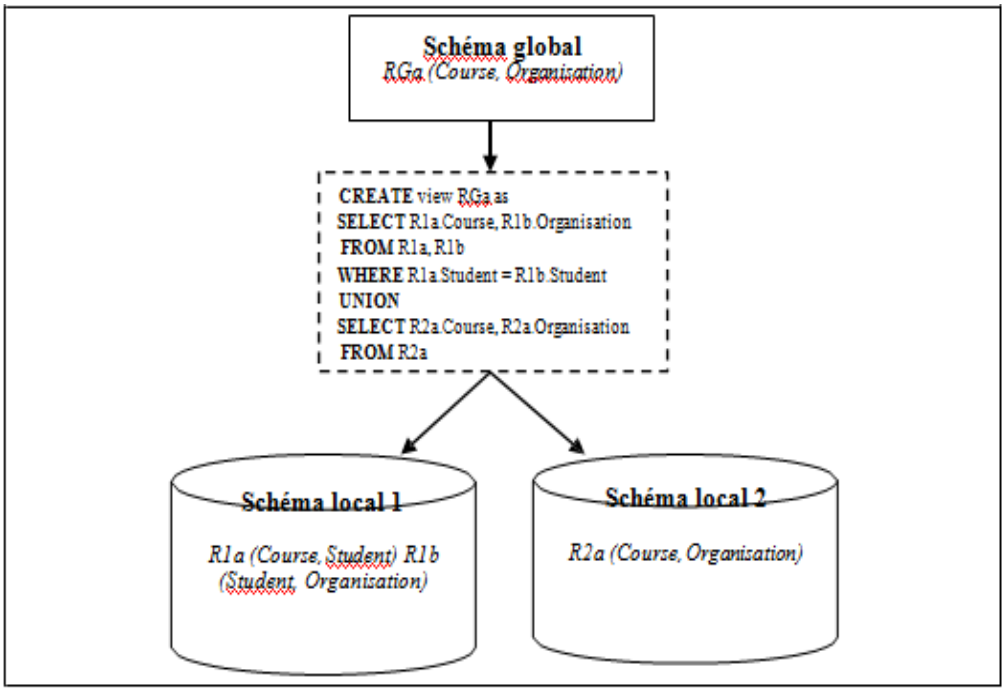


Fig. 2.3 : Définition des vues dans un s ystème GaV [46].

- **L'approche LaV (Local as View):**

L'approche LaV est l'approche duale, elle suppose l'existence d'un schéma global et consiste à définir les schémas des sources de données à intégrer comme des vues du schéma global.

Dans l'approche LaV, la requête sur le schéma global doit être reformulée suivant les schémas des sources locales. Cette reformulation (réécriture) des requêtes est une tâche complexe qui nécessite une inférence ; De plus, la complexité de cette inférence grandit avec l'augmentation du nombre de sources. En revanche, l'ajout (ou la suppression) des sources de données n'a aucun effet sur le médiateur, seules des vues doivent être ajoutées (ou supprimées). De même, un changement local de schéma est pris en compte en mettant à jour la vue locale. Autre avantage, si les données des sources locales n'ont pas le même format ceci ne pose aucun problème car en utilisant cette approche, chaque source peut être décrite séparément par un mécanisme de vue spécifique à son format. Il est à noter que l'approche LaV aura un problème de passage à l'échelle si le schéma global change, dans ce cas tous les schémas locaux doivent être redéfinis.

Les principaux systèmes développés autour de cette approche sont : Infomaster [47], PICSEL [45] et Information Manifold [48].

Exemple : Soient les deux schémas S1 et S2 de l'exemple 1. Soit un médiateur qui intègre ces deux sources suivant une approche LaV. Le schéma de ce médiateur contient les deux relations RGa et RGb.

- la relation RGa(Course, Student): {< c,s> ϵ RGa, ssi le cours *c* est donné à l'étudiant *s*}
- la relation RGb (Student , Organisation): {< s,o> ϵ RGb, , ssi l'étudiant *s* appartient l'organisation *o* }

La définition des vues pour une telle intégration est montrée dans la figure 2.4. Cette définition est obtenue par les règles de mapping suivantes :

S1 : R1a(x,y) :-RGa(x,y) et R1b(x,y) :-RGb(x,y)

S2 : R2a(x,z) :-RGa(x,y),RGb(y,z)

Reposant toujours la même requête : *Quelles sont les cours donnés aux étudiants de l'"UNIV- MASCARA"?*

```
SELECT  RG.Course
FROM RGa, RGb
WHERE  RGa.Student = RGb.Student  AND  RGb.Organisation = " UNIV-MASCARA"
```

Cette requête sera reformulée en sous-requêtes sur les deux sources comme suit:

```
SELECT  R1a.Course
FROM  R1a, R1b
WHERE  R1a.Student = R1b.Student
AND  R1b.Organisation = " UNIV-MASCARA "
UNION
```

SELECT *R2a.Course*

FROM *R2a*

WHERE *R2a.Organisation = " UNIV-MASCARA "*

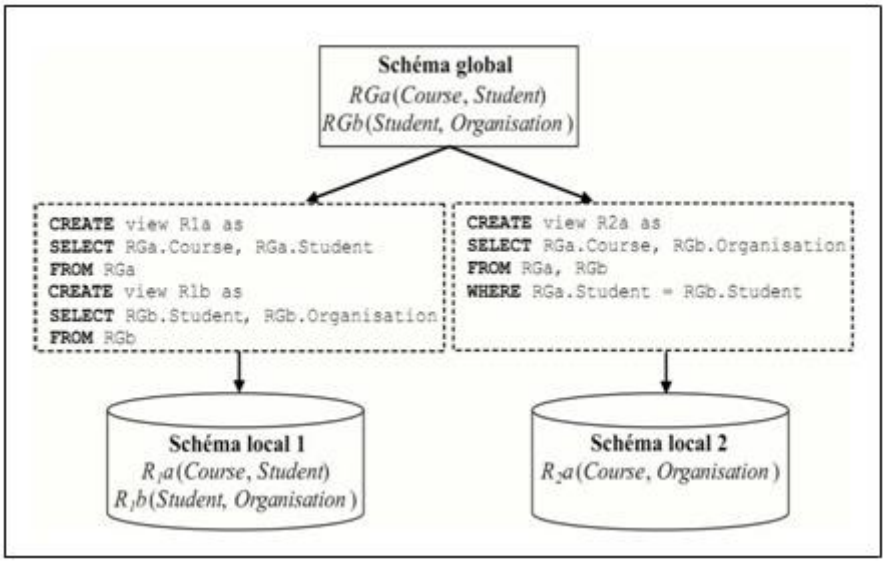


Fig 2.4 : Définition des vues dans un système LaV [47].

L'approche Generalized Local as View (GLaV):

GLaV est une variante de l'approche LaV. Elle permet de contenir une conjonction de relations d'une même source sous forme de jointure dans la tête des règles de définition de vue. Cela offre à cette approche un pouvoir d'expression strictement supérieur à celui de GaV et de LaV réunis. La complexité de réponse aux requêtes, quant à elle, n'est pas plus importante que dans LaV [49]. Comme exemple de système d'intégration adoptant l'approche GLaV on cite le projet PIAZZA [50].

2.5 Processus d'Intégration / Automaticité du mapping :

Un troisième critère important permet de caractériser l'automaticité de génération du système intégré . La notion de passage l'échelle, étant de plus en plus, un aspect essentiel, on peut caractériser cette automaticité par l'automaticité d'Intégration d'une nouvelle source au sein d'un système intégré.

- **Manuel**

Dans les Systèmes d'intégration, la synthèse des schémas locaux et les correspondances schéma global/schémas locaux sont faites manuellement. La signification des concepts utilisés, au niveau global et au niveau local étant explicite, aucun traitement automatique ne peut être envisagé.

- **Semi-automatique**

Un premier niveau d'automatisation devient possible lorsqu'on a utilisé (ensemble: synonymes, homonymes, etc.). Une telle intégration est qualifiée de semi- automatique car le domaine visé par l'intégration est suffisamment limité et formalisé, l'ontologie de domaine peut s'exprimer sous forme de prédicats de valeurs logiques. C'est le cas par exemple, dans les approches orientées relations, des Systèmes tels que Manifold.

- **Automatique**

Dans les deux types de mapping précédents, il suffit que la nouvelle source soit explicitement exprimée en fonction de l'ontologie globale pour que l'Intégration automatique soit possible.

2.6. Traitement des requêtes dans un système d'intégration :

Dans un système d'intégration, l'interrogation s'effectue généralement en utilisant des requêtes conjonctives (de type sélection-projection-jointure) à base de règles définies à l'aide du vocabulaire du schéma global qui exprime les vues sur les différentes sources. On parle alors d'interrogation basée sur les vues.

Dans un système d'intégration, les requêtes étant posées sur le schéma médiateur en utilisant un certain nombre de vues, le système essaie d'effectuer la réécriture des requêtes des utilisateurs en s'assurant que les requêtes réécrites sont soit équivalentes aux

requêtes initiales, soit incluses en essayant de trouver la meilleure solution possible par rapport aux sources disponibles.

2.7. Comparaison entre les différentes approches d'intégration :

Dans ce tableau, une classification des approches de mapping est proposée selon leur automaticité, sociabilité, traitement des requêtes extension, maintenance, avantages et limites.

Approche	**Passage à l'échelle**	**Extension**	**Avantage**	**Limites**
GAV	Etude	Maj. manuelle	Récriture facile des vues	Ajout d'une source difficile
LAV	Oui	Maj. manuelle	Ajout d' une source facile	Récriture difficile
GLaV	Oui	Maj. manuelle	Récriture facile des vues	Ajout d'une source facile

Tab. 3 : Comparaison entre les différentes approches d'intégration.

2.8. Principaux système d'intégration :

Les bases de données ont été introduites dans les entreprises au début des années

60, au départ pour assurer la gestion de petites applications commerciales. Depuis, un long chemin a été parcouru, et les applications sont devenues de plus en plus complexes et stratégiques pour les organisations. Le développement rapide d'internet et de nouvelles technologies a fait naître des applications nouvelles. Pour une organisation, le besoin d'intégrer les données issues de ces différentes applications devient une évidence.

Les premières approches d'intégration, sous forme de système fédéré, ont vu le jour dans les années 1980 [51]. Comme exemple on peut citer le système MULTIBASE [52]. Après les systèmes multibases ou fédérés, sont apparus les systèmes d'intégration à base de médiateurs (e.g., Garlic [53]), les systèmes à base d'agents (exp. InfoSleuth), et plus récemment encore les systèmes à base ontologiques (e.g., OBSERVER [54], MOMIS [46] et PICSEL [55] , les systèmes Pair à Pair (exp Edutella [35], Hyperion [56] et SomeWhere [36] et enfin les systèmes d'intégration à base de Services Web (exp. Active XML [57].

Discussion :

Dans cette section, nous décrivons sommairement un certain nombre d'entre eux (nous ne pouvons tous les présenter, mais les systèmes choisi sont représentatifs de leur famille).

Nous allons discuter rapidement des systèmes suivants : PIAZZA [58] comme exemple de système utilisant plusieurs médiateurs. SIMS [59] qui utilise une ontologie globale et l'approche centrée requête (GaV), MOMIS [46] comme exemple de système

utilisant une approche de découverte semi-automatique de correspondances, OBSERVER [54] comme exemple de système utilisant plusieurs ontologies et enfin XYLEME [60] comme exemple de système d'intégration de données semi structurées.

- **SIMS** est un système qui vise l'intégration de sources de données hétérogènes et de base de connaissances qu'il représente en utilisant le langage LOOM, basé sur la logique de description. SIMS adopte l'architecture à ontologie unique et utilise le mapping GaV. Le médiateur dans SIMS et spécialisé dans un seul domaine d'application.

- **MOMIS** est un système d'intégration qui se base sur son propre langage orienté objet dénommé ODL. Il utilise une ontologie unique globale appelée GVV (Global Virtuel View) qui est générées semi-automatiquement. MOMIS adopte l'approche GaV pour le mapping entre l'ontologie globale et les sources locales.

- **OBSERVER** est un système qui permet l'interopérabilité entre différents sources, en utilisant pour cela de multiples ontologies pour décrie les sources de données. Il se base sur CLASSIC, un langage de logique de description. Il n'y a pas d'ontologie globale dans OBSERVER ; le mapping entre multiples ontologies est réalisé à l'aide de tables de correspondance. Cependant, les relations entre ontologies sont limitées à des relations lexicales basiques telles que les synonymes, hyponymes et hyperonymes.

- **XYLEME** est un système d'intégration physique (approche entrepôt) avec XML comme modèle de données. Il adopte l'approche hybride dans son architecture.les ontologies globales et locales sont exprimées à l'aide d'arbres, avec un mapping GaV/ LaV. Le mapping est réalisé semi automatiquement par la génération de tables de correspondance entre les chemins de l'ontologie globale et les chemins de l'ontologie locales.

- **PIAZZA** en dernier, est un système qui intègre des sources qui sont, soit des ontologies, soit des données semi structurées (décrites par schéma XML ou DTD). Il s'appuie sur le modèle de données XML et une application de XQuery comme langage de requêtes. ce système suit une architecture Peer-to-Peer avec des ontologies et/ou schéma multiples. Le mapping entre les différentes ontologies est bidirectionnel (une source est décrite en fonction d'une autre et vice versa). Pour une

requête donnée, le schéma d'un nœud (source de données) peut être considéré comme étant le schéma global et ainsi, de proche en proche, les sources concernées seront interrogées.

Puis nous décrivons le rôle des services web dans les systèmes de médiation avec le détaille de deux projets : Active XML [57] (les systèmes d'intégration à base de Services Web) et PICSEL [55] (la combinaison d'un formalisme à base de règles et d'un formalisme à base de classes)

2.9 Services web DaaS (Data-as-a-Service):

Avec la démocratisation toujours croissante du Web, des sources de données de plus en plus nombreuses sont à la disposition des utilisateurs. Cependant, ces sources de données sont hétérogènes; on distingue généralement trois niveaux d'hétérogénéité :

- ***l'hétérogénéité technique*** concerne les modalités d'accès aux données : protocoles propriétaires nécessitant une application spécifique, protocoles ouverts (RPC, REST...), formulaires et tables HTML dans un navigateur Web.

- ***l'hétérogénéité syntaxique*** concerne la forme des données : formats propriétaires, données relationnelles, données semi-structurées (XML).

- ***l'hétérogénéité sémantique*** concerne la signification des schémas et vocabulaires utilisés pour décrire ces données : schémas relationnels, DTDs et schémas XML.

Une solution au problème posé par l'hétérogénéité technique réside dans l'utilisation de la technologie des services web.

On distingue deux classes de services [61]:

- **Effect-Providing Services Web (EP services)** : implémentent des processus, incarnant la logique d'application, souvent avec des effets secondaires comme, par exemple, un service de réservation de vol.

- **Services Web (DaaS)** : exposent les sources de données pour chercher des données, leurs invocation n'ont aucun effet.

Un service web DaaS est un composant qui encapsule une ou plusieurs sources de données et les rend disponibles comme interface du service web (par exemple, comme ensemble d'opérations WSDL) [62].

Les DaaS sont utilisées dans plusieurs applications et ils exploitent les outils de l'architecture orienté services[63] En bioinformatique , par exemple, ils sont utilisées pour encapsuler des bases de données et des outils pour fournir et analyser des informations de protéine possédées et conçues par des organismes autonomes de collaboration, telles que les bases de données de référence de protéine humaines HPRD et TIGRFAM [64].

2.9.1. Services web et intégration de données

Dans ce qui suit, nous allons présenter deux projets qui utilisent les services web dans la médiation des données : Active XML et le projet Picsel, puis nous allons présenter quelques travaux relatifs aux services web DaaS.

2.9.1.1. Active XML :

Active XML (AXML) [65] est un modèle déclaratif pour la gestion de données distribuées sur le Web basé sur XML et les services Web (SOAP, WSDL). Un document AXML est un document XML pouvant contenir des appels à des services Web [66].

L'approche Active XML propose une architecture pair à pair (P2P) [67], fondée sur l'échange de documents intensionnels entre différents pairs Active XML. Un pair Active XML est d'abord un entrepôt de documents AXML. Il est à la fois client (consommateur de services Web) et fournisseur de services déclaratifs définis par exemple par des requêtes (XQuery) ou des mises à jour sur les documents qu'il contient. L'utilisation d'appels de services Web inclus dans des documents permet de mettre à jour dynamiquement les données et de contrôler finement la périodicité de ces mises à jour et la pérennité des informations. la possibilité de mélanger des données et des appels de service dans un tel document

permet à chaque pair de matérialiser (c'est-à-dire remplacer l'appel par son résultat) tous ou seulement une partie des appels de service avant l'échange du document.

Les critères de ce choix sont multiples et dépendent des contraintes physiques et logiques du système et de l'application :

- Par exemple, si la bande passante entre les deux pairs est faible, il est préférable de transmettre un appel de service à la place du résultat de cet appel qui a généralement une taille plus importante.
- Un des deux pairs est incapable d'appeler le service pour des raisons de droits d'accès insuffisants.
- Un des deux pairs ne veut pas appeler des services "inconnus" pour des raisons de sécurité.

Active XML possède deux composants fondamentaux :

- Les documents Active XML qui sont simplement basés sur le fait de faire des appels de services dans des documents XML.
- Les services Active XML qui permet de faire des appels à d'autres services.

Les documents Active XML sont stockés dans les différents pairs. Ces derniers ont le rôle d'automatiser les appels de services et de mettre à jour les documents Active XML [66].

Active XML permet de faire des appels explicites aux services web, ce qui pose des problèmes lors des appels de service par adresse physique (sans annotation sémantique) qui peuvent intervenir lors d'un changement de contexte de l'appel ou disponibilité de service ou lors d'une composition dynamique de services. Ce qui implique qu'il faut adapter «manuellement» les appels en cas de changement de services disponibles.

Prenons l'exemple de l'appel explicite suivant :

```
<Inventory> Inventory of the books of city libraries
<city name="GuangZhou">
<sc>zhongshan.com\getBooks()</sc>
        <sc>GuangZhou.com\Books()</sc>
</city>
<Inventory>
Exemple I. appel de service web AXML
```

Exemple I : appel de service web AXML.

Ce qui a fortement motivé les chercheurs à définir des appels de services en terme d'ontologie indépendamment d'une adresse physique [67], c'est-à-dire faire des appels implicites où on définit le domaine du service (catégorie), des paramètres d'entrée et de sortie et le système se charge de choisir automatiquement les services appropriés. Ce qui permettrait la découverte et la composition dynamique du service (source de données).

L'exemple précédent
devient :

```
<Inventory> Inventory of the books of city libraries
<city name="GuangZhou">
<sc serviceCat="hierarchicalProfile.owl#book>
<output param_data_type="Concepts.owl#booklist" />
</sc>
</city>
</Inventory>
Exemple 2. appel de service web AXML avec annotation
```

Exemple 2 : appel de service web AXML avec annotation

2.9.1.2. Le projet Picsel :

PICSEL [68] propose une structure « médiateur » qui permet d'interroger des sources d'information multiples, hétérogènes et éventuellement réparties. Les systèmes médiateurs auxquels PICSEL s'intéresse regroupent un ensemble important de sources d'information XML relatives à un même domaine d'application. PICSEL comporte:

- le moteur de requêtes est conçu d'une façon générique pour être utilisable quel que soit le domaine d'application.

- la base de connaissances est spécifique au domaine appliqué. Elle se compose d'une ontologie du domaine et des ontologies locales.

PICSEL utilise CARIN [69] comme le langage de représentation de connaissances. Il possède également un langage de vues [70] et un langage de requêtes permettant d'exprimer, en termes de l'ontologie du domaine, respectivement, le contenu des sources et les requêtes des utilisateurs. L'ontologie globale (l'Ontologie du domaine) dans PICSEL est créée à travers l'ONTOMEDIA (Ontologie pour un MEDIAteur) [70].

Disscusion :

AXML n'aborde pas la résolution de requête par la composition de service Web. Les services Web dans un document restent inchangés tout au long de la durée de vie le document et sont sélectionnés à l'heure de création du document. Ainsi, AXML est approprié quand il y a une nécessité de gérer un ensemble prédéfini fixe des sources de données.

L'inconvénient de Picsel est qu'une fois l'ontologie partagée définie, chaque source doit utiliser le vocabulaire commun, ce qui limite l'autonomie des sources de données locales.

2.9.2. Plateformes industrielles pour les services web DaaS :

Les services de données ont gagné une attention considérable de leaders de l'industrie des logiciels SOA au cours des trois dernières années. Beaucoup de produits sont actuellement proposés ou en cours d'élaboration pour faire de la création de services de données plus facile que jamais, pour ne citer que quelques-uns, AquaLogic par BEA Systems [71], Astoria par Microsoft [72], MetaMatrix par RedHat [73], Composite Software [74], Xcalia [75], et IBM [76]. Les produits proposés ici intègrent des sources de données de l'entreprise et fournissent un accès

uniforme aux données grâce à des services de données. En plus, la plus parts des produits commerciales des bases de données incorporent des mécanismes d'exportations des fonctionnalités des bases de données comme des services web de données [77]. par exemple «IBM Document Acces Definition Extention (DADX) », technologie (Db2XMLextender) [78] et le Native XML Services Web for Microsoft SQL Server 2005 [79].

DADX fait partie du XML Extender IBM DB2, un couche de mapping XML/relational , et facilite le développement de services Web au-dessus de la base de données relationnelle qui peut, entre autres choses, exécuter des requêtes SQL et récupérer des données relationnelles au format XML.

- **BEA AquaLogic Data Services Platform :** BEA AquaLogic Data Services Platform [71] a été conçu dès le départ pour fournir un appui pour le concept de services de données. Puisqu'elle vise le monde de SOA, elle prend une vue orientée services de données. BEA AquaLogic Data Services Platform est une collection de fonctions qui ont toutes un schéma de sortie commun, acceptent différents ensembles de paramètres et sont implémentées via des expressions XQuery individuelles. Dans un exemple simplifié, un service de données exporte un ensemble de fonctions retournant des objets d'un client, où une fonction prend comme entrée le nom du client, une autre son adresse et pays, etc. AquaLogic exporte ces services de données aux développeurs des applications SOA comme un service web de données, où les fonctions deviennent des opérations.

- **Le projet « Astoria »** : En 2007, Microsoft propose le projet « Astoria » [72], connu

  aujourd'hui sous le nom d'ADO.NET Data Services. C'est un framework permettant

d'effectuer des requêtes sur des services de données disponibles sur le web ou en intranet. Les données sont transmises sous la forme de requêtes REST (Représentationnel State Transfer), identifiées par des URL. Les applications clientes utilisent bien souvent des requêtes HTTP (GET, POST, PUT et DELETE)

afin de procéder aux différentes transactions. Les formats utilisés pour représenter et transporter ces données sont en général ceux que l'on retrouve dans des formats d'échange de données tels que JSON, AtomPub et surtout XML

- **MetaMatrix :** MetaMatrix [73], fournit des outils déclaratifs pour créer une gamme importante de services de données, un référentiel pour stocker les définitions des services de données avec leurs métadonnées associées – et enfin un environnement d'exécution robuste qui assure des performances d'entreprise, l'intégrité et la sécurité des données. MetaMatrix Enterprise offre une solution plus performante et plus rapide pour répondre aux défis posés par les données SOA. La plateforme MetaMatrix Data Services supporte différentes sources de données. Les applications accèdent aux services de données au travers de SQL ou des interfaces Services Web.

- **Xcalia Intermediation Core** (XIC) [75] est une plate-forme d'intermédiation permettant à une entreprise d'accéder à l'ensemble de ses données, de déployer des Applications Métier, à partir de briques composites. XIC réalise du Mapping de Données mais aussi du Mapping de services (incluant les nouveaux Services Web, les services Mainframe, Legacy, etc.). XIC est basé sur les standards SDO (Service Data Objects), JDO (Java Data Objects), EJB (Entreprise Java Bean). XIC permet de faire coexister les applications métier avec des environnements totalement hétérogènes.

- **Composite Software :** La virtualisation de données avec Composite Software [74] permet aux entreprises d'intégrer de manière très flexible leurs données internes. Parmi les fonctionnalités clés de Composite Software', on pourra en particulier citer le fait que :

- Composite apporte aux développeurs bases de données et Java, un environnement de développement relationnel appelé 'Composite Studio'. Pour les développeurs XML / SOA

utilisant des outils de développement, orientés modélisation relationnelle pour construire des vues fédérées et des services de données, on utilisera le module 'Composite Designer'.

- Les Services d'intégration de données écrits et exécutés par le ‘Composite Information Server’ sont parfaitement adaptés pour être opérés à travers l'Internet.

    - Parce que Composite accède, fédère, résume et délivre les données à la demande, il n’est pas nécessaire de prévoir de zone de stockage de ces données sur le réseau étendu.

En outre, Composite supporte de multiples options de sécurité pour garantir un accès et une délivrance sécurisée des données idoines aux utilisateurs authentifiés et approuvés en interne ou en externe. Composite démystifie la complexité de la modélisation de données, au profit de leur utilisation dans le ‘Cloud’.

Discussion :

Dans ces plates-formes, la sémantique d'un service de DaaS est connue tant que le développeur d'applications SOA, reste dans la plate-forme (par exemple AquaLogic). Une fois dehors, par exemple lorsque les Services Web DaaS sont publiés dans un registre de service en dehors des frontières de l’entreprise, il devient difficile de distinguer entre les services que leur sémantique n’est pas définis. Il est nécessaire de compléter ces efforts industriels, en offrant un cadre intégré pour décrire la sémantique déclarative d'un Service Web DaaS (offert par les produits mentionnés) et un modèle de rechercher et de composer des services de fournisseur de données.

2.10. Conclusion :

L’objectif de ce chapitre était de définir le cadre dans lequel nos travaux vont se placer, à savoir Le DaaS permet l'accès aux données des organismes par l'intermédiaire des Services web. L'invocation d'un Service web DaaS a comme conséquence l'exécution d'une requête au-dessus des sources de données.

Nous allons discuter rapidement un certain nombre de systèmes d’intégrations des données : PIAZZA, SIMS, MOMIS, OBSERVER et XYLEME

Nous allons présenter deux projets qui utilisent les services web dans la médiation des données : Active XML et le projet Picsel, puis nous allons présenter quelques travaux relatifs aux services web DaaS.

Chapitre III :

Conception et Implémentation

3.1. Introduction :

Ce chapitre a pour objet de réaliser un ensemble des services web (DaaS) qui permettent de consulter les différentes sources de données : Microsoft SQL Server, Microsoft Access et MySQL Server.

L'objectif de notre conception est de produire un outil logiciel pour prouver et confirmer notre approche théorique. Cette partie nous permet de concrétiser cette conception en présentant les différents aspects techniques. DaaS-CNL est le nom que nous avons souhaité donner à notre application de consultation de sources des données hétérogènes. L'implémentation a été faite en Java – Netbeans 8.0.2.

L'architecture de DaaS-CNL réalisée est une interface web permettre de chercher des noms des postulants sur les différentes sources de données et le résultat est affiché sur une même page web.

Dans la première partie de cette section, nous présentons les missions et l'organisation de la Caisse Nationale du Logement (CNL) de la wilaya de Saida.

Dans la deuxième partie, nous présentons une modélisation de notre application.

Ensuite, dans la troisième partie nous décrivons l'environnement du Développement.

Puis, dans la quatrième partie, nous détaillons le principe du fonctionnement de notre application DaaS-CNL en présentant les résultats de l'évaluation des mécanismes précédemment décrits.

Enfin, nous terminons ce chapitre avec une conclusion.

3.2. Présentation de la Caisse Nationale du Logement (CNL) :

- **Création :**

La Caisse Nationale du Logement (CNL) est un établissement Public à caractère industriel et commercial (EPIC) créé par décret exécutif n° 91-145 DU 12 mai 1991 modifié par le décret exécutif n°94-111 du 18 mai1994.

- **Objectif :**

Le logement : Soutenir sa Production et Aider à son Accession.

- **Missions :**

La Caisse a pour missions et attributions :

- De gérer les contributions et aides de l'état en faveur de l'habitat, notamment en matière de loyers, de résorption de l'habitat précaire, de restructuration urbaine, de réhabilitation et de maintenance du cadre bâti et de promotion du logement à caractère social.
- De promouvoir toute forme de financement de l'habitat et notamment du logement à caractère social, par la mobilisation de sources de financement autres que budgétaires.

A ce titre, elle est chargée notamment de :

participer à la définition de la politique de financement de l'habitat et notamment du logement à caractère social.

- recevoir et gérer les ressources instituées à son profit par la législation et la réglementation en vigueur.
- proposer toutes études tendant à améliorer l'action des pouvoirs publics en direction de l'habitat et notamment du logement à caractère social.
- réaliser toutes études, expertises, enquêtes et recherches liées à l'habitat, apporter son expertise technique et financière aux institutions publiques et organismes concernés, et favoriser les actions d'informations, d'échanges d'expérience et de rencontres pour la promotion et le développement de l'habitat

- **Programmes Financés :**

Parmi les programmes qui nous avons choisissions et utilisés par notre application sont :

❖ **Habitat Rural (HR) :**

L'aide à l'Habitat Rural est destinée aux personnes physiques qui exercent ou résident en milieu rural, voulant construire une nouvelle habitation.

❖ **Logement Social Participatif (LSP) :**

Les promoteurs immobiliers ont la possibilité d'initier des projets de Logements Sociaux Participatifs (LSP) au profit de leurs clients éligibles à l'Aide de l'Etat à l'Accession à la Propriété (AAP).

❖ **Logement Promotionnel Aide (LPA) :**

Le logement promotionnel aidé (LPA) doit être réalisé par un promoteur immobilier dans le cadre de la procédure d'Appel à Manifestation d'Intérêt (AMI) telle que fixée par la réglementation en vigueur.

3.3. Modélisation :

Dans cette section, nous allons décrire avec détails l'architecture de notre application ainsi que les méthodes et les services web utilisés.

Notre application relative au domaine du contrôle fichier auprès de la caisse nationale du logement de la wilaya de Saida (CNL) via un Service web DaaS.

L'architecture réalisée est organisée en trois couches. La première couche contient Trois bases de données qui stockent les données des bénéficiaires des logements :

- **Source 1 :**est constituée d'une base de données de gestions des aides de l'Etat destinée à l'Habitat Rural (HR) créer par un système de gestion de base de données SGBD : Microsoft Access 2013.

- **Source 2 :** est constituée d'une base de données de gestions des aides de l'Etat destinée aux Logement Sociaux Participatifs (LSP) créer par un système de gestion de base de données SGBD : MySQL SERVER.
- **Source 3 :**est constituée d'une base de données de gestions des aides de l'Etat destinée aux Logement Promotionnel Aide (LPA) créer par un système de gestion de base de données SGBD : Microsoft SQL SERVER 2008.

Ces sources, représentent la masse de données relatives aux bénéficiaires des aides de l'état destinées à l'Habitat.

La seconde couche comprend une application qui accède aux bases de données de la première couche (c'est à dire qu'il exécute des requêtes paramétrées sur les bases de données).
Aussi, ces sources reliées entre elles, par un système de Contrôle de Fichier auprès de la CNL pour l'enquête et la vérification qu'un nouveau postulant bénéficié un logement ou non (services Web DaaS constituent la troisième couche).

La troisième couche comprend une interface web permettant à l'utilisateur d'exécuter ces requêtes sous forme de services Web DaaS, afin de chercher sur les trois bases de données et afficher des résultats.
Nous avons utilisé le kit de déploiement fourni avec le GlassFish Web server 4.1 [80] pour déployer nos services Web DaaS.

3.4. Environnement de développement :

Nous avons développé notre application sur une machine avec les caractéristiques suivantes :

- Processeur Intel Core(™) i3 avec une vitesse de 1,70 Ghz,
- 4 Go de mémoire vive,
- Système d'exploitation Microsoft Windows 7 (64bits).

Nous nécessitons les applications de suivi installés, configurés, et fonctionnés dans nos environnements de développement :

- Langage de programmation : Java - NetBeans 8.0.2 Open Source IDE
- Java SE Développement Kit 8 (jdk-8-windows-x64)
- GlassFish Server Open Source Edition 4.1

Les types de base de données à consulter :

- Microsoft SQL Server 2008.
- Microsoft Access 2013.
- MySQL Server 6.3CE.

3.4.1. Le langage de programmation JAVA

- **Définition :**

Java est un langage de programmation à usage général [81], évolué et orienté objet. Il a été mis au point en 1991 par la firme Sun Microsystems23. Il s'agissait de concevoir un langage bien adapté aux environnements de travail en réseau et capable de gérer des informations de nature variées (données numériques, informations sonores et graphiques).

Java est devenue aujourd'hui un phénomène incontournable dans le monde de la programmation, parmi les différentes caractéristiques qui sont attribuées à son succès:

- ✓ L'indépendance de toute plate-forme : le code reste indépendant de la machine sur laquelle il s'exécute. Il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine.
- ✓ Java est également portable, permettant l'application d'être distribuée facilement sans avoir à recompiler le code pour les différents systèmes.
- ✓ Le code est structuré dans plusieurs classes, dont chacune traite une partie différente de l'application.
- ✓ Il assure la gestion de la mémoire.
- ✓ Java est multitâche: il permet l'utilisation de threads qui sont des unités d'exécution isolés.

- **Objectif :**

Le langage JAVA participe pleinement à la création des Services Web.

Généralement, les objectifs du langage JAVA sont les suivants :

- ✓ Simple, orienté objet et syntaxiquement familier.
- ✓ Robuste et sécurité.
- ✓ Utilisant une architecture neutre et être portable (.NET).
- ✓ Disposant de haute performance.
- ✓ Interprété et disposant de processus lèges (multithreading).

3.4.2. NetBeans

- **Un Bref Historique :**

NetBeans a vu le jour en tant que projet d'étudiant en République Tchèque (appelé à l'origine Xelfi) [82], en 1996. Le but était d'écrire un EDI Java semblable à Delphi, mais en Java. Une

compagnie fut créée autour de ce projet, nommé NetBeans. Il y a eu deux versions commerciales de NetBeans, appelées Developer 2.0 et Developer 2.1. Aux alentours de mai 1999, NetBeans sorti une version béta de ce qui aurait dû être Developer 3.0. Quelques mois plus tard, en octobre 1999, NetBeans fut racheté par Sun Microsystems. Après quelques temps de développement supplémentaires, Sun sortit l'EDI FortéFro Java, Edition Communauté - le même EDI qui avait été en béta comme NetBeansDeveloper 3.0.
Il y a toujours eu un intérêt pour l'Open Source chez NetBeans. En juin 2000, Sun mis l'EDI NetBeans en open-source. Ce site est l'endroit où tout cela s'est déroulé.

- **Définition :**

Netbeans est l'environnement de Développement Intégré (EDI) supporté par SUN [82]. Il est particulièrement bien adapté pour le développement d'applications WEB.
L'EDI NetBeans fournit des outils pour construire tous les composants Java EE, ce qui inclut les Enterprise Java Beans (EJBs), les pages web, les servlets, et les Services Web.

- **Les différentes versions :**

Nous avons choisi la dernière version de NetBeans IDE 8.0.2, établie le 18 novembre 2014.

3.4.3. Microsoft Access 

Microsoft Access (officiellement Microsoft Office Access) est un SGBD relationnel édité par Microsoft. Il fait partie de la suite bureautique MS Office Pro.

MS Access est composé de plusieurs programmes : le moteur de base de données Microsoft Jet, un éditeur graphique, une interface de type Query by Example pour manipuler les bases de données, et le langage de programmation Visual Basic for Applications [83].

- **Principales caractéristiques**

MS Access est un logiciel utilisant des fichiers au format Access (extension de fichier *mdb*pour Microsoft *DataBase*(extension *.*accdb*depuis la version 2007)).

Les bases de données produites par Access restent accessibles à tous les langages de programmation qui permettent une connexion à une base ODBC, c'est le cas par exemple sous Java en se servant de la passerelle JDBC-ODBC d'Oracle.

- **Les différentes versions**

Lancement en 1992

- ✓ Access 2003
- ✓ Access 2007
- ✓ Access 2010 disponible depuis le premier semestre de 2010 en version finale
- ✓ Access 2013 disponible depuis la sortie d'Office 2013 / Office 365

Nous avons choisi la dernière version : Microsoft Access 2013.

3.4.4. MySQL SERVER

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, Informix et Microsoft SQL Server.

Son nom vient du prénom de la fille du cocréateur Michael Widenius, My. SQL fait allusion au StructuredQueryLanguage, le langage de requête utilisé [84].

- ➢ **Les différentes versions**

La première version de MySQL est apparue le 23 mai 1995.

Nous avons choisi : MySQL. Version 5.2 distribuée en février 2007

- ➢ **Installation de MySQL WorkBench**

Cet outil est très efficace et permet de manipuler MySQL de manière très simple. Il est vivement recommandé de l'installer. Cette interface graphique est en fait une couche de manipulation de MySQL (création, suppression et modification rapide des Tables).

Nous avons choisissons la dernière version de MySQL WorkBench6.3.CE **,**établie le 28/02/2015.

3.4.5. Microsoft SQL SERVER

Microsoft SQL Server est un système de gestion de base de données (abrégé en SGBD) incorporant entre autres un SGBDR (SGBD relationnel ») développé et commercialisé par la société Microsoft. Il ne fonctionne que sous les OS Windows [85].

- **Les différentes versions**

La version choisie dans notre projet : La version 2008 de SQL Server est sortie en août 2008.

3.5. Principe du fonctionnement de notre système DaaS :

Nous allons montrer les sources de données qui sont mentionné dans la section précédente :

3.5.1 Interface de Login :

Suite à l'exécution de l'application, une fenêtre s'affiche, permettant de saisir le nom d'utilisateur et le mot de passe (Fig 3.1) :

Fig 3.1 : Interface « Login ».

3.5.2. Interface principale :

En cliquant sur le bouton *Ok* de la fenêtre Login, une fenêtre principale s'apparue (Fig 3.2) :

Fig 3.2 : Fenêtre principale (Swing).

3.5.3. Interface de connexion :

En cliquant sur le menu *Outils*, puis *DB Connexions* de la fenêtre Principale, une fenêtre qui gère les connexions s'affiche, cette fenêtre permet d'établir les connexions aux différentes bases de données :

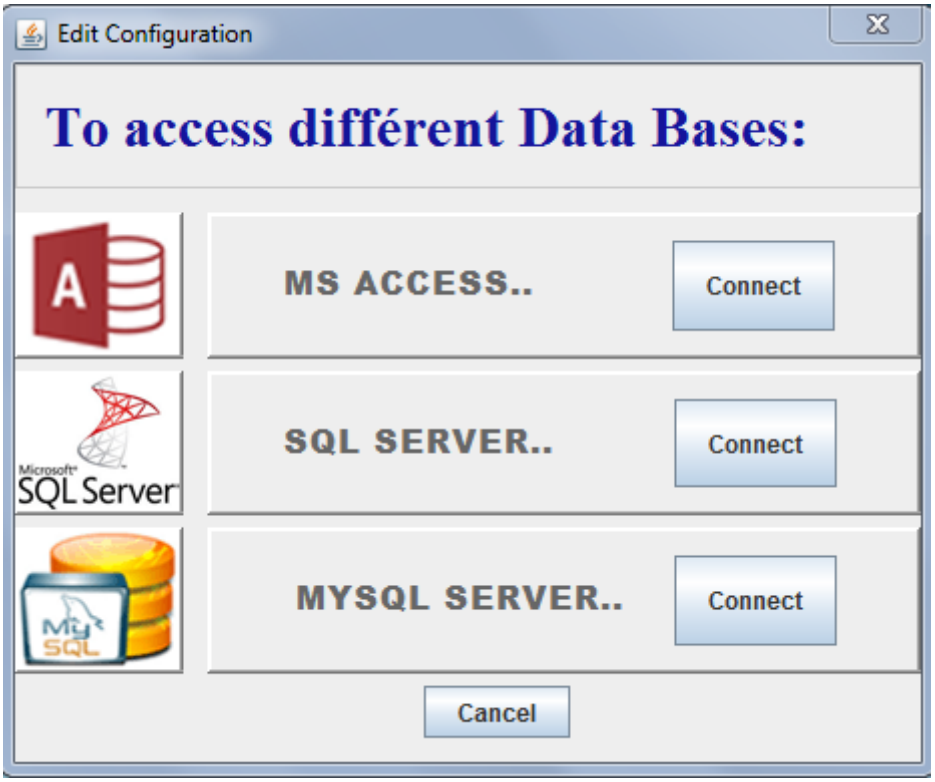


Fig 3.3 : Interface de connexion.

En cliquant sur le bouton *Connect*, la connexion a été établie avec succès (Fig 3.4) :

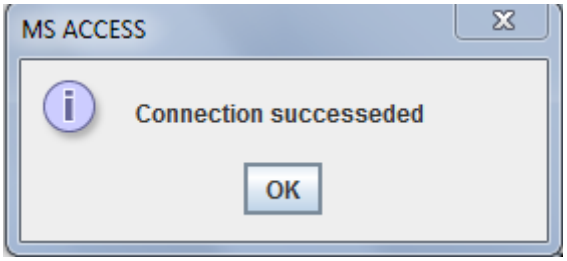


Fig 3.4 : Message de connexion.

3.5.4. Formulaire d'inscription :

Cette interface web permet d'implémenter les méthodes du CRUD (Create, Read, Update et Delete), désigne les quatre opérations de stockage d'informations en base de données Microsoft Access pour les bénéficiaires du programme Habitat Rural (HR) (Fig 3.5):

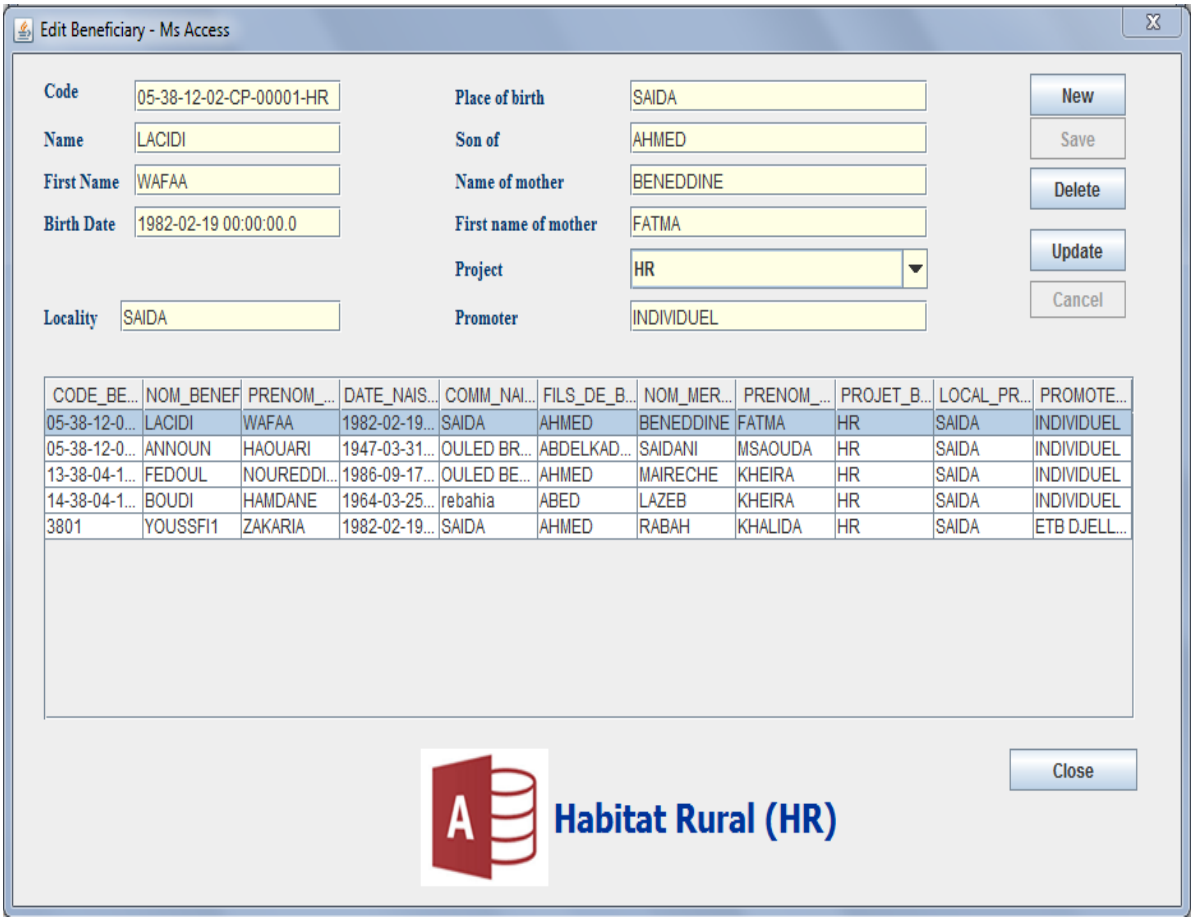


Fig 3.5 : Formulaire d'insertion d'un nouveau bénéficiaire (MS Access).

- Cette interface web permet d'implémenter les méthodes du CRUD (Create, Read, Update et Delete), désigne les quatre opérations de stockage d'informations en base de données MySQL Server pour les bénéficiaires du programme Logements Sociaux Participatifs (LSP) (Fig 3.6):

Fig 3.6 : Formulaire d'insertion d'un nouveau bénéficiaire (MySQL).

- Cette interface web permet d'implémenter les méthodes du CRUD (Create, Read, Update et Delete), désigne les quatre opérations de stockage d'informations en base de données Microsoft SQL Server pour les bénéficiaires du programme logement promotionnel aidé (LPA) (Fig 3.7):

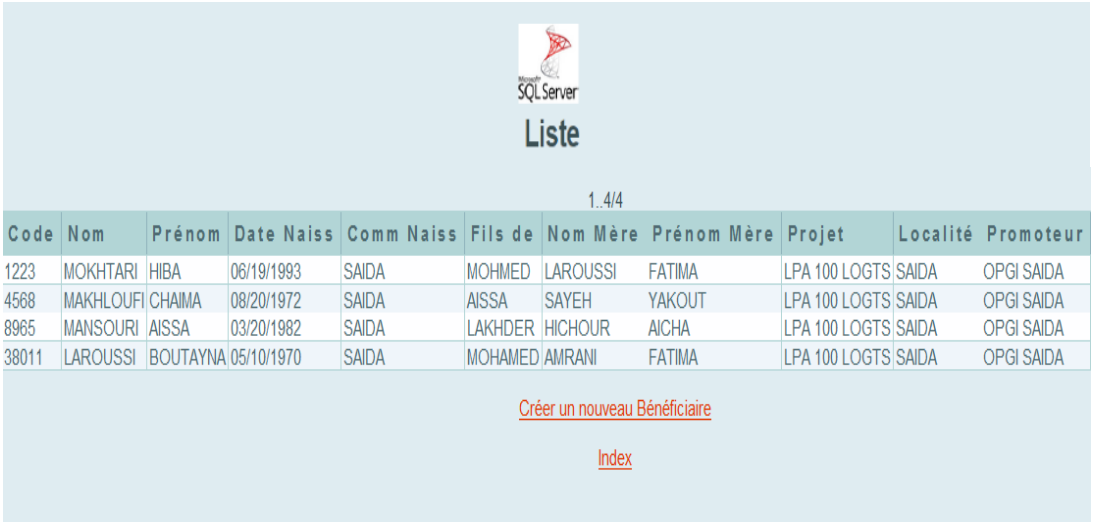


Fig 3.7 : Formulaire d'insertion d'un nouveau bénéficiaire (SQL Server).

3.5.5. Implémentation des services web en Java :

Nous avons créé trois services web, chaque service comporte deux opérations appelées:

- **AllBenef**: pour afficher tous les bénéficiaires.
- **BenByCode**: pour afficher les bénéficiaires par :
  - ✓ Nom_benef : nom de bénéficiaire.
  - ✓ Prenom_benef : Prénom de bénéficiaire.
  - ✓ Date_naiss_benef : date de naissance de bénéficiaire.
  - ✓ Code_benef : code de bénéficiaire.
  - ✓ Projet_benef : projet du bénéficiaire.
  - ✓ Local_projet_benef : localité du projet de bénéficiaire.

Aussi, chaque service web est connecté à une base de données (voir figure 3.8) :

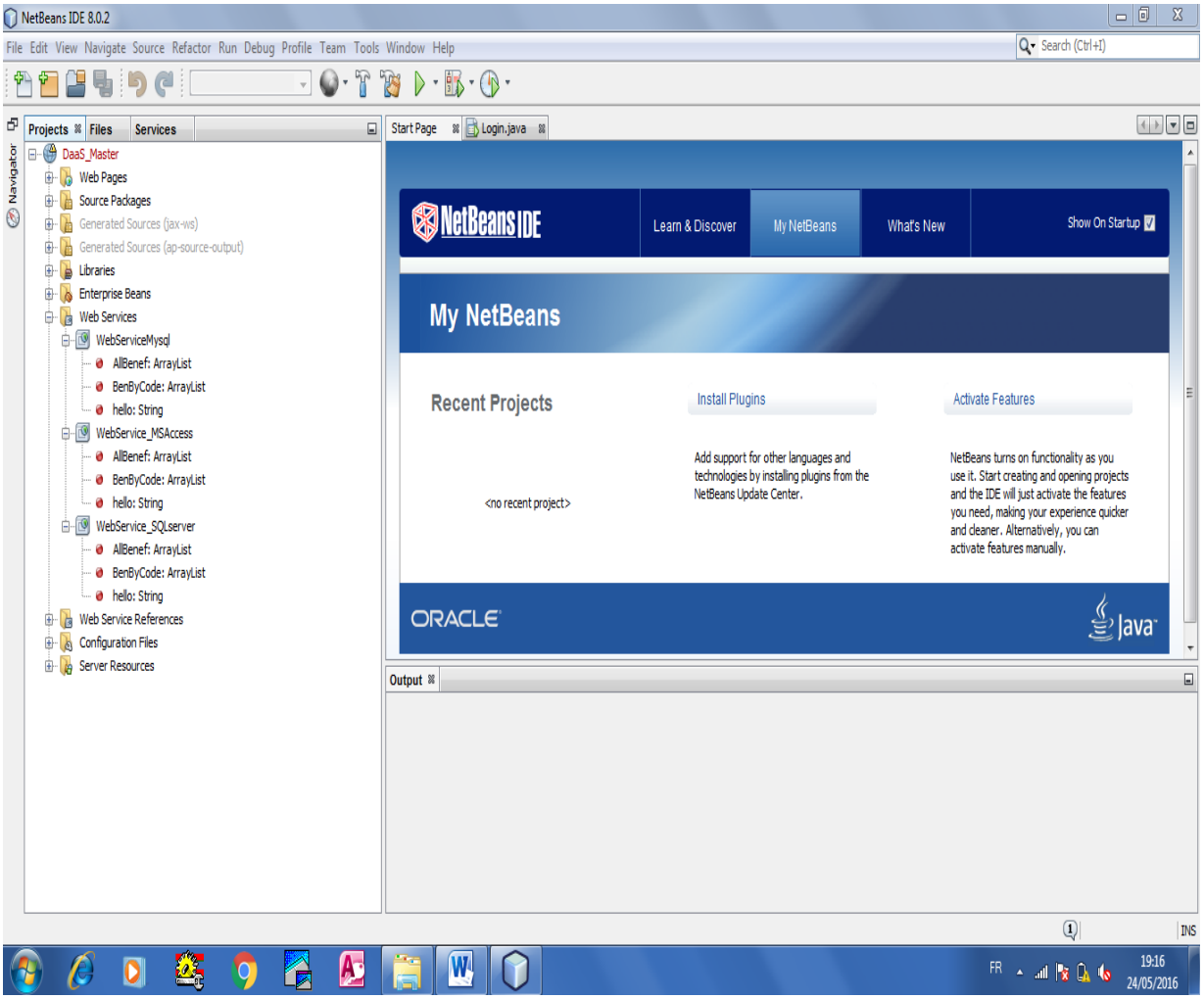


Fig 3.8: Services Web de notre prototype.

Les trois Services Web DaaS utilisés dans notre prototype :

Première Service Web : «WebServiceMysql » qui peut être testé en tapant l'adresse :

http://localhost:8080/DaaS_Master/WebServiceMysql?Tester

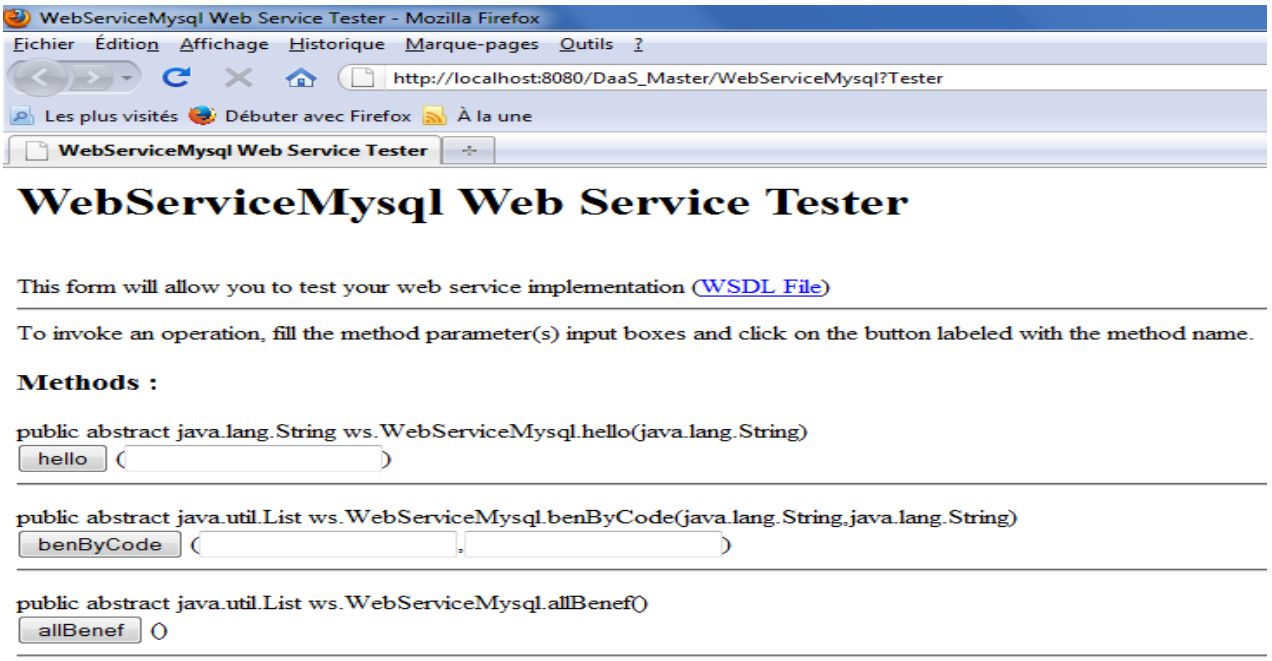


Fig 3.9: Teste de « WebServiceMySql ».

- Deuxième Service Web : «WebService_MSAcces» qui peut être testé en tapant l'adresse :

http://localhost:8080/DaaS_Master/WebService_MSAccess?Tester

Fig 3.10: Teste de « WebService_MSAccess ».

- 3ème Service web : «WebService_SQLservers» qui peut être testé en tapant l'adresse :

http://localhost:8080/DaaS_Master/WebService_SQLserver?Tester

Fig 3.11: Teste de « WebService_SQLserver ».

3.5.6. Présentation de l'interface de notre application :

Nous allons illustrer les démarches pour tester notre application.

L'exemple suivant représente comment invoquer les services web et quel sont les résultats à afficher, nous avons choisi le Service Web « WebServiceMysql », même principe pour les deux autres Services Web « WebServiceMsAccess » et « WebServiceSQLServer ».

Exemple : invoquer le Service Web «WebServiceMysql » pour chercher un nom de bénéficiaire «KADA» sur la base de données MySQL.

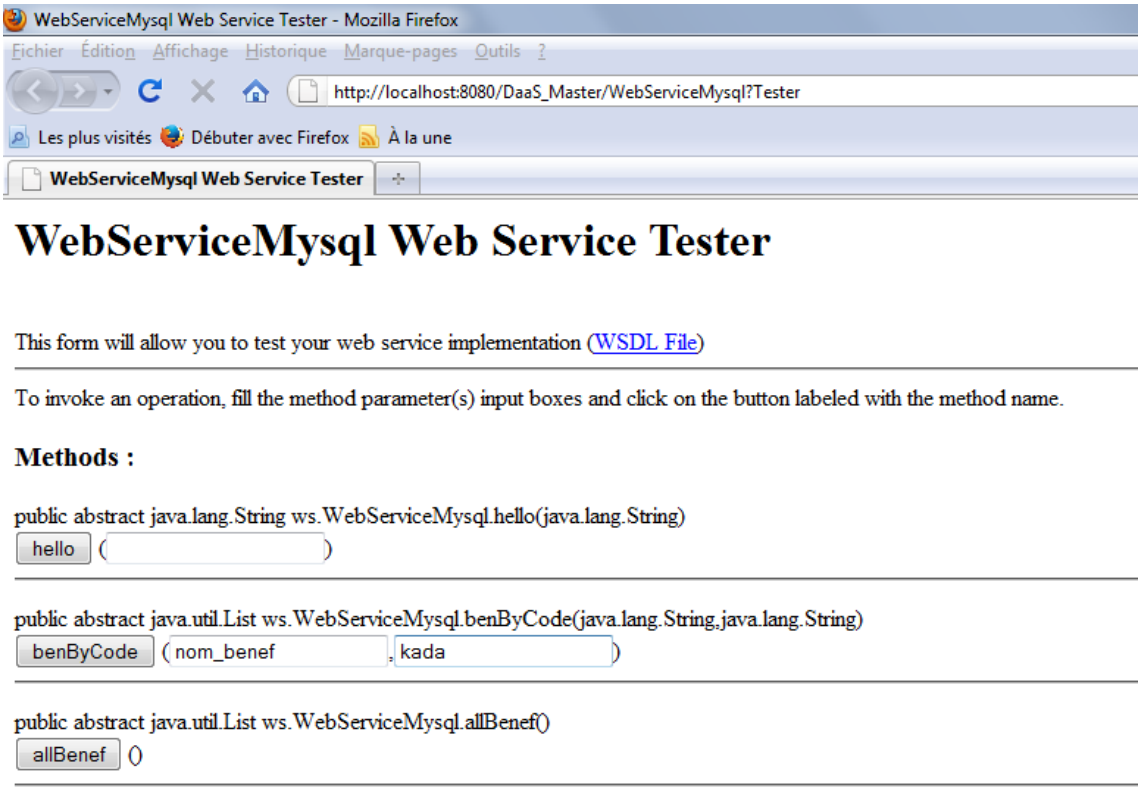


Fig 3.12: Invocation du « WebServiceMysql ».

La page de réponse qui apparut selon les champs qui existent dans la base de données comme « KADA »,

Invocation de L'opération benByCode : avec les paramètres :Nom_benef = KADA ;

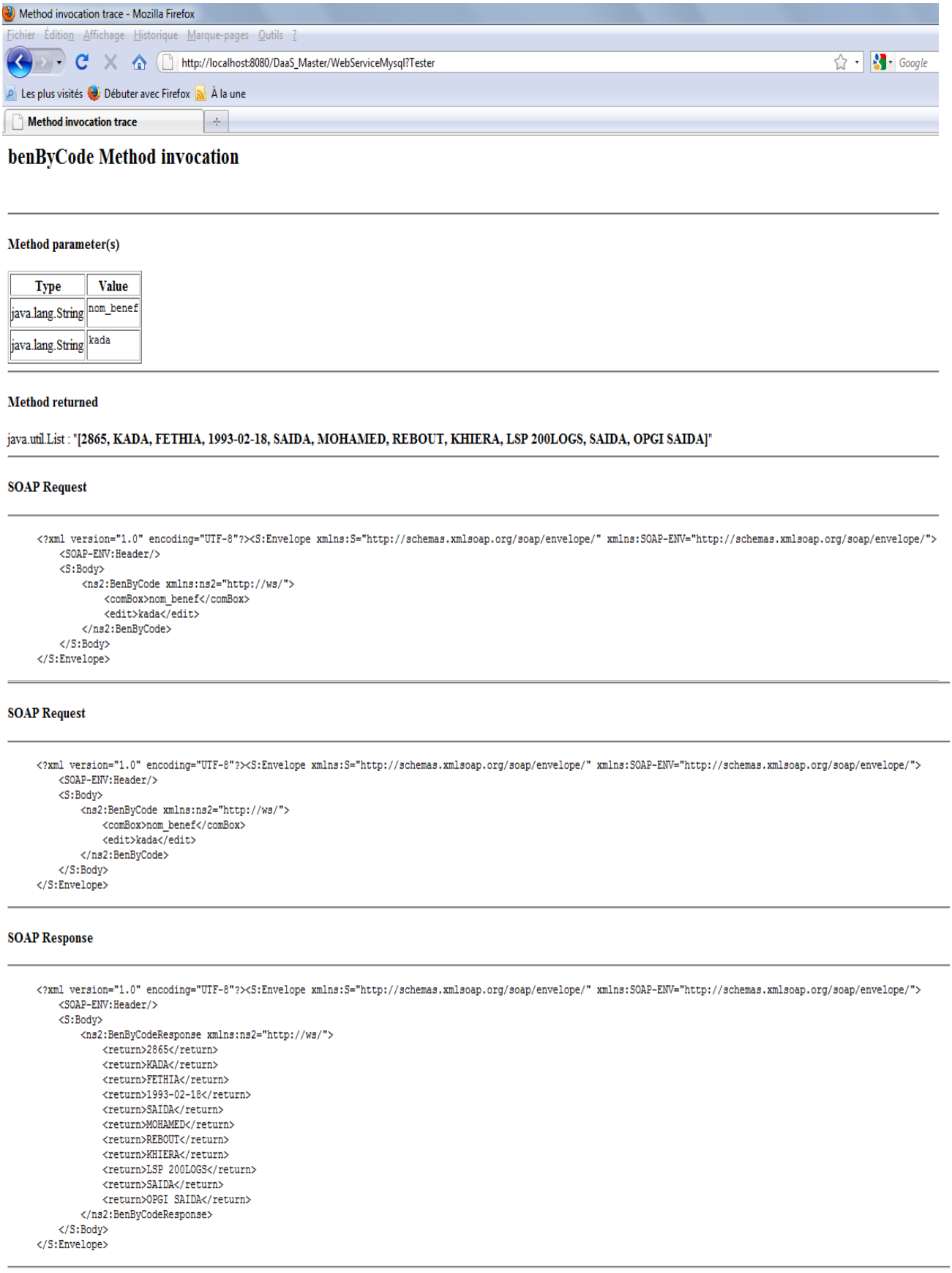


Fig 3.13 : Réponse de « WebServiceMysql ».

3.5.7. Le fichier WSDL :

Permet de décrire un Service Web, et comment l'invoquer (sa méthode d'invocation). Son objectif est :

- Décrire les services comme un ensemble d'opérations et de messages abstraits relié à des protocoles et des serveurs réseaux.

- Permet de décharger les utilisateurs des détails techniques de réalisation d'un appel.

- WSDL est un langage qui standardise les schémas XML utilisés pour établir une connexion entre émetteurs et récepteurs.

Nous pouvons accéder aux fichiers de description « WSDL » de notre Service Web WebSserviceMysql, qui est comme suit :

```
<!--
Published by JAX-WS RI (http://jax-ws.java.net). RI's version is Metro/2.3.1-b419
(branches/2.3.1.x-7937; 2014-08-04T08:11:03+0000) JAXWS-RI/2.2.10-b140803.1500 -->
<!--
Generated by JAX-WS RI (http://jax-ws.java.net). RI's version is Metro/2.3.1-b419
(branches/2.3.1.x-7937; 2014-08-04T08:11:03+0000) JAXWS-RI/2.2.10-b140803.1500 -->
<definitions
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:wsp="http://www.w3.org/ns/wspolicy"
xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://ws/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://ws/" name="WebServiceMysql">
  ▼<types>
    ▼<xsd:schema>
       <xsd:import namespace="http://ws/" schemaLocation="http://localhost:8080/DaaS_Master/WebServiceMysql?xsd=1"/>
     </xsd:schema>
   </types>
  ▼<message name="BenByCode">
     <part name="parameters" element="tns:BenByCode"/>
   </message>
  ▼<message name="BenByCodeResponse">
     <part name="parameters" element="tns:BenByCodeResponse"/>
   </message>
  ▼<message name="hello">
     <part name="parameters" element="tns:hello"/>
   </message>
  ▼<message name="helloResponse">
     <part name="parameters" element="tns:helloResponse"/>
   </message>
  ▼<message name="AllBenef">
     <part name="parameters" element="tns:AllBenef"/>
   </message>
  ▼<message name="AllBenefResponse">
     <part name="parameters" element="tns:AllBenefResponse"/>
   </message>
  ▼<portType name="WebServiceMysql">
    ▼<operation name="BenByCode">
       <input wsam:Action="http://ws/WebServiceMysql/BenByCodeRequest" message="tns:BenByCode"/>
       <output wsam:Action="http://ws/WebServiceMysql/BenByCodeResponse" message="tns:BenByCodeResponse"/>
     </operation>
    ▼<operation name="hello">
       <input wsam:Action="http://ws/WebServiceMysql/helloRequest" message="tns:hello"/>
       <output wsam:Action="http://ws/WebServiceMysql/helloResponse" message="tns:helloResponse"/>
     </operation>
```

Fig 3.14: Fichier WSDL.

3.6. Fonctionnement détaillé de l'interface Web :

Nous avons donné la possibilité à L'utilisateur de choisir la méthode de recherche des informations par l'appel du DaaS, il a le choix entre chercher par AllBenef (tous les bénéficiaires) et chercher par liste de choix (nomBenef ou Prenom_benef ou Date_naiss_benef ou Code_benef ou Projet_benef) :

1) Invocation DaaS par l'opération « AllBenef » :

Suite à d'exécution du projet java, une page web de recherche (index.jsp) est apparue (voir figure 3.15) :

Fig 3.15: Une interface web de notre application.

- En cliquant sur le bouton *Afficher Tous*, les trois Services Web DaaS sont invoqués et une page web HTML s'affiche (figure 3.16) :

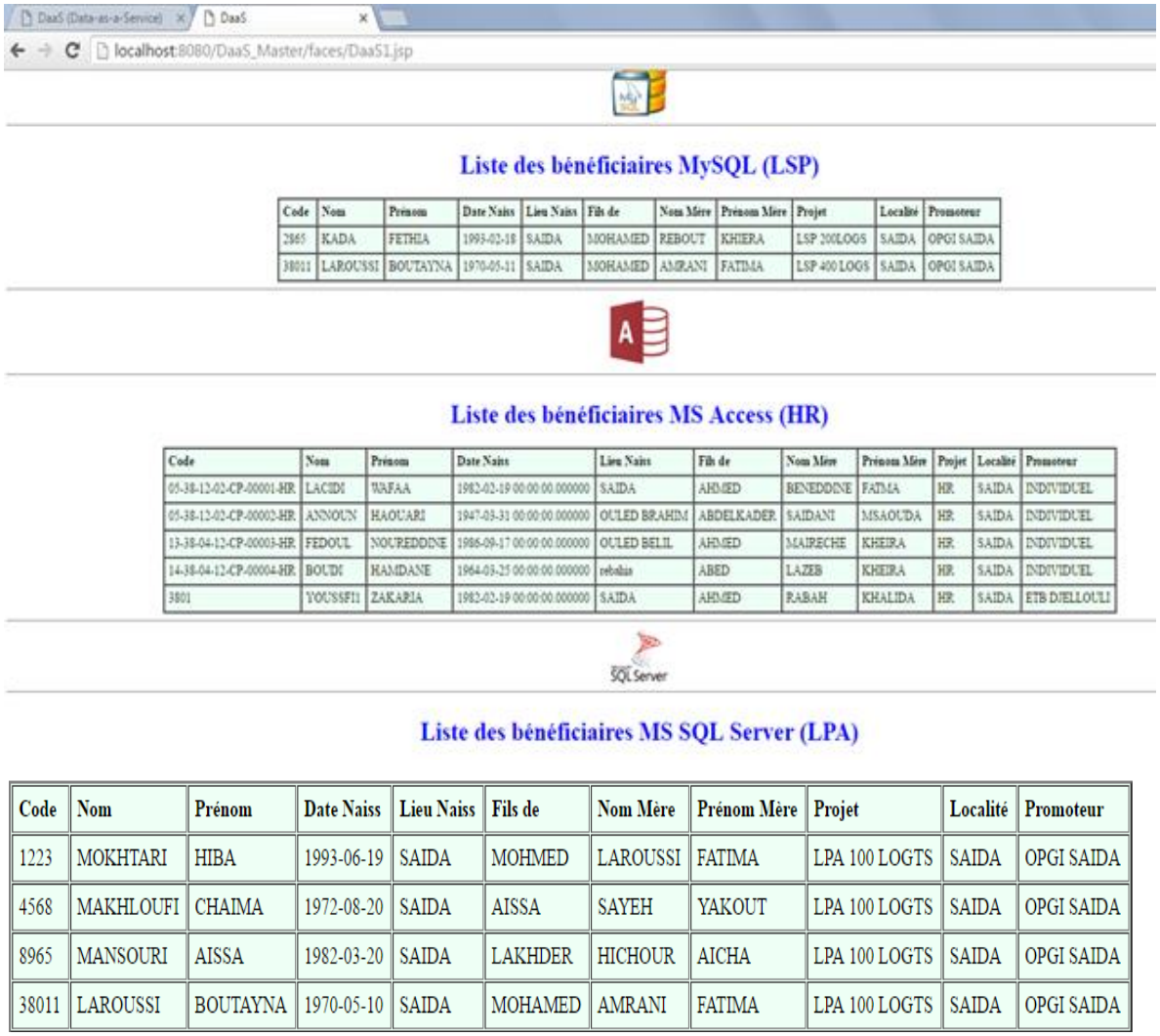


Fig 3.16: Résultat de la requête « Afficher Tous ».

Analyse des résultats obtenus :

La recherche des informations en appelant l'opération « AllBenef() » via les trois services web DaaS (WebServiceMysql, WebServiceMsAccess, WebServiceSqlserver) nous a permet d'extraire convenablement la liste de tous les « bénéficiaires » figurants dans les différents programmes de souscription et répartis sur les trois bases de données de données de notre plateforme.

2) Invocation DaaS par Liste de choix (l'opération « BenByCode ») :

Exemple : chercher le nom de bénéficiaire « KADA », dans les trois bases de données.
Nous avons choisi : *nom_benef*sur la liste de choix et nous avons saisi le mot « KADA » dans une zone de texte, puis en cliquant sur le bouton chercher (voir figure 3.17) :

Fig 3.17: Recherche par nom du bénéficiaire.

Les trois Services Web DaaS sont invoqués et une page web HTML s'affiche (voir figure 3.18) :

Fig 3.18: résultat de la recherche par nom du bénéficiaire.

Analyse des résultats obtenus :

Les résultats obtenus présentent la recherche d'informations en appelants l'opération « BenByCode » via les trois services web DaaS (WebServiceMysql, WebServiceMsAccess, WebServiceSqlserver).

L'ajout des critères de recherche (nom du bénéficiaires, date de naissance, localité,…) nous a permet de filtrer les données pour répondre aux besoins des utilisateurs en cherchant les informations selon le choix de ce critère. Si on prend l'exemple illustré ci-dessus, nous avons filtré la recherche par « nom du bénéficiaire » :KADA, et nous avons eu les informations liées à cette personne auprès des différentes bases de données.

3.7. Conclusion :

Ce chapitre représente un cadre pratique pour l'architecture globale de notre validation.

Nous avons présenté une application permettant la recherche d'information auprès de différentes sources de données hétérogènes en utilisant les Services Web DaaS, pour répondre aux besoins des utilisateurs.

Notre validation consiste à appeler des services web en utilisant des requêtes entrées par l'utilisateur. Chaque requête invoque un Service Web DaaS, et les résultats sont obtenus après l'invocation de ces services.

Conclusion Générale

Ces dernières années ont vu un intérêt croissant pour l'utilisation des services web DaaS comme un support fiable pour l'édition et le partage des données entre les organisations.

dans ce contexte, les requêtes utilisateurs sont définies de façon déclarative, et résolues par la composition des services DaaS.

Dans ce mémoire, nous avons présenté une application ainsi que les méthodes et les services web DaaS utilisés.

Notre application relative au domaine de contrôle du fichier auprès de la caisse nationale du logement de la wilaya de Saida (CNL) via les Service Web DaaS.

L'architecture réalisée est organisée en trois couches :

La première couche comporte trois sources de données : base de donnes Microsoft Access, base de données MySql Server et base de données Microsoft Sql Server. Ces sources, représentent la masse de données.

La seconde couche comprend une application qui permet d'accéder aux bases de données de la première couche (CRUD).

La troisième couche comprend une interface web permettant à l'utilisateur d'exécuter des requêtes sous forme de services Web DaaS, afin de chercher des informations dans les trois bases de données et afficher les résultats obtenus, nous avons créé trois services web DaaS, dont chaque service est connecté à une base de données particulière.

Nos perspectives pour les futurs travaux :

Nous envisagerons à appliquer ce concept dans le domaine de contrôle du fichier national (enquête et vérification d'un postulant bénéficiant un logement) auprès de Ministère de l'Habitat, de l'Urbanisme et de la Vie (MHUV) via les Services Web « DaaS ». Notre perspective est de permettre à l'utilisateur d'accéder aux bases de données de 48 wilayas (LSP, LPA, HR, AADL, Auto-Construction, OPGI, FNPOS, DUC, CNL,…etc).

Liste d’abriviation

SOA : Service Oriented Architecture

IBM : International Business Machines Corporation

W3C : World Wide Web Consortium

***URI* :**Uniform Resource Identifier

XML : eXtensible Markup Language

Http : HyperText Transfer Protocol

SMTP : Simple Mail Transfer Protocol

FTP : File Transfer Protocol

UDDI : Universal Description Discovery and Integration

API : Application Programming Interface

URL : Uniform Resource Locator

Bibliographie

[1] "Object Management Group (OMG). The Common Object Request Broker : Architecture and Spécification" *http://www.omg.org/cgi-bin/doc?formal/2011-11-01.pdf.*

[2] "Microsoft Corporation. The Component Object Model Spécification. 2011," *http://www.microsoft.com/com/.*

[3] M.Brahimi, "Contribution à la coopération dans les systèmes d'information basés Web," *PhDthesis, Université Mentouri de Constante,* 2010.

[4] H. H. D.Booth, F.McCabe, and Al, "Web services architecture. W3C Working Group Note 11 February 2014," *http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.*

[5] F. C. A.Gustavo, and Al, "Web Services: Concepts, Architectures and Applications," *edSpringer 2004*.

[6] IBM, "Présentation des services Web," *http://pic.dhe.ibm.com/infocenter/rsasehlp/v7r5m0/index.jsp?topic=%2Forg.eclipse.jst.ws.doc.user%2Fconcepts%2Fcws.html*.

[7] A. B. D.Austin, and C.Ferris, "Web Services Architecture Requirements. W3C WorkingGroup Note 11 February 2014," *http://www.w3.org/TR/wsa-reqs/*.

[8] W3C, "Extensible Markup Language (XML) 1.0 (Fourth Edition) 2006,"*http://www.w3.org/TR/2006/REC-xml-20060816/*.

[9] H.Cervantes, "Vers un modèle à composants orientées services pour supporter la disponibilité dynamique," *PhD thesis, Université Joseph Fourier - Grenoble 1, Mars 2004.*

[10] K.Heather, "Web Service Conceptual Architecture (WSCA 1.0). IBM Software Group, Mai

2011," *http://www-3.ibm.com/software/solutions/webservices/pdf/WSCA.pdf*.

[11] "K.Hubert and V.Monfort, Les Web Services techniques, démarches et outils XML, WSDL, SOAP, UDDI, Rosetta, UML. edDunod, Mars 2003,"

[12] "P.Kellert and T.Toumani, Les Web Services Sémantiques. Action spécifique 32 CNRS/STIC, Octobre 2003," *http://www.irit.fr/journal-i3/hors_serie/annee2004/revue_i3_hs2004_01_07.pdf*.

[13] S.Bhiri, "Approche Transactionnelle pour assurer des compositions fiables de services web,"

PhD thesis, Université Henri Poincaré - Nancy 1, Octobre 2005.

[14] "J.De Roey, Web Services : Beyond the peer-to-peer architecture," *PhD thesis, Université libre de Bruxelles, 2007*

[15] "K.Gottschalk, S.Graham, and H.Kreger, Introduction to Web services architecture. IBM Systems Journal, 41 (2) :170–177," 2002.

[16] C.F.Goldfard, "The SGML Handbook. 1990," *Clarendon Press*.

[17] "N.Mitra and Y.Lafon, SOAP Version 1.0 Part 0 : Primer (Second Edition) 27 Avril2007," *http://www.w3.org/TR/soap12-part0/*.

[18] "M.C.Daconta, L.J.Obrst, and K.T.Smith, The semantic Web : A guide to the future of XML, Web Services, and Know ledge Management. Wiley Technology Publishing," 2003.

[19] "E.Christensen, F.Curbera, and Al, Services Description Language (WSDL) 1.1 15 Mars2001," *http://www.w3.org/TR/wsdl*.

[20] "L.Clement, A.Hately, C.V.Riegen, and Al, UDDI Version 3.0.2, UDDI Spec Technical Committee Draft, 19 Octobre 2004," *http://uddi.org/pubs/uddi_v3.htm*.

[21] L.V.Céline, "Sélection et composition de services Web pour la génération d'applications adaptées au contexte d'utilisation," *PhD thesis, Laboratoire d'informatique de Grenoble, Novembre 2008*.

[22] D.Berardi, "Automatic service composition. Models, Techniques and Tools," *PhD thesis, Univarsité la sapienza, Décembre 2005*.

[23] Y. N. G. MIKE, "information-as-a-service: what's behind this hot new trend?," Tech. report, Forrester Research, 2007.

[24] "http://compositesoftware.com/solutions/soa.shtml."

[25] T. T. D. NGDOC, "Fédération de données semi-structurées avec XML. Thèse de

Doctorat en Informatique, de l'Universit de Versailles Saint -Quentin-en-Yvelines. Juin

2003."

[26] X. Baril, "Un modèle de vues pour l'intégration de sources de données XML : VIMIX. Thèse de doctorat en informatique, de l'université des sciences et techniques du languedoc. dec2003."

[27] A. RAHNI, "AMIDHA : Une approche médiateur d'intégration de sources de données hétérogènes et autonomes. Mémoire de stage effectué à l'ENSMA , Université de Poitiers. Juillet 2005. ."

[28] "L.Bellatrache, G.Pierra, D.Nguyen Xuan,D.Hondjack, Intégration de sources de données autonomes par articulation a priori d'Ontologies. A paraitre dans : actes du XXIIème - congrés INFORSID, Biarritz, 25-28. Mai 2004."

[29] "Christine Parent, Stefano Spaccapietra. Integration de bases de donnees :

Panorama desproblemes et des approches. Ingenierie des systemes d'information,

Volume 4, N°3, 1996.."

[30] "Goh, C. H. Representing andreasoning about semantic conflicts in heterogeneousnformation systems.PhD thesis, MITSloan School of Management. (1997)."

[31] "Ladjel Bellatreche, Guy Pierra, Dung Nguyen Xuan, Dehainsala Hondjack. An automated information integration technique using an ontology-based database approach. Proceedings of CE'2003, Special track on data integration in Engineering, Madeira, Portugal, Juillet, 2003. ."

[32] "S.S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, D. Ullman, J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources.

Proccessings of the 10th Meeting of the Information Processing Society of Japon, pages : 7-18, 1994. ."

[33] "V. H. Inmon. Building the data warehouse. 3rd edition USA. 1996 ".

[34] "Wiederhold G. Mediators in the architecture of future information systems. IEEE computers,25(3), p 38-49, March 1992.

[35] "Nejdl,W.,Wolf,B.,Qu,C.,Decker,S.,Sintek,M.,Naeve,A.,Nilsson,M.,Palmer,M.Risch,T.

2002a). Edutella: a p2p networking infrastructure based on rdf. pages 604-615."

[36] "Rousset, M.-C., Adjiman, P., Chatalic, P., Goasdoue, F. et Simon, L. (2006). Somewhere in the semantic web. In Wiedermann, J., Tel, G., Pokorny, J., Bielikova,M.et Stuller,J.,éditeurs: SOFSEM 2006 - Proceedings, pages 84-99. Springer."

[37] "Ives Z, Florescu D, Friedman M,Levy A, Weld D. An adaptative query execution engine for data integration. In proceedings of the ACM SIGMOD International Conference on Management of Data, Philadelphia, p.299-310, June2004.

[38] "H. Wache, T. V¨ogele, U. Visser, H. Stuckenschmidt,G. Schuster, H. Neumann,and S.

H¨ubner. Ontology-based integrationof information - a survey of existingapproaches.Proceedings of the InternationalWorkshop on Ontologies and InformationSharing, pages 108–117, August2001."

[39] "AKLOUF Youcef, Intégration du modèle d'ontologies PLIB et des Services Web dans les échanges inter-entreprises, Thèse Présentée pour l'obtention du diplôme de Doctorat en informatique, Université des Sciences et de la Technologie Houari Boumediene, 2007."

[40] "M. D. Siegel. Context interchange : Newfeatures and formalisms for the intelligent integration of information. ACMTransactions on Information Systems,17(3) :270–293, 1999."

[41] "E. Mena, V. Kashyap, A. P. Sheth, andA. Illarramendi. OBSERVER : An approach for query processing in global information systems based on interoperation across pre-existing ontologies. In Conference on Cooperative Information Systems, pages 14–25, 1996."

[42] "P. R. S. Visser, M. Beer, T. Bench-Capon, B. M. Diaz, and M. J. R.Shave.Resolving ontological heterogeneity in the kraft project.10th InternationalConference on Database and Expert Systems Applications (DEXA'99),pages 668–677, September 1999."

[43] E. a. B. Rahm, "Rahm, E. and Bernstein, P. (2011). On matching schemas automatically. the

VLDB Journal , 10(4):334–350.."

[44] "Chawathe, S. S., Garcia-Molina, H., Hammer, J., Ireland, K., Papa-konstantinou, Y., Ullman, J. D., and Widom, J. (1994). The tsimmis project: Integration of heterogeneous information sources. In Proceedings of the 10th Meeting of the Information Processing Society of Japan , pages 7–18.."

[45] "François Goasdoué, F., Lattès, V., and Rousset, M. C. (2000).The use of carin language and algorithms for information integration: The picsel system.International Journal of Cooperative Information Systems, 9(4):383–401.."

[46] "Beneventano, D., Bergamaschi, S., Castano, S., Corni, A., Guidetti, R., Malvezzi, G., Melchiori, M., and Vincini, M. (2000). Information integration: The momis project demonstration. In Proceedings of the 26th International Conference on Very Large Data Bases

, VLDB '00, pages 611–614, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.."

[47] "Genesereth, M. R., Keller, A. M., and Duschka, O. M. (1997). Infomaster: an information integration system. In Proceedings of the 1997 ACM SIGMOD international conference on Management of data , SIGMOD '97, pages 539–542, New York, NY, USA. ACM.."

[48] "Levy, A. Y., Rajaraman, A., and Ordille, J. J. (1996). The world wide web as a collection of views: Query processing in the information manifold. In Proceedings of the International Workshop on Materialized Views: Techniques and Applications (VIEW'1996) pages 43–55.."

[49] "Friedman, M., Levy, A., and Millstein, T. (1999). Navigational plans for data integration. In Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence , AAAI '99/IAAI '99, pages 67–73, Menlo Park, CA, USA. American Association for Artificial Intelligence.."

[50] "Halevy, A. Y., Ives, Z. G., Madhavan, J., Mork, P., Suciu, D., and Tatarinov, I. (2003). The piazza peer data management system. IEEE Transactions on Knowledge and Data Engineering."

[51] A. R. e. B. Hurson, M.W.(1991), "Multidatabase sysytemes: An advanced concept in handling distributed data. Advances in Computers, 32:149-200."

[52] T. e. R. Landers, R.L. , "An overview of multibase. H.-j. Schneider, editor, Second

International Symposium on Distributed Data Bases ,pages 153-84."

[53] W. F. Cody, Haas,L.M;, Nblack,W., Arya, M., Carey, M.j., Fagin, R., Flickner, M., Lee, D., Petkovic, D., Schwarz, P.M., II,j.T., Williams, J.H. et Wimmers, E. L.(1995). , "Querying multimedia data from multiple repositoires by content: the garlic project. pages 17-35."

[54] E. E. Mena, Illarramendi, A. Kashyap, V. et. Sheth .A.P, " OBSERVER : An approach for query processing in global information systems based on interoperation across preexisting ontologies. 8(2):223-271.."

[55] "Rousset, M.-C., et Reynaud, C. (2003). Picsel and Xyleme: Two illustrative information integration agents. In Klusch, M., Bergamaschi, S., Edwards, P. et Petta P., éditeurs: Intelligent Informatio Agents - The AgentLink Perspective, pages 50-78. Springer.."

[56] M. Arenas, Kantere, V., Kementsietsidis, A., Kiringa, I.,Miller, R. et Mylopoulos, J. (2003). , "The hyperion project: From data integration to data coordination. SIGMOD Record, 32 (3).".

[57] C. B. B. Amann, I. Fundulaki, and M. Scholl. , "Ontology-based integration of xml web resources. In International Semantic Web Conference (ISWC), Sardinia, Italy, 2002.."

[58] G. GARDARIN., "Web Services et Médiation. PriSM Laboratory, UVSQ, Versailles.."

[59] M. Arenas, Kantere, Y. Knoblock, C.A., Shen,W(1994). , "Query reformulation for dynamic information integration. Journal of intelligent information systemes. 99-130."

[60] C. B. B. Amann, I. Fundulaki, and M. Scholl. , "Query xml sources using an ontology-based mediator. In CoopIs, 2002.."

[61] B. M. a. B. D. OUKSEL Aris M "Composing and optimizing data providing web services, 17th international conference on World Wide Web, Pages 1141-1142, ACM 2008."

[62] VACULIN Roman, CHEN NERUDA Huajun, Roman, SYCARA Katia, "Modeling and Discovery of Data Providing Services," ICWS (International Conference on Web Services), IEEE, pp.54-61, , 2008."

[63] Z. W. Huajun CHEN, Heng WANG, and Yuxin "MAO.RDF/RDFS-based relational database integration.2013."

[64] S. a. W. SCHREINER, "A survey on web services composition, "IJWGS, vol. 1, no. 1, pp. 1-30, 2005."

[65] "http://www.activexml.net."

[66] O. Benjelloun, "Active XML: A data centric perspective on Web services, PhD thesis, Universtity of Paris XI, 2004."

[67] B. O. ABITEBOUL S., MILO T, "The Active XML project: an overview , VLDB Journal, Volume 17 , Issue 5, Pages: 1019 – 1040, ISSN:1066-8888, 2008."

[68] G. G. R. Ch, "Construction semi-automatique d'ontologies à partir de DTDs relatifs à un même domaine. 13èmes journées francophones d'Ingénierie des Connaissances. Rouen. 2012."

[69] A. Levy, Rousset, M.C, "Combining Horn Rules and Description Logics in CARIN", In Artificial Intelligence Journal, September 1998, Vol. 14, p. 165-209."

[70] "Gloria-Lucia GIRALDO GÓMEZ, Construction automatisée de l'ontologie de systèmes médiateurs - Application à des systèmes intégrant des services standards accessibles via le Web, THÈSE présentée pour obtenir le grade de Docteur en Sciences de l'Université Paris XI Orsay, Spécialité : Informatique, 2005."

[71] M. Carey, "Data delivery in a service-oriented world: the BEA aquaLogic data services platform. In: Proc. The 2006 ACM SIGMOD international conference on Management ofdata (2006)."

[72] "Microsoft, ADO.NET Data Services (also known as Project Astoria) (2007), http://astoria.mslivelabs.com/."

[73] "REDHAT.:MetaMatrix Enterprise Data Services Platform (2007), http://www.redhat.com/jboss/platforms/dataservices/."

[74] "Composite Software, SOA Data Services Solutions, technical report http://compositesoftware.com/solutions/soa.html,(2008),."

[75] "X. Inc, Xcalia Data Access Services (2009), http://www.xcalia.com/products/xcalia- xdasdata-access-service-SDO-DAS-data-integration-through-web-services.jsp."

[76] "Williams, K., Daniel, B.: SOA Web Services - Data Access Service. Java Developer's

Journal (2006).".

[77] C. G. Khouloud Boukadi1, Zakaria Maamar3, Djamal Benslimane2, "Transactions on Large- Scale Data- and Knowledge-Centered Systems I, Volume 5740/2009 Springer, ISBN 978-3-

642-03721-4, Pages 91-115, aug 2009."

[78]

"http://www.306.ibm.com/software/data/db2/extenders/xmlext/.

"

[79] "http://msdn2.microsoft.com/en-

us/library/ms345123.aspx."

[80] "https://glassfish.dev.java.net/."

[81] "Cay S. Horstmann et Gary Cornell: Au coeur de Java 2 volume 2 Fonctions avancées. Livre

Java.book Page I Mardi, 10. mai 2005."

[82] "http://www.netbeans.org."

[83]

"https://products.office.com/access.

[84] "http://www.mysql.com."

[85] "http://www.microsoft.com/sqlserver."