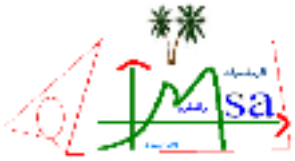
République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université de Ghardaia

Faculté des Science et de Technologie
Département de Mathématique et Informatique
&
Laboratoire de Mathématiques et Sciences Appliquées



MÉMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER académique**

En : Informatiques

Spécialité : Systèmes Intelligents pour l'Extraction de Connaissances

intitulé :

Intégration de sources de données hétérogènes dans les entrepôts de données

Rélisé par : Sara DJEBRIT

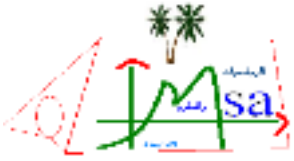
Soutenu publiquement le 04/07/2019, devant le jury composé de :

Dr.OULAD NAOUI Slimane	MCB	Univ. Ghardaia	Président
M.BOUHANI Abdelakader	MAA	Univ. Ghardaia	Examinateur
M.MAHJOUB Youcef	MAA	Univ. Ghardaia	Examinateur
M.KECHIDA Khaled	MAA	Univ. Ghardaia	Encadreur

Année universitaire : 2018/2019

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université de Ghardaia

Faculté des Science et de Technologie
Département de Mathématique et Informatique
&
Laboratoire de Mathématiques et Sciences Appliquées



MÉMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER académique**
En : Informatiques
Spécialité : Systèmes Intelligents pour l'Extraction de Connaissances
intitulé :

Intégration de sources de données hétérogènes dans les entrepôts de données

Réliser par : Sara DJEBRIT

Soutenu publiquement le 04/07/2019, devant le jury composé de :

Dr.OULAD NAOUI Slimane	MCB	Univ. Ghardaia	Président
M.BOUHANI Abdelakader	MAA	Univ. Ghardaia	Examinateur
M.MAHJOUB Youcef	MAA	Univ. Ghardaia	Examinateur
M.KECHIDA Khaled	MAA	Univ. Ghardaia	Encadreur

Année universitaire : 2018/2019

DÉDICACE

À toutes les personnes qui veulent à vivre au sommet de la montagne, dépassent toutes les obstacles et les challenges pour qu'ils grimpent.

REMERCIEMENTS

Avant tout, je remercie Allah tout puissant de m'avoir donné la force et la volonté pour achever ce travail.

Il m'est particulièrement agréable d'exprimer ma gratitude envers les personnes qui m'ont apporté leur soutien et qui m'ont aidée de près ou de loin à effectuer ce travail.

Tout d'abord à Monsieur **KECHIDA Khaled** qui m'a posé de travail sur le thème de ce mémoire.

mes plus sincères remerciements sont adressés à monsieur **BELLAOUAR Slimane** pour ses conseils et ses encouragements au durée de mes études universitaires .

Mes remerciements sont addressé aussi aux membres du jury qui m'ont honoré d'accepter l'évaluation de ce travail.
Je remercie :
monsieur **Ladjel BELLATRECHE** professeur à l'ENSMA / LISI - Université de Poitiers qui m'a fait l'honneur d'avoir accepté de voir ce mémoire et m'a guidée par ses conseils et ses encouragements.

mes enseignants à l'Université de Ghardaia qu'ils m'ont enseigné.

Enfin, je remercie, de tout mon cœur, tous mes proches pour leurs encouragements et aides.

Résumé

Des nombreuses applications sont réalisées par les méthodes d'intégration des données, qui sont dans la gestion des informations des entreprises, le domaine médical, les systèmes d'information geographiques et E-Commerce.
Le système d'intégration des données fournit des tâches pour la manipulation des sources des données d'une manière transparente et efficace.
Plusieurs essais avec des grands efforts réalisent un système d'intégration ayant un schéma global parmi des dizaines des sources des données, en assurant la qualité d'information qui extrait selon ce système et qui satisfaire les besoins des connaissances.
Dans ce mémoire nous étudions le problème du regroupement de sources de données hétérogènes qui sont différentes dans l'aspect sémantique, et dans la localisation des endroits pour construire un schéma global, puis de fournir les possibilités d'interroger ce schéma, dont le but, d'obtiennent des informations satisfaisantes et correctes. Dans ce contexte, nous réalisons un système d'intégration qui permet de fusionner des sources des données des différents formats ou des structures dans un seul schéma global, puis nous appliquons des traitements sur les requêtes pour obtiennent des informations à partir de ce schéma via une interface d'utilisateur. nous utilisons dans le système d'intégration le médiateur par l'approche LAV(Local As View), et le processus d'ETL(Extract, Transform, Load), par ailleurs, la manipulation des requêtes sur l'entrepôt de données résulte, donne une description d'accès au schéma global, et d'extraction l'information, l'optimisation et la robustesse de recherche d'informations selon les sources des données originales à l'aide d'architecture VISS(Virtual Integration Support System) avec les techniques de recherche d'information, aprés l'étude sur la perfomance des algorithmes utilisés dans notre méthodes proposées nous avons 70% comme une valeur générale qui présente la validation et l'efficacité de ces algorithmes.

Mots clés :
intégration de données, système d'intégration de données,le mediteur, ELT(Extract, Transform, Load), LAV(Local As View), algorithme de rapprochement, entrepôts de données, sources de données, données hétérogènes,

الملخص
يتم تحقيق العديد من التطبيقات من خلال دمج بيانات متضمنة في مجالات مثل:
إدارة معلومات الشركات ، المجالات الطبية ، نظم المعلومات, تطبيقات الجرافيك و التجارة الالكترونية
يوفر نظام دمج البيانات طرقا لمعالجة مصدر البيانات بطريقة شفافة وفعالة وقد تحققت عدة دراسات لتحقيق نظام دمج له مخطط كامل و عام لبعض مصادر البيانات ، لضمان جودة المعلومات المستخرجة وفقا لهذا النظام

في هذه المذكرة ندرس مشكلة تجميع مصادر البيانات الغير المتجانسة والتي تختلف في مفاهيم الدلالية ومواقع الاماكن ثم توفير إمكانيات استجواب هذا المخطط العام من أجل الحصولا على معلومات صحيحة.
في هذا السياق نبني نظام يسمح باندماج مصادر البيانات من الشكالا أو الهيا كل المختلفة في مخطط عام واحد، ثم نطبق استجوابات على المخطط العام للحصولا على معلومات منه لنزودها في واجهة للمستخدم

في مراحل الدمج نستخدم في نظام الدمج وسيط يعتمد على تقنيتي
LAV و ELT
بالضافة الى معالجة الاستجوابات على مستودع البيانات. النتائج المتحصل عليها تعطي وصفا لكيفية وصول الاستجوابات الى المخطط العام و مراحل استخراج المعلومات منه بالاستعانةبتقنية
VISS

و كذلك توظيف لتقنيات البحث عن المعلومة لنفس السياق ,بعد دراسة فعالية و كفاءة الخوارزميات المطبقة في هذه المذكرة تحصلنا على نسبة70% كمعدل لعملها بشكل صحيح

الكلمات المفتاحية:
دمج البيانات ، نظام دمج البيانات ، وسيط ،ETL ، LAV,
مصادر البيانات ، البيانات الغير المتجانسة,خوارزمية المقاربة ، مستودعات البيانات

Abstract

Many of applications realize with data intégration methods including the information of companies manager, medicals data domains, Geo-graphical informations systems and E-Commerce application.
Data integration system provide technics to manipulate transparently data source with efficient way.
there are few of tryings to get perfect global format with data intégration system which base into dozens of data sources for extract the information in height quality and satisfy the knowledge need.
The aim of this research is to study the problem of combining heterogeneous data sources wish its différent in semantic content and it's not in the same places, so we realize dataintégration systems by merge a few of data sources have a différent formats and structures into one global format, and try to treated the queries that extract informations from this format to agent interface.
The subject of this report will be a previous study followed by an implementation of mediator by local as view approach, and use the ETL processes , the query treatment on the global format is about extract the information and optimize queries effects in this case we unplug the VISS architecture with the technicals of informations retrieval,we compute the accuracy of used algorithms in our aim and we find as general value 70% wish present the validation and efficient case in proposed methods.

Key Words :
data intégration, data intégration system, mediator, ELT(EXtract,Transform,Load), LAV(Local As View), rapprochement algorithm, data warehouse.

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION

Dans ces jours-là domaine d'informatique ayant le besoin d'accéder, procéder, traiter et spécialement d'intégrer des données dans les sources diverses et variantes. Plusieurs systèmes d'intégration ont été proposés dans la littérature de gestionnaire de données, en citant : TSIMMIS développé au département de l'informatique à l'Université de Stanford, Picsel développé par l'Université Paris Sud, MOMIS développées dans l'université de Modena et Reggio Emilia et l'Université de Milan, etc. Le principe de l'intégration des données, est, d'unifier et com- biner ses différents formats et structures de sources, qui s'appellent les données hétérogènes dans un schéma global qui donne une seule interface. L'hétérogénéité de données fournit des problèmes pour l'intégration des données qui peut classer en deux parties : l'intégration de données hétérogènes, manipulation des requêtes. L'issue de sources hétérogènes produit l'objectif de fournir une vue globale de l'information, qui étant donné les entrepôts de données ou 'datawarehouse' en utilisant les concepts et le tech- niques de base pour unifier les différents formats de données et optimiser les requêtes d'extraire les informations. Ce travail étudie le sujet d'intégration de sources hétérogènes dans l'entrepôt de données, qui permet de réaliser un système d'intégration de données ayant des différents formats et structures, via une vue globale présentée par les entrepôts des données, et fournit une interface pour manipuler les requêtes. On a réalisé une médiation avec l'approche LAV en utilisant les techniques de rap- prochement pour analyser les similarités des informations de sources. En cas de traiter les requêtes nous utilisons les techniques qui constater au l'état de l'art. L'organisation de ce mémoire est comme suit : Le premier chapitre s'intéresse aux définitions et concepts qui relie aux l'intégrations des données, avec des exemples de modèles qui réaliser les systèmes d'intégration. Le deuxième chapitre donne une description sur l'évaluation de requêtes au système d'intégration, et les conditions qu'il faut vérifier pour la robustesse des requêtes. Le troisième chapitre contient des études historiques consernés avec notre sujet , en posant les résul- tats,et les grands lignes de ces études. Dans le dernier chapitre nous présentons notre travaux, qui permet de réaliser un système d'intégration, en implémentant les algorithmes de rapprochements et le processus de ETL, ainsi la construction d'un moteur de recherche à l'aide des tâches d'ar- chitecture VISS, pour évaluer

les requêtes sur l'entrepôt de données résult. Enfin, nous évaluons notre réslultats, par étudier la performance des méthodes utlisées. Finalement,ce mémoire termine par une conclusion.

CHAPITRE

1

INTÉGRATION DE DONNÉES

1.1 Introduction

L'intégration des données a été proposée comme une solution pour les problèmes des hétérogénéités de données et les diversités des sources. Un système d'intégration de données permet d'offrir à l'utilisateur une vue globale et transparente des informations issues de sources hétérogènes et distribués sans qu'il soit amené à savoir leur source ou la façon. les deux approches

- le médiation de données c'est l'approche virtuelle qui présente les étapes de traiter les sources via la vue globale,
- l'entrepôt de données c'est l'approche matérielle qui regrouper les sources des données hétérogènes fournit par une interface unifie dans le but d'interrogation des requêtes.

La modilisation(Mapping) de mediation ont été fait par plusieurs vues variantes parmi les sources des données utilisées, les plus répondus modèles sont : LAV et GAV, la plupart des études basant sur ces deux modèles avec les possibilités de faire optimiser ses modèles ou de faire la combinaison entre eux. Ce chapitre présente les concepts reliés aux intégrations de données, nous posons les définitions des approches principales et les différents types des hétérogénéités de données ; nous étudions les modèles de Mappings et leurs caractères par mettre des exemples des projets réalisés dans ce contexte.

1.2 Intégration de données

L'intégration de données est un ensemble des processus qui permet de fusionner plusieurs sources de données à travers une interface unifiée, après l'élimination de tous les conflits entre les données et de présenter d'une manière cohérente, on accéde dans les informations par une vue globale qui traite les interrogations des requêtes.[4],[5]
Un système d'intégration est composé en trois grands couches :

1. Une couche de schéma global : elle peut composer par un entrepôt de données(approche matérialisée) et par un médiateur (approche virtuelle), cette couche permet aux utilisateurs d'interroger les requêtes en accédant aux sources à travers d'un schéma global.
2. Une couche des adaptateurs/loaders ou wrappers : l'adapteur extrait les données via un schéma global, il permet d'interagir les sources selon les autres couches.L'adaptateur c'est l'unique moyen pour accéder aux sources de donnes et extraire les informations.
3. Une couche de sources données : elle compose par les sources de données sélectionnés pour intégrer.

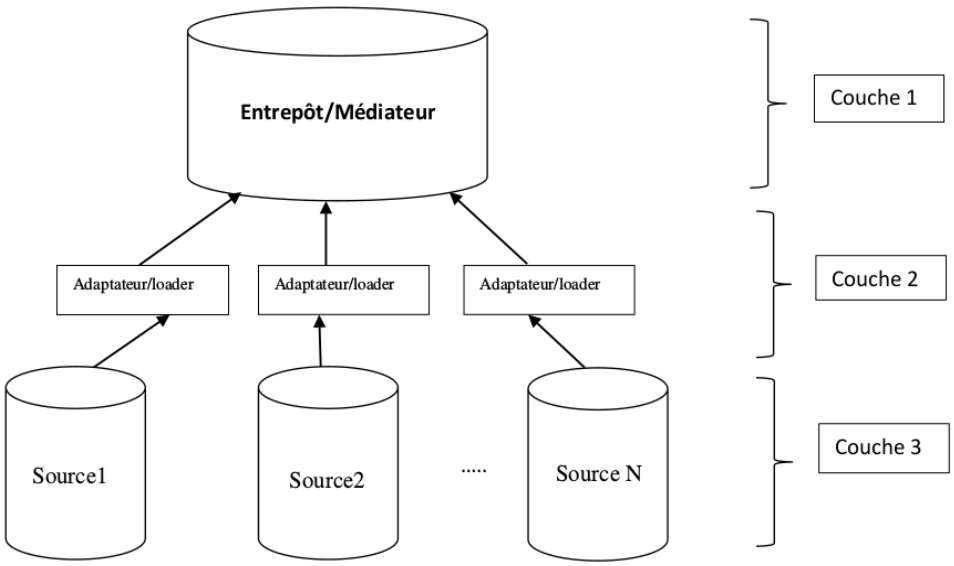


FIGURE 1.1 – L'architecture générale d'un système d'intégration

D'autre manière le système d'intégration peut être définit par un triple (G,S,M) :
G : représente le schéma global.
S : représente l'ensemble de sources données.
M : c'est le Mapping entre le schéma global et l'ensemble de sources données.

1.3 Hétérogénéité des données

nous parlons cela sur le problèmeème complexe dans les pluparts des systèmes d'intégration, telque les différences de ces systèmes basent sur les différents étapes ou les différentes vues ou les différentes manières qui ont été conçu par ce problèmeème, donc il faut éétudier les façons d'hétérogénéité des données qui effectuer lors de l'intégration. les deux grands types d'hétérogénéité :

1. **Hétérogénéité Sémantique** :
   c'est-à-dire on exprime le même concept mais avec des significations différentes[6], on peut définir deux types des hétérogénéités sémantiques ;

   — hétérogénéité sémantique lié au schéma c'est de donner même terme par des terminologies différentes ;
   — hétérogénéité sémantique lié aux données c'est le cas de données qui ayant des différentes origines, des différents contenants ; et différentes structures et utilisent des conventions différentes.[7]

2. **Hétérogénéité structurelle** :
   on l'appelle aussi hétérogénéité des schémas, c'est-à-dire il existe des mêmes concepts avec de différentes présentations, où en utilisant des modèles différents pour décrire les mêmes données ou d'une manière inverse.

Il existe quatre types principaux d'hétérogénéité sémantiques(conflits sémantiques) [2] :

— Conflits de représentation : c'est le cas d'utiliser des différentes schémas ou des différentes propriétés pour décrire le même concept,
— Conflits de noms (termes) : Ces conflits se trouvent dans le cas où on utilise soit des noms différents pour le même concept ou propriété (synonyme), soit des noms identiques pour des concepts (et des propriétés) différents (homonyme),
— Conflits de contextes : dans ce cas on donne des différentes représentations d'un seul objet dans les sources de données tel que chaque source ayant un contexte local pour ce objet,
— Conflits de mesure de valeur : on trouve dans ce cas l'utilisation des unités différentes pour mesurer les valeurs des mêmes concepts.

1.4 Entrepôt de données

un entrepôt de données a été défini comme une méthode de stockage de données intégrées pour être utilisées dans les systèmes, en offrant des méthodes d'analyse comme OLAP(On-Line Analytical processing), un cube OLAP contient des données servant à faire des analyses de données provenant de différentes sources hétérogènes et distribuées. Cette analyse est effectuée en organisant les données de manière multidimensionnelles .[8],[9]
L'entrepôt de données a été réalisé par passer des étapes qui correspondent au processus ETL (Extract,Transform and Load)[10], c'est a dire :

— l'extraction des données à partir d'un source,
— la transformation des données : effectuer un ensemble des opérations sur lesquelles on élimine tous les conflits des sources de données par nettoyer et formater et filtrer ces sources,
— Chargement / Load de données : c'est-à-dire charger les données qui sont préparées par l'état précédent au niveau de l'entrepôt de données, ces données peuvent être utilisées par des outils décisionnels tels qu'OLAP.

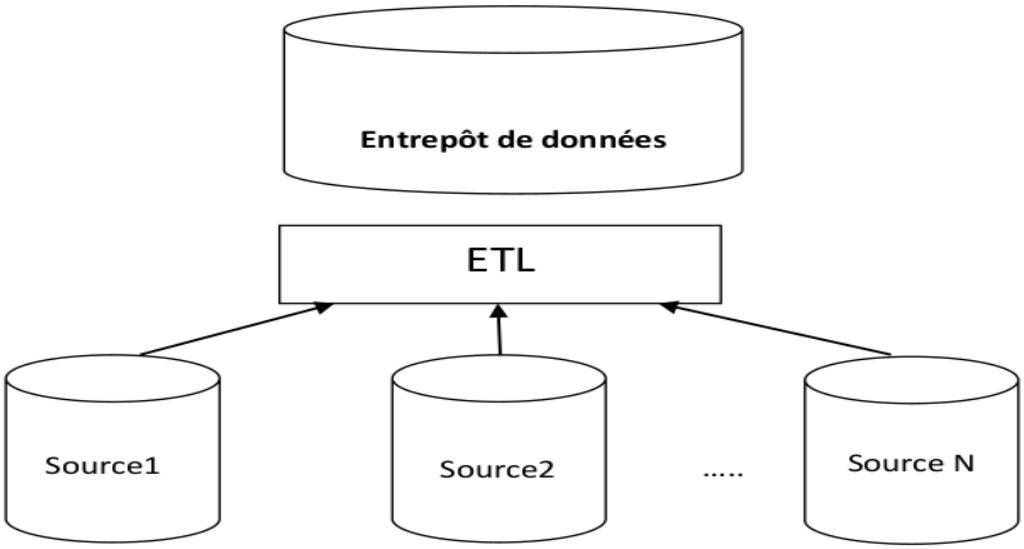


FIGURE 1.2 – Architecture d'un entrepôt de données

1.5 Médiation

L'approche médiation c'est l'une des composantes plus importantes dans les systèmes d'intégration il contient les pluparts des tâches du systèmeème d'intégration, il a été introduire par wiederhold en 1992. [8]. La construction de médiation constitue généralement par ;

— Les sources de données.
— Les wrappers (adaptateurs) qui permettent de traduire les requêtes dans la façon compréhensible par les sources de données et retourne les réponses dans le format de médiateur, il joue le rôle d'intermédiaire entre les sources et le médiateur.
— Le médiateur c'est le principal élément dans la médiation, il permet de collecter les données depuis les sources et appliquer la fusion entre eux, ainsi il valide l'interrogation de schéma global.

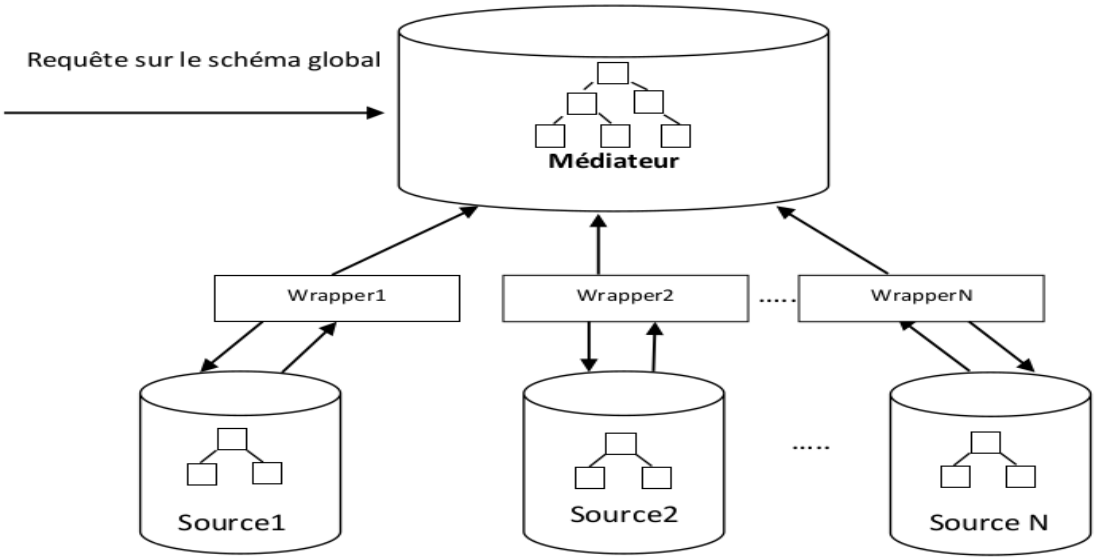


FIGURE 1.3 – Architecture d'une médiation

1.6 Mapping (Correspondance)

C'est la modélisation des liens entre le schéma global et les sources de données, il existe quatre types des modélisations pour définir les correspondances entre les sources et le schéma global.[11],[12],

1.6.1 Types de mappings

on a quatre types de mappings pour la modélisation des correspondances :

Global-As-View (GAV)

L'approche GaV a été la première qui être proposée pour intégrer des informations. Elle consiste sur la façon semi-automatique en fonction des sources de données à intégrer puis elle connecte aux différentes selon les prédicats du schéma global proposé, L'approche GaV appelle les relations globales données comme des vues sur les prédicats des schémas des sources ont intégré. [2]
l'inconvénient de cette approche c'est que la mise à jour au niveau de sources de données nécessite d'appliquer la mise à jour au schéma global qui est plus coûteuse.

Local-As-View (LAV)

cette approche est contrairement à l'approche GAV, elle consiste de représentée tous les sources de données comme un schéma global pour cela il n'y a pas une séparation entre les sources de données et le schéma global.
l'inconvénient de cette approche c'est que la formulation et la réécriture des requêtes peuvent être

plus difficile pour obtenir une réponse, l'interrogation se fait dans un taux plus long pour l'union des réponses dans une seule réponse satisfaire.

Global-Local-as-View (GLAV)

c'est la combinaison entre les deux précédentes approches selon leurs avantages, tels qu'on dispose un schéma global et local dans la vue générale.

Both as view (BAV)

c'est permis de faire la correspondance dans les deux directions c'est-à-dire de schéma global vers les sources de données et de manière inverse, l'interrogation des requêtes se réaliser par l'approche GAV.

1.7 Traitement des requêtes dans l'intégration de données

Le système d'intégration de données hétérogènes fait partie en deux majoraux buts :

— la validation de schéma d'intégrer de sources hétérogènes ;

— le traitement de requête a travers le schéma global.

Le traitement de requêtes est un ensemble des opérations concernées d'amélioration la qualité de requête d'une manière mathématique pour optimiser la pertinence de reponcés et de réduit le taux d'obtention les informations, la manipulation de requête se fait par deux parties : répondeur de requête (Query answering) pour effectuer le gain de répondeur et réécriture de requête (Query Rewriting)dans le but d'optimiser le contexte de requête. les fonctions qui focalisent le traitement des requêtes présenter comme des formules mathématiques, elles calculent les possibilités d'augmenter la pertinence de réponce et réduire les bruits(les conflits entre les requêtes et les réponces). Ce chapitre introduit un préliminaire sur le traitement de requête dans le domaine d'intégration de données, puis nous étudions la fonction de réécriture de requête en présentant l'aspect mathématique, ainsi nous posons les études sur la fonction répondeur de requête.

1.7.1 Préliminaire

les formules mathématiques donnent des définitions pour présenter les deux fonctions de traitements de requêtes : répondeur de requête, réécriture de requête.

Soit :

A : un alphabet ou signature qui contient un ensemble de relations qui s'appelle : n-aire relation telle que $n \subseteq \mathbb{N}$ c'est un sous-ensemble de produit cartésien de A^n qui lui donne un ensemble de n-tuple des éléments de A, donc c'est un ensemble finie de symboles reliés entre eux.

σ : est une instance de A, présente une structure relationnelle qui ayant un paire : $I = \langle \Delta^i, \cdot^i \rangle$, tel que Δ^i c'est le domaines des objets et $\cdot^i$ c'est une fonction d'association de chaque symbole de relation.

$\mathcal{R} = \{R_1, R_2, R_3, ...R_n\}$: est un base de données de signatures qui contiennent les symboles de signatures, $\mathcal{R} - extension$ est un base de données des instance notées par $\mathbf{D}$.
$\mathcal{V} = \{V_1, V_2, V_3, ...V_k\}$: est une vues de symboles qui n'appartient pas dans $\mathcal{R}$, chaque V associé par une vue de définition $V^{\mathcal{R}}$.
$V^{\mathcal{R}}$: est une formule d'un langage $\mathfrak{L}$ pour donner une expression de V par $\mathcal{R}$ selon les termes de base de données de symboles. $\mathcal{V} - extension$ sont des instances de vue notés $\mathbf{E}$.
$\mathcal{Q}$: requête, est une fonction de structure relationnelle depuis S signatures celui qui associer chaque structure relationnelle I par la relation $\mathcal{Q}(I)$ qui s'appelle la réponce de $\mathcal{Q}$ par I.
$\mathcal{Q}$ référence au $\mathcal{R}$ comme un base de requêtes et ainsi elle référence au $\mathcal{V}$ comme de vue de requêtes.
Les réponces de requêtes dans le système d'intégration de données vérifiées selon l'ensemble de base de données, le traitement de requêtes basant sur $\mathcal{V} - extension$.
Il existe deux tâches dans le traitement de requêtes : répondeur de requête (Query Answering) et réécriture de requêtes (Query Rewriting).[13]

1.7.2 Répondeur de requête(Query Answering)

C'est une fonction donne un ensemble des termes dans une base de données évalué selon les instances de vue ($\mathcal{V} - extension$), cette fonction fait l'extraction de termes et utiliser les instances de $\mathcal{V} - extension$ pour filtrer les termes les plus pertinents par apport la requête. Le gain de cette fonction s'étudier dans les deux façons :

- Cas exact : c'est á dire le cas de termes contenants dans la base de données et qui sont équivalents avec les instances de $\mathcal{V} - extension$ sans l'application de filtrage.
- Cas bruit : c'est le cas opposite c'est á dire $\mathcal{V} - extension \subseteq les\ termes\ extrées$.

Les réponces de requête $\mathcal{Q}$ sous le cas bruit c'est l'existence de tuples t dans l'ensemble $\mathbf{E}$ tel que $t \subseteq \mathcal{Q}(\mathbf{D})$, ce cas donne :

$$AN^{bruit}_{\mathcal{Q},\mathcal{V}} = \bigcap\{\mathcal{Q}(\mathbf{D})|\mathbf{D}, \mathbf{E} \subseteq \mathcal{V}_{\mathcal{R}}(\mathbf{D})\} \tag{1.1}$$

Les réponces de requête $\mathcal{Q}$ sous le cas exact, c'est l'existence de tuples t un ensemble $\mathbf{E}$ tel que $t \subseteq \mathcal{Q}(\mathbf{D})$,[?] ce cas donne :

$$AN^{exact}_{\mathcal{Q},\mathcal{V}} = \bigcap\{\mathcal{Q}(\mathbf{D})|\mathbf{D}, \mathbf{E} = \mathcal{V}_{\mathcal{R}}(\mathbf{D})\} \tag{1.2}$$

1.7.3 Réécriture de requêtes

C'est la fonction qui fait la reformulation de requêtes $\mathcal{Q}$, c'est á dire une transformation de langage utilisé dans $\mathcal{Q}$ via le langage de base de données $\mathbf{D}$ pour faciliter la tâche de répondeur de requêtes.
la fonction de réécriture des requêtes consacre sur deux cas qui donne une description de son travail, il existe donc deux cas pour effectuer la réécriture des requêtes[13] :

Réécriture maximale

Soit : $V^{\mathcal{R}}$ une vue appartient dans $\mathcal{V}$, $\mathbf{E}$ c'est le $\mathcal{V} - extension$, la réécriture maximale se donne comme suit :

$$\mathcal{Q}'_r(\mathbf{D}) \supset \mathcal{Q}_r(\mathbf{D}) \leftrightarrow \mathcal{Q}'_r(V^{\mathcal{R}}(\mathbf{D})) \supset \mathcal{Q}_r(V^{\mathcal{R}}(\mathbf{D})) \tag{1.3}$$

tel que :
$\mathcal{Q}'_r$: c'est la réécriture de requête en langage de vue $\mathcal{V} - extension$.
$\mathcal{Q}_r$: c'est la réécriture de requête en langage de base de données .

Réécriture exact

c'est le cas d'équivalence entre la réécriture de requête en langage vue et le réécriture de requête en langage de base de données, il considère comme un cas idéal ou le gain d'information est pertinent. la formule descriptive dans ce cas :

$$\mathcal{Q}_r(V^{\mathcal{R}}(\mathbf{D})) = \mathcal{Q}(\mathbf{D}) \tag{1.4}$$

Le traitement de requête dans le système d'intégration de données se compose en deux tâches principales : la réécriture de requête et le répondeur de requête, telque on présente sous forme mathématique qui donne une description pour ces tâches. la réécriture de requête ê applique la reformulation de requête appartient de langage utilisé dans la base de données en mesurant la performance de cette opération depuis les instances de vues. Le répondeur de requête fournit une reponce aux requêtes doit être des termes filtrés de base de données via les instances de vues. Les traitements de requête sont des tâches expliquées par les formules mathématiques qui calcule les gains de ces tâches il génère donc deux cas pour chaque tâche : un cas d'exact, et un cas de bruit ou de maximal.

1.8 Exemples de projets d'intégration de données

1.8.1 Le projet TSIMMIS

c'est le premier projet a étè réalisé dans l'intégration de données, il base sur l'approche GAV en donnant un moyen d'intégrer des informations hétérogènes . dans ce projet le wrapper assemblé un modèle de données orienté objet appelé Modèle d'échange d'objet (Objet échange Model : OEM), il est chargé de réécrire une requête en sous-requêtes selon l'interrogation qui utilise un langage de règles (MSL, Mediator Spécification langage).[13] Ce projet utilise plusieurs médiateurs pour reçoivent les réponses de wrappers et générer le plan de requête.[14] Ce projet utilise plusieurs médiateurs pour reçoivent les réponses de wrappers et générer le plan de requête.

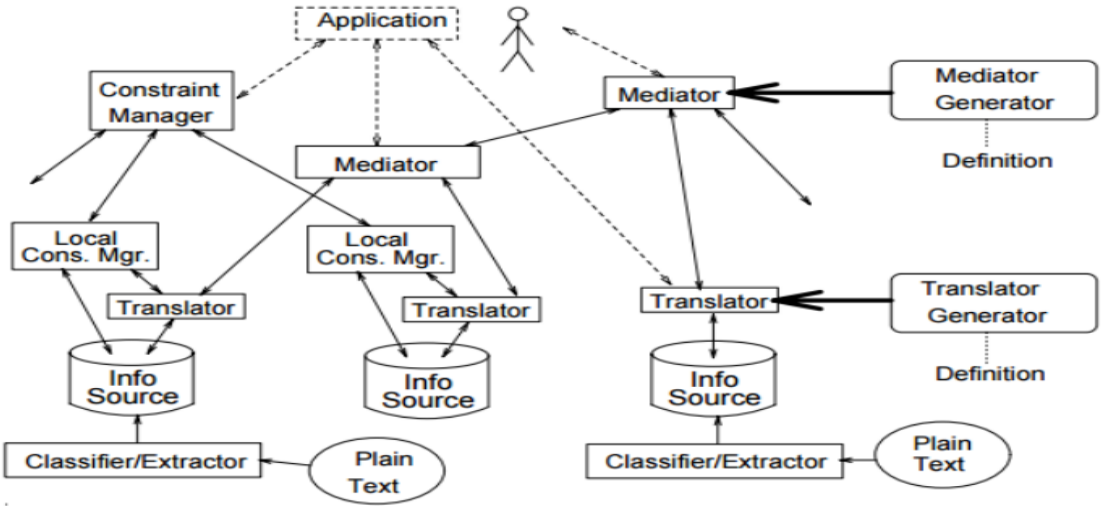


FIGURE 1.4 – Architecture générale de TSIMMIS

1.8.2 Le système MOMIS (Mediator Environnement for Multiple Information Sources)

l'intégration de données a étè réalisé basant sur un thesaurus dérivé de la base de données lexicale *WordNet* qui vise à intégrer les données de manière semi-automatique. dans ce système le wrapper assemble les données et traduit les informations des sources vers une représentation commune basée sur le modèle (ODL-I3).[15]

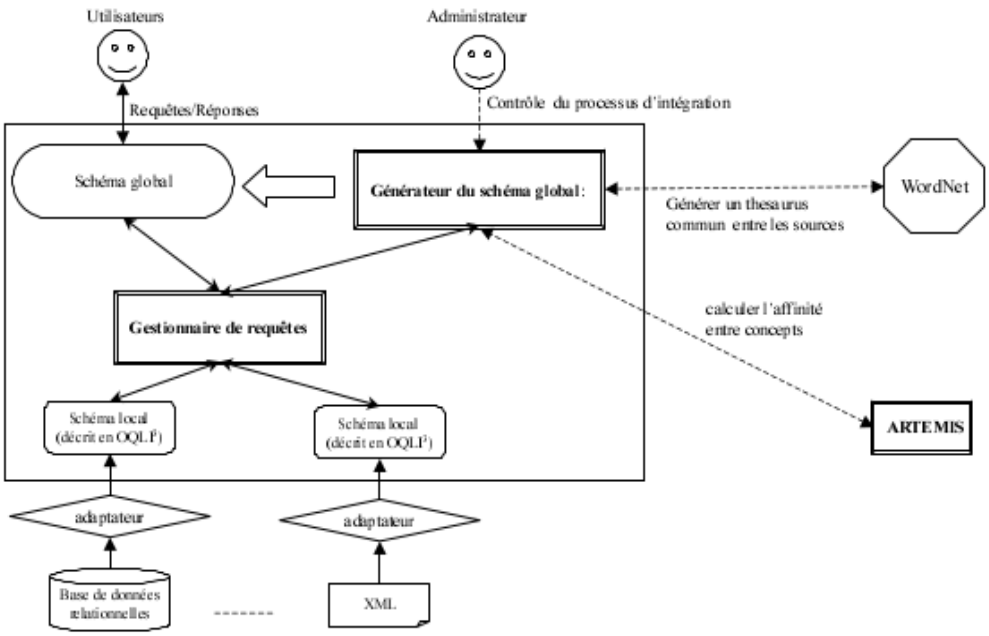


FIGURE 1.5 – Architecture générale de MOMIS

1.8.3 Le système HERMES

C'est un système qui suit l'approche de GAV, il propose un langage de règle pour exprimer les requêtes. le médiateur est basé sur deux tâches principales :

— intégration de domaine qui permet de lier physiquement les sources d'information ;

— intégration sémantique qui permet d'extraire et de combiner les informations des sources.[16]

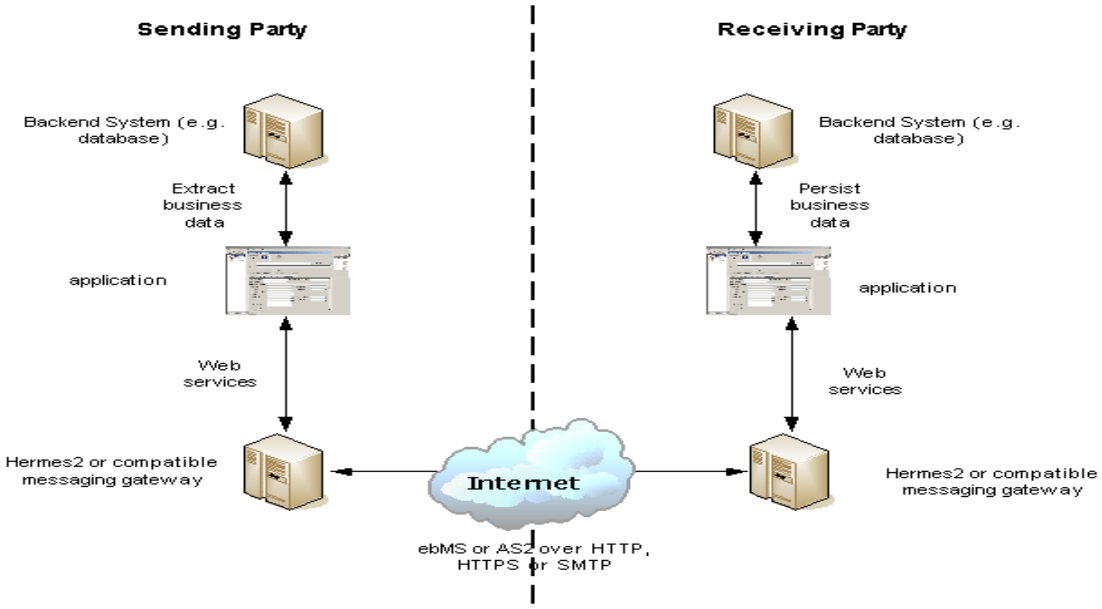


FIGURE 1.6 – Architecture général de système HERMS

1.8.4 Information manifold

Ce projet suit l'approche LAV, tel qu'on pose des relations qui décrivent les sources de données sous forme des requêtes á travers le schéma global. Le traitement de requêtes a étè réalisé par deux phases :

— utilisation des relations des sources qui contenues dans les requêtes des utilisateurs,

— pose un plan sémantique qui génère parmi les requêtes d'une façon conjonctives.

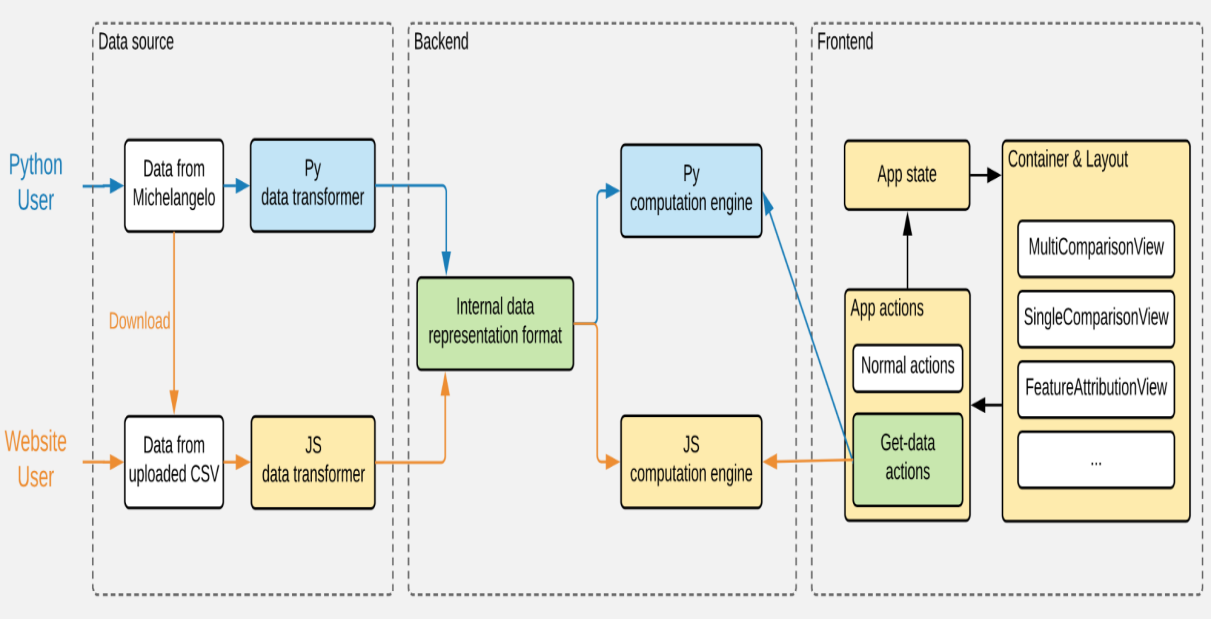


FIGURE 1.7 – Architecture Information manifold

1.8.5 Infomaster

ce projet permet d'accéder à diverses sources hétérogènes, il utilise un format d'échange de connaissances (KIF : knowledge interchange format) pour représentation le contenu de sources . Ce projet pose trois types de relations :

1. relation d'interface pour traiter et formuler les requêtes d'utilisateur,
2. relation des sources pour décrire les données qui contiennent dans les sources,
3. Relations globales qui représente le schéma global.

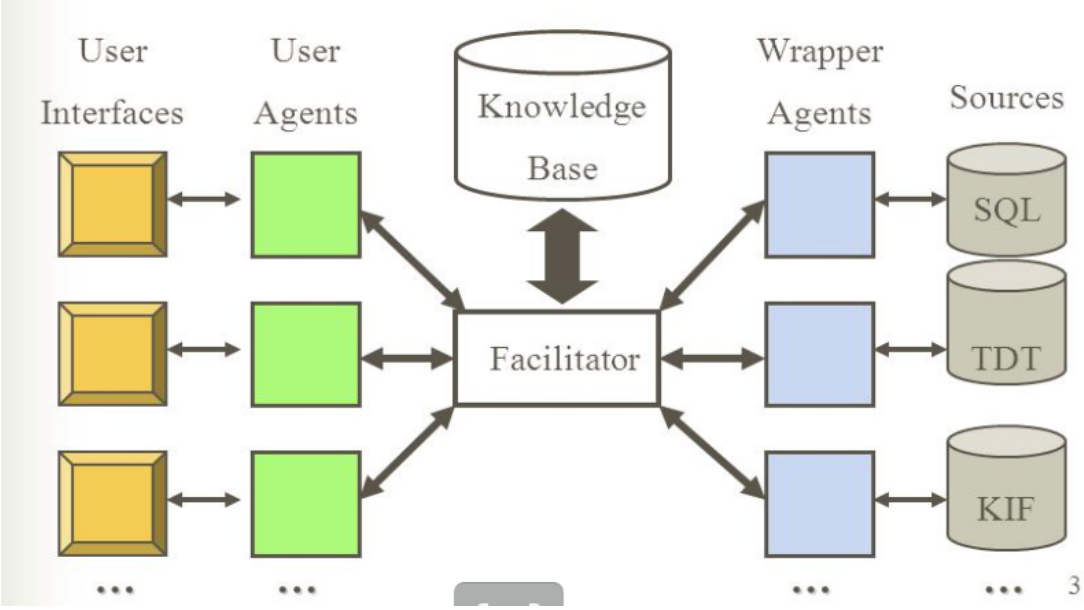


FIGURE 1.8 – Architecture général de Infomaster

1.9 Conclusion

Dans ce chapitre nous présentons des phases et des définitions reliées aux les intégrations de données hétérogènes. Parmi les problèmes qui se focalisent dans l'intégration de données c'est l'hétérogénéité sémantique qui consacre d'étudier les cas de conflits dans la phase de significations par railleuse hétérogénéité structurelle permet d'analyser les présentations diverses de données. La validation d'un système d'intégration de données basées sur la réalisation des structures et de composants qui objectivent depuis des sources hétérogènes via un schéma global, ces principaux composants : Entrepôt de don- nées, Médiation qui contient le médiateur et les rappeurs(adaptateurs), Mapping (Correspondance) qui ont des types majoraux : LAV(Local as View), GAV(Global as View), BGLAV(Bilan global Local AV). Enfin nous présentons des prototypes des projets qui réalisaient le système d'intégration de données.

CHAPITRE

2

ETAT DE L'ART D'INTÉGRATION DE DONNÉES

2.1 Introduction

Dans les années passées, étaient faites des études et des essais dans le domaine d'intégration des données. La réalisation de systèmes d'intégration de données traiter deux tâches principales :

— poser un modèle idéal pour intégrer les sources de données diverses ;

— manipulation des requêtes.

Dans le contexte d'intégration de données, on étudie les problèmes d'hétérogénéité des données avec ses types, tel que on mit les solutions de ces problèmes par des méthodes comme le rapprochement qui basant sur la similarité de formats de données, ou dans autre phase on utilise les notions d'ontologies de ces sources. La manipulation des requêtes a étè fait en plusieurs façons au niveau du schéma global, la technique fréquente c'est diviser les requêtes à travers les sources de données pour rendre les réponses de chacune source dont le médiateur de schéma global reformule et filtre ces réponses pour doivent être plus satisfaire.

La modélisation des mappings qui produit ce système d'intégration a étè posé par les deux majeurs approches GAV et LAV, en consistant dans ce chapitre sur l'approche LAV qui assure la récupération des sources de données, la facilité de les intégrer, mais un peu de complexe de traiter les requêtes.

Ce chapitre introduit quelques efforts d'intégrer les sources de données hétérogènes à intégrer via un

schéma global avec le modèle LAV(Local as View), le premier travail concerne sur les techniques de grille informatiques comme le rapprochement des champs d'un source avec des différentes sources indépendamment, un autre travail étudie l'utilisation d'ontologie dans l'intégration de données par la description des scénarios contiennent des fonctions basées sur le traitement de notion sémantique .Par ailleurs la manipulation des requêtes propose l'architecture de VISS qui est composé par des proessus et structures suivre l'opération d'interrogation en première étape à donner les requêtes jusqu'aux répondre son satisfaires.

2.2 Grille informatique :Rapprochement de données

Grille informatique rassemble les sources des données diverses, dispersées qui ayant le domaine administratif multiple pour réaliser une commune fonction.
L'une des techniques plus valides dans l'intégration des données hétérogènes c'est : la technique de rapprochement qui cherche les similarités entre les sources de données dans la phase sémantique, si on dispose qu'il y a aucune information commune entre eux.[1]

2.2.1 Techniques de comparaison

La comparaison de deux bases différentes doit passes en deux étapes :

1. une étape de comparaison de tous les champs communs entre les deux bases,
2. une étape de compilation et d'analyse des résultats de comparaisons pour la prise de décision sur le rapprochement.

Méthodes empiriques

Cette méthode correspond de mesurer la similarité entre deux champs.
la comparaison de deux valeurs entières *s1* et *s2* est présentées par cette méthode : comparateur champ par champ.

2.2.2 Méthodes évoluées de comparaisons de chaînes de caractères

Dans cette partie on a des méthodes de comparaison entre les chaînes de caractères, telque on distingue deux grandes familles :

— les algorithmes de mesure de similarité appelés aussi « pattern matching », on utilise les deux algorithmes :**LCS** qui donne la plus grande sous sequence commune de chaînes de caractères,

et l'algorithme de **Jaro-Winkler** qui calcule la distance de caractères communes dans les deux chaînes de caractères.

— les algorithmes phonétiques, s'appuyant sur la prononciation des mots, on utlise dans ce cas l'algoritme **Soundex**.

2.2.3 Expérimentation sur données

On a mener le test d'algorithme de rapprochement sur les champs d'un enregistrement de données :Nom, Prénom, Adresse .

Résultat brut

On présente le résultat qui indiquer pour chaque champ des données par le taux de vrais positifs (VP), faux négatifs (FN) et faux positifs (FP).Enfin on calcule la Précision qui correspond les performances des méthodes,tel que Précision=$VP/(VP+FP)$,en présentant sur la figure suivante :

Champ - Méthode	VP	FN	FP	Résultat	Précision
Nom – Jaro-Winkler	11.53	1.21	0.06	96.08	90.08
Nom – Soundex-US	9.33	1.14	0.11	77.75	88.19
Prénom – Jaro-Winkler	13.11	2.21	0.09	109.25	85.07
Prénom – Soundex-US	10.37	1.93	0.13	86.42	83.43
Adresse – Jaro-Winkler	9.82	1.72	0.11	81.83	84.29
Adresse – Soundex-US	7.41	1.72	0.19	61.75	79.51

FIGURE 2.1 – Résultats bruts de data linkage[1]

dans un autre cas on ajoute les études sur les données avec l'insertion **des biais**, plusieurs types des biais qui sont :

— suppression / ajout de caractère ;

— inversion de caractères ;

— substitution de caractères.

De cette manière, on présente les résultats de rapprochement des champs de données en montrant tous les cas possibles :

Nom + Prénom + Adresse	VP	FN	FP	Résultat	Précision
Jaro-Winkler	11.63	0.24	0.01	96.91	97.8
Soundex-US	9.93	1.08	0.04	82.75	92.08

FIGURE 2.2 – Résultats combinés de data linkage[1]

On présente les performances des 3 champs :Nom, Prénom, Adresse, parmi la robustesse de algorithmes ***Pattern Matshing*** et les algorithmes phonétique.

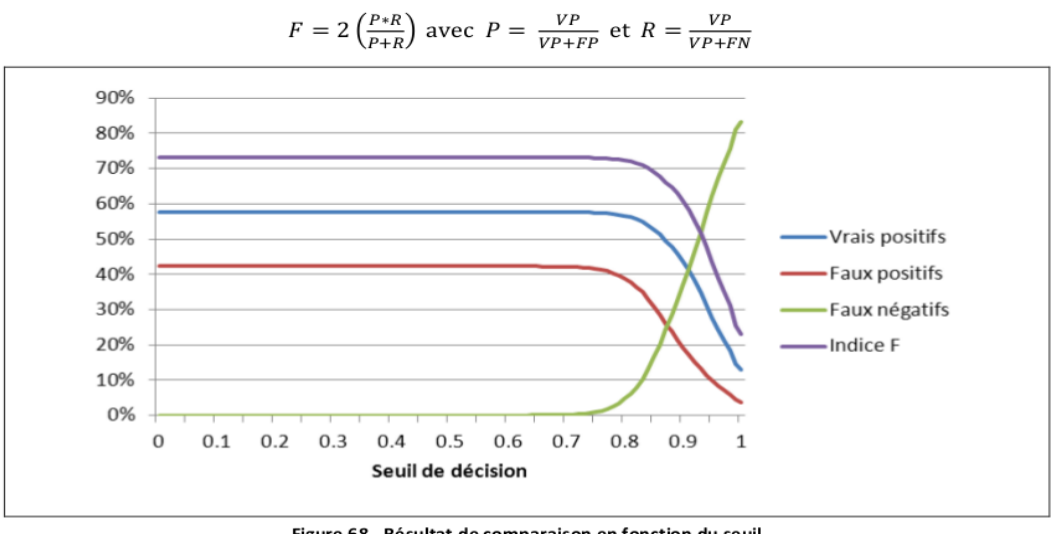


FIGURE 2.3 – Résultat de comparaison en fonction du seuil [1]

Dans ce travail, on a réalisé un processus de rapprochement,en indiquant deux étapes :

— Utilisation des algorithmes de rapprochement Jaro-Winkler et LCS qui s'appuient sur l'évolution de comparaison de chaînes de caractères,
— La mesure de performances et de rapidité des algorithmes de comparaison utilisées avec quelque amélioration des parties comme la conjointe des algorithmes phonétiques et algébriques.

Les difficultés rencontrées lors de réalisation de cette travail c'est l'ordre légal et de fournir un infructueuse des données hétérogènes réel[1].

2.3 Système d'intégration de BDBOs(Base de Données de Base Ontologiques)

Le système d'intégration en général traite les possibilités de combiner un ensemble de sources hétérogènes dans une seule vue globale c'est l'entrepôt de données, l'un des problèmes plus complexes dans ce contexte c'est l'interprétation automatique de la signification, la sémantique, des données hétérogènes et autonomes ce qui donne lieu à différents conflits, ce qui dispose que les concepts d'ontologies font une partie très important parmi des taux des systèmes d'intégration proposés.

Le système d'intégration de BDBOs définit des opérations dans la phase d'intégration qui basant sur les ontologies extraites depuis les sources de données utilisés, sur lequel on essaie se propose une ontologie partagées dans tous les sources utilisés pour faciliter la combinaison entre eux. dans ce contexte on pose un scénario d'intégration de données basant sur l'ontologie en présentant des étapes ou des opérations dans ce travail, puis on donne des phases visent à décrire les trois scénarios

d'intégration correspondant à trois opérateurs algébriques de composition de BDBO.[2]

2.3.1 Scénarii d'intégration de données

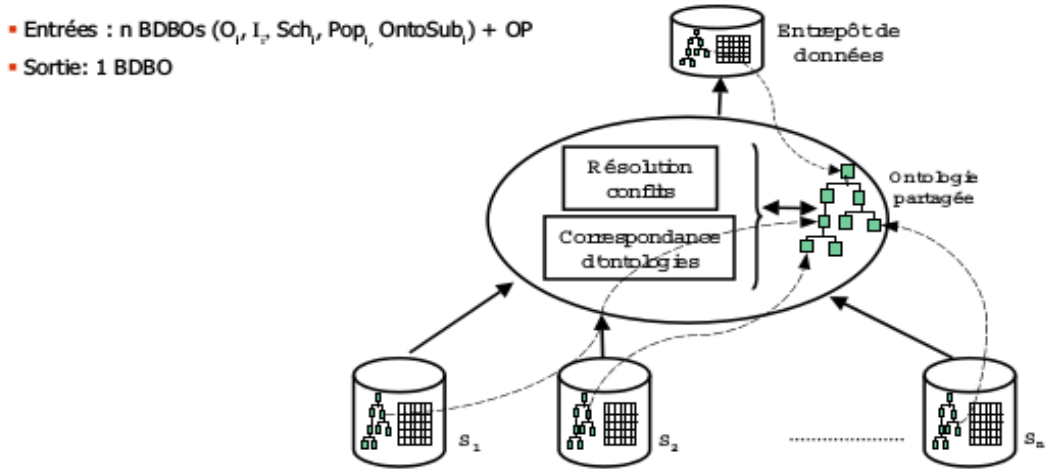


FIGURE 2.4 – Architecture général de BDBOs [2]

Soit :
S = $\mathcal{S}_1$, $\mathcal{S}_2$, ..., $\mathcal{S}_n$, l'ensemble de sources de données utilisés.
chaque $\mathcal{S}_i$ est définie comme suit
$\mathcal{S}_i :< \mathcal{O}_i , \mathcal{I}_i , Sch_i , Pop_i >$.
Dans ce travail qui base sur l'articulation a priori d'ontologie, on suppose que il existe un part de $\mathcal{O}_p$ un ontologie prtagé dans les sources utilisés, (DBA) est un adminstrateur pour chaque source qui propose sa propre ontologie.
L'ontologie partagés $\mathcal{O}_p$ est conçue en six étapes :

1. le DBA choisit la hiérarchie de classes ($\mathcal{C}_i$, Sub_i) de sa propre ontologie $\mathcal{O}_i$, tel que $\mathcal{C}_i$ est le classe de source, et Sub_i est subsomption de $\mathcal{O}_i$,
2. le DBA articule cette hiérarchie de classes avec celle de l'ontologie partagée $\mathcal{C}_p$ en définissant les relations de subsomption $OntoSub_{i,p}$ entre C_i et C_p ,
3. Le DBA importe dans $Applic_i(c_i)$ les propriétés de $Applic_p(OntoSub^{-1}_{i,p}(c_i)) \subset P_p$ qu'il souhaite utiliser dans sa propre ontologie. Ces propriétés appartiennent alors à P_i,
4. Le DBA complète éventuellement les propriétés importées par des propriétés supplémentaires, propres à son ontologie définissant ainsi l'ontologie locale : $O_i :< C_i, P_i, Sub_i, Applic_i >$,
5. Le DBA de chaque source choisit pour chaque classe feuille les propriétés qui seront évaluées en définissant $Sch_i : C_i \longrightarrow 2^{P_i}$,

6. Le DBA choisit une implémentation de chaque classe feuille, et il définit ensuite $Sch(c_i)$ comme une vue sur l'implémentation de c_i .

Dans ce travail on pose trois scénarii d'intégration qui correspond l'articulation d'ontologie parmi l'ontologie propre de source ou l'ontologie partagé entre tous les sources.On détaille ces trois scénarii selon ses algorithmes ci-dessous. [2]

FragmentOnto

Dans ce scénarii on suppose que l'ontologie propre de chaque sourceO_i est un sous ensemble de ontologie partagé O_p.

Algorithme

Soit : Sch_{Int}, le schéma du système intégré est défini pour chaque classe, tel que on suppose que l'intégration fait par l'intersection.

$$Sch_{Int}(c) = (\bigcap_{i \in 1..n | Sch_i(c) = \Phi} Sch_i(c)) \tag{2.1}$$

Dans ce calcul en prise en compte seul les classe de chaque source qui ayant l'ontologie locale Oi en prend pas les valeurs nulles.
L'imprémentation de FragmentOnto fait par l'union de tous les classes qui ont s'appelle des populations telles que :

$$Pop_{Int}(c) = \bigcup_i pop_i(c) \tag{2.2}$$

soit : $Pop_{Int}(c)$ c'est la population de chaque classe.

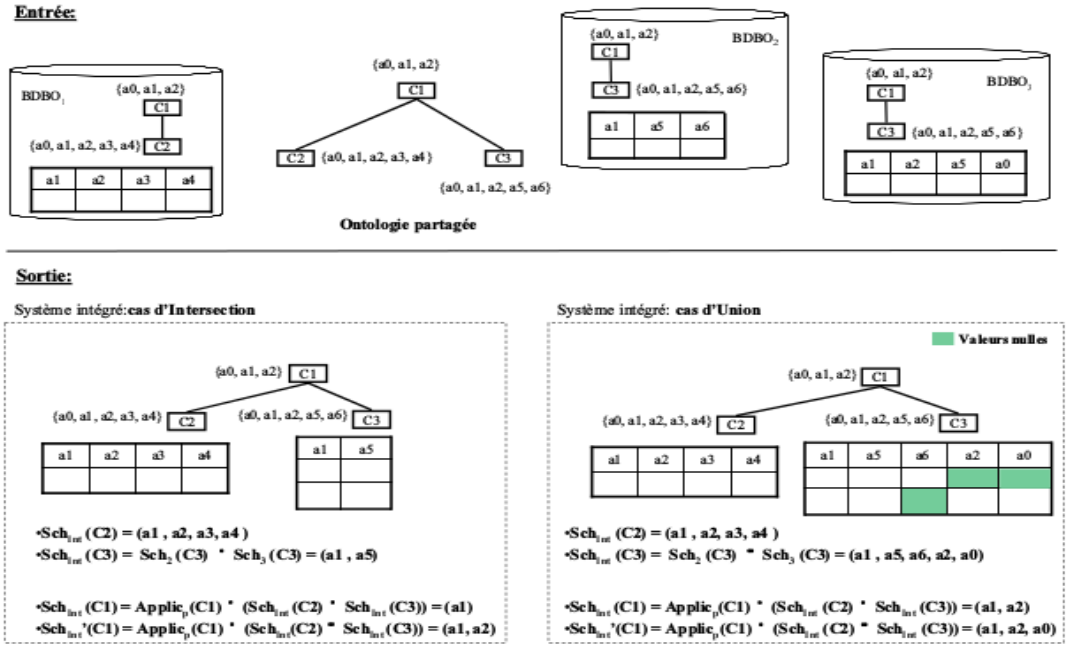


FIGURE 2.5 – Exemple d'intégration par le scénario de FragmentOnto [2]

ProjOnto

Dans ce scénarii l'ontologie partagée a étè produit par les instances de chaque ontologie propre d'un source.

Algorithme

Soit : $Sch'_{Int}(c)$, le schéma integré des instances de classes, il obtient par l'intersection des propretés Applic(c) d'ontologie partagée avec un sous schéma integré pour chaque classe $(Sch'_{Int}(c_k))$ union par le schéma de population des instances de classes $Sch*(c)$, tel que :

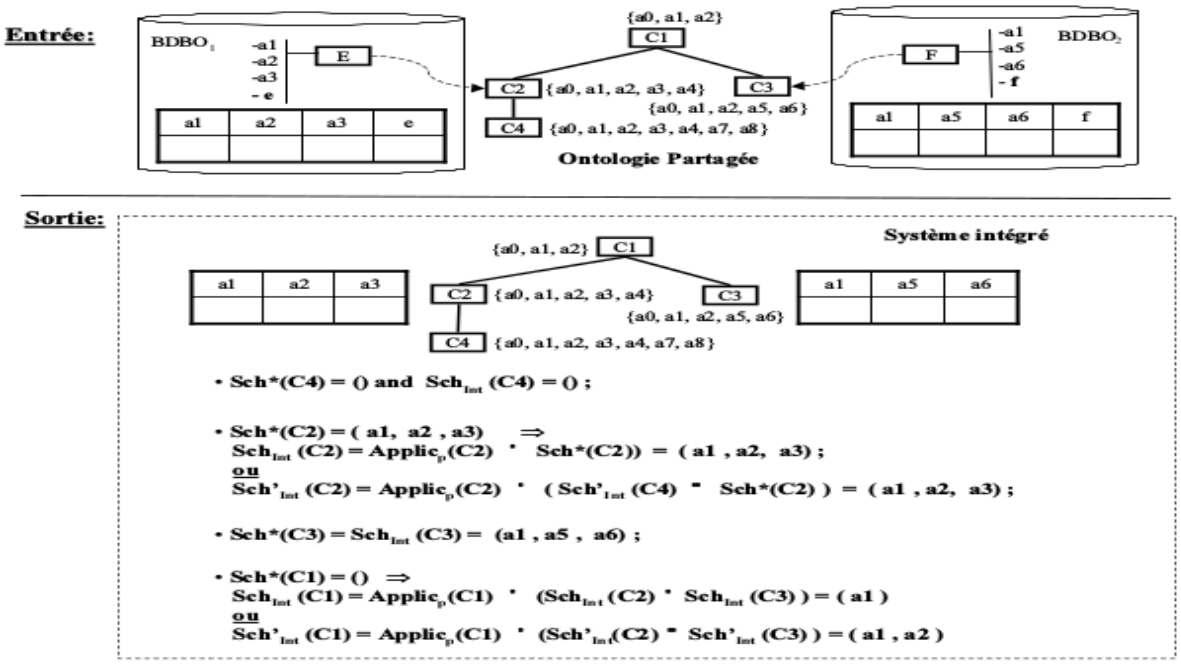


FIGURE 2.6 – Exemple d'intégration par le scénarii ProjOnto [2]

ExtendOnto

L'ontologie propre de chaque classe définit comme le principe de scénari ProjOnto, l'ontologie partageé propose depuis les Schéemsa integrés de chaque instance de classes.

Algorithme

Soit : $sch * (c)$ le schéma des instances de classe c, on peut calculer par l'intersection des propriétés d'ontologie partagé par le schéma propre pour chaque classe c,tel que :

$$Sch_{Int}(c) = Applic_{Int}(c) \bigcap \Big(\bigcap_{(c_i \in Sub_{Int}(c)) \wedge (Pop_{Int}(c_i) \neq \phi)} Sch_{Int}(c_i) \Big)$$

ou

$$Sch'_{Int}(c) = Applic_{Int}(c) \bigcap \Big(\bigcup_{c_i \in Sub_{Int}(c)} Sch'_{Int}(c_i) \Big)$$

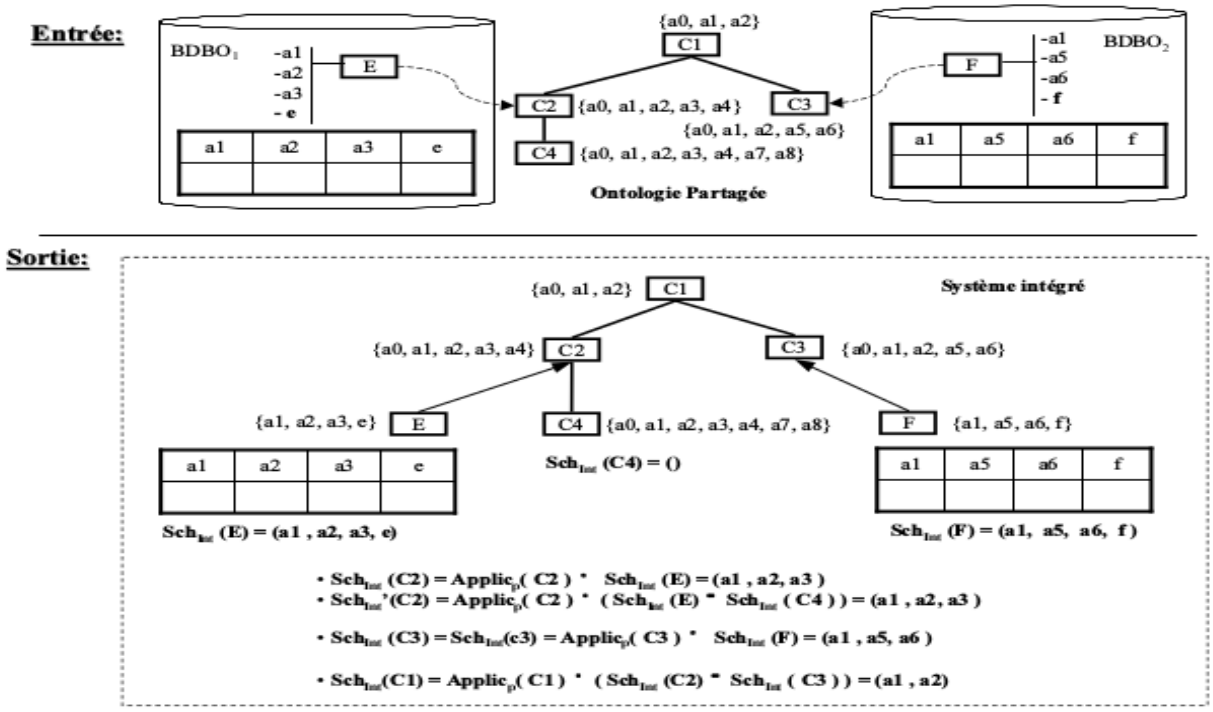


FIGURE 2.7 – Exemple d'intégration par le scénarii ExtendOnto [2]

2.4 Validation d'architecture VISS (Virtual Integration Support System)

Depuis des beaucoup de prototypes proposées dans l'intégration de données, la plupart des projets réalisés basant sur l'approche LAV, sur lequel il fournit d'architectures plus simples de combiner les sources de données mais un peu complexe de traiter les requêtes. L'architecture VISS fait optimiser des anciens autres architectures qui permettent d'utiliser l'approche LAV mais avec le concept de GAV au niveau de traiter les requêtes. dans ce contexte nous étudions plus proche sur les processus proposés au niveau de manipuler les requêts. [17].

2.4.1 Architecture VISS

Le médiateur VISS [17] permet d'analyser les requêtes pour déterminer les sources de données correspondantes pour retourner les réponses, une plateforme appellé : metadata, elle lance des commandes d'accès sur ces sources sélectionnées et importer les informations demandées pour les stocker il existe des principaux processus font la production de médiateur VISS qui est :

1. **Processus User Query**
   Il existe une tâche dans VISS qui calcule les réponses des requêtes les plus performantes(des

réponces plus proche au contexte de requête) qui sont nommées :**Monotone queries**.Certains **Monotone queries** reliées par les expressions algébriques : l'union(disjunctive) ; produit cartésien ; conjunctive où ils peuvent être récursifs, les requêtes disposent sous termes des prédicats selon le schéma global.

2. **Processus Management and Communication Module (MCM)**
   La fonction principale de MCM c'est gérer les exécutions des queries à travers les sources des données jusqu'à il donne les réponses satisfaires aux besoins d'utilisateurs.
   MCM combine le travail de processus User Query parmi les commandes d'importer, en basant sur les motsèles sémantique qu'ils utilisent la technique de Logic-based brogramming system.
3. **Processus Metadata Validation**
   L'architecture de VISS utilise le format XML pour les sources de données, les processus de Validation métadonnées évalue le cas d'intégration des sources et la fusion de ces sources via une seule vue de forme XML, puis il envoie des messages d'échec vers le processus MCM pour la validation d'opération d'intégration
4. **Processus Metadata Store**
   ce processus sauvegarde les sources de données et fait les gestions au niveau d'accéder et importer les informations selon ces sources.
5. **Processus Query Execution Engine**
   ce processus consiste sur les opérations qui reliées á l'exécution de requêtes, il permet ainsi de traiter les commandes d'import de MCM et contrôle les accès aux sources de données
6. **Processus Program Builder**
   ce processus a étè déterminé par le processus MCM, il permet de traiter les prédicats et les opérations logiques qui génèrent appartiennent les contextes des requêtes, l'une des taches vérifiées c'est la simplification des requêtes dont le but d'optimiser le temps de réponse.
7. **Processus Logic-based Programming System (LPS)**
   Ce processus fait les déductives et les opérations logiques sur les sources de données en coopérant avec tous les récents processus, il peut collecter les informations dans les sources de données et génère les réponses selon les informations demandées et les retourner au MCM pour le contrôle de ces reponce et envoyer aux interfaces d'utilisateur.
8. **Les wrappers**
   ces architectures ayant les relations directes aux sources de données et avec le médiateur celui qui il contient des tâches pour un passage idéal des requêtes a travers les sources, ces taches sont : le wrapper stocke une ontologie spécifiée par la médiation et par le source sur lequel il traduit les requêtes aux sous-structures de chacune source et il reformule les réponses ce qui est compatible parmi le schéma global

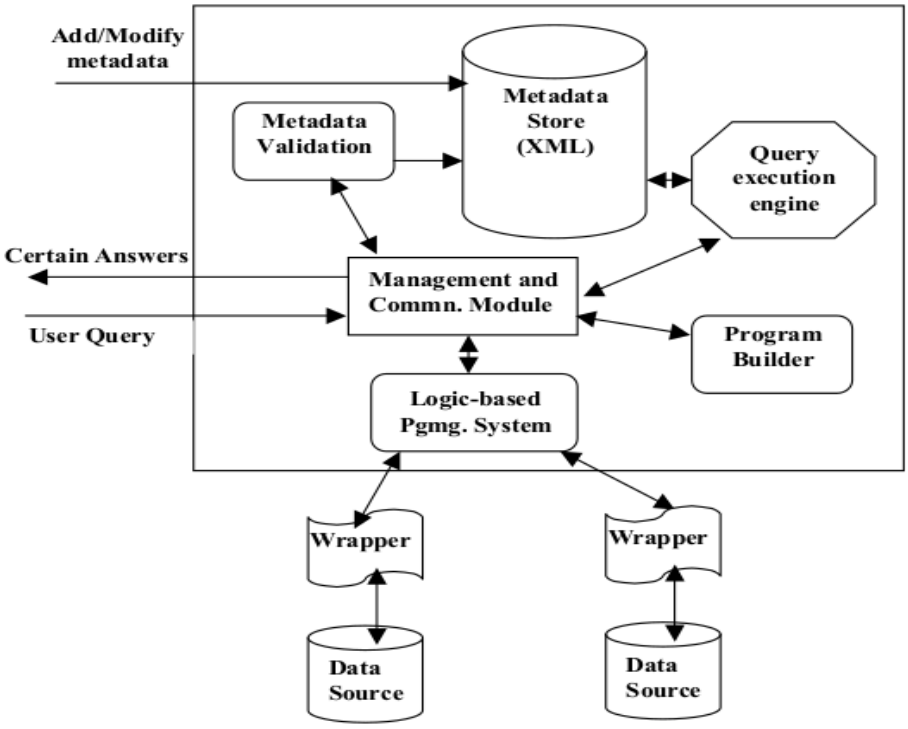


FIGURE 2.8 – Architecture général de VISS

2.4.2 Implémentation de VISS

Dans ce contexte l'implémentation produite par réaliser les processus d'architecture VISS en posant des descriptions pour chaque processus parmi les outils et les fonctions principaux.

Validation de schéma XML

cette tache réalise le processus Metadata validation qui fait l'opération de combinaison entre les sources de données valables hétérogènes via des documents de formats XML.

```
1 <xsv xmlns="http://www.w3.org/2000/05/xsv" docElt="{None}VirInt"
2      instanceAssessed="true" instanceErrors="0" schemaErrors="0"
3      target="file:///mappings.xml" validation="lax"
4 </xsv>
```

FIGURE 2.9 – Exemple d'output d'un schéma XML aprés l'intégration

Berkeley DB XML (BDBXML)

c'est une librairie open source sous C++ dévelopée par Oracle, elle permet de stocker et gérer le schéma xml, elle utilise une fonction :XQilla pour implémenter le processus Query Execution

Engine sur les documents xml dont il devient XQuery Execution Engine.
Viss utilise BDBXML comme un storage de metadonnées.

Interface metadonnées

C'est l'interface qui présente le schéma global de VISS, il utilise l'approche **POSL translator** pour traduire les requêtes d'utilisateur vers les sources de données, ainsi il dialogue l'utilisateur par évaluer les réponses satisfaisantes.

Analyseur metadonnées

Il a étè implémenté par SAX Parser sous C++ , il permet d'analyser les sources de données sur lequel pour générer les déductives pour optimiser les réponses, il inclut généralement dans le wrapper.

DLVDB

C'est la présentation de processus MCM il évalue tous les autres processus , ainsi il contrôle la manipulation de requêtes.

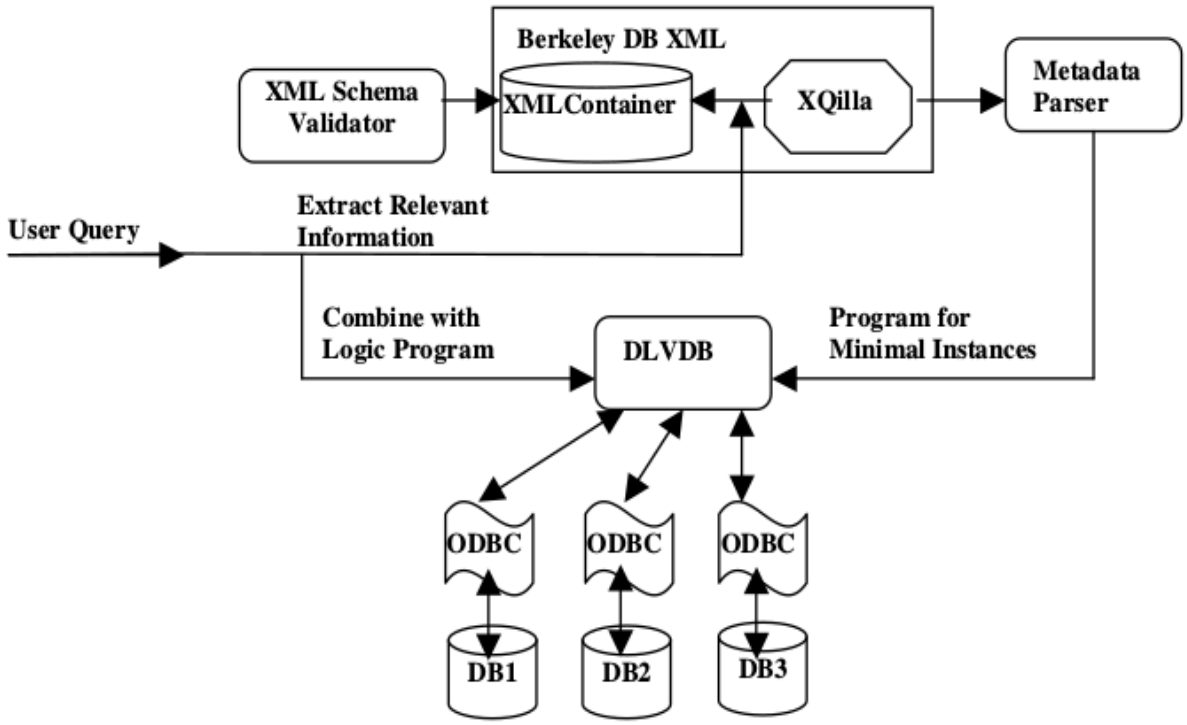


FIGURE 2.10 – Implémentation de VISS

2.5 Conclusion

Le problème d'intégration de données présente par traiter les deux majors taches :

— poser un modéle idéal pour intégrer les sources de données diverses,

— manipulation des requêtes.

On étudie le modèle LAV comme un prototype dans notre travail en basant sur comment on élimine les hétérogénéités de données en consistant sur les techniques de rapprochement qui donner des méthodes performantes et un peu robustesse dans la variante de données, par railleur, on étudie l'architecture de VISS qui dispose des plus des techniques pour faciliter la manipulation de requêtes et gestionnaire les sources de données d'une manière de programmation.

CHAPITRE

3

CONSTRUCTION D'UN SYSTÈME D'INTÉGRATION DE DONNÉES

3.1 Introduction

Le système d'intégration de données incluse sous plusieurs applications qui sont reliées par les gestionnaires des sources des données hétérogènes et ainsi qui correspondants avec le domaine de recherche les sources d'information, dans ce travail nous utilisons le domaine d'entreprise selon la gestion des données des employeurs.
Les mesures de performances des dizains des prototypes proposées de ce système d'intégration c'est l'optimisation des deux majors tâches :

- poser un modèle conceptuel pour intégrer les sources de données hétérogènes ;
- proposer une approche d'interrogation et manipulation des requêtes selon les sources de données via une interface unique.

Dans le chapitre précédent de l'état de l'art, nous avons étudié des solutions pour chaque tâche.
La technique de rapprochement donne des algorithmes riches qui regroupe des sources indépendants selon les mesures de similarités en basant sur les notions sémantiques, ces algorithmes sont robustesses dans la manière de conflits d'informations en prise en compte tous les cas des conflits (nous avons déjà vu dans le premier chapitre), elles posent plusieurs possibilités de cas de sources de données en basant sur les notions de probabilité mathématique comme : le biais.
L'architecture VISS est généralement disposée pour optimiser les traitements des requêtes et ré-

duire les problèmes de Mapping entre le schéma global et les sources de données, il propose des processus d'une manière organisée pour suivent tous les étapes de manipulation de requête et détectent l'erreur précisément grâce à ses fonctions qui ont été posées.
ce chapitre consacre comme suit : Dans la première tâche nous implémentons l'architecture de médiation selon l'approche LAV, en utilisant les algorithmes de rapprochement sur des sources de données des employeurs du types XML, HTML, JSON, SQL, et nous appliquons les techniques de fusion. nous réalisons les wrappers pour chaque source pour extraire les données, stocker, et poser des dictionnaires de données. Dans la deuxième tâche nous utilisons les processus d'architecture VISS accompagnés avec les techniques de recherche d'informations pour réalisons un traitement de requêtes efficace. L'implémentation du système d'intégration se fait en langage de programmation JAVA sous l'IDE d'*Éclipse* en Linux.

3.2 Construction de médiation selon l'approche LAV

L'approche LAV consiste à regrouper tous les sources de données hétérogènes dans un schéma global homogène unifie.
Le problème général étudié par mon travail de mémoire est le regroupement de données prévenant de sources hétérogènes et faisant fusionner dans un seul schéma accessible via une interface unique et conviviale.
L'objectif de notre travail c'est réaliser un système d'intégration de donnes qui fournir une vue globale contient tous les différents sources des données.
Cette partie se focalise sur l'implémentation de l'approche LAV en prend le compte de réaliser tous ses composants : le médiateur, le wrapper ; cette approche contient des implémentations des algorithmes du technique de rapprochement ainsi de réaliser les étapes de processus ETL (Extract, Transform, Load), dernièrement on analyse et évalue le schéma globe résulté, en utilisant les calculs des performances et la matrice de confusion.

3.2.1 Préparation de sources de données

Dans ce contexte, nous proposons des sources de données réelles d'une entreprise d'assurance des États-Unis Américaine, ces données des employeurs sont dans des endroits différents par les filières localisées dans les états : Mechegan, Los Angeles, New York, Seattle.[3]

Les sources de données qu'on a utilisées, sont hétérogènes, les notions sémantiques de ces sources sont dispersé selon leurs endroits, nous basons sur la diversité des formats pour qu'on sélectionne quatre sources différents, tels que les sources sont structurées et non structurées. On a : une source du type SQL, du type HTML, du type XML, du type JSON.

Schéma de SQL

```
CREATE TABLE mytable(
   MonthYear          VARCHAR(22) NOT NULL PRIMARY KEY
  ,Employee_Name      VARCHAR(24) NOT NULL
  ,Department         VARCHAR(3) NOT NULL
  ,Outside_Employer   VARCHAR(58) NOT NULL
  ,Job                VARCHAR(91) NOT NULL
  ,Conditions         VARCHAR(970)
  ,Expiration_Date    DATE  NOT NULL
);
MERGE INTO mytable t
USING (
SELECT '06/01/2016 12:00:00 AM' AS MonthYear,'Lucia Seyranyan' AS Employee_Name,'HHS' AS Department,'Faith and Family Counseling, LLC' AS
Outside_Employer,'Marriage and Family Therapist' AS Job,'In this outside employment activity, you may not provide services to anyone who you are
providing services to or have provided services to as a County employee or who is or has been a client of your division at DHHS, and you may not
as a County employee make  referrals to the outside business or solicit other County employees to make such referrals.' AS
Conditions,'06/15/2019' AS Expiration_Date
UNION ALL
SELECT '06/01/2016 12:00:00 AM','Steven Alderton','FRS','Premiere Improvements LLC','Owner',NULL,'06/15/2019'
UNION ALL
SELECT '06/01/2016 12:00:00 AM','Carlton Lewis','HHS','SRA International','Network Administrator',NULL,'06/15/2019'
UNION ALL
SELECT '01/01/2016 12:00:00 AM','Biodun Ayo-Durojaiye','FRS','Primerica','Independent Representative',NULL,'01/13/2019'
UNION ALL
SELECT '06/01/2016 12:00:00 AM','Roland Moore','FRS','Ben Lewis Plumbing','Truck Driver',NULL,'06/15/2019'
UNION ALL
SELECT '06/01/2016 12:00:00 AM','Richard Harris','OMB','Race DC Timing','Assistant Timer',NULL,'06/15/2019'
UNION ALL
SELECT '05/01/2016 12:00:00 AM','Yujia Chen','LIB','TopOne Mortgage, Inc','loan loan originator and processor',NULL,'05/19/2019'
UNION ALL
SELECT '05/01/2016 12:00:00 AM','Brent Frain','FRS','Frederick Community College','Adjunt Instructor','The employee may not focus on Montgomery
```

Schéma de HTML

agerange	name	sex	race	businesstitle	deptid	jobcode	positionnbr	location	jobentrydt	paygroup	stdhours	fte	salad
51-60	Johnson,Lester	M	BLACK	Laborer	2526000	601	00601084	PS HMD STR	1478390400	GEN	40	1	D1
61-70	Ferraris,Steven L	M	WHITE	Heat Ventilation & Air Condti	2553000	619	00619004	GS FAC ADM	1482019200	MGM	40	1	DOC
31-40	Gilbert,Heather N	M	WHITE	Recreation Specialist	2020000	909	00909032	PARKS WATE	1493424000	GEN	29	0.73	D4
OVER 70	Winston,James Edward	M	BLACK	School Crossing Guard	2221191	920	00920103	POL SX-ING	1184457600	GEN	29	0.28	D4
41-50	Strong,Micah H.	M	WHITE	Fire Fighter	2715000	352	00352193	FIRE3ENG32	1364688000	FIR	48	1	F48
51-60	Griffith,Michael	M	WHITE	Automotive Mechanic	2560000	667	00667040	PS FLT MGM	1519516800	GEN	40	1	D1
51-60	Spitznagel,Lisa	F	WHITE	Administrative Specialist-EXM	2553000	001	00001226	GS FAC ADM	1542499200	MGM	40	1	DOC
OVER 70	Haynie,Mary C	F	BLACK	Parks/Recreation Program Ldr	1919410	953	00953203	CRC-E296	1442102400	GEN	29	0.73	D4
26-30	Jefferies,John L.	M	WHITE	Police Officer	2220000	377	00377607	POL VICE	1399766400	POL	40	1	POL
31-40	Riefle,Charles W	M	WHITE	Fire Recruit 80 hr	2715000	352	00352330	FIRE3ENG35	1391904000	FIR	48	1	F48

Schéma de XML

```
<EmployeeName>Brown, Mia</EmployeeName>
<EmployeeNumber>1103024456</EmployeeNumber>
<State>MA</State>
<Zip>01450</Zip>
<DOB>11/24/1985</DOB>
<Age>32</Age>
<Sex>Female</Sex>
<MaritalDesc>Married</MaritalDesc>
<CitizenDesc>US Citizen</CitizenDesc>
<Latino>No</Latino>
<RaceDesc>Black or African American</RaceDesc>
<DateofHire>10/27/2008</DateofHire>
<ReasonForTerm>N/A - still employed</ReasonForTerm>
<EmploymentStatus>Active</EmploymentStatus>
<Department>Admin Offices</Department>
<Position>Accountant I</Position>
<PayRate>28.50</PayRate>
<ManagerName>Brandon R. LeBlanc</ManagerName>
<EmployeeSource>Diversity Job Fair</EmployeeSource>
<PerformanceScore>Fully Meets</PerformanceScore>
</row>
<row>
<EmployeeName>LaRotonda, William </EmployeeName>
<EmployeeNumber>1106026572</EmployeeNumber>
<State>MA</State>
<Zip>01460</Zip>
<DOB>4/26/1984</DOB>
<Age>33</Age>
<Sex>Male</Sex>
<MaritalDesc>Divorced</MaritalDesc>
<CitizenDesc>US Citizen</CitizenDesc>
<Latino>No</Latino>
<RaceDesc>Black or African American</RaceDesc>
<DateofHire>1/6/2014</DateofHire>
<ReasonForTerm>N/A - still employed</ReasonForTerm>
```

Schéma de JSON

```
FullName:                "Aarhus, Pam J."
Gender:                  "F"
CurrentAnnualSalary:     "70959.79"
GrossPayReceived2017:    "71316.72"
OvertimePay2017:         "0.00"
Department:              "POL"
DepartmentName:          "Department of Police"
Division:                "MSB Information Mgmt and Tech Division Records Management Section"
AssignmentCategory:      "Fulltime-Regular"
EmployeePositionTitle:   "Office Services Coordinator"
PositionUnderFilled:     ""
DateFirstHired:          "09/22/1986"
1:
FullName:                "Aaron, Marsha M."
Gender:                  "F"
CurrentAnnualSalary:     "110359.00"
GrossPayReceived2017:    "108040.82"
OvertimePay2017:         "0.00"
Department:              "HHS"
DepartmentName:          "Department of Health and Human Services"
Division:                "Adult Protective and Case Management Services"
AssignmentCategory:      "Fulltime-Regular"
EmployeePositionTitle:   "Supervisory Social Worker"
PositionUnderFilled:     ""
DateFirstHired:          "11/19/1989"
2:
FullName:                "Ababio, Godfred A."
Gender:                  "M"
CurrentAnnualSalary:     "55950.24"
GrossPayReceived2017:    "62575.19"
OvertimePay2017:         "7649.19"
Department:              "COR"
DepartmentName:          "Correction and Rehabilitation"
```

FIGURE 3.1 – Sources de données [3]

3.2.2 Validation de processus ETL (Extract, Transform, Load)

Le processus d'ETL(Extract, Transform, Load) consacre de passer les données dans le cas d'extraire depuis ses sources vers se transformer les reformulations des formats, des types, des structures de ces sources pour une meilleure présentation via le chargement de ces données épurées dans une vue globale. Danc cette partie nous réalisons les trois fonctions (Extract, Transform ; Load) dans notre architecture LAV proposée telle que chaque composante de l'approche LAV fait une seule fonction d'ETL[10]

Validation des wrappers

Le wrapper (adaptateur) est un composant intermédiaire entre les sources de données et le schéma global, d'abord il extrait les données depuis les sources proposées, et il stocke ces données pour l'utilisation dans le cas d'interrogation parmi le schéma global.[18]
dans ce contexte nous créons quatre plateformes qui nous présentons les wrappers selon les quatre sources que nous proposons, ces plateformes effectuent la première fonction de processus ETL c'est : **Extraction** qui fait extrait les attributs avec ces valeurs et les stockent les quatre plateformes que nous avons utilisé, c'est du type *Hash Map*, nous avons implémenté la fonction d'extraction dans quatre manières différentes selon les structures de sources.

Wrapper de source XML

Le wrapper de source XML où Xwrapper fait la fonction d'extraction en basant sur des outils spéciaux, pour lire le contenant de XML et d'obtention les informations. Pour implémenter la fonction d'extraction de ce wrapper nous utilisons l'outil DOM qui permet d'analyser un source XML d'une façon arborisante.

Outil DOM(Document Object Model)

DOM (pour modèle objet de document) est une interface de programmation pour les documents HTML, XML et SVG. Elle fournit une représentation structurée du document sous forme d'un arbre et définit la façon, dont la structure peut être manipulée par les programmes, en matière de style et de contenu. Le DOM représente le document comme un ensemble de nœuds et d'objets possèdent des propriétés et des méthodes. Quelques fonctions d'Outil DOM utilisées dans notre travail(en donnant les codes JAVA) :

- La lecture de fichier XML comme un Document : *DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance() ;*
  DocumentBuilder dBuilder = dbFactory.newDocumentBuilder() ;,
- analyse les contenents de cette fichier(detecter les Tags, et les information entre eux) : *Document doc = dBuilder.parse(notre XML fichier(Employee.xml)) ;*

doc.getDocumentElement().normalize() ;
doc.getDocumentElement().getNodeName() ;,

— création une liste de Noeuds et se poser les tags comme des attributs, création une liste de éléments et se pose les valeurs de ces tags(Liste de Noeuds) :
NodeList nList = doc.getElementsByTagName(Le nom de Noeud principale) ;
Node nNode = nList.item(i ∈ 0 · · · n n c'est le nombre de noeuds) ;
Element eElement = (Element) (nNode) ;
String name= (eElement).getElementsByTagName.

Implémentation de Xwrapper

Algorithm 1 Algorithme de Xwapper .

La source de XML fichier :Employee.xml
Array List<String> Valeurs ;
String attributs ;
HashMap <String, ArrayList<String> > hash_xml ;

DOM libraries pour lecture et manipuler XML fichier :

— lire XML fichier comme un document,
— analyse le contenu de ce fichier(détecter les Tags, et les information entre eux),
— création une liste de Nœuds et en se pose les tags comme des attributs, création une liste de éléments et se pose les valeurs de ces tags(Liste de Nœuds).

Node Noeuds ;
$N \leftarrow lenombredeNoeuds$;

for $i \leftarrow 0$ N **do**
$Noeuds \cdot i \leftarrow chaquetagdeceXMLfichier$;
$attributs \leftarrow Noeuds \cdot nom_i$

for $j \leftarrow 0$ les nombres des atrributs **do** $valeurs \leftarrow Noeuds \cdot valuedeelements$;

end for
$hash\_xml \cdot put < attributs, valeurs >$;

end for

Le résultat de cet algorithme pour réaliser Xwrapper c'est dans la figure suivante :

```
Xwrapper
_____________________________________________________________________________________
{Zip=[01450, 01460, 02703     ], PayRate=[28.50, 23.00, 29.00], CitizenDesc=[US Citizen, US Citizen, US Citizen],
Position=[Accountant I, Accountant I, Accountant I], ManagerName=[Brandon R. LeBlanc, Brandon R. LeBlanc, Brandon R.
LeBlanc], Sex=[Female, Male, Male], EmployeeNumber=[1103024456, 1106026572, 1302053333], EmploymentStatus=[Active,
Active, Active], DateofHire=[10/27/2008, 1/6/2014, 9/29/2014], Department=[Admin Offices, Admin Offices, Admin Offices],
State=[MA, MA, MA], DOB=[11/24/1985, 4/26/1984, 9/1/1986], Latino=[No, No, No], PerformanceScore=[Fully Meets, Fully
Meets, Fully Meets], MaritalDesc=[Married, Divorced, Single], ReasonForTerm=[N/A - still employed, N/A - still
employed, N/A - still employed], EmployeeName=[Brown, Mia, LaRotonda, William  , Steans, Tyrone  ], Age=[32, 33, 31],
RaceDesc=[Black or African American, Black or African American, White], EmployeeSource=[Diversity Job Fair, Website
Banner Ads, Internet Search]}
```

FIGURE 3.2 – Implémentation de Xwrapper

Le xwrapper permet d'extraire les données depuis le source XML parmi DOM d'une manière arborisant celui qui fait le stockage facile. Xwrapper base sur la structure **Hash Map** pour stocker les données par les attributs avec ses valeurs, nous assurons dans ce cas les informations sont récupérées avec un taux faible de perte.

Wrapper de source HTML

Le wrapper de source HTML où Hwrapper permet d'extraire les données de source HTML d'une façon spécielle de codage d'information dans HTML, nous utilisons l'outil **jsoup** pour analyser et obtient le contenu de source HTML.

Outil Jsoup

L'outil jsoup contient plusieurs fonctions pour traiter le fichier HTML et spécialement d'extraire les informations, nous utilisons dans ce Hwrapper les fonctions : *Select, Parse, Node* ; qu'ils font la lecture de source HTML et disposent les attributs comme des nœuds sur lesquels nous prenons les valeurs avec ces attributs..

Implémentation de Hwrapper

nous implémentons le Hwrapper sous Java en utilisant le package de jsoup et nous stockons les contenants de source HTML dans le HashMap.

Algorithm 2 Algorithme de Hwapper .

La source de HTML fichier :Employee.htm

Array List<String> Valeurs ;

String attributs ;

HashMap <String, ArrayList<String> > hash_html ;

Outil Jsoup pour traiter HTML fichier :

- detecter tous les têtes de cette fichier qui s'appelle *body* tel que le format c'est :<body>....<\body>,
- pour chaque tête en prennent les noms de Tags qui ayant la forme :<nom de Tag> et contennent dans la tête de HTML fichier,
- la sélection de valeurs entourées par les Tags et qui s'appelle *Child* tel que :<nom de Tag> Child <\nom de Tag>.

Têtes=$t_1, t_2, ..., t_n$

$N \leftarrow lenombredeTags$;

for $i \leftarrow 0$ **to** le nombre de têtes **do**

$t \cdot i$;

for $j \leftarrow 0$ **to** N **do** $attributs \leftarrow t_i \cdot nomdeTag_j$;

$valeurs \leftarrow nomdeTag_j \cdot Child_j$

end for

$hash\_html \cdot put < attributs, valeurs >$;

end for

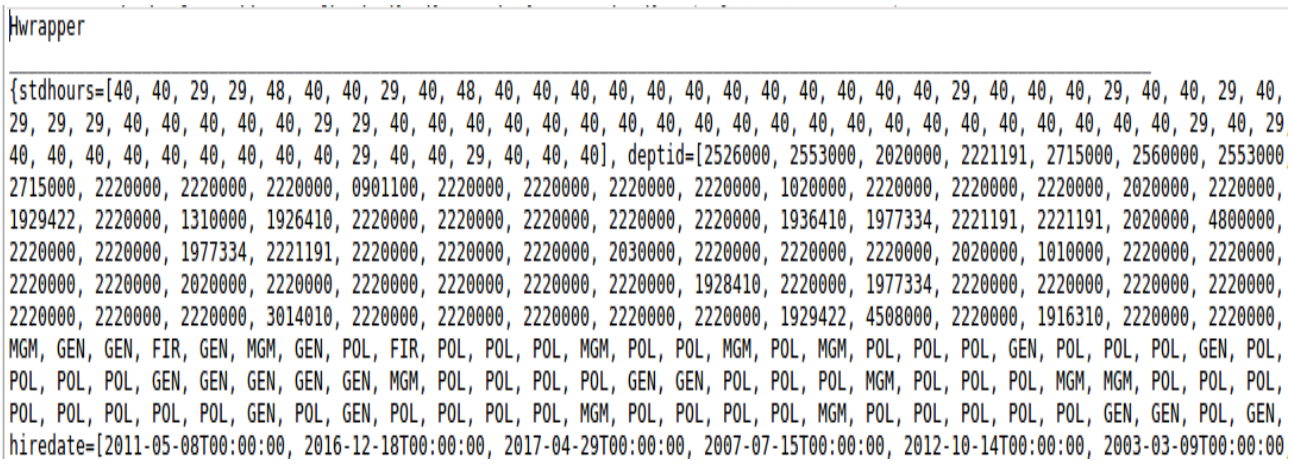


FIGURE 3.3 – Implémentation de Hwrapper

Dans la Fig3.3 l'exécution de Hwrapper affiche une partie des données de source HTML, tel que le HashMap affiché enregistre les attributs avec ses valeurs d'une manière organisée et correcte.

Wrapper de source Json

Le source qui contient le fichier Json qu'est de type structuré, il formalise comme un tableau organisé, la fonction d' extraire a étè utilisée par l'outil **Gson** qui est spécielle pour traiter le fichier Json, dans ce contexte nous réalisons un wrapper de json ou Jwrapper qui extrait les données depuis ce source dans le HashMap.

Outil Gson

parmi les plusieurs libraries pour traiter le source Json nous trouvons que **Gson** est la meilleure selon la lecture performante et la manipulation des données de ce source.
Gson est une librarie développée par *Geogle* pour traiter les données dans le cas où on utilise dans la page web.

Implémentation de Jwrapper

Le jwrapper implémenté comme les restes précidents wrappers sous Java, tel que nous utilisons **Gson** pour lire le source Json et extraire les données de ce source, nous implémentons les mêmes étapes d' extraction des données avec ses valeurs de Xwrapper, la différence consacrée dans l'implémentation d'Outil Gson dans le manière de lire et analyser ainsi de obtiennent les données et ses valeurs.

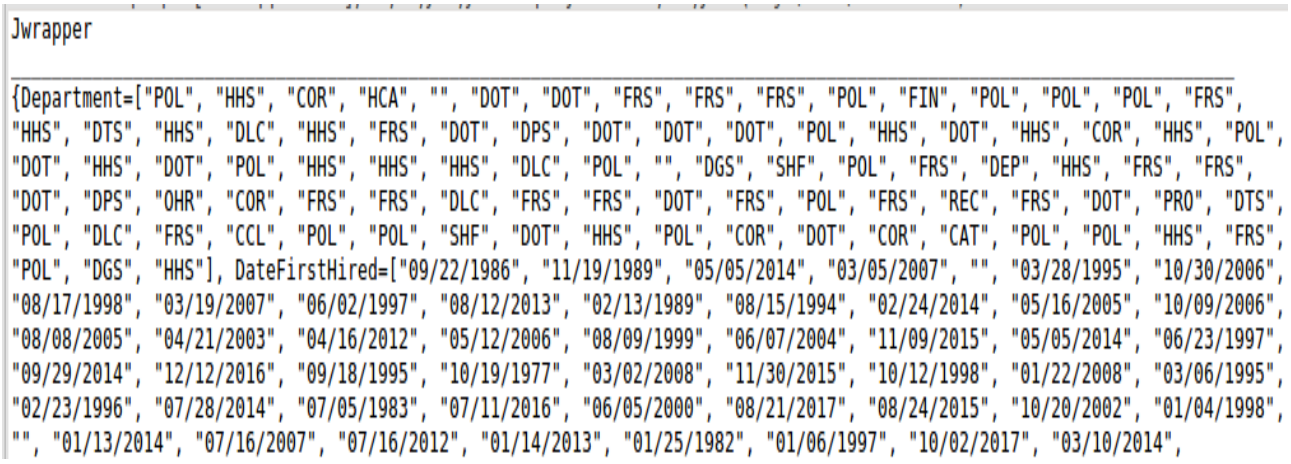


FIGURE 3.4 – Implémentation de Jwrapper

la Figure3.4 affiche l'exécution de Jwrapper nous posons un capture d'une parte des valeurs extraites depuis le source Json dans le HashMap.

Wrapper de source SQL

La validation de ce wrapper est totalement différente parmi les précidents implémentations, nous avons besoin de connecter avec un environnement de SQL pour traiter le source SQL et

d'importer dans la plateforme sous JAVA que nous appliquons l'extrait de données de ce source dans le HashMap.

MySQL Workbench

c'est un environnement développé sous Linux tel que nous l'utilisons pour traiter le source SQL et d'extraire les attributs de table SQL avec ses valeurs d'une façon organisée et correcte.

Implémentation de Swrapper

L'implémentation de ce wrapper fait la première phase de connecter l'environement **MySQL Workbench** avec la plateforme sous JAVA , puis nous utilisons les outils de c'environnement pour lire le source SQL et extrait les données qui sont les attributs avec ses valeurs, en dernier nous importons les données extraites dans le HashMap.

Algorithm 3 Algorithme de Swrapper .

Le chemin de location de stocker le SQL fichier :employee.sql, et le port de connexion avec *MySQL Workbench* par défault (8080)
Array List<String> Valeurs ;

HashMap <String, ArrayList<String> > hash_sql ;
String [ ] columns = connect(//home/MySQLWorkspace/employee.sql,8080)·(*getlesnomsdescolumnes*);
N ← le nombre de columnes dans employee.sql

for $i \leftarrow 0$ **to** N **do**
$Query \leftarrow$ "$SELECTcolumn[i]FROMemployee.sql$"
pour $j \leftarrow 0$ **to** M **faire**
$valeurs_j \leftarrow excution(//home/MySQLWorkspace/employee.sql, 8080, Query)$

end for
$hash\_sql \cdot put < columns, valeurs >$;

3.2.3 Validation de médiateur

'Le médiateur c'est le composant qui présente la vue globale d'intégration des sources de données, dans l'implémentation de processus ETL(Extract, Trasform, Load) le médiateur fait les deux fonctions : **Transformation** et **chargé**.

1. **Transformation** :
   dans l' étape de transformation de processus ETL,nous essayons de valider l'intégration des sources de données qui sont dans les wrappers telque :
   nous appliquons les techniques de rapprochements qui permettre d' analyser les wrappers et de trouver la simulation entre les données dans la façon sémantique.

ainsi nous utilisons les concepts d'ontologies en basant sur les sémantiques des informations dont nous obtenons les synonymes, en construction un dictionnaire des données contient les termes et leurs sens, sur lequel nous fusionnons les termes qui ayant la même sémantique.

2. **Charge** :
   Cette fonction fait après l'intégration de sources de données dans un schéma global, tel qu'un source global homogène résulte depuis l'intégration des sources hétérogènes a étè chargée dans un entrepôt de données, dans notre travail c'est le XML fichier global.

Technique de rapprochement

Les algorithmes de technique de rapprochement consacrent de trouver les simulations entre les caractères des termes. Dans notre travail nous implémentons un algorithme qui est l'optimisation des deux algorithmes de cette technique pour faire la comparaison dans l'objectif de trouver les termes semblables dont nous validons l'intégration entre les sources qui contiennent ces termes.

— **Algorithme LCS « Longest Common Susbtring »**
  LCS problème consiste à trouver la plus longue sous séquence commune entre les deux chaînes de séquences, l'algorithme de LCS permet de comparer deux chaînes de caractères pour trouver la divergence entre eux selon les caractères trouvés, jusqu'à trouver la plus longue chaîne commune.
  Cet algorithme est plus performant pour le cas d'une divergence ou une simulation des caractères entre les termes.
  La mesure de similarité se calcule par ;

$$LCS(X_i, Y_j) = \begin{cases} \emptyset & \text{if } i = 0 \text{ or } j = 0 \\ LCS(X_{i-1}, Y_{j-1})\hat{}\, x_i & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max\{LCS(X_i, Y_{j-1}), LCS(X_{i-1}, Y_j)\} & \text{if } i, j > 0 \text{ and } x_i \neq y_j. \end{cases}$$

  Telque :
  X_i, Y_j : deux chaînes de caractéres.
  i,j : les deux longueurs de X,Y par ordre, tel que.

— **Algorithme de Jaro-Winkler**
  Cet algorithme correspond de calculer la distance de simulation entre deux termes, Jaro propose une formule de calcul basé sur le poids de caractères dans la longueur des termes parmi les deux chaînes de caractères.
  La mesure de similarité se calculer par :

$$Jaro(S_1, S_2) = \frac{1}{3}\left(\frac{C}{|S_1|} + \frac{C}{|S_2|} + \frac{C-P}{C}\right)$$

  Telque :
  C :le nombre de caractères communs.
  P : le nombre de permutations.

L'amélioration de ce formule se fait par Winkler tel que il prise en compte le nombre N de caractères communs au début des deux chaînes pour réduire le taux de comparaisons.

$$Jaro - Winkler(S_1, S_2) = Jaro(S_1, S_2) + \frac{N}{10}(1 - Jaro(S_1, S_2))$$

Utilisation d'ontologie

nous utilisons les concepts d'ontologie pour augmenter les performances de calcul de similarité, les techniques de rapprochement que nous les utilisons ne supportent pas les cas d'existence des synonymes ayant des termes différents, donc dans ce contexte nous utilisons un dictionnaire de données pour chaque source de données en représentant les attributs comme des termes avec ses synonymes, nous implémentons la comparaison entre les dictionnaires et les attributs pour obtiennent la similarité.

Dictionnaire de données

C'est une structure qui contient des termes compagne avec les synonymes, ces termes réprésentent les attributs de chaque source de données référencées par des mots similaires. Dans notre travail nous proposons un dictionnaire creé d'une façon manuelle qui contient un en- semble des attributs référencés par ses synonymes, a l'aide de Word Net, nous pose pour chaque attribut des quatre sources un ensemble de trois synonymes obtient de Word Net.[19]

WordNet

Est un base de données lexicales. Les termes y sont organisés sous formes d'ensembles de synonymes, les synsets. Chaque synset est un concept lexicalisé. Ces concepts lexicalisés sont reliés par des relations linguistiques. WORDNET est un énorme dictionnaire hypermédia de l'anglais-américain (plus de 100 000 synsets). Sa richesse et sa facilité d'accès le positionnent comme un intéressant outil pour la recherche d'information ou d'autres tâches comme le traitement du langage naturel mais ce n'est pas un ontologie car les relations ne sont en aucun cas formelles. L'utiliser tel quel, dans un système formel est donc voué à l'échec. Sa seule utilisation dans le cadre de l'intégration ne peut donc être que d'assister un expert humain.[20]

3.2.4 Implémentation de Médiateur

La réalisation de médiateur est d'effectuer les deux fonctions principales de processus ETL : **transformation** pour rassembler les sources de données participées et **charger** le schéma global XML résulte dans l'entrepôt de données.
Notre travail est de valider un algorithme qui suit les principes de la fonction de transformation, nous utilisons les techniques de rapprochement et le dictionnaire de données.

Implémentation d'Algorithme de Wrinkler-Jarro

L'algorithme de Wrinkler-Jarro c'est l'un des plus algorithmes meilleurs de rapprochement, elle s'agit de mesurer la similarité entre deux séquences sur lesquelles, elle pose des pourcentages supérieurs s'ils sont les mêmes sens, nous implémentons cet algorithme dans les comparaisons entre les attributs de sources de données différents en prise en compte tous les types de conflits.

Algorithm 4 Algorithme de Wrinkler-Jarro

S_1 : $chainesdecaracteres.$
S_2 : $chainesdecaracteres.$
C_1 : $chainedecaracteres$;
C_2 : $chainedecaracteres$;
$intt \leftarrow 0$;
$C_1 \leftarrow caracterecomunes(S_1, S_2)$
$C_2 \leftarrow caracterecomunes(S_2, S_1)$
$N \leftarrow lalongueurde C_1$

for $i \leftarrow 0$ N **do**

if $C_1[i] \neq C_2[i]$ **then** $t \leftarrow t + 0.5$

$$réelR \leftarrow \frac{N}{S_1 \cdot longueur} + \frac{C_2 \cdot longueur()}{S_2 \cdot longueur} + \frac{\frac{C_1 \cdot longueur() - t}{C_2 \cdot longueur}}{3}$$

Implémentation d'Algorithme de LCS(Longuest Common Subsequences)

Nous utilisons l'algorithme de LCS pour plus précise les opérations de comparaison sur lesquelles en assurent les comparaisons entre les attributs caractère par caractère en prise en compte tous les cas posés sur les attributs n'en objectivent que :

Algorithm 5 Algorithme de LCS(Longuest Common Subsequences)

S_1 : $chainesdecaracteres.$
S_2 : $chainesdecaracteres.$
N : $LalongueurdeS_1.$
M : $lalongueurdeS_2$

function LCS($<S_1, S_2$,N,M $>$) :chaine de caracteres

if $N > 0 et M > 0$ et $S_1[N] == S_2[M]$ **then** retourner LCS($S_1[N-1], S_2[M-1]$,N-1,M-1)

else if $N > 0 et M > 0$ et $S_1[N] \neq S_2[M]$ **then**Max((N,M-1),(N-1,M)
retourner null

Implémentation des dictionnaires de données

La création du dictionnaire de données permet de prendre les attributs de chaque wrapper depuis les quatre (Xwrapper, Hwrapper, Jwrapper, Swrapper) comme des termes référencés par ses synonymes extraits de Word net. Nous implémentons une fonction qui prend les attributs de données pour chaque wrapper et d'importer que trois synonymes parmi *Word Net*, enfin nous collections tous les attributs de quatre wrappers avec ses synonymes dans un seul dictionnaire.

Algorithm 6 Construction d'un dictionnaire de données

$wrappers = Xwrapper, Jwrapper, Hwrapper, Swrapper$
word $\leftarrow impoterle fichiercomplé WordNet$
$HashMap < String, ArrayList < String >> dictionnaire$
$w \leftarrow 0$
while wrapper[w]< le nombre de wrappers **do**
w $\leftarrow w + 1$

for $i \leftarrow 0$ le nombre de attributs de chaque wrapper **do**

for $j \leftarrow 0$ le nombre de word **do**

if $wrapper[w] \cdot attribut_i == word_j$ **then** dictionnaire $\cdot put <$wrapper[w]$\cdot attribut_i, word_j >$

Implémentation d'Algorithme de médiateur

Notre algorithme proposé mit tous les avantages de ces précidentes algorithmes, telque nous essayons de combiner les parts des algorithmes qui vérifie ces avantages dans notre algorithme. nous importons aussi les exécutions des algorithmes :

- Algorithme de Wrinkler-Jarro : pour donner la mesure de similaritée entre deux attributs ;
- Algorithme de LCS : pour obtenir la plus longue sous-séquence entre deux attributs sur lesquels nous posons la chaîne caractères résulte comme un attribut Global qui fusionne les valeurs de ces deux attributs comparés ;
- Le dictionnaire de données résulte parmi l'algorithme de construction d'un dictionnaire de donnée.

Algorithm 7 Algorithme de fusion

$wrappers = Xwrapper, Jwrapper, Hwrapper, Swrapper$

distance ← *un réel.*

attribut_G ← une chaîne de caractère pour poser la chaîne de caractère résultat de comparaison.

Noeuds ← *les listes des attributs pour créer un XML fichier.*

*valeur_G ← les valeurs d'***attribut_G**.

éléments ← *contient valeur_Global référencer par Nœuds.*

w ← 0

while *wrapper*[*w*] < *le nombre de wrappers* **do**

for *i* ← 0 (*le nombre de attributs de wrappers*[*w*] *ou wrappers*[*w* + 1]) **do**

distance ← $Wrinkler - Jarro(attribut_i$ *de wrapper*[*w*], $attribut_i$ *de wrapper*[*w* + 1])

if distance>80.70 and max(distance) **then**

attribut_G ← LCS(attribut$_i$ *de wrapper*[*w*],$attribut_i$ *de wrapper*[*w* + 1]) ;

valeur_G←attribut$_i$ *de wrapper*[*w*] · *valeurs* ∪ $attribut_i$ *de wrapper*[*w* + 1] · *valeurs* ; **end if**

else

synonymes1 ← *dictionnaire* · (attribut$_i$ *de wrapper*[*w*]) ;

synonymes2 ← *dictionnaire* · (attribut$_i$ *de wrapper*[*w* + 1]) ;

if synonymes1==synonymes2 **then**

attribut_G ←attribut$_i$ *de wrapper*[*w*] ;

valeur_G ←attribut$_i$ *de wrapper*[*w*] · *valeurs* ∪attribut$_i$ *de wrapper*[*w* + 1] · *valeurs* ; **end if**

else

attribut_G←attribut$_i$*dewrapper*[*w*] ;

attribut_G ←attribut$_i$*dewrapper*[*w* + 1] ;

valeur_G ←attribut$_i$*dewrapper*[*w*] · *valeurs* ;

valeur_G ←attribut$_i$*dewrapper*[*w* + 1] · *valeurs* ; **end else end else**

end For

end While

Noeuds ← *attribut_G*;

elements ← *valeur_G*;

Création XML fichier Global selon les noms de liste nœuds c'est **Nœuds**, et les valeurs contenant dans ces Nœuds c'est **éléments**.

L'exécution d'algorithme de médiateur donne un schéma global qui représente par un XML fichier contient tous les valeurs et les informations avec les attributs nouveaux résultats par le rapprochement entre deux attributs.

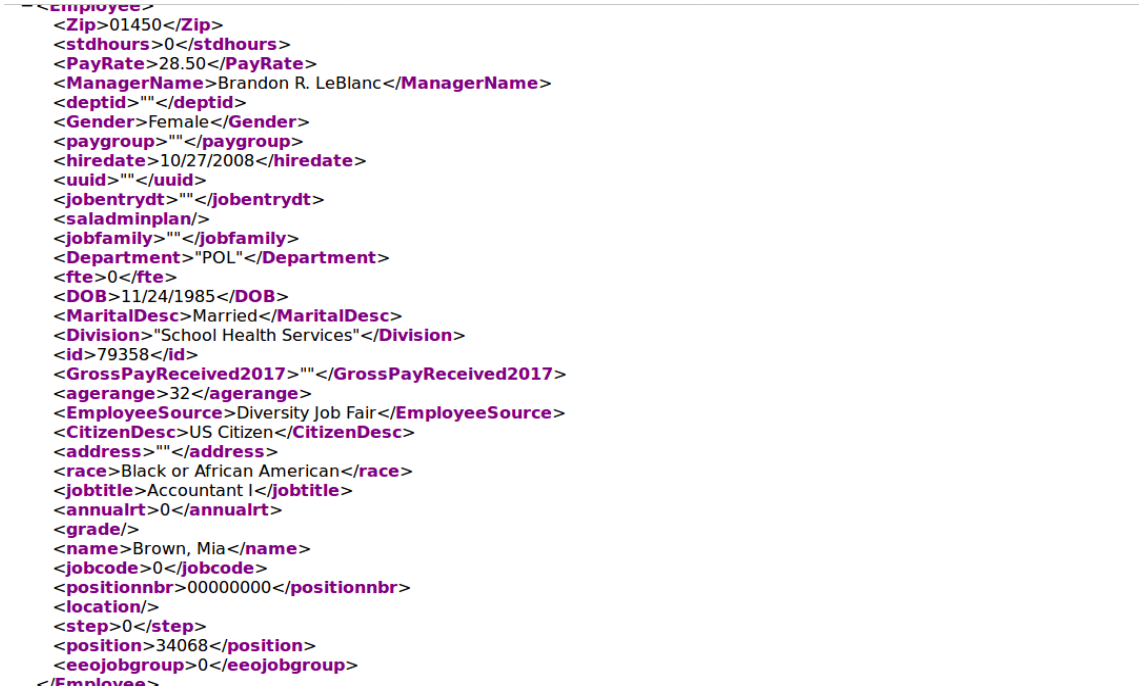


FIGURE 3.5 – Une partie de fichier global en XML

La médiation LAV se focalise de réaliser un ensemble de processus qui permet de poser les étapes dans le but de traiter les données par implémenter un wrapper pour chaque source de données.
Ces précidentes étapes évaluent par utiliser le processus d'ETL(Extract, Transform, Load) qui permet se contient un ensemble de processus pour intégrer des sources de données hétérogènes de différents formats et de différentes sémantiques.
dans cette partie nous réalisons une médiation LAV par implémenter des algorithmes concernés par le processus ETL tel que : nous avons quatre sources de données : XML fichier, HTML fichier, JSON fichier, SQL fichier.
Nous créons quatre wrappers pour chaque source de données, chaque wrapper stocker les données extraites depuis un source de données.
Pour valider un médiateur nous utilisons les algorithmes de rapprochement Jarro-Wrinkler et LCS et le dictionnaire de données construit a l'aide de WordNet pour ayant les synonymes avec les attributs de notre source de données.
L'implémentation de wrappers et de médiateur résulte un schéma global représenté par un XML fichier qui contient tous les données extraites depuis les sources référencées par les attributs nouveaux qui disposées par les algorithmes de rapprochement.

3.3 Traitement de Requêtes dans le médiation LAV

Cette section est concernée de donner une méthode efficace pour satisfaire le besoin de recherche des informations pertinentes selon le traitement des requêtes posées et la fourniture une façon de recherche plus performante. Le problème secondaire étudie par notre travail c'est de traiter les requêtes qui interroge le schéma global et de manipuler les réponses disposées par ce schéma. L'objectif de cette partie c'est de réaliser une architecture valide pour traiter les requêtes qui interroge au schéma global XML en utilisant les techniques de recherche qui doivent améliorer la réponse de requête et sa pertinence.

3.3.1 Implémentation de Moteur de recherche par VISS

L'architecture VISS fournit cinq tâches principales pour traiter : l'interrogation de requêtes depuis l'interface d'utilisateur, l'accès dans le XML metadata, l'amélioration de contexte de requête (modification de contexte de requête) jusqu'au traitement de réponse retourné à partir de XML metadata est satisfait le besoin de recherche. La construction d'un moteur de recherche contient la validation des trois processus majoraux :[21]

- **Indexation de données** : c'est un processus qui donne une description de contenants de données d'une façon de filtrage du schéma de données pour contient des termes les plus importants pour l'opération de recherche, en éliminant l'espace vide entre les termes, les caractères spéciaux, les mots vides.
- **traitement de réponces** : celui qui permet d'analyser le niveau de performances de réponces parmi les requêtes et de disposer sur une interface les réponces d'un format valide et pertinent.

L'implémentation que nous validons c'est de réaliser les trois processus précidentes, pour chaque processus nous utilisons les tâches de VISS qui vérifie l'objective de ce processus.

Indexation de données

nous utilisons Le schéma XML qui est l'entrepôt de données que nous avons besoin d'interroGuer, nous utilisons le schéma XML sans indexation. Ce processus consacre de traiter et de préparer le schéma de données pour être donné les reponce aux requêtes, l'architecture VISS contient deux tâches qui évalue cette objective :

1. **Validation de schéma XML**
   cette tâche et la préparation de schéma global résulte, nous avons XML fichier StylePaper.xml qui collecte tous les sources de données intégrées. La figure3.6 affiche une capture de ce XML fichier global.
2. **Berkeley DB XML (BDBXML)**
   Nous implémentons cette tâche pour gérer et faciliter l'accès au XML fichier, nous utilisons l'outil DOM qui permet d'analyser le contenant de ce fichier.

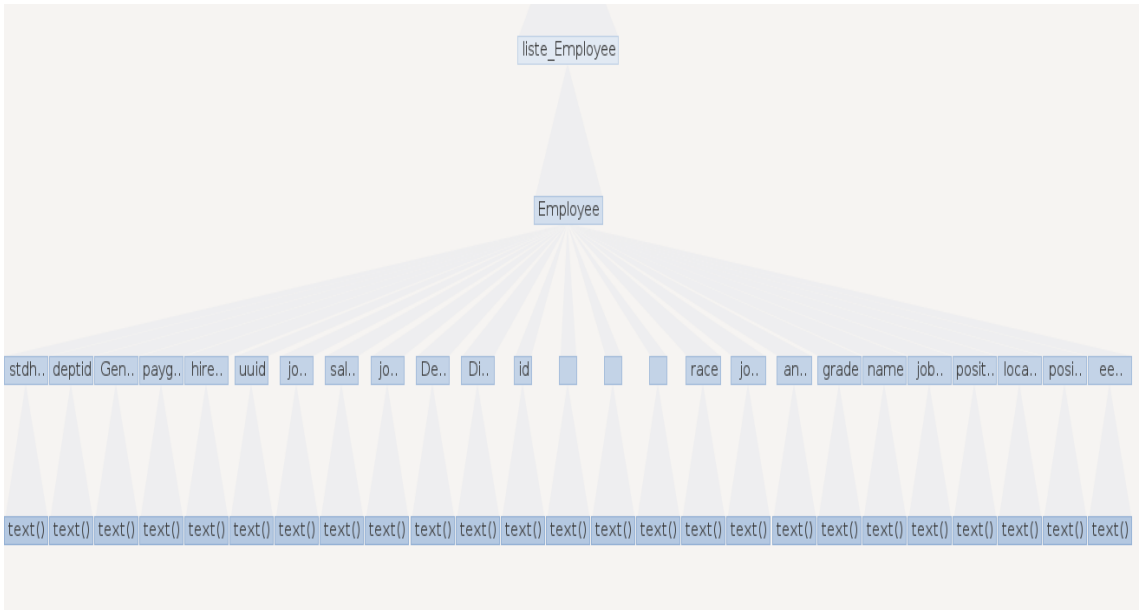


FIGURE 3.6 – Schéma XML global sous forme d'arbre par l'Analyseur DOM

Traitement de requêtes

Ce processus est effectué par trois fonctions :

— **Analyser le contexte de requête** : pour identifier les termes composés, dans ce cas, en éliminant les termes vides, les espaces vides, présenter pour chaque terme de requête son ontologie dans l'index.

— **Interrogation de requête** : c'est l'opération de recherche d'accès dans le schéma XML, elle concerne de recherche l'existence des termes d'une requête dans une partie du schéma XML, ou de trouver des informations sémantiquement simulées avec les termes de requête.

Les tâches de VISS concernées dans ce processus sont :

Berkeley DB XML (BDBXML)

il contient une sous tâche s'appelle : <*Execution de requête de recherche* > qui facilite la recherche dans le schéma XML par appliquer les trois fonctions suivantes.

Implémentation de Berkeley DB XML (BDBXML)

Algorithm 8 Algorithme d'analyser la requête

Query : chaînes de caractères ;
StopWord : Fichier texte qui contient les mots vides.
Query= $\sum_{i \subseteq 0...n} term_i$
term : c'est le sous chaînes caractères
Qindex $\leftarrow$ *le document qui contient les termes de Qyery.*
for $i \leftarrow 0 \rightarrow Query.longueur$ **do**

for $word : chaque\ mot\ vide \subset StopWord \rightarrow la\ fin\ de\ StopWord$
if term$_i$ = *word* **then**
$term_i = espace\ vide$
enfin en éliminant les caractères spéciaux par exemple : l'espace vide.
Qindex.chaque ligne $\leftarrow$ *Query.*
end if

end for

end for

L'implémentation d'algorithme d'analyser la requête résulte donne un document index qui contient les termes de requêtes composées et filtrées depuis les mots vides et les caractères spéciaux.

Algorithm 9 Interrogation de requête

le schéma XML *StylePaper.xml* :analysé par l'outil DOM et formater sous forme des Nœuds qui référencer ses valeurs .
index : qui contient l'indexation de requête.
index = $\sum_{i=0}^{n} term_i$
Node$\leftarrow$ *la liste de n*$\sphericalangle$*uds de XML fichier.*
Éléments $\leftarrow$ *les attributs qui contient dans chaque n*$\sphericalangle$*uds.*
résultat $\leftarrow$ *les n*$\sphericalangle$*uds qui contient les termes de requête.*
for *chaque terme dans index* **do**

for j$\leftarrow$ 0**to** *le nombre d'éléments pour chaque Node*
if term$_i$ contient dans $Elements_j.valeur$ **then**
résultat $\leftarrow$Node.Elements$_j$
end if
end for
end for

Traitement de réponces

Dans ce processus il s'agit de valider une interface qui afficher la résultat de recherche, le traitement de réponces pour faire l'ordre des noeuds des XML fichier qui contient le maximum de termes de requêtes.

Interface metadonnées

L'architecture VISS fournit l'interface métadonnées, elle utilise elle utlise l'approche *POSL translator* pour traduire les requêtes d'utilisateur vers les sources de données, ainsi elle évalue les réponses satisfaires et présentables.

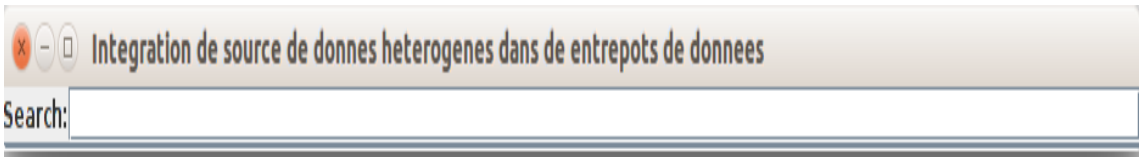


FIGURE 3.7 – Interface d'utilisateur pour fait la recherche

La figure3.7 affiche l'interface de dialogue avec l'utilisateur qui entre les requêtes surlequlle en traitant les requêtes et retournant les informations satisfaites disposées par le schéma XML.

Search: US

Zip	std...	Pay...	Ma...	dep...	Gen...	pay...	hire...	uuid	job...	sal...	jobf...	Dep...	fte	DOB	Mar...	Divi...	id	Gro...	age...	Em...	Citiz...	add...	race	jobt...	ann...	grade	name	job...	pos...	loc...	step	pos...	eeo...
014...	0	28.50	Bra...	""	Fe...	""	10/...	""	""		""	"POL"	0	11/...	Mar...	"Sc...	793...	""	32	Div...	US ...	""	Bla...	Acc...	0		Bro...	0	00...		0	340...	0
014...	0	23.00	Bra...	""	Male	""	1/6/...	""	""		""	"HHS"	0	4/2...	Div...	"Sc...	793...	""	33	We...	US ...	""	Bla...	Acc...	0		LaR...	0	00...		0	340...	0
027...	0	29.00	Bra...	""	Male	""	9/2...	""	""		""	"COR"	0	9/1/...	Sin...	"Sc...	794...	""	31	Inte...	US ...	""	White	Acc...	0		Ste...	0	00...		0	205...	0
000...	29	00.00	""	222...	"M"	GEN	200...	6FE...	118...	D4	D4PT	"DOT"	0.28	""	""	"Aff...	205...	"96...	OV...	""	""	htt...	BLA...	Sch...	18...	909	Win...	909	00...	POL...	0	340...	7
000...	40	00.00	""	222...	"M"	POL	201...	0EA...	146...	POL	D2...	"POL"	0.73	""	""	"Tr...	341...	"11...	31-40	""	""	htt...	WHI...	Poli...	69...	377	Be...	377	00...	POL...	4	205...	0
000...	40	00.00	""	480...	"F"	GEN	201...	284...	148...	DOC	D0...	""	0.73	""	""	"Chi...	205...	"94...	26-30	""	""	htt...	WHI...	Ass...	15...	909	Bra...	909	00...	MS...	0	341...	0
000...	40	00.00	""	222...	"M"	POL	199...	""	144...	POL	D2...	"SHF"	0.28	""	""	"ISB...	341...	"63...	51-60	""	""	htt...	WHI...	Poli...	69...	377	Sa...	377	00...	POL...	4	205...	2
000...	40	00.00	""	301...	"M"	POL	199...	A91...	124...	DOC	D0...	"DTS"	1	""	""	"FS...	341...	"98...	51-60	""	""	htt...	WHI...	Poli...	69...	377	Kra...	377	00...	WA...	4	341...	4

FIGURE 3.8 – Exemple de résultat d'une requête

La figure 3.8 affiche une implémentation de moteur de recherche à l'aide d'architecture VISS pour traiter la requête d'accéder au schéma XML et retournez-la reponce pertinante et satisfaire aux besoins d'utilisateurs.

3.4 Expériment de résultats

Le système d'intégration que nous réalisons contient deux tâches principaux : un médiation avec l'approche LAV celui qui compose à deux structures tel que :

1. les wrappers,
2. le médiateur.

Dans cette section nous intéressons à étudier la performance d'algorithmes de fusion dans la structure médiateur tel qu'elle valide le concept d'intégration.

3.4.1 Évaluation d'algorithme de fusion dans le médiateur

C'est l'algorithme le plus important dans notre système d'intégration celui qui contient plusieurs algorithmes qui influence au fonctionnement de médiateur, nous étudions les évaluations de deux algorithmes que nous posons dans cet algorithme

— les techniques de rapprochement : nous avons l' algorithme Wrinkler-Jarro .
— Comparaison avec dictionnaire de données.

Les techniques de rapprochement

nous implémentons dans ce contexte l'algorithme Wrinkler-Jarro dont nous analysons la performance d'algorithme Wrinkler-Jarro que nous basons pour mesurer la similarité entre les données, dans cette phase la performance de Wrinkler-Jarro est étudiée par la table de contingence et les mesures d'efficacité.
pour afficher les mesures de similarité selon Wrinkler-Jarro nous sélectionnons une comparaison entre deux attributs de sources de données comme un exemple, la table suivante affiche les valeurs.

Attribut	**Zip**	**PayRate**	**CitizenDesc**	**Position**	**ManagerName**	**Sex**
Department	89.0	93.0	85.0	88.0	87.0	89.0
DateFirstHired	85.0	87.0	85.0	84.0	83.0	87.0
FullName	89.0	89.0	85.0	86.0	89.0	91.0
AssignmentCategory	81.0	83.0	79.0	82.0	83.0	83.0
OvertimePay2017	85.0	89.0	83.0	84.0	87.0	85.0
Division	91.0	85.0	87.0	92.0	83.0	91.0
Gender	91.0	89.0	91.0	88.0	91.0	93.0
GrossPayReceived2017	79.0	83.0	77.0	78.0	77.0	81.0
EmployeePositionTitle	84.0	84.0	84.0	93.0	80.0	86.0
PositionUnderFilled	80.0	80.0	82.0	89.0	78.0	82.0
CurrentAnnualSalary	78.0	78.0	78.0	77.0	78.0	80.0

Attribut	**EmployeeNumber**	**EmploymentStatus**	**DateofHire**	**Department**
Department	88.0	88.0	88.0	100.0
DateFirstHired	82.0	94.0	86.0	87.0
FullName	86.0	86.0	88.0	91.0
AssignmentCategory	84.0	82.0	82.0	87.0
OvertimePay2017	86.0	86.0	96.0	87.0
Division	86.0	84.0	86.0	86.0
Gender	88.0	90.0	90.0	91.0
GrossPayReceived2017	74.0	78.0	78.0	81.0
EmployeePositionTitle	83.0	83.0	85.0	88.0
PositionUnderFilled	77.0	81.0	79.0	82.0
CurrentAnnualSalary	83.0	75.0	79.0	80.0

Attribut	**State**	**DOB**	**MaritalDesc**	**ReasonForTerm**	**EmployeeName**	**Age**	**RaceDesc**
Department	91.0	89.0	90.0	82.0	87.0	87.0	92.0
DateFirstHired	85.0	86.0	82.0	83.0	83.0	84.0	87.0
FullName	86.0	86.0	88.0	91.0	97.0	81.0	
AssignmentCategory	81.0	84.0	78.0	81.0	81.0	84.0	85.0
OvertimePay2017	85.0	86.0	80.0	85.0	85.0	92.0	87.0
Division	89.0	93.0	90.0	80.0	85.0	85.0	88.0
Gender	93.0	90.0	86.0	87.0	89.0	94.0	95.0
GrossPayReceived2017	79.0	78.0	76.0	77.0	75.0	80.0	81.0
EmployeePositionTitle	84.0	85.0	79.0	82.0	84.0	87.0	84.0
PositionUnderFilled	80.0	81.0	77.0	78.0	78.0	79.0	80.0
CurrentAnnualSalary	78.0	79.0	77.0	80.0	78.0	81.0	80.0

TABLE 3.1 – Les valeurs de similarite par wrinkler-jarro

Les trois tables affichent les distances de similarité calculées par wrinkler-jaro sur 2 sources de données par sélectionner leurs attributs, dans le but d'évaluer la performance de cet algorithme dans notre algorithme de méditer [Algorithme de médiateur, algorithme 7] nous utilisons la matrice de confusion.

Matrice de confusion

La matrice de confusion est un sommaire pour prédire les résultats d'une performance pour un algorithme, elle contient les valeurs de quatre classifications :[22]

- les vrais positifs VP :sont les valeurs correctes toutes les valeurs,
- les vrais négatifs VN : sont les valeurs incorrectes mais en posant comme des valides valeurs,
- les faux positifs FP : ce sont les valeurs qui sont incorrectes et considèrent comme des incorrectes valeurs,

— les faux négatifs FN : sont les valeurs incorrectes mais en considérant comme des valeurs correctes.

Selon les trois tables précidentes nous produisons la matrice de confusion et calculer leur matrice de classifications.

Nous basons dans notre algorithme une expression qui s'agit de poser que les deux attributs sont simulés sémantiquement ou non par presise si la distance de wrinkler-jarro supérieur au 80.7

Pour calculer les quatre valeurs VP,VN,FP,FN nous utilisons le langage *R* pour donne des résultats bien présises, le logiciel *R* utilise la librarie *scikit-learn* qui ayant la fonction de calculer la matrice de confusion prédéfinit nous trouvons comme suit :

```
Reference
Prediction simule nonsimule
simule 4 1
nonsimule 2 3

Accuracy : 0.7
95% CI : (0.3475, 0.9333)
No Information Rate : 0.6
P-Value [Acc > NIR] : 0.3823

Kappa : 0.4
Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.6667
            Specificity : 0.7500
         Pos Pred Value : 0.8000
         Neg Pred Value : 0.6000
             Prevalence : 0.6000
         Detection Rate : 0.4000
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.7083

       'Positive' Class : 0
```

FIGURE 3.9 – Résultats de la matrice de confusion par *R*

la figure3.9affiche les résultats de matrice de confusion avec des valeurs très importantes sur la performance de notre algorithme, la valeur Accuracy c'est-à-dire quelles sont les cas ou l'algorithme travaille dans la manière correcte, dans notre algorithme nous avons la valeur 0.70 qui donne 70%.

Comparaison avec dictionnaire de données

Le dictionnaire de données se produit par extraction des ontologies pour chaque source de données et de prendre des synonymes parmi Word Net, nous utilisons ce dictionnaire dans notre algorithme comme un autre cas dans la fusion si les techniques de rapprochement ne travaillent pas.

Nous étudions le cas d'utilisation de dictionnaire de données, la fonction principale dans l'algorithme c'est l'obtention des synonymes corrects parmi les attributs de notre source de données, pour calculer sa performance.

Matrice de confusion

Dans ce contexte nous étudions la performance de comparaison avec le dictionnaire de données dans les mêmes étapes que l'analyse d'algorithme Wrinkler-Jaro, tel que nous utilisons le langage R pour calculer la performance et la matrice de confusion tel que nous avons les résultats suivants.

```
Reference
Prediction synonyme nonsynonyme
synonyme 8234  1340
nonsynonyme25 401

Accuracy : 0.829
95% CI : (0.3475, 0.9333)
No Information Rate : 0.6
P-Value [Acc > NIR] : 0.3823

Kappa : 0.5
Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.4530
            Specificity : 0.8400
         Pos Pred Value : 0.9200
         Neg Pred Value : 0.0800
             Prevalence : 0.6000
         Detection Rate : 0.4500
   Detection Prevalence : 0.7500
      Balanced Accuracy : 0.8325

       'Positive' Class : 0
```

FIGURE 3.10 – Matrice de confusion des comparaisons avec dictionnaire de données

3.4.2 Discussion

En général, notre algorithme ayant le but de fusion les attribtuts simulées puis le regroupement de leurs valeurs, par conséquent nous obtenons la complexité générale de notre algorithme en basant sur les études précidentes.

Premier cas

C'est le cas que l'algorithme de fusion base totalement sur la distance de Wrinkler-jaro pour obtient la similarité entre les attributs dans ce cas nous considérons que la performance de notre algorithme étudiée selon la performance d'algorithme Wrinkler-Jaro, on obtient le résultat suivant :

```
Reference
Prediction s n
s 18 2
n  10 2

Accuracy : 0.720
95% CI : (0.3475, 0.9333)
No Information Rate : 0.6
P-Value [Acc > NIR] : 0.3823

Kappa : 0.4
Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.6667
            Specificity : 0.7500
         Pos Pred Value : 0.8000
         Neg Pred Value : 0.6000
             Prevalence : 0.6000
         Detection Rate : 0.4000
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.7083

       'Positive' Class : 0
```

FIGURE 3.11 – Matrice de confusion 1 d'algorithme de fusion(Logiciel R)

Deusième cas

Nous posons dans ce cas que notre algorithme de fusion base sur la comparaison avec la dictionnire de données pour obtennir la distance de similarité, nous étudions la perfomance dans ce cas par la matrice de confusion nous avons la résultat suivante

```
Reference
Prediction s n
s 23 1
n  0 8

Accuracy : 0.718
95% CI : (0.3475, 0.9333)
No Information Rate : 0.25
P-Value [Acc > NIR] : 0.4232

Kappa : 0.4
Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.4225
            Specificity : 0.8590
         Pos Pred Value : 0.8900
         Neg Pred Value : 0.2200
             Prevalence : 0.6000
         Detection Rate : 0.4000
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.7083

       'Positive' Class : 0
```

FIGURE 3.12 – Matrice de confusion 2 d'algorithme de fusion(Logiciel R)

Cas général

Dans ce contexte nous étudions la performance de notre algorithme de fusion parmi la combinaison entre les deux cas précédentes tels que nous utilisons la matrice de confusion implémentée

en langage R pour obtenir la performance totale, on obtient les résultats suivants :

```
Reference
Prediction s n
s 21 1
n  5 5

Accuracy : 0.725
95% CI : (0.3475, 0.9333)
No Information Rate : 0.12
P-Value [Acc > NIR] : 0.65

Kappa : 0.4
Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.4756
            Specificity : 0.7950
         Pos Pred Value : 0.7600
         Neg Pred Value : 0.3500
             Prevalence : 0.6000
         Detection Rate : 0.4000
   Detection Prevalence : 0.5000
      Balanced Accuracy : 0.7183

       'Positive' Class : 0
```

FIGURE 3.13 – Matrice de confusion global d'algorithme fusion

Résultats

Les expéréments de données parmi les algorithmes réalisés donnent des études très importantes telles que :

- les techniques de rapprochement : algorithme de Wrinkler-Jaro ayant la performance de 70% en prise en compte tous les types de données, tel qu'il donne des meilleures valeurs pour les distances de similarité ;
- la comparaison avec le dictionnaire de données donne la maximale performance selon l'algorithme de Wrinkler-jaro tel qu'elle est 82.9% et presque trouvé la plupart valeurs simulées ;
- la performance globale de notre algorithme base sur ces deux cas d'étude

Par ailleurs nous étudions le temps d'exécution de notre algorithme de fusion parmi l'effective de machine pour valider les résultats. Nous observons le changement de temps dans certains nombres de données, tel qu'en mesurant tanque l'algorithme de fusion arrive l'exécution nous avons les valeurs suivantes.

Nombre de données	temps d'execution
l0	2.1
20	2.23
l00	3.06
200	3.234
400	3.79
l000	5.029
2000	6.53
4000	7.033
l0000	l2.25
25000	l7.62
35000	l7.83

TABLE 3.3 – Table de temps d'exécution selon le nombres de données

Une présentation graphique est donnée dans la Fig3.14

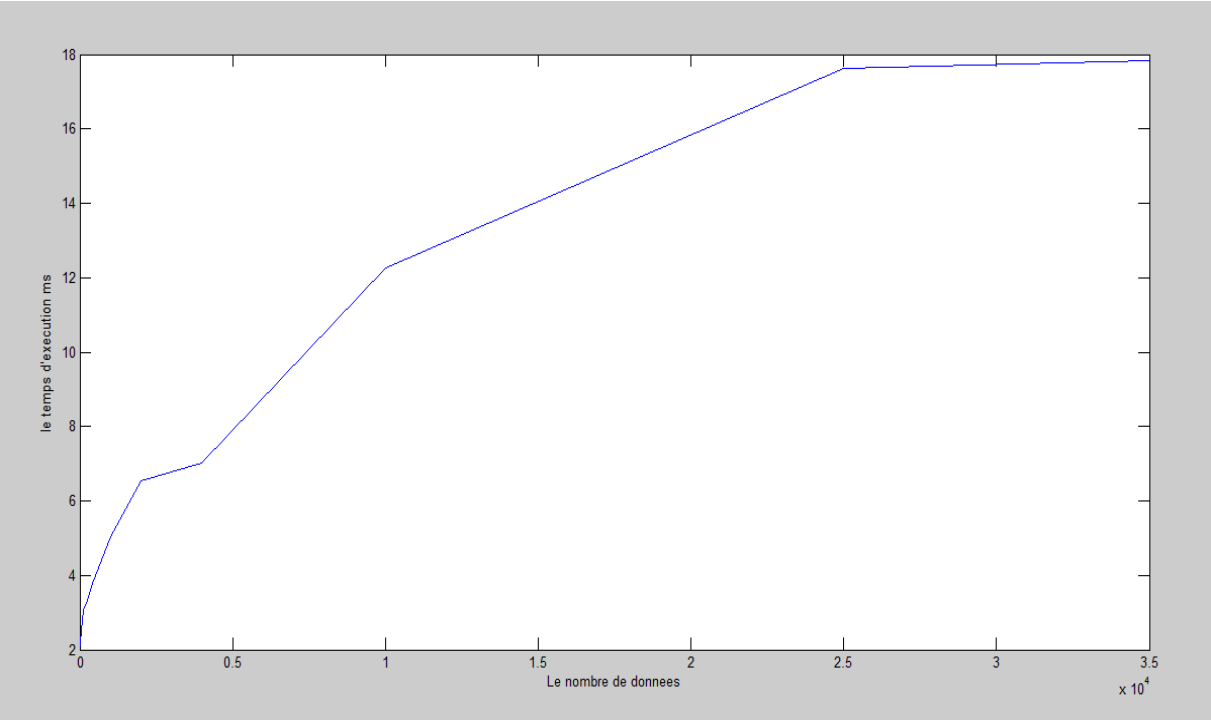


FIGURE 3.14 – Le temps d'exécution en ms depuis les nombres de données

3.5 Conclusion

Dans ce chapitre nous introduisons la partie contribution avec l'implémentation qui étudie les deux problématiques générales dans l'intégration de données. nous implémentons une médiation

d'intégration selon l'approche LAV qui permet de réaliser des fonctions pour regrouper les sources de données hétérogènes dans l'aspect sémantique, telque nous réalisons la structure médiation par trois fonctions de processus ETL, chaque fonction représente un processus d'une structure incluse dans cette médiation, la première structure c'est le wrapper celui qui fait l'extraction des données selon les sources que nous choisissons, dans ce contexte nous réalisons quatre wrappers depuis les sources que nous les sélectionnons nous avons ces quatre wrappers : Xwrapper, Hwrapper ; Jwrapper, Swrapper pour nos sources de données suivantes : XML, HTML, JSON, SQL. une autre structure c'est le médiateur qui permet de transformer les données depuis des sources hétérogènes via un schéma XML global telque il collecte toutes les données de ces sources, nous implémentons le médiateur par réaliser l'algorithme de fusion qui utilise l'algorithme de rapprochement : Wrinkler-Jaro ainsi nous utilisons un dictionnaire de donnés dans le but de trouver les attributs de données qui sont synonymes. Par ailleurs nous implémentons un moteur de recherche a l'aide d'architecture VISS et les techniques de recherche d'information dans le but d'interroger le schéma global XML et obtient les informations pertinentes.

CONCLUSION

Dans le domaine d'intégration des sources des données hétérogènes, il existe plusieurs méthodes de l'intelligence artificielle pour valider un système complet et il vérifie les concepts d'une bonne combinaison de ces sources.
les projets qui valident des systèmes d'intégration des sources de données étudient deux majors problématiques

— proposer une tâche d'intégration de données hétérogènes via un schéma global unique ,

— proposer une approche d'interroguation des requêtes en accédant vers le schéma global .

Dans ce mémoire on a introduit les conceptions d'intégration de sources de données hétérogènes auxquelles on a les utilise pour réaliser une tâche d'intégration, qui est : les sources de données hétérogènes en détaillant sur les aspects d'hétérogénéité : aspect sémantique et l'aspect structurel, les wrappers, le médiateur, l'entrepôt de données, les types de mappings : GAV, LAV, BGLAV.

Ensuite, on a prévenu les fonctions principales de traitement de requêtes qui sont : le répondeur de requête et la réécriture des requêtes dont le but d'optimiser le contexte de requête et faciliter d'obtenir l'information pertinente.

nous avons implémenté un médiation selon l'approche LAV, sur quatre sources de données hétérogènes qui ayant des formats différentes : XML, HTML, JSON, SQL. Ce médiation réalisé par l'implémentation des quatre wrappers et un médiateur, parmi les algorithmes de rapprochement que nous avons implémenté, nous trouvons *Wrinkler-Jarro* et *LCS* acompagné avec le dictionnaire de données que nous avons le validé selon l'ontologie de source de données et *WORDNET* dont le but de programmer un algorithme qui fusionner entre les sources de données en basant sur la mesure de similarité. Enfin, nous avons implénenté un moteur de recherche à l'aide de l'architecture VISS supporté par les techniques de recherche d'informations, pour interroger le schéma global xml.

Notre implémentation des deux algorithmes : Wrinkler-Jarro et LCS sont valides pour tous les cas possibles des données et elles sont robustes pour l'information qui incluse dans les sources. Par ailleurs dans la tâche de traitement de requêtes, l'architecture VISS contient de plusieurs composantes pour analyser le contexte d'interrogation dans le but d'optimisation et réduit le taux de reponcer, les techniques de recherche d'information permettent de filtrer le contenu d'information en prise en compte beaucoup plus sur la pertinence de cette information et d'augmenter le pourcien- Tage de similarité entre les termes de requête et les contenus de l'information choisie.

Comme perspectif, il est meilleur d'optimiser les structures de comparaisons qui sont inclus dans les techniques de rapprochement, tel que, l'explosion de données avec d'importants volumes se traiter par Wrinkler-Jarro et LCS peuvent perdre un pourcentage de validation sans prise en compte les contenants où les contextes des données, l'architecture VISS fournit des structures de traiter les requêtes qui interroger que le schéma global de formats XML, celui qui limite cette architecture de travail aux autres formats de données. Donc, il peut être mieux, de modifier quelques structures ou des fonctions pour généraliser l'architecture VISS.

BIBLIOGRAPHIE

[1] DE VLIEGER Paul. *Création d'un environnement de gestion de base de données « en grille ». Application à l'échange de données médicales*. PhD thesis, Université d'Auvergne, 2011.

[2] Dung Xuan Nguyen. *Intégration de bases de données hétérogènes par articulation à priori d'ontologies : application aux catalogues de composants industriels*. PhD thesis, Université de Poitiers, 2006.

[3] data.org. open data. `www.data.org`.

[4] Maurizio Lenzerini. Data integration : A theoretical perspective. pages 233–246, 2002.

[5] Mohand-Said Hacid and Chantal Reynaud. L'intégration de sources de données. *Revue Information-Interaction-Intelligence*, 3 :4, 2004.

[6] Assia Soukane. *Génération automatique des requêtes de médiation dans un environnement hétérogène*. PhD thesis, Versailles-St Quentin en Yvelines, 2005.

[7] Isabel F Cruz and Huiyong Xiao. The role of ontologies in data integration. *Engineering intelligent systems for electrical engineering and communications*, 13(4) :245, 2005.

[8] Gio Wiederhold. Mediators in the architecture of future information systems. *Computer*, 25(3) :38–49, 1992.

[9] Marc Friedman, Alon Y Levy, Todd D Millstein, et al. Navigational plans for data integration. *AAAI/IAAI*, 1999 :67–73, 1999.

[10] Panos Vassiliadis, Alkis Simitsis, and Spiros Skiadopoulos. Conceptual modeling for etl processes. In *Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP*, pages 14–21. ACM, 2002.

[11] Yannis Katsis and Yannis Papakonstantinou. View-based data integration. *Encyclopedia of Database Systems*, pages 3332–3339, 2009.

[12] Peter McBrien and Alexandra Poulovassilis. Data integration by bi-directional schema transformation rules. In *Proceedings 19th International Conference on Data Engineering (Cat. No. 03CH37405)*, pages 227–238. IEEE, 2003.

[13] Piotr Wieczorek Paolo Guagliardo. *Query Processing in Data Integration*, chapter 05. Dagstuhl Publishing, 2013.

[14] J. Hammer K. Ireland Y. Papakonstantinou J. Ullman S. Chawathe, H. Garcia-Molina and J. Widom. The tsimmis project : Integration of heterogenous information sources.

[15] Bergamaschi S.-Castano S. Corni A. Guidetti R. Malvezzi G. Vincini M Beneventano, D. Information integration : the momis project demonstration.

[16] Brink A. Emery R. Lu J. J. Rajput A. Ward C Subrahmanian V. D., Adali S. Hermes : A heterogeneous reasoning and mediator system.

[17] Gayathri Jayaraman. A mediator-based data integration system for query answering using an optimized extended inverse rules algorithm. Master's thesis, Department of Systems and Computer Engineering, Carleton University, 2010.

[18] Lahmar Fatima. *Une approche Hybride d'intégration de sources de données hétérogènes dans les datawarehouses*. PhD thesis, Université Mentouri de Constantine, 2011.

[19] Shokoh KERMANSHAHANI. *IXIA (IndeX-based Integration Approach) A Hybrid Approach to Data Integration*. PhD thesis, Université Joseph-Fourier -GrenobleI, 2009.

[20] A. Miller. Wordnet : A lexical data-base for english. *Communications of the ACM*.

[21] Hinrich Sch ütze Christopher D. Manning, Prabhakar Raghavan. *Introduction to Information Retrieval*. CAMBRIDGE UNIVERSITY PRESS, 2008.

[22] Pankaja R Maria Navin J R. Performance analysis of text classification algorithms using confusion matrix. *International Journal of Engineering and Technical Research (IJETR)*, 2016.

[23] Vincent Labatut and Hocine Cherifi. Evaluation of performance measures for classifiers comparison. *arXiv preprint arXiv :1112.4133*, 2011.

[24] J. D Ullman. Information integration using logical views. in database theory—icdt'97. *Springer Berlin Heidelberg*, 1997.

[25] April Reeve. *Managing Data in Motion .Data Integration Best Practice Techniques and Technologies*. Elsevier, 2013.