Université Saâd DAHLAB de Blida

Faculté des sciences

Département d'Informatique

Mémoire présenté par :

M^lles **KEDDAR Manel et KHELLADI F.Zohra**

Pour l'obtention du diplôme de Master

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Ingénierie des logiciels

Sujet :

Génération automatique des requêtes de médiation dans un environnement hétérogène

Soutenu le : 02 Juillet 2011, devant le jury composé de :

M^me **FAREH**	Promotrice
M^r **M.BALA**	Président
M^r **OULED AISSA**	Examinateur
M^lle **N.**	Examinatrice

PROMOTION 2011/2012

Dédicaces

Au terme de ce travail nous tenons à remercier tout d'abord ALLAH tout puissant de nous avoir donné le courage et facilité la réalisation de ce travail.

C'est avec un immense plaisir que je dédie ce travail

A mes très chers parents qui sont toute ma vie et tout ce que j'ai de plus cher au monde, en témoignage de ma reconnaissance infinie pour les nombreux sacrifices qu'ils n'ont cessé de déployer pour moi et dont je serais à jamais redevable.

Qu'ils trouvent en ce travail la preuve de mon éternel amour et ma reconnaissance envers eux.

Que dieu les garde et leur procure la santé et le bonheur.

Ainsi qu'à mes frères bien aimés Amine et Nadjib qui sont ma fierté, en témoignage de ma grande affection pour eux.

Puisse dieu les protéger.

Aussi à toute ma famille mes tantes mes oncles et ma grand-mère.

Aussi à mes copines d'or Mellyara, Halima,Affaf,Ibtissem,Yasmine et Imène.

Sans oublier mon amie Fatma zohra (mon acolyte dans cette épreuve) ainsi que toute sa famille.

Mlle KEDDAR Manel

Remerciement

Tout d'abord, nous tenons à rendre grâce à dieu tout puissant pour nous avoir donné le courage et la détermination nécessaire pour finaliser ce travail et le mener à terme.

En second lieu et puisque une seule main ne lie pas un fagot de bois, Nous tenons à remercier chaleureusement et affectueusement tous ceux et toutes celles qui de près ou de loin ont contribué à la réalisation de notre projet.

On ne saurait ne pas remercier encore une fois nos parents respectifs qui, par leur amour et leur affection nous ont permis d'arriver là où nous somme aujourd'hui.

Nous tenons également à remercier M^{me} Fareh qui a endossé son rôle de promotrice de la meilleure façon qui soit. Nous retiendrons son aide précieuse, ces conseils avisés, ces idées riches mais aussi sa sympathie et ses encouragements. Nous prions Allah de lui rendre grâce pour avoir fait de notre travail avec elle, un réel honneur et grand plaisir.

Par ailleurs ; nous rendons un vibrant hommage à l'ensemble du corps professoral du département d'informatique de l'université Saâd DAHLAB de Blida qui ont contribué activement et vaillamment à notre formation pendant ces cinq dernières années.

Dédicaces spéciale à tous nos camarades de la promotion 2011/2012Aussi à l'ensemble du personnel de la bibliothèque centrale qui se sont montrés très professionnels et serviables et nous ont fournis des conditions de travails optimales.

A tous ceux et à toutes celles dont les noms n'apparaissent pas sur cette page, qu'ils demeurent convaincus, que nous ne les avons point oubliés et qu'ils soient assurés de notre profonde gratitude.

Merci.

Mlle KEDDAR Manel et Mlle Fatma Zohra Khelladi.

Sommaire

Chapitre I :Etat de l'art des système d'integration

chapitre II :Etat de l'art sur la génération des requêtes de médiation

Chapitre III:Approche proposée

Chapitre IV: Analyse des besoins et conception

Chapitre V: Implémentation

Chapitre VI: Tests et validation

Conclusion Générale

Les figures

RESUME

Résumé

Les systèmes de médiation sont aujourd'hui largement développés et connus. Cependant, leur mise en œuvre pose un certain nombre de problèmes, en particulier la définition de requêtes de médiation en présence d'un grand nombre de sources de données, et d'un volume important de métadonnées les décrivant. Ce problème est d'autant plus complexe que les sources sont hétérogènes.

Face à cette problématique, nous proposons dans ce projet pour le contexte relationnel (MySQL) et les métadonnées qui sont présentées par XML et les connaissances OWL. Et d'un schéma global OWL, une approche de génération automatique de requêtes de médiation. A partir de la description d'un ensemble de sources de données hétérogènes

Notre but est de générer d'une manière automatique avec la prise en compte de la sémantique, à partir d'une requête globale (écrite en termes de schéma global), des sous requêtes écrites en modèle des sources. Il s'agit de produire un ensemble de requêtes de médiation possibles à partir d'un ensemble de sources de données distribuées et hétérogènes (sources de données, de métadonnées et connaissances).nous avons développé Un outil, permettant de générer, automatiquement, des requêtes de médiation.

Mots clés :

Système d'intégration de données, médiateur, adaptateur, sources de données hétérogènes, schéma global,requête globale, requête locale, sémantique, ontologie.

Abstract

Nowadays, mediationsystems are widelyused. However, theirimplementationraisesseveral problems, especially the definition of mediationquerieswhenthereis a high number of sources, and an important amount of metadata. The heterogeneity of the data sources makesthisproblemeven more complex.

Facedwiththisproblem, we propose in thisproject for the relationalcontext (MySQL) and the metadatathatispresented by (XML) and knowledge(OWL). And a global schemaOWL, an approach for the automaticgeneration of mediationqueries. From the description of a set of heterogeneoussources.

RESUME

Our goal is to generatewithan automaticwaywith the inclusion of semantics, from a global query (written in terms of global schema), subquerieswritten in model sources. This is to producea set of mediationqueries possible from a set of an heterogeneous (data sources, metadata and knowledge). We have developedatool to generate, automatically, mediationqueries.

Keywords:

Data integration system, mediator, adapter, heterogeneous data sources, global schema, global query, query local semantics,Ontologie.

ملخص

أنظمة وساطة معروف على نطاق واسع وتطويرها. ومع ذلك،فإن تنفيذها يثير عددا من المشاكل،وخاصة تعريف من الاستفسارات وساطة في ظل وجود عدد كبير من مصادر البيانات،وكمية كبيرة من البيانات الوصفية واصفا إياها. هذه المشكلة أكثر تعقيد الآن المصادر هي غير متجانسة .

في مواجهة هذه المشكلة،نقترح في هذا المشروع لسياق العلائقية (الخلية)،والفوقية التي ترد من قبل XMLوالمعرفة .OWL وعالميOWLمخطط، وهو نهج لتوليد التلقائي للاستعلامات وساطة. من وصف مجموعة من مصادر البيانات غيرا لمتجانسة هدفنا هوإنشاء طريقة تلقائية مع إدراج دلالات،من استعلام العالمية (مكتوبة من حيث المخطط العالمي)،والاستعلامات الفرعية في كتابي مصادر نموذج.

هذا هو إنتاج مجموعة من الاستفسارات وساطة محتملة من مجموعة من مصادر البيانات الموزعة وغير المتجانسة (مصادر البيانات،والفوقيةوالمعرفة)،ونحنقدوضعتأداةلتوليد،تلقائيا،وساطةاستفسار.

كلمات البحث:

بيانات نظام التكامل، وسيط، محول، غير متجانسة مصادر البيانات، ومخطط عالمي، الاستعلام العالمية، والاستعلام المحلية، وعلم الوجود الدلالي.

Introduction générale

Introduction générale

1. Contexte de travail

De nos jours, les systèmes multi-source sont de plus en plus développés. Ils sont définiscomme l'intégration de plusieurs sources hétérogènes et distribuées. Parmi ces systèmesd'informations, nous distinguons les entrepôts de données, les systèmes d' informations baséssur le web, les systèmes de bases de données fédérées, ou encore les systèmes de médiation.

Notre étude s'inscrit principalement dans le contexte des systèmes de médiation. Un système de médiation est un système qui permet d'inter opérer sur un ensemble de sourceshétérogènes et distribuées. Ses composants essentiels sont : le schéma global appelé schéma

de médiation, les mappings du schéma global avec les sources, les fonctions de réécriture de requêtes et les fonctions de composition des résultats. Les mappings du schéma global avec les sources sont des requêtes, appelées requêtes de médiation, dont l'expression varie selon l'approche choisie : 1) approche descendante (Global As View ou GAV) où chaque objet duschéma global est défini par une requête sur les sources, 2) approche ascendante (Local As View ou LAV) où chaque objet d'une source de données est défini par une requête sur le Schéma global.

2. Problématique et motivation

Plusieurs problèmes de conception émergent lors de l'utilisation de ces médiateurs. L'une des principales difficultés rencontrée dans un système de médiation est la définition du schéma global et la définition des mappings (requêtes de médiation) qui relient le schéma global aux sources de données.

L'écriture manuelle des requêtes de médiation donne sans doute le résultat le plus pertinentau regard des besoins des utilisateurs. Cependant il est difficile de l'entreprendre en raison dugrand nombre de sources de données qui peuvent être impliquées (des centaines ou des milliers) et du volume important de méta-données les décrivant (description des schémas

dessources et du schéma global, assertions de correspondance linguistique, assertions intra-source et inter-source, etc.). La question principale est de savoir comment automatiser lagénération de requêtes de médiation ?

Par ailleurs, comme les données sont hétérogènes, il est important de garantir leur conformitémutuelle et leur conformité avec le schéma global. Deux types de conflits liés àl'hétérogénéité des sources sont distingués:

1) les conflits sémantiques liés au schéma dus àl'utilisation d'une terminologie différente pour décrire un même concept (ex. Nom et Nom_Pers).

. La non prise en compte de ces conflits peut fausser la sémantique des requêtes de médiation et retourner à l'utilisateur desrésultats non conformes à ceux définis dans son schéma de médiation. Les opérations quicomposent une requête de médiation ne sont valides que si les conflits liés à l'hétérogénéitédes données sont détectés et résolus. La seconde question que l'on se pose est comment détecter et résoudre lesconflits liés à l'hétérogénéité des sources de données ?

Les difficultés principales d'intégration qu'on doit la surmonter sont :

- L'hétérogénéité des sources (XML, Relationnelle, et OWL), leur mise en œuvre pose un certain nombre de problèmes (sémantique syntaxique et structurel), tant en ce qui concerne la génération des liens sémantique entre le schéma de médiation et les sources (requête de médiation), qu'en ce qui concerne l'adaptation de l'accès aux besoins des utilisateurs.
- La définition du schéma global et la définition de mapping qui reliant le schéma globale aux sources de données.

De manière générale, la problématique peut se résumer dans l'hétérogénéité de métadonnées qu'il est nécessaire de résoudre pour permettre de mettre en correspondance les sources et autoriser l'interrogation et la réponse aux requêtes de façon transparentes.

Introduction générale

3. Objectifs de travail

Au vu des articles de recherche étudiés, deux catégories de travaux se distinguent. Celle qui vise à interroger les sources de données distribuées et hétérogènes et à la définition derequêtes de médiation, et celle qui vise à intégrer les données et à construire le schéma global.

Très peu de travaux se sont intéressés à la génération de requêtes de médiation et à la priseen compte des conflits liés à l'hétérogénéité des sources durant le processus de génération, laplupart se sont focalisés sur la détection et (ou) sur la résolution d'un conflit particulier dansle contexte de l'intégration de données. Nous présentons un état de l'art dans le chapitresuivant qui illustre le manque de travaux dans le contexte de l'interrogation.

Les systèmes de médiation étant de nos jours des systèmes de plus en plus développés etconnus, méritent une exploration plus approfondie dans le domaine de la recherche. Le peu detravaux a motivé notre étude en vue d'explorer plus en détails la problématique liée à cessystèmes et d'apporter une solution originale, adaptée aux attentes des utilisateurs.

Face à la problématique de définition de requêtes dans un système de médiation, nous avons Opté pour une approche de génération automatique des requête de médiation qui permet à l'utilisateur de poser une requête globale sans se soucier de la structure des sources de données à interroger ni de leur localisation. Notre proposition pour la générationde requêtes de médiation se décompose en sept étapes :

- Extraction et traduction des schémas sources en (OWL).
- Annotation de chaque classe et propriétés des schémas sources locaux.
- Extraction des relations sémantiques et structurelles entre les classes et propriétés.
- Regroupement des classes qui se correspondent.
- Construction des classes représentatives.
- Calcul des mesures de similarité.
- Calcul et génération de la table de mapping

Nous avons développé un prototype qui permet de générer automatiquement des requêtes

(SQL, XQuery, et des requête pour l'interrogation des sources OWL) et qui tient compte aussi de l'hétérogénéité des sources de données.

4. Organisation du mémoire

Afin d'atteindre les objectifs cité ci-dessus, notre mémoire s'articulera autour des chapitres suivants:

- **Chapitre I: Etat de l'art des systèmes de médiation** :

  Dans ce chapitre nous avons présenté les différents problèmes d'intégrations ensuite nous avons présentés l'approche médiateur.

- **Chapitre II :**Etat de l'art des différentes approches existantes pour la génération des requêtes : lors de ce chapitre nous avons présenté une étude sur les travaux de recherche existants dans les contextes de la génération de requêtes de médiation et de l'intégration de données. Il présente aussi unétat de l'art sur les outils commerciaux présents sur le marché industriel.

- **Chapitre III:**Approche proposée, qui sera consacrée à la méthode que nous avons proposé.

- **Chapitre IV:** Analyse des besoins et conception, dans lequel, nous parlerons de la démarche de modélisation utilisée, nous présenterons la conception du système suivant cette démarche. Partant de l'analyse des besoins fonctionnels et techniques, ensuite, l'analyse du système et présentation des scénarios et enfin, la présentation de l'architecture statique.

- **Chapitre V:** Implémentation du système, dans lequel nous présenterons l'environnement de développement (langages et outils utilisés). Ensuite, le diagramme d'accessibilité, l'architecture de déploiement du système et enfin les impressions d'écran.

- **Chapitre VI:** Tests et validation, oùnous évaluerons les expérimentations du système et la qualité des résultats qu'il fournit comparée aux objectifs initiaux, et ce, à travers des

expérimentations et des tests sur deux requête une manuelle et l'autre générée automatiquement par le système.

En fin, la conclusion de ce mémoire synthétisera nos principales contributions et donnera quelques perspectives à notre travail.

Chapitre I :
Etat de l'art sur les systèmes de médiations

Chapitre I: Etat de l'art des systèmes de médiation

1. Introduction

Avec l'avènement de l'internet et l'intranet, et aussi la diversité des supports de stockage, l'interopérabilité et l'accès à ces différents sources de données est devenue un problème, d'où l'issue de l'idée d'intégration de données pour permettre un accès cohérent. Pour cela différentes approches d'intégrations ont été mises en œuvres pour homogénéiser les différentes formats trouvées.

2. C'est quoi l'intégration

L'adoption massive des bases de données entraine un grand besoin d'intégrer de différentes données avec une possibilité qu'ils soient incompatibles. Il existe deux grandes techniques liées au processus d'intégration de différentes données résidant dans différentes sources :

- L'entrepôt de données qui visent à regrouper des données en vue de leur analyse.
- Les médiateurs permettent l'accès transparent à différents types de données hétérogènes sans toucher à leurs sources.

L'autonomie des sources de données à intégrer représente différents défits comme la création d'une interface appropriée pour l'accès à ces sources de données, faire le mapping des vues à intégrer avec les schémas sources, faire différentes requêtes sur ces sources de données selon leurs schéma.

➢ L'autonomie

Les données qui participent au scenario d'intégration sont autonomes ce qui veut dire que les outils qui gèrent et contrôlent ces sources sont indépendants. En conséquence, pour la création d'un système d'intégration d'information, le problème d'autonomie doit être pris en considération.

Les différents aspects de l'autonomie sont:

Autonomie de conception: chaque source d'information locale a son propre modèle, langage de requête, schéma de conception, sémantique d'interprétation de données…etc.

Autonomie de communication : chaque source d'information locale décide quand elle va communiquer avec le système d'intégration c.à.d. quand elle répond à la requête.

Autonomie d'exécution: c'est pas le système d'intégration qui gère et contrôle les transactions locales ou les opérations externes de la source locale. En d'autre terme, une source d'information dans un scenario d'intégration de données n'as pas besoin d'informer les autres sources de l'ordre de son exécution et ses opérations.

Autonomie d'association : les sources locales ont la possibilité de s'associer ou de se désassocier du système d'intégration c.à.d. elles décident comment faire participer leurs fonctionnalités et données dans le système d'intégration. [SHO,09]

- Hétérogénéité

Un des problèmes les plus complexes lors de l'intégration de plusieurs sources de données autonome est l'hétérogénéité de ces sources. Les systèmes d'information qui doivent être intégrés peuvent être développés dans des environnements différents, avec des schémas conceptuels et définitions de données différents. Cela conduit à une hétérogénéité à différents niveaux du système. L'hétérogénéité est indépendante de la distribution physique des données.

3. Les conflits liés à l'intégration

Il existe beaucoup de problèmes liés à l'intégration des données hétérogènes. Les sources de données qui vont être interrogé sont différentes que ce soit au niveau structurel ou sémantique **[1].** Pour réussir à intégrer ces différents sources il faut prendre en considération les conflits liés à l'hétérogénéité des sources.

Les données de différents systèmes d'intégration sont, pour la plupart, issues des différentes sources de données hétérogènes, il existe différentes raisons qui expliquent cette hétérogénéité, par exemple différences dans le matériel, le système logiciel, les systèmes des SGBD, et les données…etc.

Le schéma suivant présente les conflits de données en trois niveaux : conflits syntaxiques, conflits structurels et conflits sémantiques **[2].**

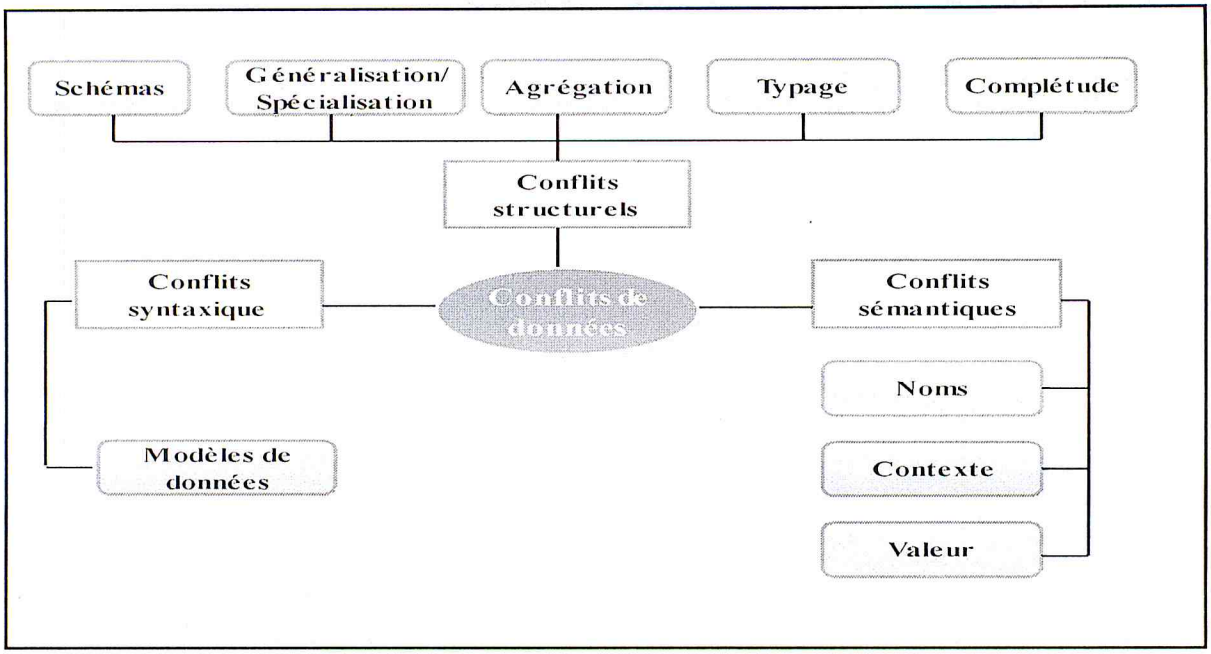


Figure 1 : Classification des conflits de données ***[2].***

3.1. Les conflits sémantiques

L'interrogation de plusieurs sources de données hétérogènes peut donner lieu à différent types de conflit sémantiques. Plusieurs chercheurs ont identifié trois types de conflits sémantiques: conflit de contexte, conflit de valeur et conflits de noms. **[2]**

3.1.1. Conflits de noms

Se trouvent lorsque les différents schémas utilisent des noms différents pour représenter le même concept ou propriété (synonyme), ou des noms identiques pour des concepts ou des propriétés différents (homonymes) **[2].**

3.1.2. Conflit de mesure ou de valeur

Ce type de conflit survient quand deux systèmes expriment la même valeur à l'aide d'unités différentes. Par exemple si on veut intégrer deux tables « Produit », dont l'attribut « Prix » du premier a pour unité de mesure l'Euro et l'attribut du deuxième composant utilise le dollar **[2]**

3.1.3. Conflit de contexte

Se retrouvent dans le cas ou les concepts semblent avoir la même signification dans deux schémas mais sont différents a cause de leur contexte.par exemple le poids d'une personne dépend de la date ou elle se pèse **[2].**

3.2. Les conflits syntaxiques

Résultant de l'utilisation de modèles de données différents d'un système à l'autre.

Des concepts différentes sont utilisés pour structurer la même information (relation dans le relationnel, classe dans le modèle objet, balise XML,...etc.) **[2].**

Exemple : « représentation d'une entité *Thèse* dans différents modèles »

- Schéma relational

  CREATE TABLE [DBO]. [thèse]

  ([Numthese][varchar](10)NOT NULL ,

  [titre_t] [varchar] (20) NOT NULL,

  [domaine_t][varchar](30)NOT NULL).

- XML : Définition de Type de Données (DTD)

  < !DOCTYPE Thèse[

  < !ELEMENT Thèse (Numthese,titre_t,domaine_t)>

  < !ELEMENT Numthèse(#PCDATA)>

  <!ELEMENT titre_t(#PCDATA)>

  <!ELEMENT domaine_t(#PCDATA)>].

- Schéma orienté objet: Public Thèse (String Numthese, String titre_t, String domaine_t).

3.3. Les conflits structurels

Résultant d'une structuration et classification différentes des informations. Ils sont étroitement liés aux choix de conception **[2].**

Exemple : représentation de l'information « Téléphone » de manière différente dans le même modèle(Relationnel).

Les numéros de téléphone sont représentés par des attributs dans la relation

Entreprise (Code, Nom, Adresse, tél1, tél2, tél 3, Fax, e_mail).

Les numéros de téléphone sont regroupés dans une relation à part :

Entreprise (Code, Nom, Adresse, Fax, e_mail).

Tél_Entreprise(Code, Tél).

- Les conflits de schémas

Résultent de l'utilisation de différents concepts pour représenter le même objet. Une information peut être représentée par une entité dans un système (S1) et par une relation dans autre système (S2) **[2].**

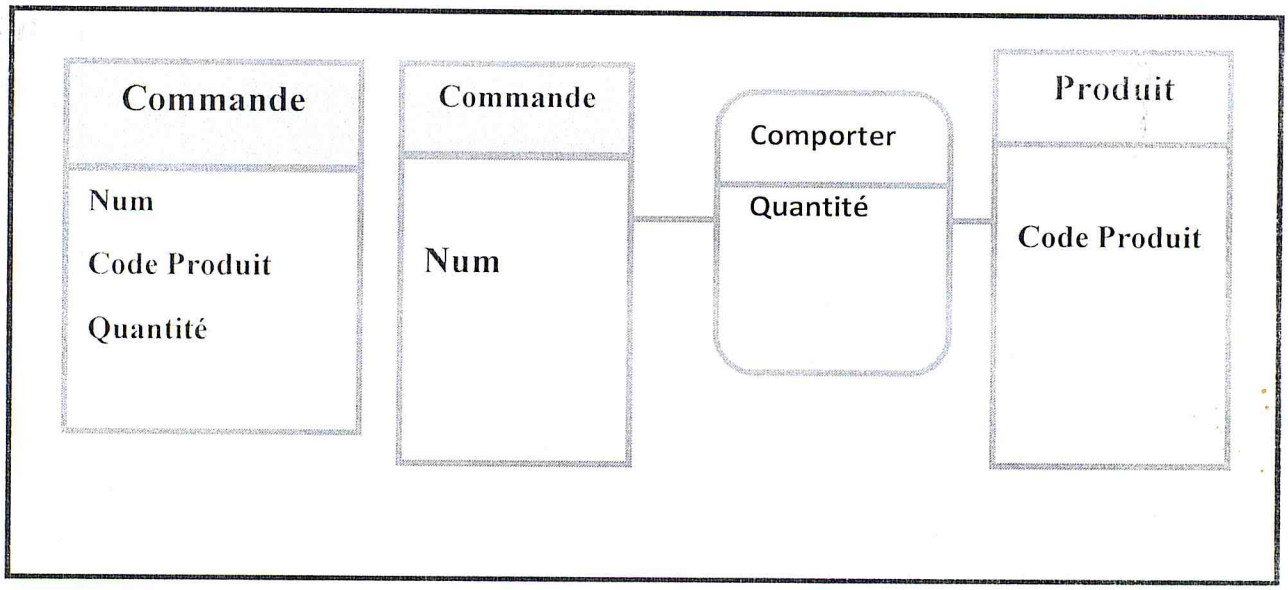


Figure 2 : Exemple représente les conflits Schématiques .

- Les conflits de généralisation /spécialisation

Résultant des différences de hiérarchisation des informations **[2].**

Exemple : dans un système(S1) les personnes sont regroupées dans un seul objet PERSONNE alors que dans (S2) on utilise deux objets HOMME et FEMME pour représenter les personnes.

- Les conflits d'agrégation

Résultant d'un niveau de granularité différent de deux systèmes.

La valeur d'un attribut sur un système correspond à une agrégation des valeurs de plusieurs attributs sur un autre **[2].**

Exemple : nous disposons des moyennes modulaires des étudiants dans un système(S1) et le détail des notes obtenues au cours des examens dans le deuxième(S2).

- Les conflits de typage

Résultant des différences de typage pour le même objet dans les différents systèmes de la coopération **[2].**

Exemple : le mois est représenté par une chaine de caractère sur le système (S1), est représenté comme entier sur le système(S2).

- Les conflits de complétude

Apparaissent lorsque des objets se correspondent partiellement sur les différents systèmes.

C'est-à-dire qu'une partie d'un objet de système(S1) trouve une correspondance sur le système(S2) **[2].**

Exemple

(S1) : ETUDIANT (matricule, Nom, Prénom, Adresse).

(S2) : STAGIAIRE (N°ins, Nom, Adresse, Ville, Pays).

Dans (S1) l'attribut Adresse contient Rue, Ville et Pays alors que dans(S2) l'adresse contient Seulement la Rue, Dans (S2) le nom contient à l fois nom et prénom.

4. Définition d'un système de médiation

Un médiateur est un système puissant qui répond à des besoins collectés à partir des sources dispersées et hétérogènes d'une manière transparente. Il permet d'homogénéiser l'accès à ces sources en résolvant le problème des conflits afin de pouvoir fournir à l'utilisateur une vue centralisée et uniforme.

L'approche médiateur présente l'intérêt de pouvoir construire un système d'interrogation de sources de données sans toucher aux données qui restent stockées dans leurs sources d'origine.

5. Les composants d'un système de médiation

L'approche de médiation consiste à utiliser un médiateur et un ensemble d'adaptateurs chacun correspond à une source de données pour faciliter l'accès aux bases de données hétérogènes.

Un médiateur comprend un schéma global ou ontologie dont le rôle est central. L'ontologie fourni un vocabulaire structuré servant comme support pour exprimer des requêtes.

L'interrogation se fait réellement à travers les adaptateurs qui traduisent les requêtes.

La figure 3 illustre l'architecture d'un système de médiation en générale.

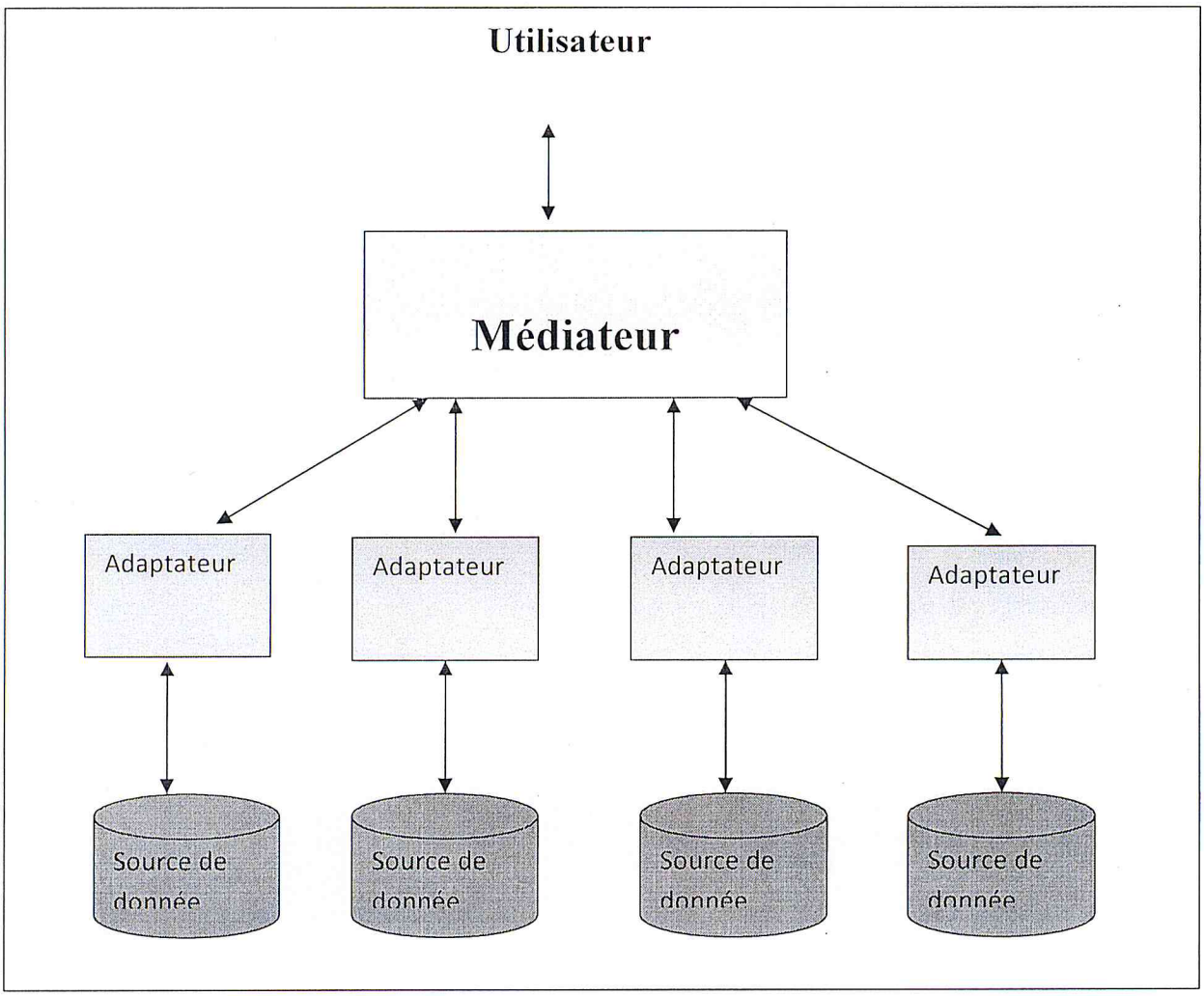


Figure 3: Architecture d'un système de médiation

- **Le niveau sources**

Comporte les différentes sources de données ; à l'aide d'un adaptateur (wrapper), il est capable de communiquer avec le médiateur et facilitateur du niveau supérieur, en leur fournissant une vue homogène de la source à laquelle il est associé. Un adaptateur accepte une requête donnée dans le langage commun du médiateur, la transcrit dans le langage natif de la source et exécute la requête. Le résultat de la requête transmis sous forme native est alors transformé suivant le modèle de données global du médiateur et renvoyé à celui-ci.

- **Le niveau médiateur**

Comporte des médiateurs permettant d'intégrer les données en provenance de différentes sources afin de répondre aux requêtes des utilisateurs. Ce module joue un rôle actif dans la couche entre les applications utilisateurs et les sources de données. Son rôle est de fournir à la couche supérieure une vue centralisée des sources qui sont hétérogènes et distribuées.
On trouve aussi à ce niveau des facilitateurs permettant d'identifier les sources qui peuvent être des sources de données ou des médiateurs, et de composer les réponses pour les utilisateurs.

- **Le niveau adaptateur**

✓ convertit les requêtes exprimées dans le modèle de médiation provenant d'un ou de plusieurs médiateurs vers les modèles de sources de données locales.

✓ convertit les données résultantes d'une requête, du modèle de base de données locale vers le modèle de médiation

✓ un wrapper de données offre une interface homogène locale d'accès aux données sources (résolution des conflits syntaxiques) **[2].**

6. Les approches d'intégration utilisées par les médiateurs

Les systèmes de médiation sont classifiés suivant la relation entre les schémas des sources locales par rapport au schéma global du médiateur.

On distingue deux approches : un schéma global comme vue sur des schémas locaux GAV (Global As View) ou des vues locales comme vues du schéma global LAV (Local As View). Dans l'approche GAV, la transformation d'une requête sur le schéma global en requête sur le schéma local est une simple opération faite par le gestionnaire de vues. Dans le cas d'une

approche LAV, la requête sur le schéma global doit être reformulée suivant les schémas des sources locales. D'un autre côté, dans une architecture GAV, une modification sur l'ensemble des sources locales ou sur leur schéma entraîne une reconsidération complète du schéma global. Dans l'architecture LAV, chaque source est spécifiée de manière indépendante. Un changement local de schéma se prend en compte en mettant à jour la vue locale. De plus, si les données des sources locales n'ont pas le même format (relationnel, semi structuré), il est difficile de définir le schéma global comme vue des sources de différents formats. En utilisant une approche LAV, chaque source peut être décrite séparément par un mécanisme de vue spécifique à son format.

Les architectures les plus souvent utilisées sont les architectures GAV.

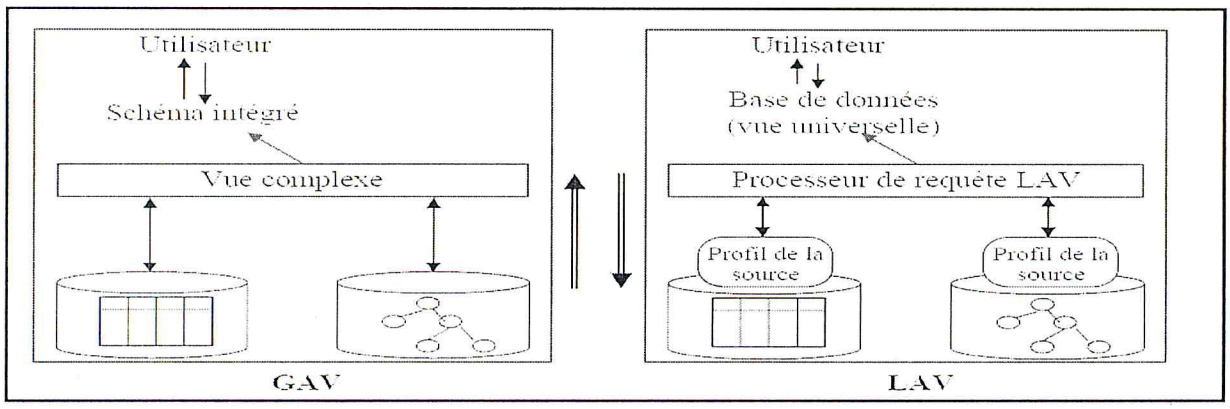


Figure 4 : Comparaison des architectures GAV et LAV ***[3].***

7. Avantage et inconvénient des deux approches

- Pour l'approche LAV : elle facilite l'ajout d'une nouvelle source, par contre la traduction des requêtes est complexe.
- Pour l'approche GAV : contrairement à l'autre approche, cette approche nous facilite la traduction des requêtes, mais l'ajout d'une nouvelle source implique la modification du modèle globale.

8. Les types de médiation

Selon la manière avec laquelle le médiateur traite les requêtes et utilise la sémantique des sources de données, nous pouvons distinguer deux types de médiation : médiation de schéma et médiation de contexte **[2].**

8.1. Médiation de schéma

L'intégration des schémas des sources locales permet de déterminer la structure du schéma global et de définir les correspondances entre ces schémas et le schéma global et aussi de localiser et intégrer les informations pertinentes. Les requêtes sont exécutées sur ces connaissances (préétablies) qui indiquent au médiateur la localisation physique des données et les correspondances entre les données pour permettre leur combinaison et transformation afin de restituer des résultats homogènes et exploitables.

Ce type de médiation doit résoudre les conflits sémantiques et structurels afin d'obtenir un schéma global **[3].** Une fois le schéma global défini, il reste de déterminer pour chacune des sources locales, les correspondances avec le schéma global.

8.2. Médiation de contexte ou de source

Cette approche repose sur une intégration dynamique des informations. Chaque application et chaque source de données locale doivent exprimer leur domaine sous forme de contexte d'utilisation des informations et ce pour permettre au médiateur, lors du traitement d'une requête, de comparer le contexte de l'application aux contextes des systèmes participant à la coopération. Le médiateur est guidé alors par des informations à caractère sémantique pour résoudre dynamiquement une requête sans connaissance préalable des SI participants.

9. Langage de médiation

La médiation adopte un langage unique appelé langage de médiation pour permettre aux différents utilisateurs et applications d'interroger les différentes sources de données de manière uniforme, en masquant les détailles d'hétérogénéité et de localisation **[2].**

10. Le traitement des requêtes

Le traitement des requêtes suppose le passage du schéma global aux schémas des sources locales, autrement dit **[4]** :

- La transformation des requêtes exprimées à partir du schéma global vers des requêtes destinées aux sources locales.

- La transformation et le recoupement des résultats retournés par les sources locales en vue de les présenter μa l'aide du schéma global.

Le traitement des requêtes est organisé en cinq étapes :

1. L'analyse sémantique et structurelle de la requête.
2. La réécriture de la requête. Cette étape consiste à décomposer la requête en une ou plusieurs sous-requêtes sur les schémas des sources locales et une requête de recomposition. La réécriture est réalisée grâce aux assertions établies dans le schéma global pour définir les correspondances entre les schémas des sources locales et le schéma global.
3. La décomposition. Cette étape a pour but d'optimiser le traitement de la requête par l'affectation des sous-requêtes aux différentes sources locales en tenant compte de leurs capacités d'exécution.
4. La planification. L'objectif de cette étape est de définir l'ordre d'exécution des sous-requêtes afin d'obtenir les résultats attendus par le client, certaines sous-requêtes pouvant dépendre du résultat d'autres sous-requêtes.
5. L'exécution. Les sous-requêtes sont envoyées aux sources locales suivant le plan d'exécution déterminé à l'étape précédente. Le médiateur recompose les résultats à l'aide de la requête de recomposition déterminée dans la seconde étape, avant de retourner au client un résultat global.

11. Les ontologies dans les systèmes de médiation

Une définition possible de l'ontologie peut être énoncée comme suit **[5]** :

« Une ontologie est une spécification explicite d'une conceptualisation ».

En ce qui concerne l'intégration des sources de données, les ontologies peuvent être utilisées pour l'identification des correspondances entre les objets sémantiquement liés.

Nous pouvons distinguer trois approches d'utilisation des ontologies.

- Approches mono-ontologie (mono-domaine)

Utilisent une seule ontologie globale fournissant un vocabulaire commun pour la spécification de la sémantique. Toutes les sources de données sont reliées à cette ontologie, ce qui permet

d'identifier les règles de correspondances entre les objets des différentes sources de données. SIMS est un exemple de système qui implémente cette approche. Il utilise une ontologie globale pour représenter le domaine d'application **[2].**

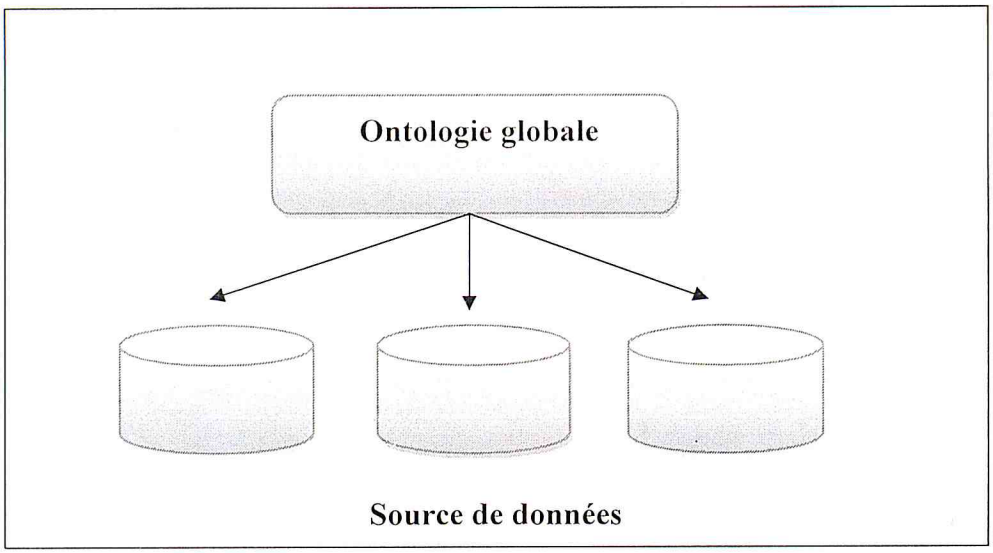


Figure 5 : Approche mono-ontologie

-Approches multi-ontologies (multi-domaines)

Dans cette approche, chaque source de données est décrite par une ontologie locale pour exprimer le contexte d'utilisation de ses informations. Les principaux avantages de cette approche sont l'autonomie dans la construction de l'ontologie du site local ainsi que l'extensibilité de la solution face à l'ajout et le retrait des SI. Par contre vu l'absence d'un vocabulaire commun les ontologies locale construites de manière autonome et indépendante peuvent présenter beaucoup de divergences dans la représentation de la sémantique. Ceci compliquera l'identification des correspondances entre les concepts des ontologies locales et donc l'identification des correspondances entre les objets des différentes sources de données sera aussi difficile. Pour résoudre ce problème, cette approche prévoit généralement des relations inter-ontologies (mappings) pour définir les correspondances entre les concepts des ontologies locales. Dans le cas où les SI ont des vues très différentes sur le domaine d'application et sachant que les ontologies locales correspondantes ont été construites sans vocabulaire commun, la construction des relations inter-ontologies (mappings) revient à résoudre les problèmes d'hétérogénéité sémantique entre les différentes ontologies **[2].**

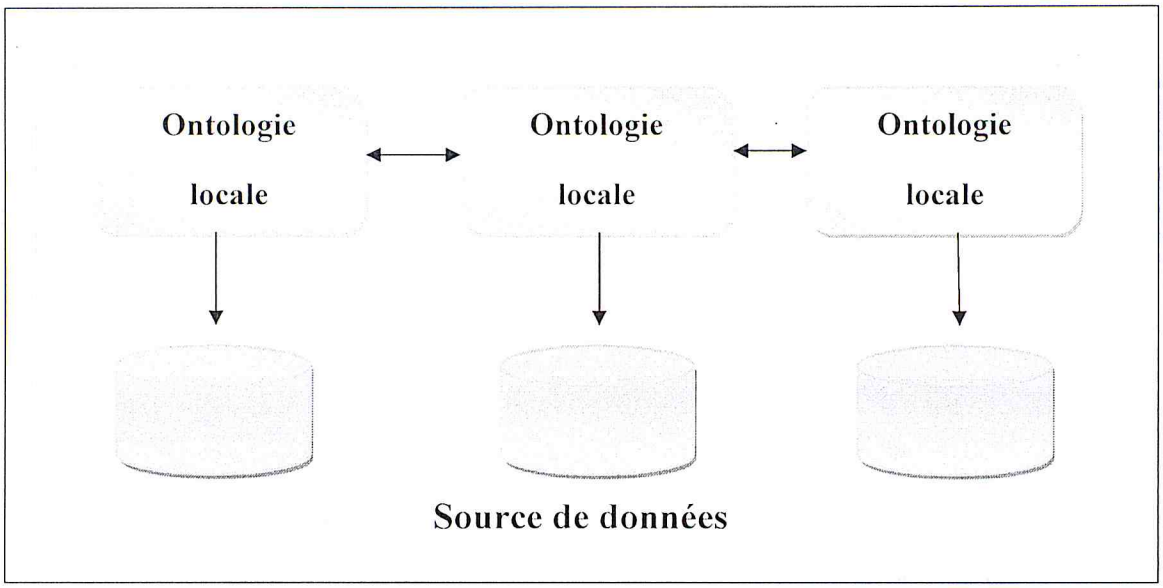


Figure 6 : Approche multi-ontologies

- Approches hybrides

Pour répondre aux inconvénients des approches mono-ontologie et multi-ontologies, des approches hybrides ont été développés. Semblables aux approches multi-ontologies,(la sémantique de chaque source de données est décrite par une ontologie locale), les approches hybrides consistent à rendre les ontologies locales facilement comparables et ce en les construisant sur un vocabulaire global commun. Le vocabulaire commun contient les limites de base (les primitifs) d'un domaine. Il peut être lui aussi une ontologie **[2].**

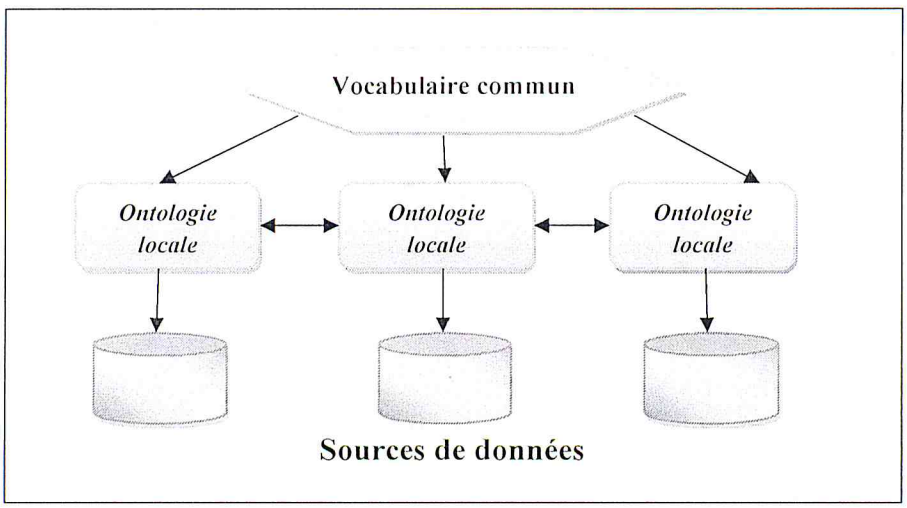


Figure 7 : Approche hybride

11. Conclusion

Les systèmes d'intégration sont basés sur le mécanisme de vues pour présenter une vue unifiée de données provenant de sources hétérogènes. Cette vue unifiée est appelée schéma médiateur du système.

Il existe principalement deux approches pour construire le schéma médiateur d'un système d'intégration. L'approche GAV et l'approche LAV.

Lorsque les vues sont virtuelles, on parle de système médiateur, ces vues sont extraites de différentes sources de données (base de données, fichier texte, page web ...etc.), dans notre cas on a 3 modèles de sources hétérogènes.

Chapitre II:

Etat de l'art sur la génération des requêtes de médiations

1. Introduction

Le système de médiation est un outil très intéressant dans le domaine d'intégration de données hétérogènes mais leur mise en place pose beaucoup de problèmes. Le problème majeur est la définition des requêtes de médiation en tenant compte l'hétérogénéité des sources qui doivent être interrogé à partir du schéma global.

Parmi les difficultés qu'on peut rencontrer lors de la génération des requêtes est l'identification des sources pertinentes pour répondre à la requête. Il faut aussi résoudre tous types de conflit en utilisant les correspondances (mapping) entre le schéma de médiation et les schémas des sources dont chacun possède une structure et une description sémantique différente.

La génération de requêtes consiste à générer des requêtes de médiation qui calculent une instance du schéma global à partir d'un ensemble de sources distribuées et hétérogènes. Très peu d'approches se focalisent sur l'interrogation des sources (bases de données réparties, bases de données fédérées,...etc.), et à la génération des requêtes de médiation qui traduisent les mappings entre le schéma global et les sources de données. Cette section présente ces approches :

2. Le projet MEDIAGRID

MEDIAGRID est un projet pluridisciplinaire réalisé en 2002 de l'ACI GRID (Actions Concertées Incitative - Globalisation des Ressources Informatiques et des Données).Dont l'objectif est de contribuer à la définition d'un canevas ouvert de système de médiation pour l'accès transparent aux sources largement distribuées. Il met l'accent sur la gestion de métadonnées, la génération de requête de médiation, et l'évaluation adaptative et interactive de requêtes d'utilisateur. Dans ce qui suit nous s'intéresserons qu'à la gestion de métadonnées et la génération de requêtes de médiation, on ne s'est pas donné une vision concernant l'évaluation des requêtes adaptative et interactive car ce n'est pas le but de notre projet **[6].**

2.1. Objectif du projet

Le terme grille de données (data grid) caractérise la globalisation de données réparties sur un ensemble non figé d'équipements et accessibles au plus grand nombre. Cette globalisation des données nécessite la mise en place d'une infrastructure de médiation entre les applications et de multiples sources de données préexistantes, autonomes et potentiellement hétérogènes **[6].**

L'objectif du projet MEDIAGRID est de spécifier et implanter des éléments d'un système de médiation tout en tenant compte des spécificités de la globalisation des données :

(i) Supporter plus de sources disponibles en considérant des sources contenant des donnés faiblement structurées.

(ii) Autoriser des résultats partiels pour des requêtes dans le cas des sources indisponibles et/ou pour satisfaire l'intérêt d'utilisateur.

(iii) Un générateur de requêtes de médiation.

(iv) Un évaluateur de requête adaptatif et interactif.

2.1.1. Architecture générale du projet

L'architecture générale d'un système de MediaGrid suit l'architecture de trois-couches classiques. Des utilisateurs ou des applications visitent des sources par le niveau de médiation. Des requêtes sont formulées en schéma de médiation (schéma global), et sont réécrite en schéma exporté. Les deux genres de schémas sont définis avec la syntaxe de XML. Le schéma de médiation décrit des données intégrées manipulées dans le niveau de médiation. Le schéma exporté décrit les sources abonnées au système. Pendant ce procédé, les sources sont adaptées et des descriptions de données sont traduites en schéma exporté. Et alors, les requêtes réécrites sont évaluées par un évaluateur **[6].**

Dans MEDIAGRID, les requêtes de médiation sont générées selon l'approche (Global-as-view) et stockées comme partie de métadonnées **[6].**

Les activités de recherche du projet se sont focalisées sur :

L'automatisation de la construction de requêtes de médiation. En effet, ces requêtes sont, dans la plupart des systèmes d'intégration de sources, construites manuellement car complexes à définir compte tenu de l'hétérogénéité des types de données sources et des liens sémantiques entre les données des niveaux global et local (dépendances fonctionnelles, contraintes de valeurs et référentielles, compatibilité de domaines, équivalence sémantique entre les attributs et les instances des attributs clés, etc). Le fait d'automatiser le processus d'intégration permet de faciliter la gestion de la forte dynamicité du système (ajout, retrait de sources, mise à jour des schémas des sources).

La figure 8 illustre l'architecture générale du système MEDIAGRID.

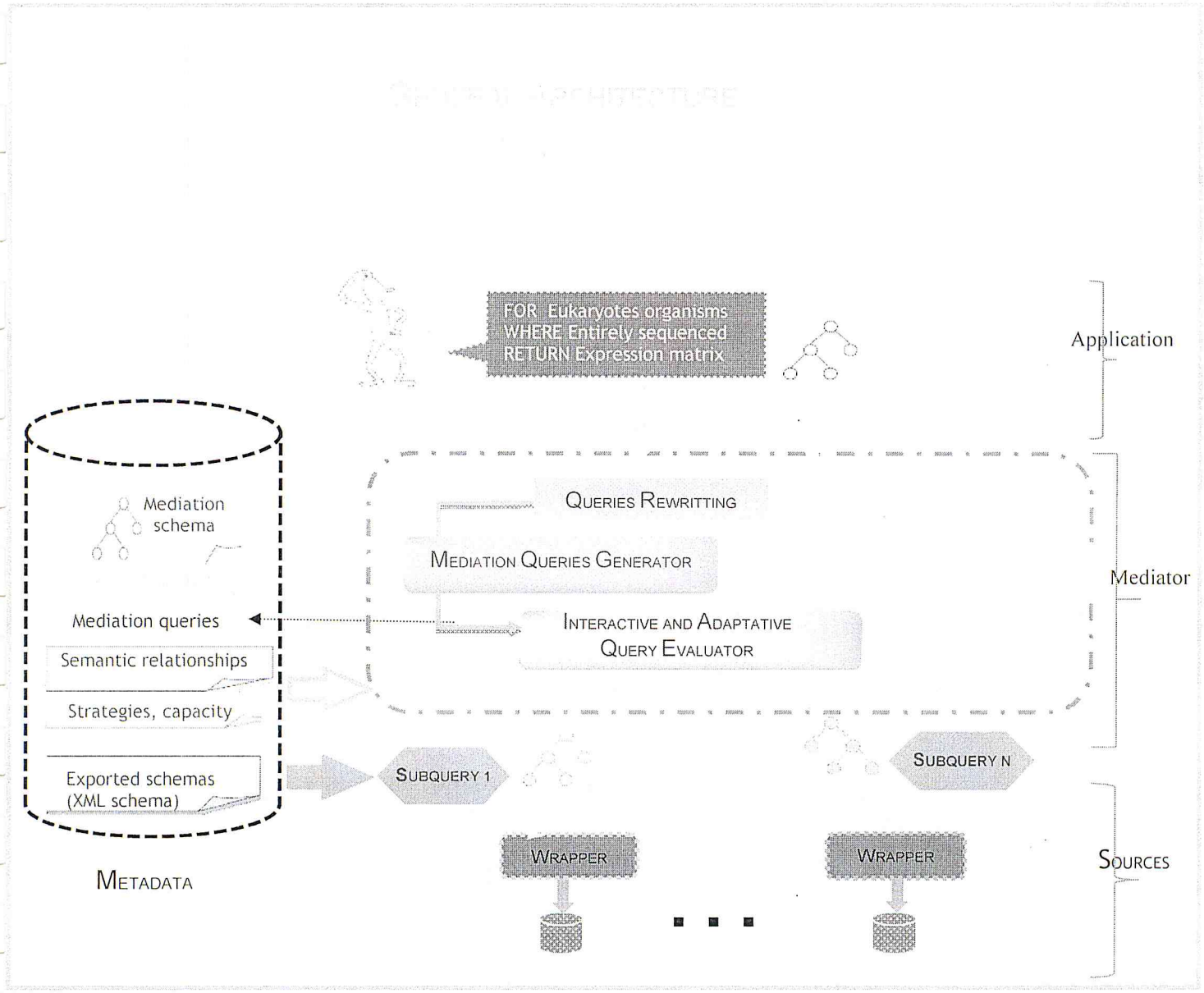


Figure 8:Architecture générale de système MEDIAGRID **[7].**

2.1.2. Démarche

La démarche de recherche a consisté à réaliser, dans un premier temps, un état de l'art sur les infrastructures de médiation en se focalisant sur les deux principaux aspects du projet : l'intégration de données et le traitement de requêtes ; Ils ont également identifié les spécificités de ces fonctions dans le cadre des applications du domaine de la biologie. C'est dans ce cadre que ils se sont focalisé sur l'intégration des données XML et l'évaluation de requêtes sur des sources XML. « Les systèmes de médiation XML sont encore peu nombreux aujourd'hui (en 2002) » **[6].**

2.1.3. Solution de MEDIAGRID

1. Gestion des métadonnées

- Selon l'approche MEDIAGRID, les métadonnées utilisées pour la génération de requêtes de médiation et l'évaluation de ces requêtes sont stockées dans une méta-base. Les métadonnées décrivent le schéma de médiation, les schémas exportés, les requêtes de médiation, les correspondances sémantiques, les capacités des sources ainsi que des statistiques sur les données (cardinalités, distribution, etc.).
- Les schémas exportés ainsi que le schéma de médiation sont représentés en utilisant XML schéma. Chaque schéma est un graphe comprenant un ensemble de nœuds.
- Les liens sémantiques entre les nœuds de chaque schéma exporté et ceux du schéma de médiation sont exprimés au moyen de correspondances sémantiques.
- Requêtes de médiation : Décrivent des mappings entre le schéma exporté et le schéma de médiation.
- Correspondances sémantiques : Des nœuds de schéma exporté sont sémantiquement reliés avec des nœuds de schéma de médiation par des correspondances sémantiques.
- Capacités des sources : Pour éviter un flux énorme transféré sur le réseau.
- Statistiques : Des statistiques font un rôle très important dans l'évaluation de requêtes. Elles peuvent être obtenues des sources quand elles se sont abonnées à un système de médiation, ou elles peuvent être obtenues pendant l'exécution.

2. Génération des requêtes

En raison de la nature semi-structurée des schémas dans le système MADIAGRID, il était difficile de définir les requêtes de médiation pour l'ensemble du schéma de médiation en une seule étape. L'idée de leur approche est **[8]:**

- De décomposer le schéma de médiation en un ensemble de sous arbres appelés sous arbres cibles.
- De trouver les requêtes permettant de dériver les instances de chaque sous arbre à partir des schémas exportés, appelées mappings partiels,
- Et enfin de combiner les mappings partiels pour produire les requêtes de médiation.

2.1.Préliminaires

L'approche MEDIAGRID a considéré une version simplifiée de XML Schéma. Chaque schéma est représenté par un arbre. La figure 10 montre un exemple de deux schémas exportés et un schéma de médiation. Dans la suite, pour éviter les confusions, chaque nœud sera suffixé par le nom du schéma auquel il appartient. Ainsi, (AuthorIds1) désignera le nœud (AuthorId dans le schéma exporté S1) et (Authors2) désignera le nœud Author dans le schéma exporté S2. Chaque nœud dans l'arbre est soit un nœud texte (AuthorIds1), soit un nœud interne (Authors1) .Les feuilles de l'arbre sont toujours des nœuds texte **[8].**

La cardinalité de chaque nœud est décrite par les attributs minOccur et maxOccur, représentant respectivement le nombre minimum et maximum d'instances de ce nœud dans l'arbre pour chacune des occurrences de son nœud père **[8].** Chaque nœud est :

monovalué (maxOccurs = 1) ou multivalué (maxOccurs >1).

Il peut également être optionnel (minOccurs = 0) ou obligatoire (minOccurs > 0).

Dans la figure 9:

- Le symbole '+' représente un nœud multivalué et obligatoire (Books2).
- Le symbole '*' représente un nœud multivalué et optionnel (Bookts).
- Le symbole '?' représente un nœud monovalué et optionnel.
- L'absence de symbole représente un nœud monovalué et obligatoire (Ids1).

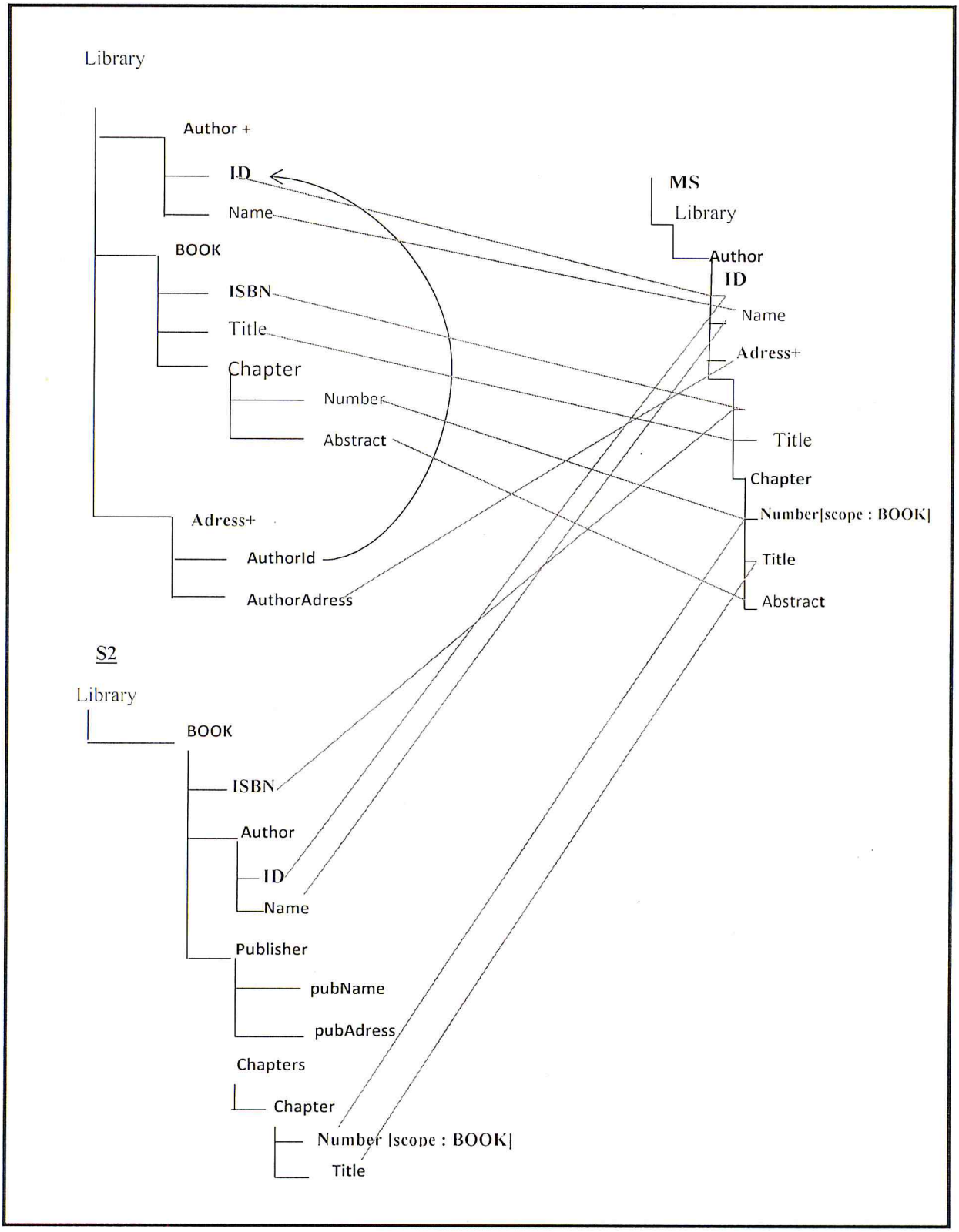


Figure 9:Schémas et correspondances sémantiques ***[8].***

Les clés sont définies soit dans tout le schéma soit dans un sous arbre. Dans le premier cas, la clé est dite *absolue*. Dans le second cas, la clé définie pour un nœud ***n*** est dite *relative* et sa portée est un nœud ascendant de *n* et différent de la racine de l'arbre. Dans la figure 9, les clés sont représentées en gras. Si le nom de la clé est suivi par des crochets, cela signifie que la clé est optionnelle et que sa portée est le nœud figurant entre crochets ; par exemple, *Numbers2* est une clé relative dont la portée est *Books2*. Dans le cas contraire, la clé est absolue, comme *ISBNs1*. Un schéma peut contenir des références, chacune étant un ensemble de nœuds référençant un autre ensemble de nœuds définis comme clé. Dans la figure 9, les références sont représentées par des flèches : ainsi, *AuthorIds1* est une référence sur *Ids1* **[8].**

Dans XML Schéma, la cardinalité d'un nœud est donnée par rapport à son nœud père ; La définition de cardinalité à une paire quelconque de nœuds dans un schéma est généralisée comme suite : étant donnés deux nœuds *n* et *n'* dans un schéma, *n* est multivalué (respectivement monovalué) par rapport à *n'* s'il peut exister plusieurs instances (respectivement une seule) de *n* pour chaque instance de *n'*. Cette cardinalité peut être déduite des cardinalités associées aux arcs composant le chemin de *n* à *n'*. Par exemple, dans *S1, IdS1* est multivalué par rapport à *ISBNS1* et *ISBNS1* est monovalué par rapport à *NumberS1* **[8].**

2.2.Correspondance sémantique

Un ensemble de correspondances est supposé comme étant fourni entre chaque schéma exporté et le schéma de médiation. Chaque correspondance relie un nœud *n* du schéma exporté à un nœud *n'* du schéma de médiation et signifie que ces deux nœuds représentent le même concept ; cette correspondance est notée $n \cong n'$; dans notre exemple, on a $Ids1 \cong Idts$. Dans la figure 9, les lignes rouges représentent des correspondances.

Des correspondances sont également considérées entre ensembles de nœuds. Etant donnés deux ensembles de nœuds *s1 et s2*, il y a une correspondance entre *s1 et s2* si

(i) *s1 et s2* ont le même nombre de nœuds.

(ii) Pour chaque nœud *n1* dans *s1*, il y a un nœud *n2* dans *s2* tel que $n1 \cong n2$. La correspondance entre les deux ensembles *s1 et s2* est notée $s1 \cong s2$ *(par exemple,* $\{Ids1, Names1\} \cong \{Idts, Namets\}$*).*

2.3.Détermination des sous -arbres cibles

Etant donné un schéma de médiation, chaque sous arbre est tel que :

• la racine *r* du sous-arbre *t* est soit un nœud multivalué soit la racine du schéma de médiation.

• tous les autres nœuds dans t sont des descendants monovalués de *r*.

• pour chaque paire de nœuds *n1 et n2* dans *t*, si un arc existe dans le schéma de médiation entre ces deux nœuds, alors cet arc est également dans *t*.

• il y a au moins un nœud texte dans *t*.

Dans le schéma de médiation de la figure 9, il y a trois sous arbres cibles, décrits dans la figure10 :

- le sous arbre *t1* composé du nœud multivalué *Authorts* et de ses trois fils monovalués *Idts, Namets* et *Addressts*.
- Le sous arbre *t2* composé du nœud *Bookts* et de ses deux fils monovalués *ISBNts* et *Titlets*.
- Le sous-arbre *t3* composé des nœuds *Chapterts, Numberts, Titlets* et *Abstractts*.
- Le nœud *Libraryts* n'est dans aucun sous arbre cible puisqu'il n'a aucun descendant monovalué qui soit un nœud texte.

2.4.Détermination des mappings partiels

Lorsque les sous-arbres cibles sont déterminés, les mappings partiels pour chacun des sous-arbres sont recherchés. Chaque mapping partiel représente une façon de dériver les instances d'un sous-arbre cible à partir des instances des schémas exportés de façon à respecter les contraintes du schéma de médiation. Il peut y avoir plusieurs mappings partiels pour chaque sous-arbre cible. La détermination des mappings partiels consiste à :

- Identifier les parties des schémas exportés qui sont pertinentes pour le sous-arbre cible considéré, appelées parties pertinentes.
- Rechercher les opérations de jointures possibles pour combiner les parties pertinentes.
- Déterminer les mappings partiels à partir des parties pertinentes et des jointures possibles.

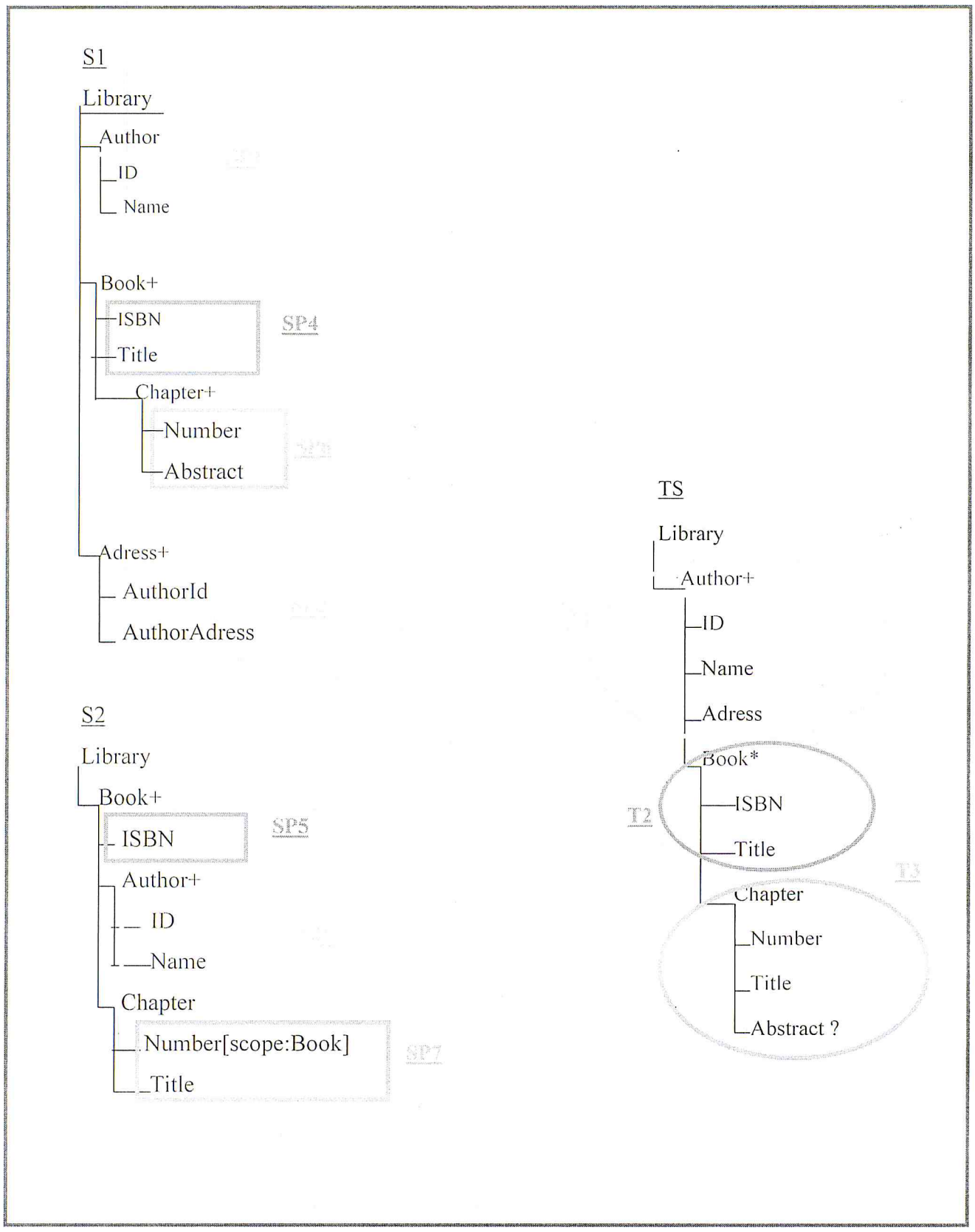


Figure 10 :Sous-arbres cibles et parties pertinentes ***[8].***

le sous arbre cible T3. Il possède deux parties pertinentes : une dans le schéma exporté *S1*, qui contient les deux nœuds *Numbers1* et *Abstracts1*, et l'autre dans le schéma exporté *S2* composé de

Numbers2 et *Titles2*. Une jointure est possible entre ces deux parties, avec le prédicat *Numbers1* = *Numbers2*. Il y a deux mappings partiels pour *t3* :

- Le premier pm1 est une projection des instances de *Numbers2* et *Titles2* pour dériver des instances pour *Numberts* et *Titlets* ; il ne produit pas d'instances pour *Abstractts*, ce qui n'est pas un problème puisqu'*Abstractts* est optionnel.
- Le second mapping partiel pm2 consiste à joindre les deux parties pertinentes précédentes.

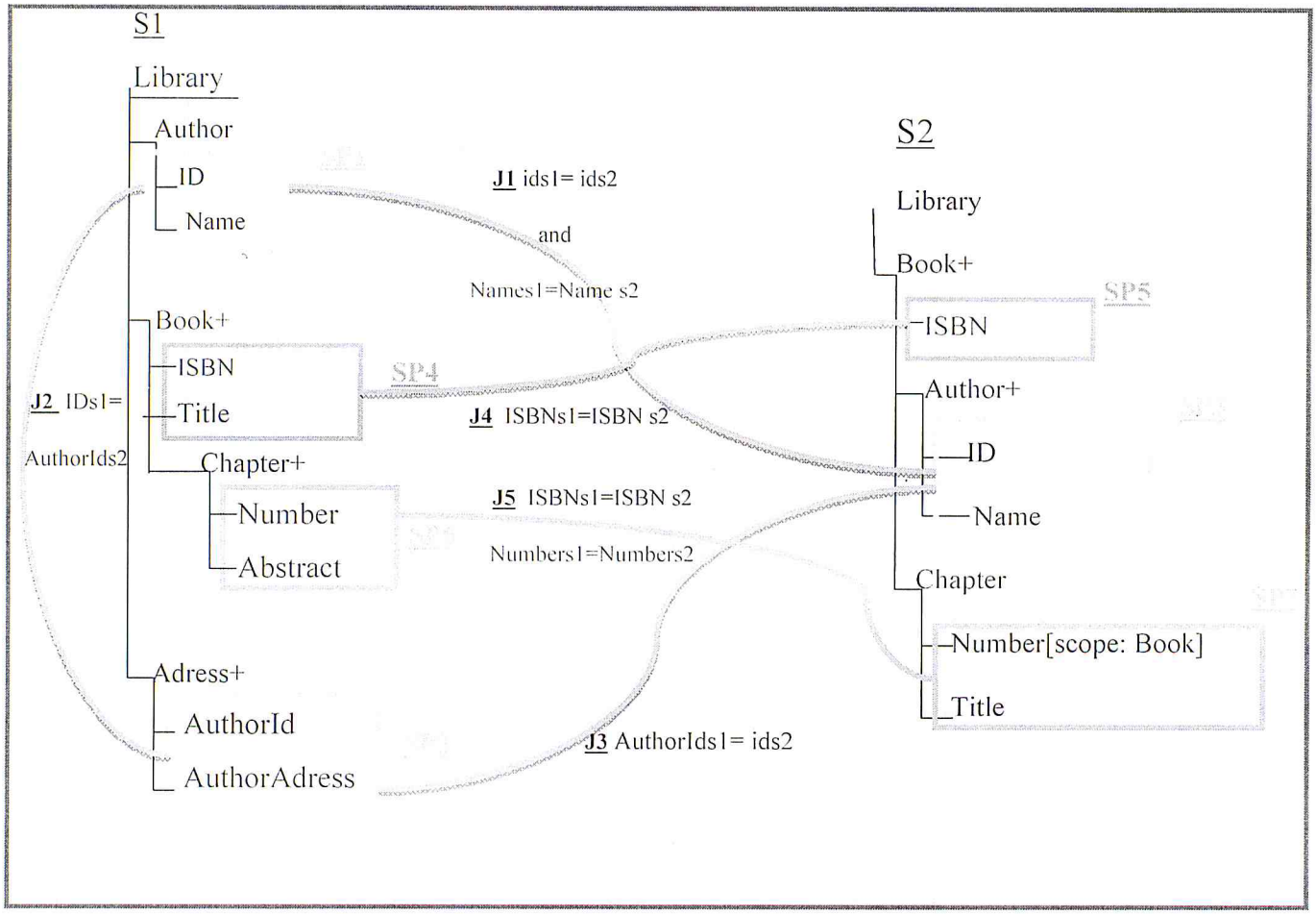


Figure 11:Jointures possibles entre parties pertinentes ***[8].***

Toutes les parties pertinentes dans notre exemple sont représentées en figure 10. Toutes les jointures possibles entre elles sont représentées en figure 11, où chaque jointure est représentée par un arc numéroté entre les deux nœuds représentant les parties pertinentes concernées par la jointure ; chaque arc porte un label qui est le prédicat de la jointure correspondante

2.5. Génération de requêtes de médiation.

La génération de requêtes pour l'ensemble du schéma de médiation se fait en deux étapes : des requêtes candidates sont d'abord produites en combinant les mappings partiels des différents sous-

arbres cibles ; puis pour chaque combinaison, on vérifie si les relations père/fils entre sous-arbres cibles sont respectées par la requête candidate ; si c'est le cas, une requête de médiation est générée.
Une requête candidate pour un schéma de médiation est un ensemble de mappings partiels tel qu'il existe au plus un mapping partiel pour chaque sous-arbre cible ; il peut se produire qu'aucun mapping partiel ne corresponde à un sous-arbre si celui-ci est optionnel.
Considérons les mappings partiels suivants : pm3 = {sp1} et pm4 = {j3} pour le sous-arbre cible *t1* ; pm5 = {j4} pour le sous-arbre cible *t2* et pm2 = {j5} pour le sous-arbre cible *t3*. Puisque *t2* est optionnel, que *t3* est obligatoire, et comme *t3* est fils de *t2*, alors une requête candidate peut ne comporter aucun mapping partiel pour *t2* et *t3* (comme les requêtes candidates Q1 = {pm3} et Q2 = {pm4}), ou contenir un mapping partiel pour chacun des sous-arbres *t2* et *t3* (comme les requêtes candidates m3 = {pm3, pm5, pm2} et m4 = {pm4, pm5, pm2}).
Pour chaque requête candidate, on vérifie que les liens père/fils entre les sous-arbres cibles sont vérifiés.
Soient un sous-arbre cible *t* dans un schéma de médiation, son sous-arbre père *t'* et leurs mappings partiels respectifs pm et pm' ; ces deux mappings partiels préservent la relation entre
t et t' si les conditions suivantes sont vérifiées :

- Il y a une partie pertinente sp dans pm et une partie pertinente sp' dans pm' telles que sp et sp' soient dans le même schéma exporté.
- Il existe un nœud n dans sp qui soit monovalué par rapport à tous les autres nœuds dans sp', ou un nœud n' dans sp' qui soit monovalué par rapport à tous les autres nœuds dans sp.

Considérons la requête candidate Q4 = {pm4, pm5, pm2}, la relation père/fils entre *t1* et *t2* est satisfaite car *ISBNs2* dans sp5 (qui figure dans pm5) est monovalué par rapport à *Ids2* et *Names2* dans sp3 (qui figure dans pm4).
La relation père/fils entre t2 est t3 est également satisfaite car *ISBNs2* dans sp5 (qui figure dans pm5) est monovalué par rapport à *Numbers2* et *Titles2* dans sp7 (qui figure dans pm2).
Une requête de médiation est générée pour chaque requête candidate qui préserve les liens entre sous-arbres cibles. Dans notre exemple, les requêtes de médiation sont Q1, Q2 et Q4.
De nouvelles requêtes peuvent être produites en appliquant des opérations ensemblistes (union, intersection, différence) à deux requêtes ou plus (exemple Q1 U Q4).
Chaque requête de médiation peut être traduite dans une expression des langages XQuery ou XSLT.
Pour traduire la requête en XQuery, chacun de ses mappings partiels est transformé en une expression FWR (For-Where-Return). L'expression FWR correspondant à un sous-arbre cible t, fils du sous-arbre cible t', est imbriquée dans l'expression FWR de t'. La relation entre t et t' est

représentée dans l'expression FWR de t. Une opération de groupement est ajoutée pour chaque clé dans le schéma de médiation.

2.1.4. Présentation du prototype pour la génération de requêtes de médiation

Un prototype pour la génération automatique de requêtes a été implémenté ; la figure 12 représente les différents modules du prototype. L'identification de schéma pertinent a pour but d'isoler dans un schéma exporté la partie pertinente par rapport au schéma de médiation. Cela permet de manipuler des descriptions de sources plus restreintes et de ne pas considérer les nœuds qui ne peuvent contribuer à la génération de requêtes. Le module de décomposition de schéma prend en entrée un schéma de médiation et fournit un ensemble de sous-arbres cibles. Chaque sous-arbre cible constitue l'entrée du module de détermination des mappings partiels, avec l'ensemble des schémas pertinents et les correspondances sémantiques ; la sortie de ce module est l'ensemble des mappings partiels existants pour le sous-arbre cible considéré, chacun représentant une façon de dériver les instances de ce sous-arbre à partir des instances des schémas pertinents en respectant les contraintes définies sur le schéma de médiation. A partir des mappings partiels correspondants aux différents sous-arbres cibles, le module de génération de requêtes produit des requêtes de médiation en considérant toutes les combinaisons possibles de mappings partiels. Chaque requête produite doit satisfaire les contraintes de cardinalités posées sur le schéma de médiation ainsi que les liens père/fils entre les sous-arbres cibles. Ce module produit un ensemble de requêtes ayant chacune une sémantique propre. Ces requêtes sont des requêtes abstraites exprimées dans un langage interne indépendant d'un langage de requêtes particulier. Elles peuvent être ensuite traduites à l'aide des modules de traductions XQuery ou XSLT **[6].**

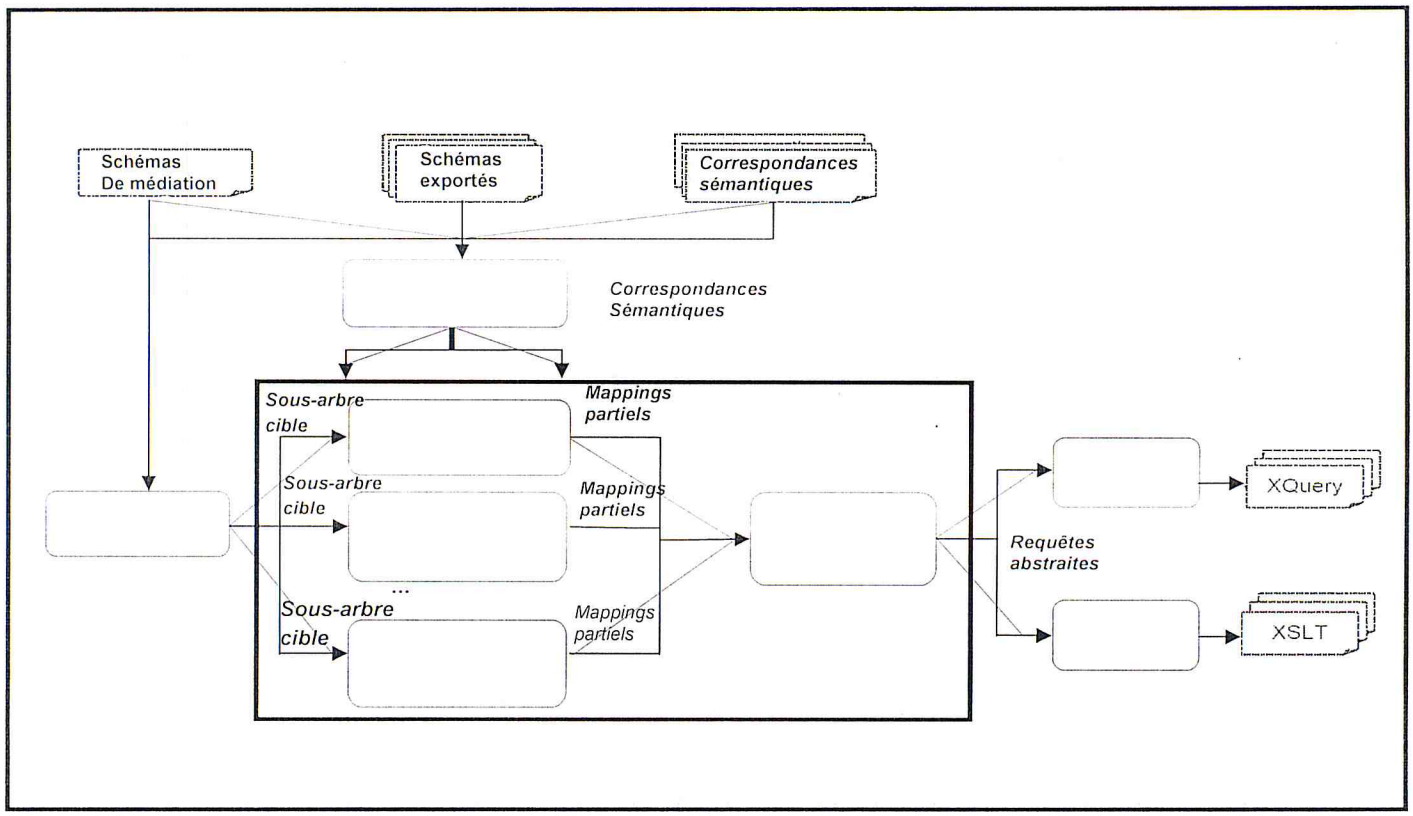


Figure 12 : Présentation du prototype pour la génération de requêtes de médiation ***[8].***

2.1.5. Conclusion et critiques

Cette partie présente le projet de MediaGrid dont l'objectif est de contribuer à la définition d'un canevas ouvert de système de médiation pour l'accès transparent aux sources largement distribuées. Il met l'accent sur la gestion de métadonnées, la génération de requête de médiation. XML est utilisé dans le niveau de médiation et XQuery est utilisé pour formuler requêtes de médiation. Sur l'article du projet, les auteurs ont proposé une approche de générer automatiquement des requêtes de médiation (étant donné le schéma exporté et le schéma de médiation). L'implantation des instances du canevas du projet a eu besoin de construire le système de médiation spécifique provident un accès transparent aux sources biologiques GOLD (source XML), SMD et SGD (se sont des sources relationnelles) étaient considérées. Mais ce n'est pas suffisant de démontrer que ce projet peut atteindre son objectif de fournir un canevas ouvert de système de médiation pour l'accès transparent aux sources hétérogènes et largement distribuées. En effet, cet article était publié en 2004 où ce projet n'avait pas été tout à fait fini. Il restait encore des problèmes comme la capacité de passer à grande échelle et l'évaluation de performance à valider. Jusqu'au présent, il n'y pas encore un article publié qui fournit une conclusion finale pour ce projet.

Dans le futur, des systèmes de médiation pourraient être extrêmement dynamiques: ils devront gérer l'évolution de sources de données et l'insertion et la suppression de sources. La disponibilité de sources devrait être considérée par le traitement de requête, où des requêtes ont peut-être besoin de dynamiquement changer leur plan d'exécution, pour produire des résultats partiels ou matérialiser des résultats. Finalement, des utilisateurs et des applications pourraient contrôler le traitement de requête.

3. Le projet CLIO

3.1. Description

Dans le cadre d'une collaboration entre le centre de recherche de la société « IBM Almaden » et l'université de «Toronto», un outil appelé CLIO est développé. Il permet de générer automatiquement des requêtes calculant une instance du schéma global à partir de sources de données structurées ou semi structurées. Cet outil a fait l'objet de plusieurs travaux de recherche, parmi ces travaux nous citons dans ce qui suit les deux articles les plus pertinents : **Miller [10]** proposent dans le contexte relationnel une approche de génération de requêtes de médiation qui permet de calculer automatiquement des requêtes SQL. La figure suivante résume leur approche :

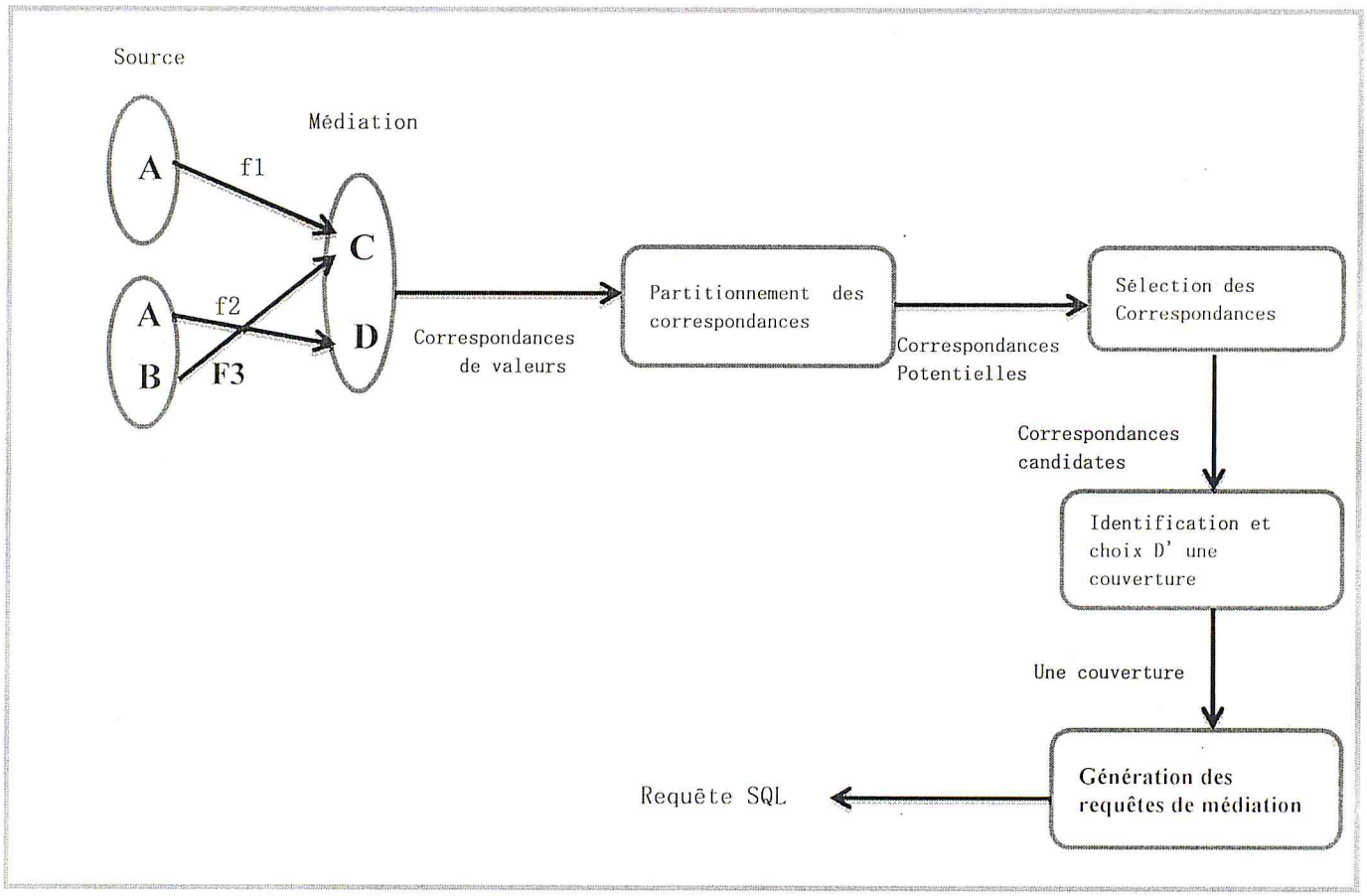


Figure 13 : Approche de génération de requêtes dans le contexte relationnel ***[9].***

Cette approche est basée sur des correspondances de valeurs définies entre les attributs sources et les attributs du schéma global. Par exemple dans la figure suivante l'attribut caller de la relation calls et l'attribut *artifact* dans la relation du schéma global *references* sont liés par la correspondance de valeur *f2*.

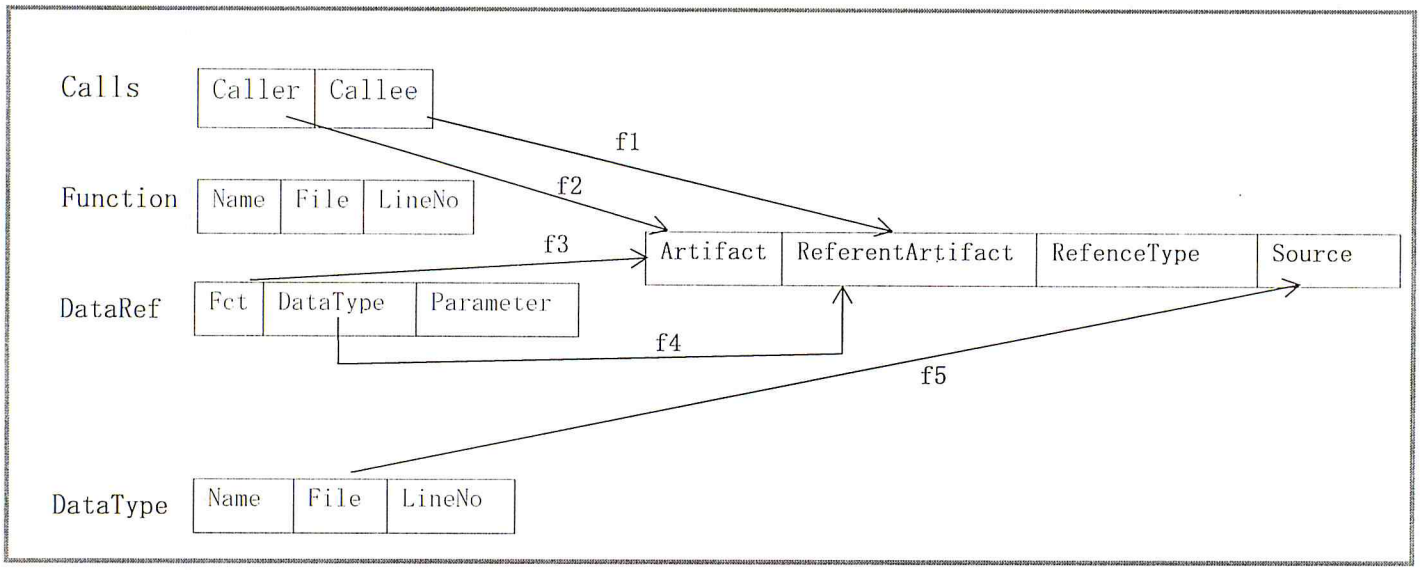


Figure 14 : Utilisation des correspondances de valeurs pour lier deux attributs ***[9].***

Les relations sources et la relation du schéma global représentent les nœuds du graphe, et les liens entre un attribut source et un attribut du schéma global représentent les arcs du graphe. La correspondance de valeur fi assignée à chaque arc est une correspondance de valeur qui permet de lier deux attributs. Elle est soit déterminée par une technique de matching particulière (d'apprentissage, statistiques, ontologie,...etc.), soit définie au préalable manuellement par le concepteur du système. Lorsque les attributs ont des valeurs, ces correspondances sont recherchées par des algorithmes basés sur des techniques de matching permettant de détecter les conflits sémantiques liés aux données. Par exemple si l'attribut A1 a pour valeur V1 et l'attribut A2 a pour valeur V2 une technique de matching est utilisée pour identifier un lien entre la valeur
V1 et la valeur V2. Dans le cas où aucune valeur n'est associée à un attribut ces liens sont recherchés en se référant à des algorithmes basés sur une ontologie ou sur des dictionnaires linguistiques qui permettent d'identifier les conflits sémantiques liés au schéma. L'approche de génération de requêtes de médiation proposée dans cet article compte essentiellement quatre phases :

1- partitionnement des correspondances : cette phase prend en entrée l'ensemble de correspondances de valeurs noté V, il partitionne cet ensemble et il génère un autre ensemble P= {C1, C2, Ci,..., Cn} où chaque ci contient une seule correspondance possible pour un attribut de la relation du schéma global. En d'autres termes si pour le même attribut du schéma global il existe plus d'une correspondance avec les attributs sources, une seule correspondance est prise en compte. Le résultat de cette phase est un ensemble de correspondances potentiel au calcul d'une relation du schéma global.

2- Sélection des correspondances : cette seconde phase prend en entrée l'ensemble de correspondances potentiel et cherche des combinaisons candidates entre les relations sources pour le calcul de la relation du schéma global. Elle est basée sur un algorithme qui utilise les clés Étrangères. Ces clés sont définies au préalable ou recherchés par un algorithme de data mining.
Le résultat de cette phase est un ensemble de correspondances candidates noté C Í P.

3- Identification et choix d'une couverture : cette troisième phase prend en entrée l'ensemble C de combinaisons candidates et cherche un sous ensemble de C dit couverture où toutes les correspondances de valeurs de l'ensemble V apparaissent une seule fois. Si plusieurs couvertures sont identifiées la couverture qui contient le moins de combinaisons candidates est choisie.

4- Génération de requêtes : cette dernière phase prend en entrée la couverture sélectionnée.
Pour chaque combinaison candidate une requête SQL est construite. Les requêtes produites par le système CLIO et qui permettent de calculer la relation references sont les suivantes :

```
Q1: Select C.Caller, C.Callee, relname (c), F.File
From Function F,left outer join Calls C
Where C.Caller = F.Name
Q2: Select C.Caller, C.Callee, relname (c), F.File
From Function F,left outer join Calls C
Where C.Callee = F.Name
```

Dans **[11]** est proposée une généralisation de l'approche de génération de requêtes dans le contexte XML. Elle permet de générer des requêtes mais à partir d'un seul schéma source et dans un format particulier ad-hoc. Etant donné un schéma global et un schéma source.ils définissent des requêtes en utilisant des contraintes référentielles. Leur algorithme prend en entrée les correspondances de valeurs et produit en sortie des requêtes dites (mappings logiques).

3.2. Critiques

Clio permet le mapping depuis et vers des schémas relationnels ; et prend en considération les contraintes qui lient les attributs dans les schémas sources et schéma globale .Mais ce système fait le mapping que s'il dispose auparavant des correspondances entre les attributs des schémas sources et ceux du schéma globale qui sont déterminées manuellement ce qui le rend semi-automatique.

4. L'approche de NORA MAIZ

La thèse **[13]** a été consacrée pour la conception d'un système de médiation à base d'ontologie. Cette méthode est basée sur une approche sémantique permettant à la fois de résoudre l'hétérogénéité des données et d'améliorer la qualité des résultats de la recherche des données. Pour cela, ils ont utilisé les ontologies comme moyen de résolution des conflits structurels et sémantiques où chaque source de données est accompagnée de son ontologie locale.

4.1. L'architecture du système

L'ontologie globale est extraite à partir des concepts utilisés dans les ontologies locales c'est meilleur résolution des conflits.

1) la présentation des ontologies locales

Cette étapes consiste à construire une ontologie à partir des sources, c'est la représentation des sources qui ont été choisi comme pertinents dans un langage d'ontologie (par exemple : OWL).

2) la construction d'un vocabulaire partagé sous forme d'une ontologie globale

Cette étape consiste à construire l'ontologie globale en deux phases principales :

- La première étape implique une analyse complète des ontologies construites à partir des sources locales.
- La deuxième étape est une sélection des concepts en localisant les problèmes d'hétérogénéités sémantiques. Dans cette étape la détection des conflits sémantique liés aux données sont les plus probables.

3) la définition des différentes correspondances entre les sources, les ontologies locales et l'ontologie globale

Les correspondances entre les deux niveaux d'ontologies sont les règles de passage d'un niveau à un autre. Chacune de ces règles précise l'emplacement d'origine de l'entité ainsi que sa définition. Les règles de correspondances sont utiles dans la réécriture des requêtes. Elles permettent de réécrire la requête exprimée dans le langage de l'ontologie globale, en un ensemble de sous-requêtes exprimées dans les schémas locaux.

Pour garder la source locale de chaque concept dans le schéma globale, l'approche a utilisé le principe d'annotation. L'annotation est un type de propriété qui peut être employé pour ajouter des informations sur des classes. . L'annotation se fait selon un schéma de méta donnée prédéfini.

Par exemple :

Pour représenter par exemple que la classe "Professeur" a été créée par la personne "x" lors de la déclaration de la classe "Professeur", on ajoute l'annotation sur la classe :

< owl : Class rdf : about = "Professeur" >

< rdfs : subClassOf rdf : ressource = "Personne"/ >

< dc : creator xml : lang = "fr" > x < /dc : creator >

< /owl : Class >

4.2. Traitement des requêtes

L'utilisateur compose sa requête sous forme de conjonction de concepts et de propriétés du vocabulaire de l'ontologie globale. Ensuite, un mécanisme de réécriture de requêtes doit assurer la décomposition et la recomposition des résultats élémentaires de la requête décomposée.

4.3. La réécriture des requêtes

L'ontologie globale est un ensemble de concepts classés en une hiérarchie. Cette ontologie ne contient pas de propriétés reliant les différents concepts. Ces propriétés existent chacune dans leur ontologie locale. L'utilisateur peut donc créer une requête qui contient des concepts de l'ontologie globale et éventuellement des propriétés des ontologies locales. La requête exprimée en termes de concepts et propriétés doit être réécrite de manière à obtenir des résultats qu'on peut agréger. Tout concept qui ne garantit pas la combinaison des données obtenues est exclu. Du point de vue sémantique cette exclusion tend à rendre une requête cohérente. Une requête cohérente est une requête décomposable en sous-requêtes exécutables et dont les résultats sont composables. Cette réécriture peut être vue comme une correspondance entre l'ontologie globale et les ontologies locales, puisqu'elle permet de rendre le traitement de la requête utilisateur direct de l'ontologie globale vers les ontologies locales. Une requête utilisateur de base est sous la forme :

Concept ^ Propriété ^ Concept ou tout simplement *Concept*.

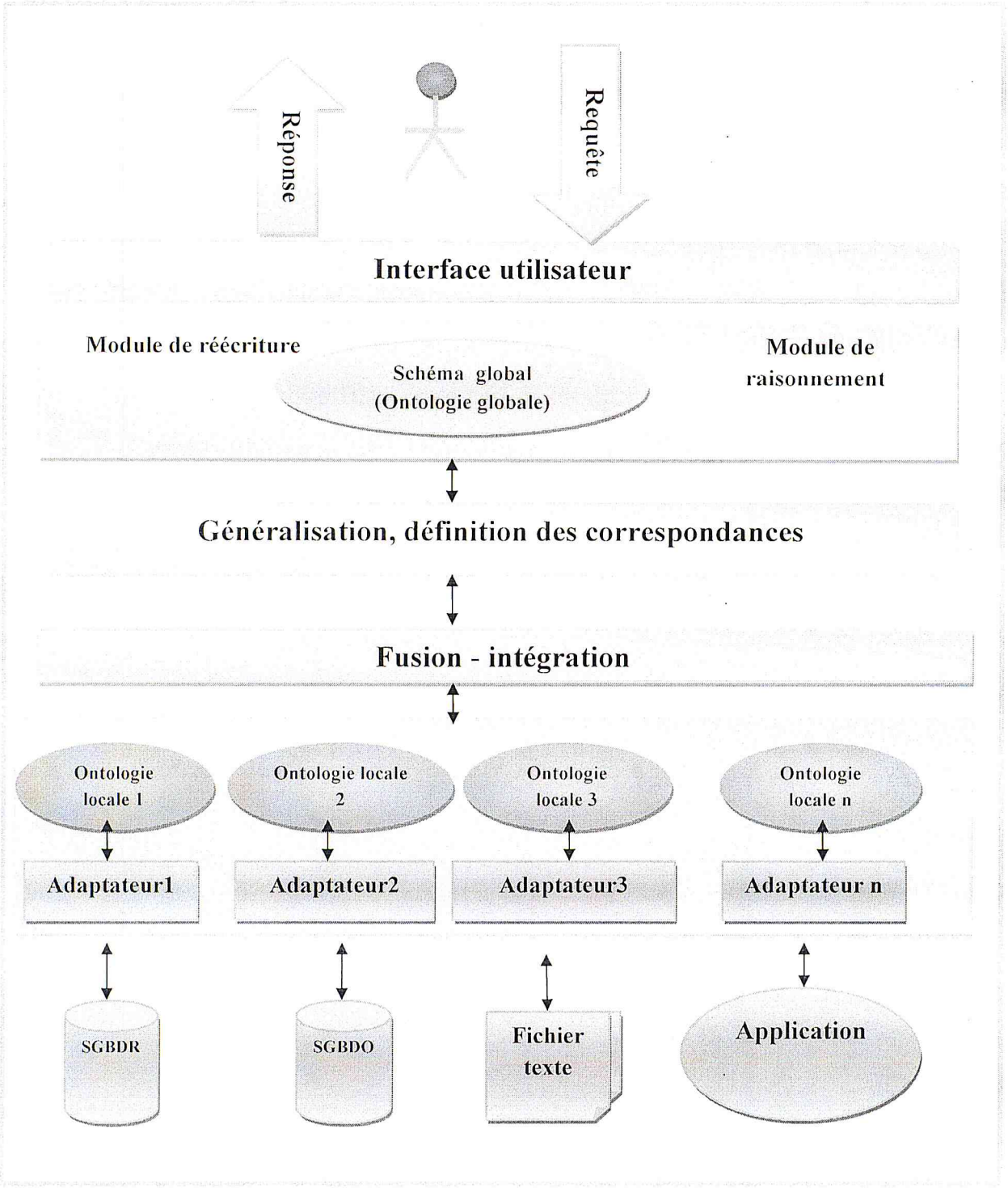


Figure 15 : Architecture d'un système de médiation à base d'ontologie ***[13].***

5. L'approche A.SOUKANE

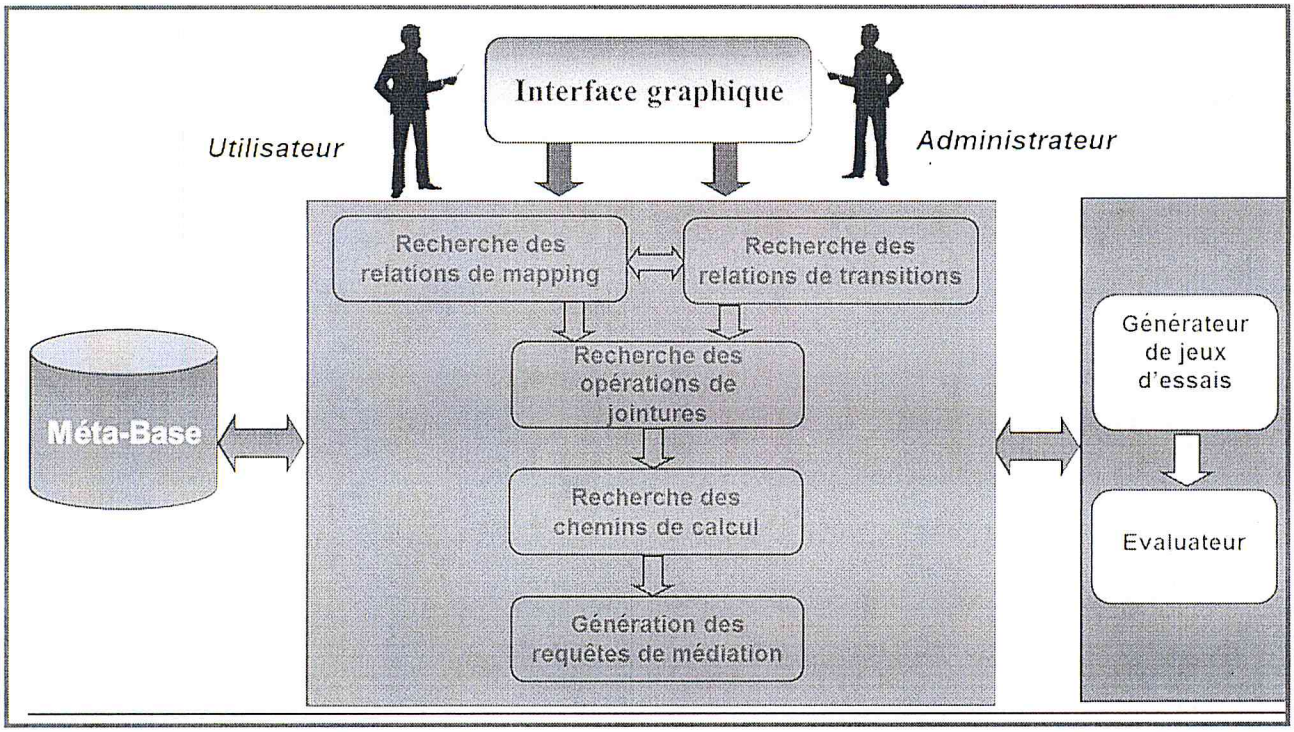


Figure 16: l'architecture générale du système.

La méthode qui a été proposée dans la thèse **[1]** est une méthode de génération de requêtes de médiation dans un contexte relationnel avec la prise en compte des conflits sémantiques liés aux données et aux schémas.

Les étapes suivantes résument le processus de génération automatique de requêtes de médiation dans cette méthode:

1) Identification des relations sources pertinentes pour la définition d'une requête de médiation du schéma de médiation (globale), et génération de relations de mapping qui sont obtenues par la projection des relations sources sur leurs attributs communs avec la relation de médiation.

2) Identification des opérations relationnelles possibles entre les relations de mapping en fonction de leur schéma et de leurs clés, et génération du graphe d'opérations.

3) Recherche des chemins de calcul à partir du graphe d'opérations pour calculer la relation de médiation.

4) Génération de requêtes de médiation déduites à partir des chemins de calcul de la relation de médiation.

Pour que le processus de génération des requêtes de médiation prenne en considération l'hétérogénéité des données, ils ont intégré des procédures, qui détectent et résolvent les conflits liés à l'hétérogénéité des données, dans chaque étape de l'algorithme de la génération des requêtes.

1) Dictionnaire linguistique

Le dictionnaire est une solution aux problèmes de synonymies et d'homonymies, il permet de détecter les correspondances linguistiques entre les attributs.

Dans cette thèse ils ont utilisé les méta-données pour décrire et donner plus d'information sur les sources, le schéma globale et les relations entre les sources et le schéma de médiation.

2) Type étendue d'un attribut

Le type étendu d'un attribut est un ensemble de métadonnées qui sert comme une description de sa valeur. En plus de son nom et de son type, le type étendu est un ensemble de donnée qui donne plus de signification à un attribut.

Le type étendu est défini comme un tableau associatif d'éléments à deux colonnes.

<nom-attribut> : [Type = type de base,

Element = {elem1, elem2,.....,elemn}]

La première ligne décrit le nom de l'attribut et la deuxième ligne décrit sa valeur.

3) Librairie de fonction

Une fonction de transformation transforme la valeur d'un élément du type étendu d'un attribut en une autre valeur.

Une fonction de transformation est décrite comme suit :

<nom-fonction> : [élément = < élément à traiter >

valeur_in = < valeur de l'élément en entrée >

valeur_out = <valeur en sortie de l'élément>]

Nous avons présenté dans cette section un état de l'art sur les techniques utilisées par les approches qui visent à interroger des sources distribuées et hétérogènes, et à générer automatiquement des requêtes de médiation. Certaines approches se focalisent sur l'identification des correspondances entre les attributs en utilisant plusieurs techniques telle que les ontologies , d'autres supposent que ces correspondances sont déjà définies ou appellent des techniques existantes pour les déterminer, et se focalisent sur la génération des requêtes et aussi sur la sémantique de ces dernières pour garantir un retour correct à l'utilisateur.

6. Etude comparative des différentes approches précédentes

Les deux tableaux suivant résument les similarités et les différences entre les approches déjà cité précédemment :

		MedaiGrid	*CLIO*	*A. Soukane*	*N. Maiz*
Système de Médiation		Oui	Non	Oui	Oui
Correspondance		1-n	1-n	1-1	1-n
Système mono-source		Non	Oui	Non	Non
Conflits syntaxiques		Oui	Non	Non	Oui
conflits sémantiques	***Schéma***	Oui	Oui	Oui	Oui
	Données	Oui	Non	Oui	Oui
Correspondances linguistiques	***Automatique***	Oui	Non	Oui	Non
	semi-automatique	Non	Oui	Non	Oui
Schéma Global		XMLs	relationnel	Relationnel	OWL
Schémas sources (pivot)		XMLs/REL	relationnel	relationnel	OWL

Tableau 1:Etude comparative des différentes approches précédentes.

Méthodes	Description
MediaGrid	**Problématique :** -Génération des requêtes de médiation. - Prise en compte de l'hétérogénéité des sources. **Approche :** ➢ -Décomposer le schéma de médiation en un ensemble de sous arbres cibles. ➢ -Trouver les requêtes permettant de dériver les instances de chaque sous arbre à partir des schémas exportés, appelées mappings partiels. ➢ -Combiner les mappings partiels pour produire les requêtes de médiation. **Points faibles :** - La non autorisation des résultats partiels pour des requêtes dans le cas des sources indisponibles. -Le système ne gère pas l'évolution des sources. **Point forts :** -prise en compte des correspondances 1-n. -prise en compte des conflits sémantiques liés aux données et au schéma. -prise en compte des conflits syntaxiques -les correspondances linguistiques sont définies automatiquement.
CLIO	**Problématique :** Génération de requêtes de médiation. **Approche :** 1-Partitionnement des correspondances. 2- Sélection des correspondances. 3- Identification et choix d'une couverture. 4- Génération des requêtes de médiation.

	Points faibles : ➢ Détection manuelle des correspondances linguistique. ➢ La non prise en compte des conflits sémantiques liés aux données. ➢ Prise en compte des conflits sémantiques liés au schéma. **Point forts :** -prise en compte des correspondances 1-n. -prise en compte des requêtes imbriquées.
A. SOUKANE	**Problématique :** -Définition des requêtes de médiation. - Prise en compte de l'hétérogénéité des sources. - Prise en compte des besoins des utilisateurs en termes de contraintes d'intégrité. **Approche :** o Recherche des relations de mapping. o Recherche des opérations candidates. o Recherche des chemins de calcul. o Génération des requêtes de médiation. **Point faibles :** - La non prise en compte des correspondances 1-n, seules les correspondances 1-1 sont considérées. - La non prise en compte des requêtes imbriquées. -non prise en compte des conflits syntaxiques. **Point forts :** -Détection automatique des correspondances linguistiques. -Prise en compte des conflits sémantiques liés aux données et au schéma

N.MAIZ	**Problématique :** -Définition des requêtes de médiation en utilisant les ontologies **Approche :** ➢ présentation des ontologies locales. ➢ construction du schéma global. ➢ définition des correspondances entre les sources. **Point forts :** -prise en comptes des conflits sémantiques et syntaxiques. -Traitement automatique des requêtes. **Point faibles :** -L'annotation des concepts des sources locales qui se fait manuellement.

Tableau 2: Description des différentes approches précédentes.

7. Conclusion

Dans ce chapitre nous avons fait une étude sur des différentes approches utilisées dans le domaine d'intégration des données et aussi la génération automatique des requêtes de médiation.

Nous avons présenté un tableau d'étude comparative des différentes méthodes citées précédemment pour bénéficier de leurs points forts et points faibles.

Généralement tous les systèmes de mapping que nous avons vu sont dépendants du modèle de représentation de schémas sources et schéma globale ; concernant notre approche nous devons concevoir un système de mapping qui est indépendant du modèle de représentation des schémas sources et schéma globale, nous nous basons pour cela sur le concept d'*Ontologie*.

Chapitre III:
Approche proposée

Chapitre III : Approche proposée

1. Introduction

Après avoir donné un aperçu général sur les différentes approches que nous avons étudies dans le chapitre précédent concernant la génération des requêtes de médiation ainsi que les conflits liés à l'hétérogénéité, une petite étude comparative des différentes approches a été présentée , nous allons maintenant présenter le processus générale qu'on a mis au point après constat des points forts et points faibles de chacune de ces dernières .

2. Proposition d'une nouvelle approche de génération de requêtes de médiation GRMOH :

Après avoir fait une étude comparative de plusieurs méthodes de génération de requêtes de médiation, nous avons constaté qu'il n'existe pas une méthode qui génère automatiquement des requêtes de médiation en résolvant tous les types de conflits qui existent. De plus généralement tous les systèmes de mapping que nous avons vu sont dépendants du modèle de représentation de schémas sources et schéma global.

Pour remédier à ce problème nous avons imaginé et mis au point le système GRMOH (Générateur de requête de médiation pour environnement hétérogène) qui se concentre sur deux architectures :

2.1. Génération des mappings :

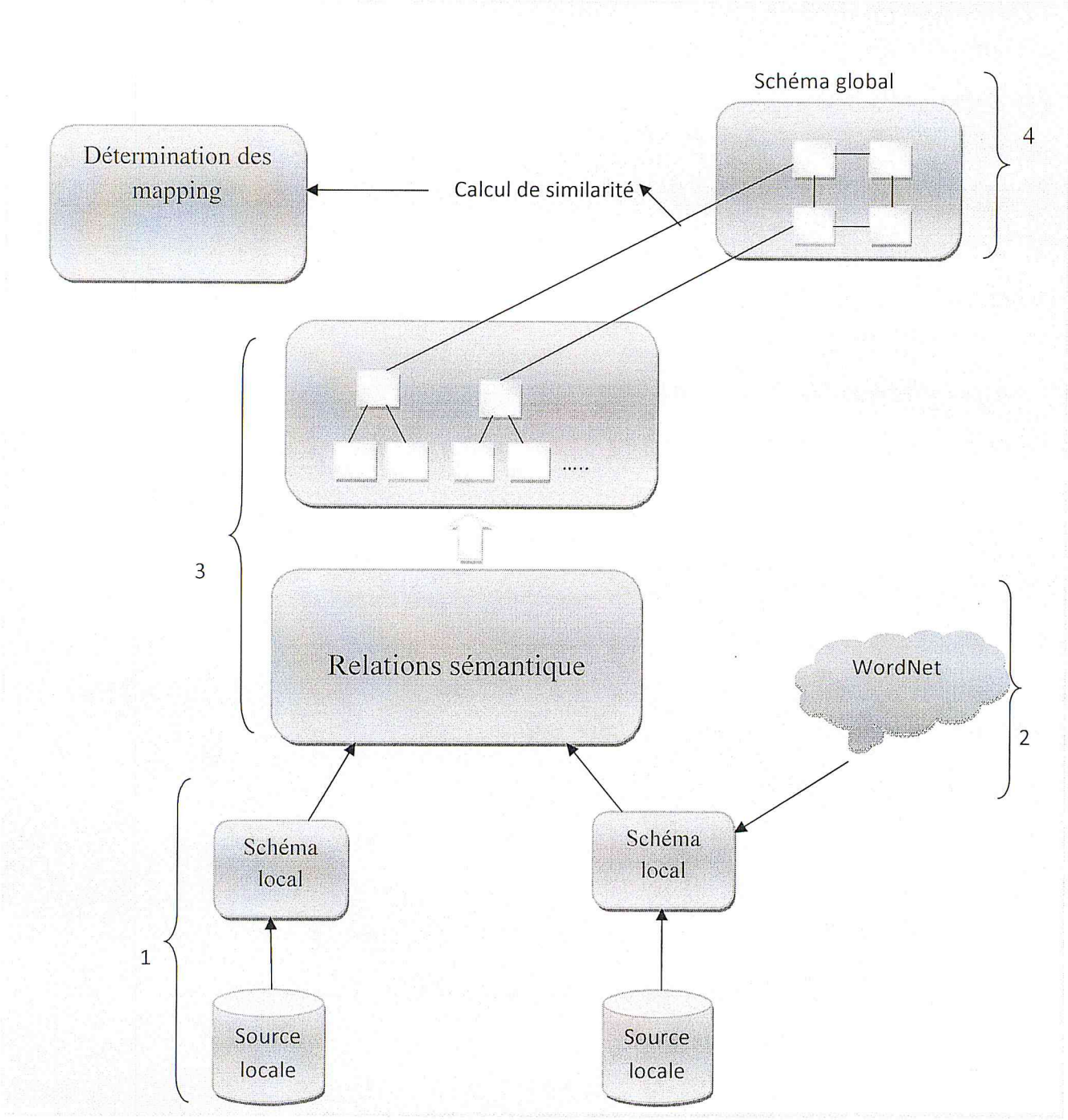


Figure 19 : processus de génération de la table de mapping .

1. Ce système peut générer automatiquement des requêtes de médiations à partir d'une requête globale (écrite en termes de schéma global) avec la prise en compte des conflits liés à la sémantique.
2. Des sous-requêtes écrites en langage de requête des sources mères. Dans notre cas, on s'intéresse aux : données relationnelles, métadonnées XML ou RDF, et aux connaissances OWL.
3. Le schéma global du système est exprimé en OWL (et qui est supposé comme étant déjà définit).
4. Chaque classe du schéma global est une classe globale qui regroupe un ensemble de classes, qui appartiennent aux schémas locaux, tel que ces classe ont des caractéristiques en commun (des synonymes ou bien ils contiennent entre eux des relations structurelles).

Le processus de notre système se décompose en 4 phases principales :

1. L'extraction d'un schéma OWL pour chaque source qui contient les classes, les sous classes et les attributs.
2. Annotation automatique : Cette deuxième étape consiste à faire associer à chaque classe des schémas sources une définition, dans ce cas nous allons utiliser un thesaurus qui regroupe un ensemble de termes avec leur définition.
3. Extraction des relations sémantiques : Á partir de la deuxième étape, où chaque source est associée à une définition, notre système dérive des relations sémantique inter et intra schéma c.-à-d. il met en correspondance les classes qui sont reliées par des relations structurelles ou lexicales. Une relation structurelle est une relation qui lie des classes qui contiennent entre eux une clé étrangère ou bien des propriétés de type (objectProperty), une relation lexicale dépend des définitions qui sont associées aux classes dans la deuxième étape (l'annotation).

   Ensuite, notre système regroupe toutes les classes qui contiennent des caractéristiques en commun, et pour chaque groupe, le système doit créer un représentant.
4. Après avoir construit les groupes, notre système est censé à calculer une mesure de similarité entre les représentants de chaque groupe et les classes du schéma global.

   Une fois les mesures sont calculées, notre système peut calculer les mapping entre le schéma global et les schémas locaux en mettant en correspondance les classes dont la

mesure de similarité et supérieure ou égale à un certain niveau.la figure suivante décrit les étapes du processus :

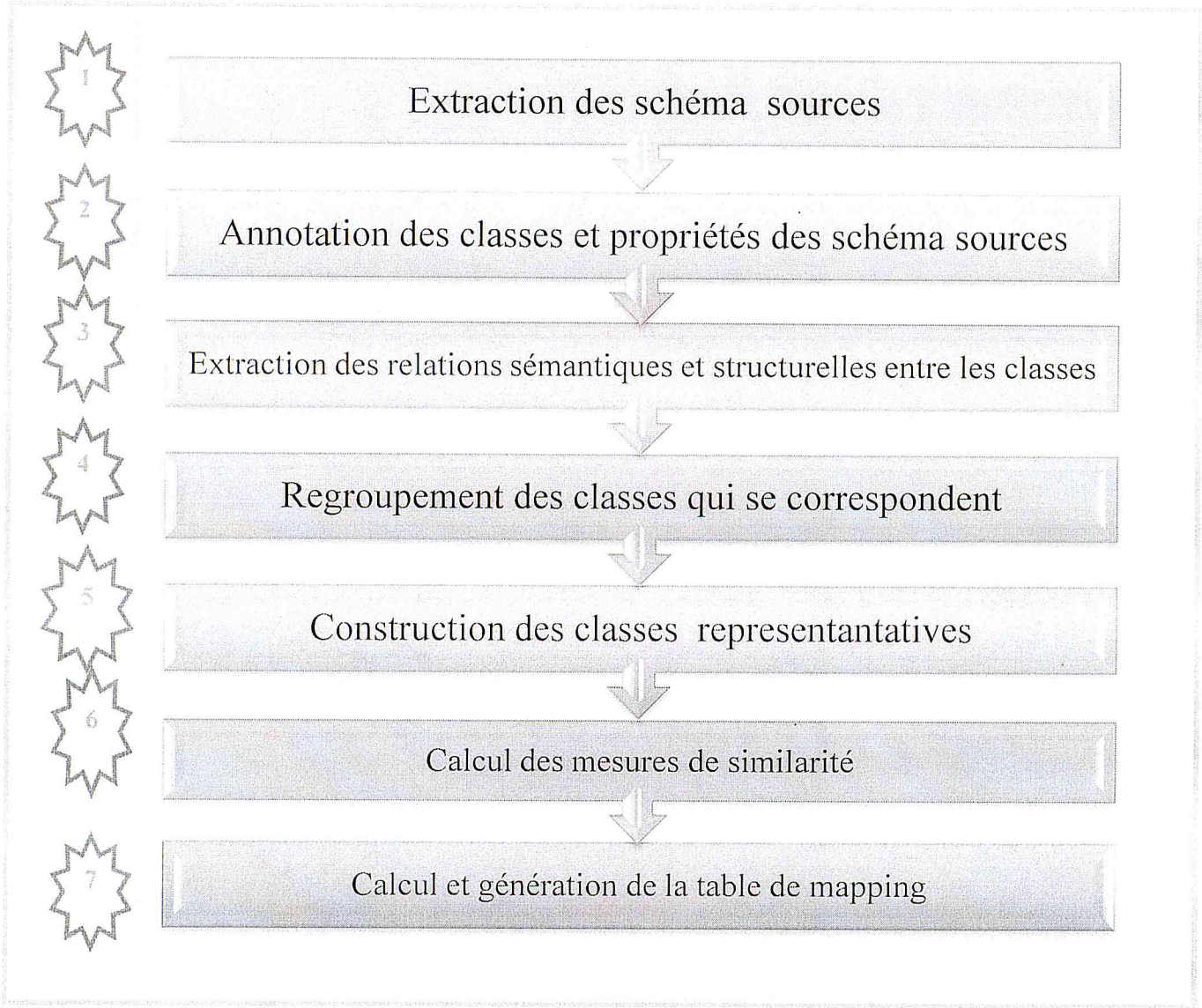


Figure 17: Processus global pour la génération de la table de mapping

2.2 .Traitement de requêtes :

En achevant ces étapes, l'utilisateur peut maintenant poser sa requête sur le schéma global pour récolter les informations désirées à partir des sources locales.

L'utilisateur pose sa requête en utilisant les classes et les attributs du schéma global, le système ***GRMOH*** cherche dans la table de mapping les classes et les attributs locaux associés à chaque classe et attribut globale dans la requête.

La requête doit être décomposée en deux parties, une partie pour l'ensemble des classes et l'autre pour les propriétés.

Puis le système cherche dans la table de mapping les correspondances sémantique entre les concepts et propriétés globales sélectionnées et les celle du locales

Ensuite, le système doit effectuer deux opérations : « l'enrichissement de la requêtes » et « la vérification de la requête ».

Pour l'enrichissement de la requête, notre système doit faire une combinaison entre les classes, deux à deux, afin de trouver toutes les relations qui peuvent exister entre les classes de la requête et qui ne sont pas déjà sélectionné par l'utilisateur, pour les ajouter à notre requête.

Pour la vérification de la requête, il faut combiner les classes deux à deux avec toutes les propriétés qui existent dans la requête, et aussi chaque classe avec toutes les relations afin de prouver la validité de la requête.

Puis le système doit réécrire la requête globale en sous requêtes locales écrite en termes de schémas locales : Pendant la vérification, pour chaque combinaison valide, notre système génère une sous requête selon la syntaxe de la source qui lui correspond.la figure suivante illustre

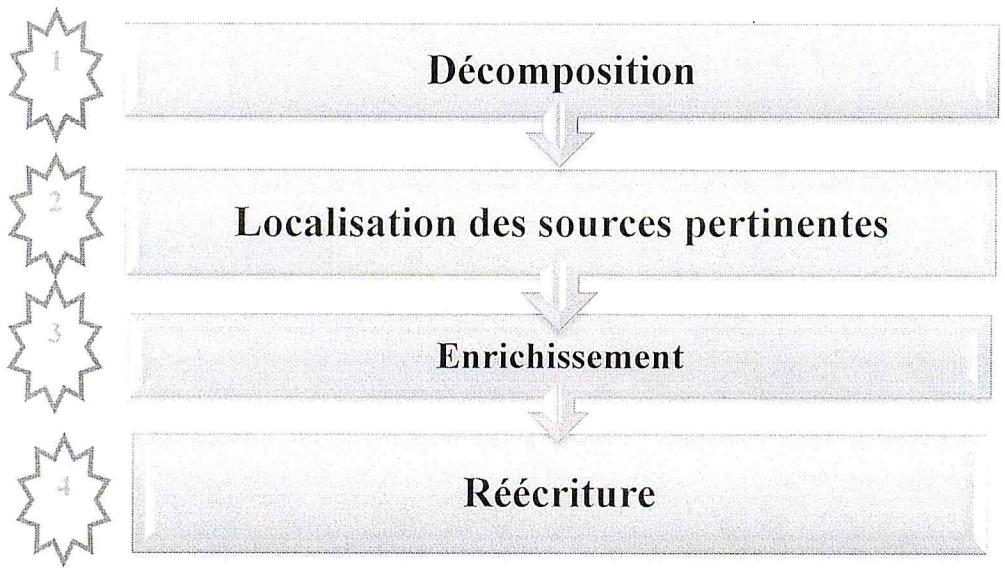


Figure 18: Processus global pour le traitement de la requête

Une requête de médiation ne peut retourner des résultats justes que si les conflits présentés dans les sources seront résolus. Pour pallier à ces contraintes les solutions suivantes sont

prises en compte : Pour les conflits sémantiques, nous avons utilisé un dictionnaire lexical « wordnet ». Cette base lexicale contient les catégories grammaticales suivantes : noms, verbes, adjectifs et adverbes. Elles sont organisées selon leur sens et non leur forme. Wordnet va nous aider à décrire chaque classe des sources selon leurs contextes, ce qui nous facilitera par la suite l'extraction des synonymes.

Les mesures de similarité nous permettent aussi de déterminer dans quelle mesure deux mots sont proches sur le plan sémantique qui les unit.

3. Conclusion

Au cours de ce chapitre nous avons essayé de donner le processus global de notre approche. Nous consacrerons la section qui suit à la conception du système GRMOH.

Chapitre IV:

Analyse des besoins et conception

Chapitre IV: Analyse des besoins et conception

1. Introduction

Après avoir donné un aperçu sur GRHoM, nous allons à présent entamer la partie conception de ce dernier. En passant par une brève définition de la démarche de conception adoptée. Ensuite, on étudiera les besoins techniques et fonctionnels du système GRHOM, on en fera l'analyse et on conclura le chapitre, en parlant de l'architecture statique du système.

2. Présentation de la démarche utilisée

2.1. UP *(Unified process)*

UP est une méthode générique de développement de logiciels, qui couvre tout le cycle de vie du développement basée sur cinq caractéristiques essentielles :

- UP est piloté par les cas d'utilisation.
- UP est centré sur l'architecture.
- UP est itératif et incrémental,
- UP utilise le langage UML,
- UP est à base de composants (Orienté Objet).

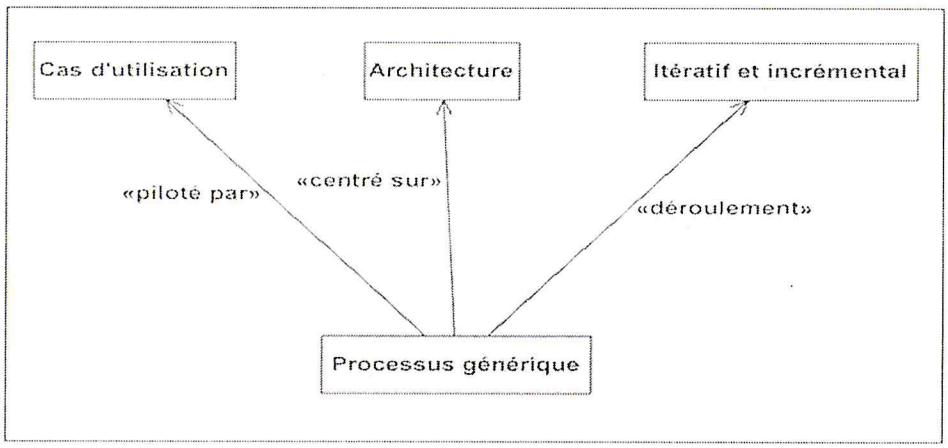


Figure20:Processus Générique UP.

✓ **Le Processus unifié est piloté par les cas d'utilisation**

- Le Processus unifié est piloté par les cas d'utilisation.

- A partir des cas d'utilisation, les développeurs créent une série de modèles UML.

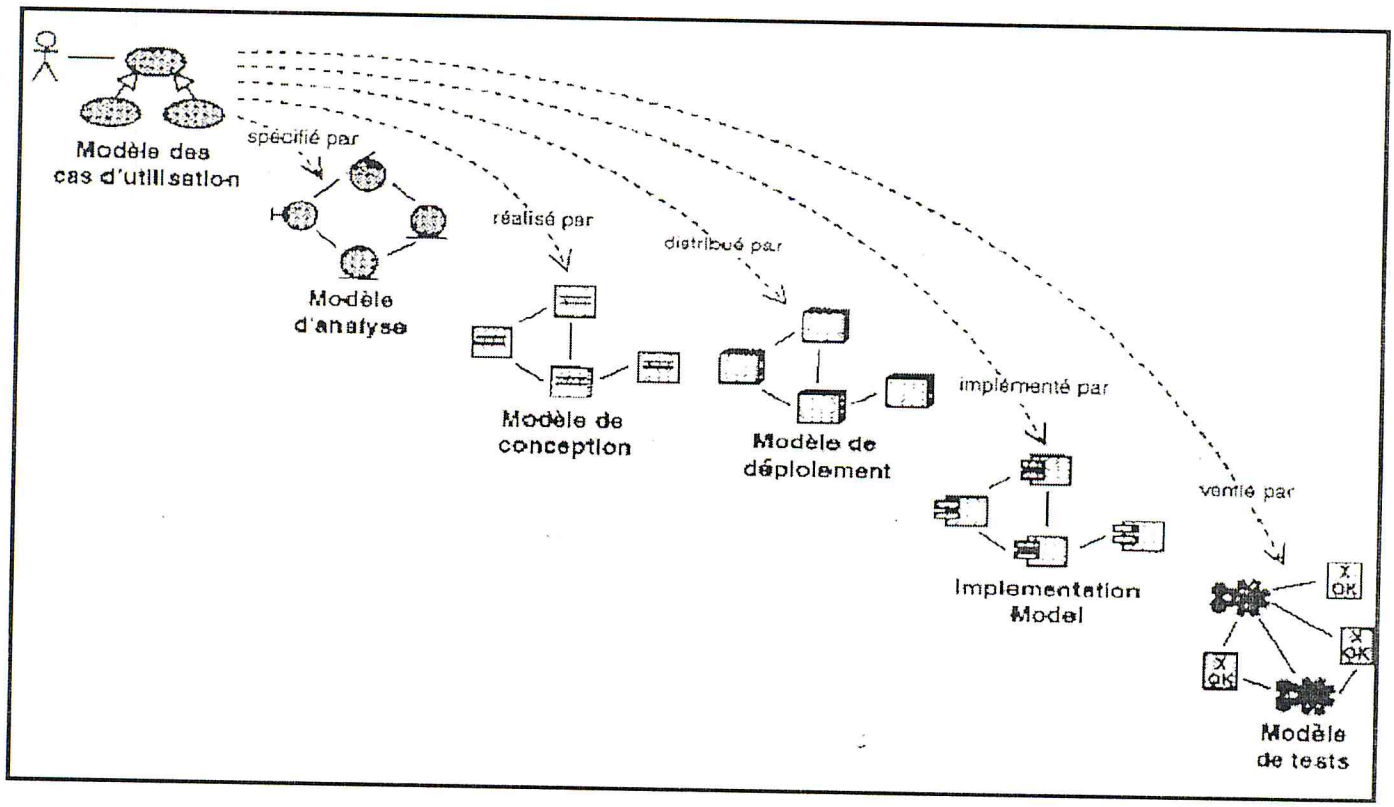


Figure21:UP est centrée par les cas d'utilisations

✓ **Le Processus unifié est piloté par les cas d'utilisation centré sur l'architecture**

- L'architecture regroupe les différentes vues du système qui doit être construit.
- Elle doit prévoir la réalisation de tous les cas d'utilisation.
- Marche à suivre:
  - Créer une ébauche grossière de l'architecture.
  - Travailler sur les cas d'utilisation représentant les fonctions essentielles.
  - Adapter l'architecture pour qu'elle prenne en compte ces cas d'utilisation.
  - Sélectionner d'autres cas d'utilisation et refaire de même.
- L'architecture et les cas d'utilisation évoluent de façon concomitante.

Chapitre IV: Analyse des besoins et conception

✓ **Le Processus unifié est itératif et incrémental**

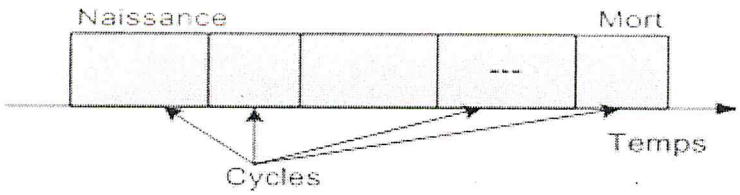
- Découpe du projet en "mini-projet" : Des *ITÉRATIONS* qui donne lieu à un *INCRÉMENT*.
- Chaque itération comprend un certain nombre de cas d'utilisation et doit traiter en priorité les risques majeurs.
- Une itération reprend les livrables dans l'état où les a laissé l'itération précédente et les enrichit progressivement (incrémental).
- Les itérations sont regroupées dans une phase. Chaque phase est ponctuée par un jalon qui marquera la décision que les objectifs (fixés préalablement) ont été remplis.

✓ **Avantages d'un processus itératif**

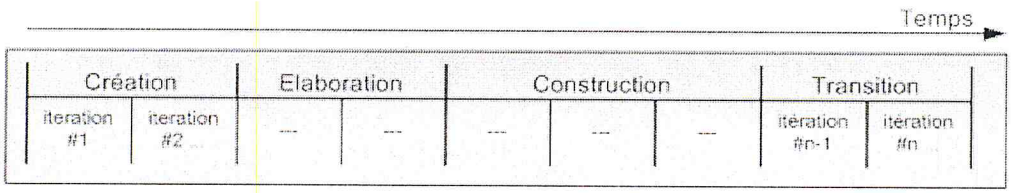
- Permettre de limiter les coûts, en termes de risques, aux strictes dépenses d'une seule itération.
- Permettre de limiter les risques de retard grâce à un feed-back rapide.
- Accélérer le rythme de l'ensemble du développement: travail plus efficace vers des objectifs à court terme.
- Prendre mieux en considération les besoins des utilisateurs qui se dévoilent au fur et à mesure des itérations.

✓ **Le cœur du processus unifié**

➢ Le processus unifié répète un nombre de fois des cycles qui se conclut par une nouvelle version du produit:



➢ Chaque cycle compte quatre phases qui se subdivisent à leur tour en itérations:



Chapitre IV: Analyse des besoins et conception

➢ Chaque phase est repose sur cinq enchaînements d'activités: l'expression des besoins, l'analyse, la conception, l'implémentation et le test.

2.1.Les activités

2.1.1. EXPSSION DES BESOINS

L'expression des besoins comme son nom l'indique, permet de définir les différents besoins :

- inventorier les *besoins* principaux et fournir une liste de leurs fonctions.
- recenser les *besoins fonctionnels* (du point de vue de l'utilisateur) qui conduisent à l'élaboration des modèles de cas d'utilisation.
- appréhender les *besoins non fonctionnels* (techniques) et livrer une liste des exigences.

Le modèle de cas d'utilisation présente le système du point de vue de l'utilisateur et représente sous forme de cas d'utilisation et d'acteur, les besoins du client.

2.1.2. Analyse

L'objectif de l'analyse est d'accéder à une compréhension des besoins et des exigences du client. Il s'agit de livrer des spécifications pour permettre de choisir la conception de la solution.

Un modèle d'analyse livre une spécification complète des besoins issus des cas d'utilisation et les structure sous une forme qui facilite la compréhension (scénarios), la préparation (définition de l'architecture), la modification et la maintenance du futur système.

Il s'écrit dans le langage des développeurs et peut être considéré comme une première ébauche du modèle de conception.

2.1.3. Conception

La conception permet d'acquérir une compréhension approfondie des contraintes liées au langage de programmation, à l'utilisation des composants et au système d'exploitation.

Elle détermine les principales interfaces et les transcrit à l'aide d'une notation commune.

Elle constitue un point de départ à l'implémentation :

- elle décompose le travail d'implémentation en sous-système
- elle crée une abstraction transparente de l'implémentation

2.1.4. Implémentation

L'implémentation est le résultat de la conception pour implémenter le système sous formes de composants, c'est-à-dire, de code source, de scripts, de binaires, d'exécutables et d'autres éléments du même type.

Les objectifs principaux de l'implémentation sont de planifier les intégrations des composants pour chaque itération, et de produire les classes et les sous-systèmes sous formes de codes sources.

2.1.5. Test

Les tests permettent de vérifier des résultats de l'implémentation en testant la construction. Pour mener à bien ces tests, il faut les planifier pour chaque itération, les implémenter en créant des cas de tests, effectuer ces tests et prendre en compte le résultat de chacun.

3. Le langage UML

UML ou Unified Modeling Language, est un langage de modélisation qui est né au milieu des années 90 de la fusion de trois méthodes objets: OMT (Object Modeling Technique), BOOCH1 (GRADY BOOCH le concepteur de la méthode) et OOSE (Object Oriented Software Engineering). L'idée de cette fusion est partie du constat qu'à l'époque ils existaient plusieurs méthodes objets liées par un consensus autour d'idées communes: objets, classes, sous-systèmes etc. **[14].**

Chapitre IV: Analyse des besoins et conception

C'est à partir de 1997 que l'OMG2 (Object Management Group) qui standardise les technologies de l'objet, s'est attachée à la définition d'un langage commun unique, utilisable par toutes les méthodes objets dans toutes les phases du cycle de vie et compatible avec les techniques de réalisation du moment. D'où la naissance d'UML.

UML offre des éléments pour décrire les différents aspects d'un système: les diagrammes. Ses points forts sont les suivants :

- C'est un langage formel et normalisé : il permet un gain de précision, de stabilité et encourage l'utilisation d'outils.
- C'est un support de communication performant : il cadre l'analyse et facilite la compréhension de représentations abstraites complexes. De plus, son caractère polyvalent et sa souplesse en font un langage universel **[14]**.

UML 2.0 comporte ainsi treize types de diagrammes représentant autant de vues distinctes pour représenter des concepts particuliers du système d'information. Ils se répartissent en trois grands groupes **[14]**:

- ***Diagrammes structurels ou diagrammes statiques (UML Structure)***
  - diagramme de classes (Class diagram)
  - diagramme d'objets (Object diagram)
  - diagramme de composants (Component diagram)
  - diagramme de déploiement (Deployment diagram)
  - diagramme de paquetages (Package diagram)
  - diagramme de structures composites (Composite structure diagram)
- ***Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior)***
  - diagramme de cas d'utilisation (Use case diagram)
  - diagramme d'activités (Activity diagram)
  - diagramme d'états-transitions (State machine diagram)
- ***Diagrammes d'interaction (Interaction diagram)***
  - diagramme de séquence (Sequence diagram)

- diagramme de communication (Communication diagram)
- diagramme global d'interaction (Interaction overview diagram)
- diagramme de temps (Timing diagram)

UML est un langage qui permet de représenter des modèles, mais il ne définit pas le processus d'élaboration des modèles. Cependant, dans le cadre de la modélisation d'une application informatique, les auteurs d'UML préconisent d'utiliser une démarche.

Pour standardiser les démarches, plusieurs modèles de démarches ont été décrits et parfois formalisés, parmi ces derniers, UP.

4. Expression des besoins

4.1. Recueil des besoins fonctionnels (diagramme de cas d'utilisation)

4.1.1. Identification des acteurs

La première étape de cette phase est d'énumérer les Acteurs susceptibles d'interagir avec le système. Un **Acteur** représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système), qui interagissent directement avec le système étudié.

Le tableau ci-dessous identifie les acteurs de notre système et décrit la mission de chacun d'eux.

- Les Acteurs

Acteur	Désignation
Utilisateur tiers	Les utilisateurs tiers sont des utilisateurs non privilégiés.
Expert	L'expert est un utilisateur privilégié qui joue un rôle primordial pour l'annotation des classes.
Médiateur	Acteur principal qui s'occupe du traitement de la requête et reproduire le résultat

Tableau 3 : Les acteurs du système.

- Description des besoins fonctionnels

Acteur	Description des besoins fonctionnels
Utilisateur tiers	L'application doit permettre aux utilisateurs tiers de : • Sélectionner un schéma global et les sources à interroger • Lancer une requête globale • Visualiser les résultats finaux
Expert	L'application doit permettre aux experts de : • S'authentifier : saisir un nom et un mot de passe • Sélectionner un schéma global et les sources à interroger • Effectuer une annotation. • Un expert peut lancer une requête globale • Visualiser le résultat de la requête

Tableau4: Description des besoins fonctionnels.

4.1.2. Identification des cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre **[14].**

Un cas d'utilisation (use case) représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier. Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée « notable » à l'acteur concerné **[14]**.

L'identification des cas d'utilisation, donne un aperçu des fonctionnalités futures que doit implémenter le système. Cependant, il nous faut plusieurs itérations pour ainsi arriver à constituer des cas d'utilisation complets. D'autres cas d'utilisation vont

Chapitre IV: Analyse des besoins et conception

L'identification des cas d'utilisation, donne un aperçu des fonctionnalités futures que doit implémenter le système. Cependant, il nous faut plusieurs itérations pour ainsi arriver à constituer des cas d'utilisation complets. D'autres cas d'utilisation vont apparaître au fur à mesure de la description de ceux-là, et l'avancement dans le « recueil des besoins fonctionnels ».

Dans notre cas d'étude nous avons défini les diagrammes de cas d'utilisation généraux et détaillés. Pour les cas globaux, on distingue:

- Lancer une requête (pour les utilisateurs /Expert).
- Sélection du schéma global et les sources locales.
- Effectuer une annotation.
- Traitement de la requête.
- Visualisation des résultats.

Nous détaillerons chaque cas d'utilisation qui doit faire l'objet d'une définition a priori qui décrit l'intention de l'acteur lorsqu'il utilise **le système**, et les séquences d'actions principales qu'il est susceptible d'effectuer. Ces définitions servent à fixer les idées et n'ont pas pour but de spécifier un fonctionnement complet et irréversible.

Remarque : Les descriptions détaillées vont être organisées dans des tableaux de la façon suivante :

Elément	Signification
Cas d'utilisation	Le nom du cas d'utilisation
Acteur	L'acteur qui réalise le cas d'utilisation
But	Le but du cas d'utilisation
Description	Une explication du cas d'utilisation
Pré condition	Les conditions qui doivent être vérifiées afin d'effectuer le cas d'utilisation

Post condition	Les résultats du cas d'utilisation
Exception	Les informations entrées par l'acteur

Tableau 5:Modèle de représentation des descriptions détaillées des cas d'utilisations.

4.1.3. Diagrammes de cas d'utilisation

- *Diagramme de cas d'utilisation global*

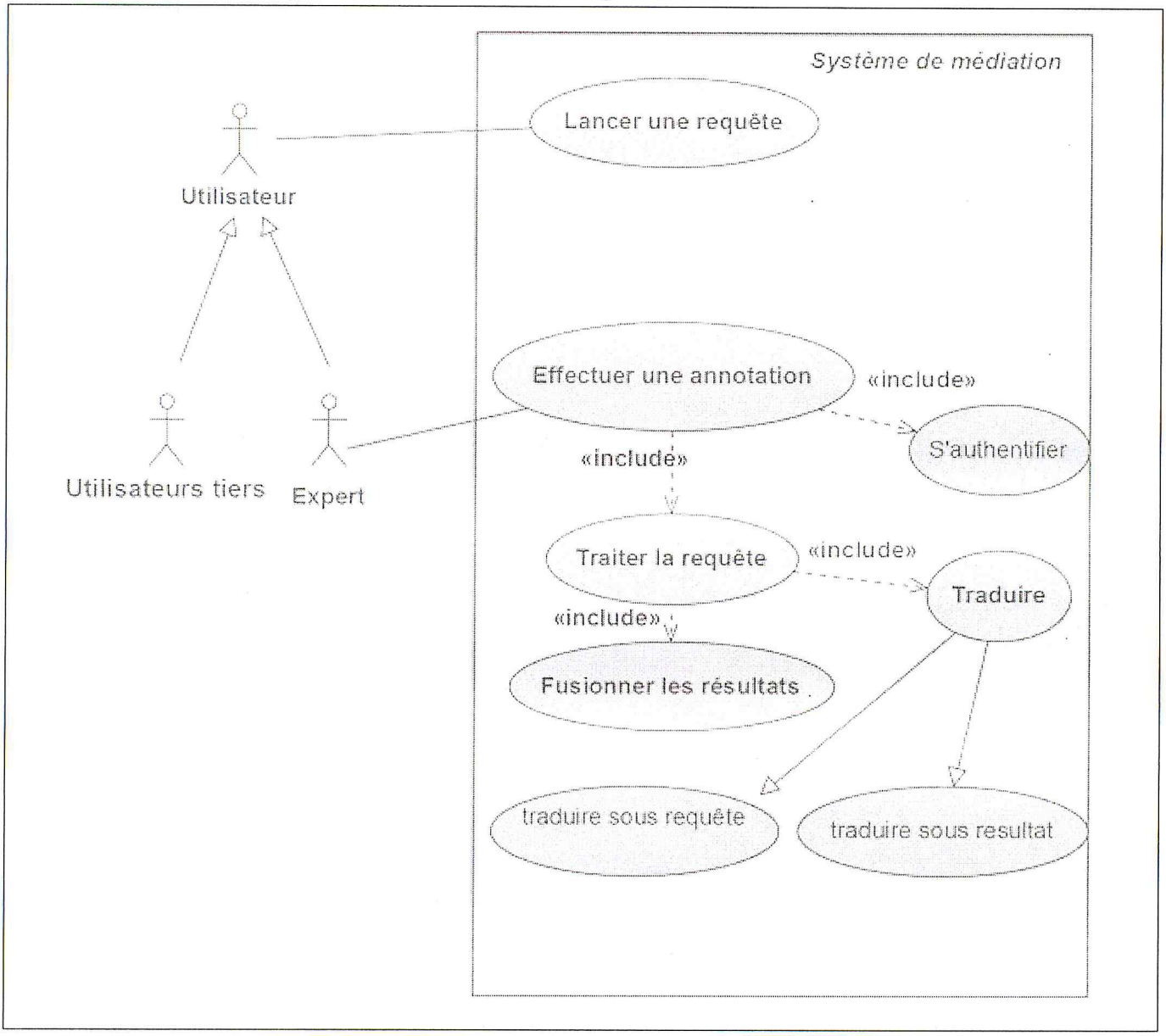


Figure22:Diagramme de cas d'utilisation global

Authentification	
Acteur	Expert
But	Permettre à l'utilisateur de s'authentifier pour ouvrir une session et accéder à ses privilèges
Description	L'accès à l'espace privilégié doit passer par l'interface d'authentification qui contient les renseignements adéquats.
Pré condition	L'expert doit fournir les informations correctes (le nom d'utilisateur et le mot de passe).
Post condition	L'expert peut effectuer des tâches qui lui sont permises
Exception	Annulation ; si l'expert saisit un nom d'utilisateur ou un mot de passe incorrecte, le système affiche un message d'erreur.
Lancement de la requête globale	
Acteur	Utilisateur
But	Obtention des résultats après avoir effectué une requête globale valide
Description	Le lancement d'une requête doit passer par la sélection du schéma global et des sources locales.
Pré condition	L'utilisateur doit sélectionner un schéma global et des sources locales et formuler une requête globale valide écrite en termes de schéma global.
Post condition	Les résultats sont affichés.
Annotation des sources locales	

Acteur	Expert
But	Associer à chaque source locale non annotée une définition qui lui convient ceci pourra être effectué à l'aide d'une base de données lexicale tel que Wordnet.
Description	L'annotation des sources ne peut être effectuée que par un Expert du domaine.
Pré- condition	L'expert, après s'être authentifié peut : - Sélectionner des classes locales non annotées. - Ajouter et associer des définitions aux classes locales.
Post condition	Des sources de données sont annotées : chaque classe ou propriété appartenant à une source de données locale est associée à une définition qui lui convient.
Traitement de la requête	
Réalisée par	Médiateur
But	Obtention des sous-requêtes prête à être traduite par l'adaptateur
Description	Le médiateur doit décomposer la requête globale en sous-requêtes exprimées en termes de schéma global et localiser les sources pertinentes en se basant sur la table de mapping puis il se charge à réécrire les sous-requête exprimées en terme des schémas sources à fin d'obtenir des sous-requêtes prêtes à être traduite par l'adaptateur.
Pré- condition	Une requête globale valide
Post condition	Des sous-requêtes exprimées en termes des schémas sources locaux sont produites.

	des sources mères.
Description	L'adaptateur doit traduire les sous-requêtes reçues en langage de requêtes des sources locales qui lui est associées
Pré- condition	Une sous-requête exprimée en termes des schémas des sources locales.
Post condition	Une sous-requête traduite en langage de requêtes des sources locales
Traduction des sous-résultats	
Acteur	Expert
But :	Obtention des sous requêtes exprimées en langage de requêtes des sources locales.
Description	L'adaptateur doit résoudre les conflits sémantiques liés aux données qui sont généralement causés par la provenance des données de diverses origines .et reformuler un sous-résultat qui a la même précision que celle définit en schéma global.
Pré- condition	Recevoir un sous-résultat de la part de la source locale.
Post condition	Sous-résultat transformé.
Fusion	
Réalisée par	Médiateur
But	Regrouper les sous-résultats afin d'obtenir un résultat unie
Description	Le médiateur doit rassembler le sous-résultat envoyé par l'adaptateur et les regrouper pour les retourner à l'utilisateur
Pré- condition	Recevoir un sous-résultat
Post condition	Un résultat global.

Tableau 6: Description détaillée du cas d'utilisation global

Description	Le médiateur doit rassembler le sous-résultat envoyé par l'adaptateur et les regrouper pour les retourner à l'utilisateur
Pré- condition	Recevoir un sous-résultat
Post condition	Un résultat global.

bleau6 : Description détaillée du cas d'utilisation global

- *Diagramme de cas d'utilisation détaillé pour le « processus de génération des mapping »*

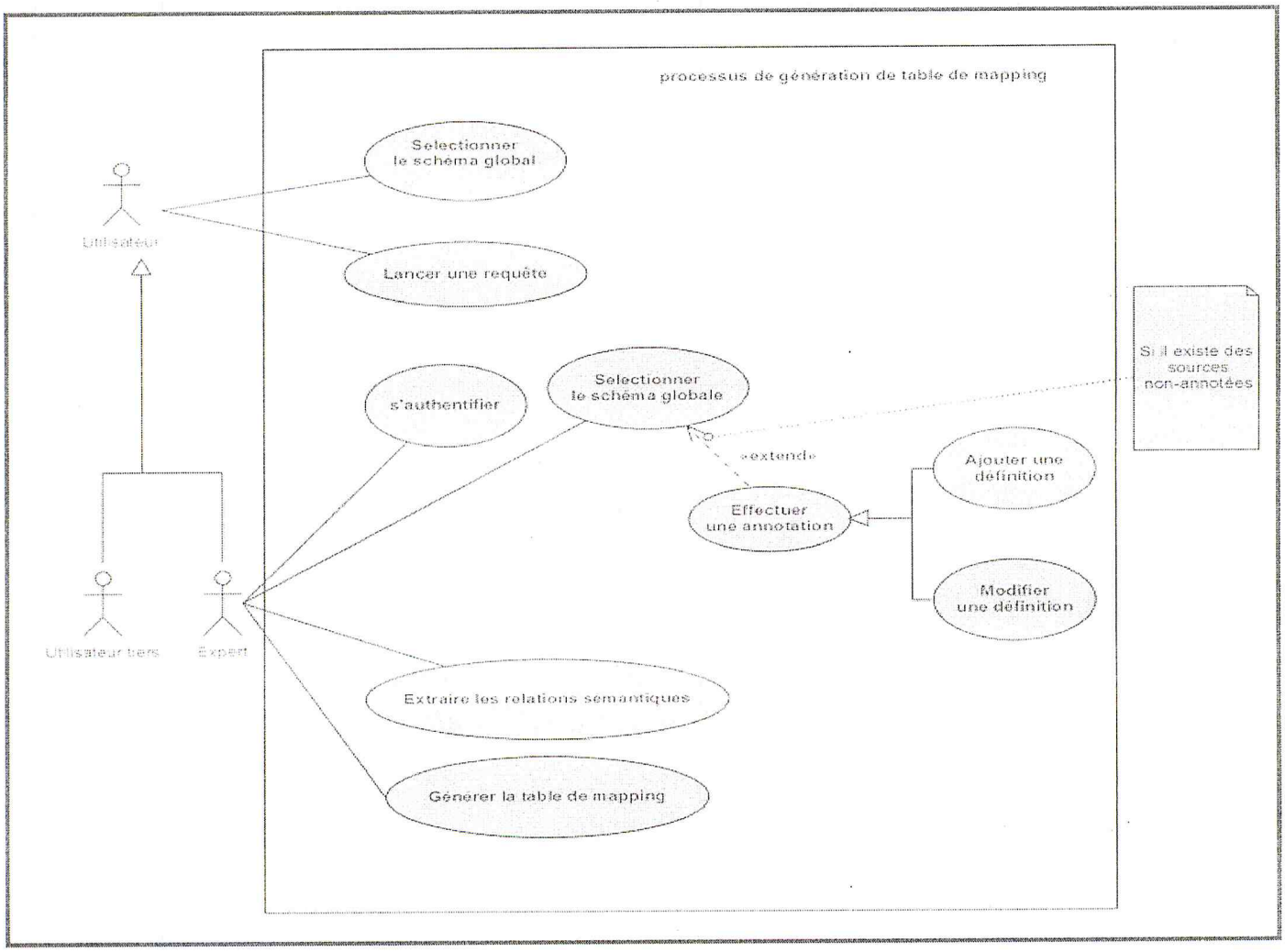


Figure 23:Diagramme de cas d'utilisation du processus de génération des mapping

Sélection d'un schéma global et les sources locales déjà existantes	
Acteur	Utilisateur
But	Pouvoir effectuer des requêtes exprimées en termes de schéma global.
Description	L'utilisateur peut sélectionner un schéma global, automatiquement les sources locales qui lui corresponds sont sélectionnées aussi et affichées.
Pré condition	L'existence d'un schéma global et des sources locales et qu'elles ne soient pas vide.
Post condition	Visualisation d'un schéma global et des sources locales.
Exception	Annulation, Si l'utilisateur ne trouve pas le schéma global
Sélection d'un schéma global et les sources locales déjà existantes	
Acteur	Expert
But	Pouvoir effectuer des requêtes exprimées en termes de schéma global. Ainsi que la possibilité d'effectuer une annotation aux sources locales.
Description	L'expert peut sélectionner n'importe quelle source et peut ajouter une définition qui lui corresponde.
Pré condition	L'existence d'un schéma global et des sources locales.
Post condition	Visualisation d'un schéma global et des sources locales (avec statut) des couleurs spécifiques pour des sources annotées et non annotées.
Exception	Annulation, Si l'utilisateur ne trouve pas le schéma global, ou annule le processus.
Effectuer une annotation	
Acteur	Expert
But	Attribuer manuellement à des classes locales non annotées des

	définitions données par sa part. afin d'enrichir la base de données lexicale et aider le système à effectuer l'extraction des relations sémantiques entre toutes les classes locales.
Description	L'expert, après s'être authentifié peut : Associer à une classe globale une définition donnée par lui-même
Pré condition	S'authentifier et la classe locale soit avec un statut non-annotée.
Post-condition	Classe locale annotée
Extraction des relations sémantiques inter-schéma et intra-schéma et relations structurelles et classification des classes	
Réalisée par	Le Système
Lancée par	Expert
But	Extraction des relations sémantiques et structurelles entre les classes
Description	Le système doit extraire des relations sémantiques à partir des définitions qui sont associées à chaque classes locales ainsi des relations structurelles .afin de pouvoir les classifier et les regrouper en plusieurs groupes.
Pré condition	Des sources locales annotées.
Post condition	Des classes locales classifiées en plusieurs groupes selon leurs relations sémantiques et structurelles.
Génération de la table de mapping	
Réalisée par	Le système
Lancée par	L'expert
But	Associer chaque classe et propriétés globale aux ensembles des classes et propriétés locales qui lui correspondent.
Description	Après avoir calculer les mesures de similarités entre les classes globales et les représentants .le système établit les correspondances entre

	chaque classe globales et le groupes de classes locales qui lui correspond en suite une table de mapping est générée comprenant les classes globales et les classes locales et propriétés globales et locales et leurs correspondances.
Pré condition	L'annotation a été déjà effectué plus l'extraction des relations sémantiques plus l'existence d'une correspondance entre le schéma global et les groupes des schémas sources locales.
Post condition	Une table de mapping comprenant les correspondances sémantiques.

Tableau7:Description détaillée pour le lancement d'une requête.

- *Diagramme de cas d'utilisation détaillé « Traitement d'une requête »*

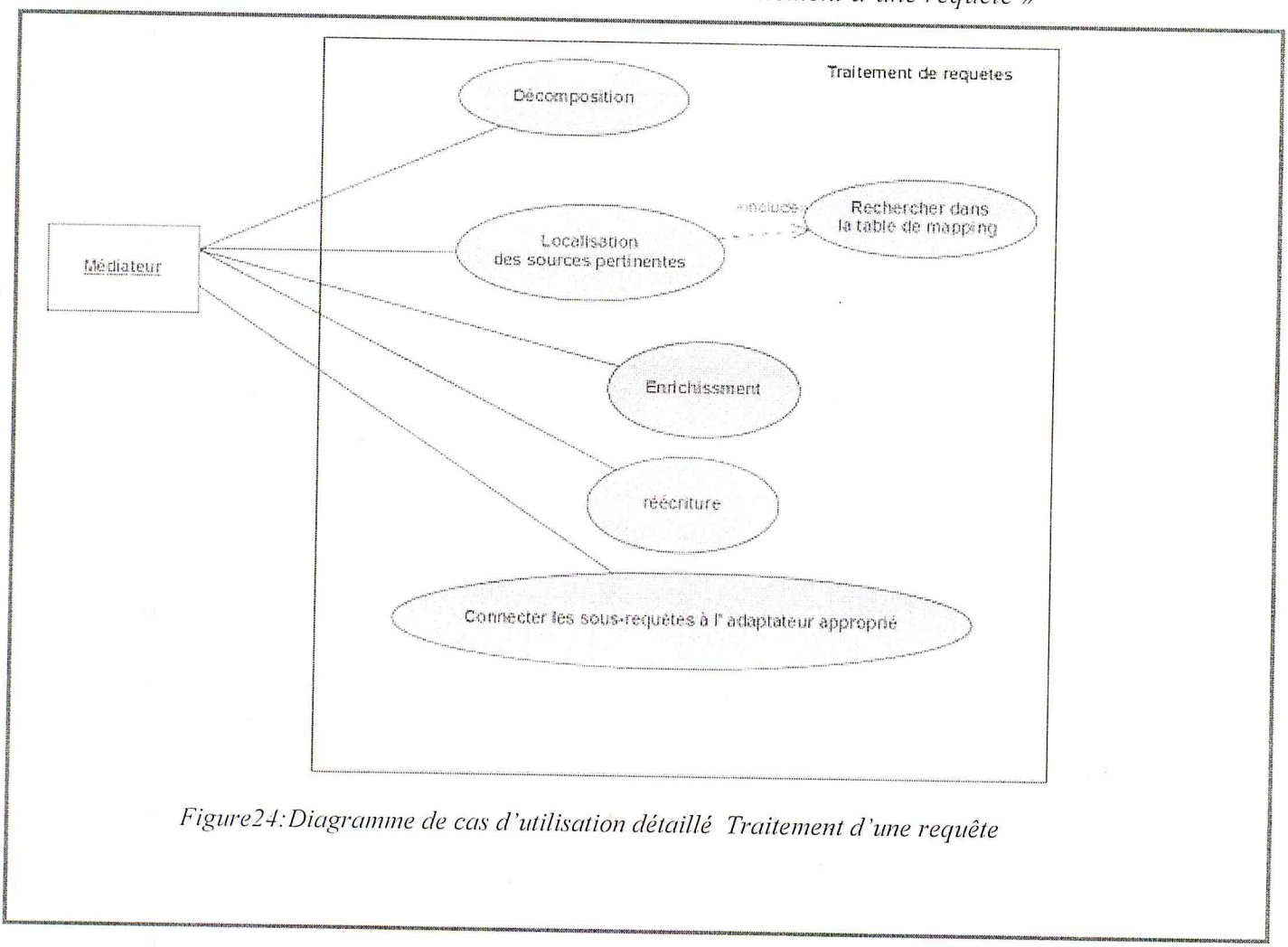


Figure24:Diagramme de cas d'utilisation détaillé Traitement d'une requête

Décomposition de la requête	
Réalisée par	Médiateur
But	Obtention des requêtes adaptées aux sources
Description	Décomposer la requête globale en deux parties une partie pour les classes globales et une partie pour les propriétés.
Pré condition	Lancement d'une requête valide
Post condition	Création des sous-requêtes
Localisation des sources pertinentes	
Réalisée par	Médiateur
But	Localiser la source pertinente pour réécrire les sous requêtes
Description	En se basant sur la table de mapping le médiateur peut localiser les sources pertinentes et les classes locales qui se correspondent avec les classes globales interrogées et les propriétés locales qui se correspondant avec les propriétés globales.
Pré condition	L'existence d'un lien de correspondances entre une classe globale et une classe locales et entre une propriété globale et une propriété locale.
Post condition	Des sources pertinentes sont localisées
Enrichissement	
Réalisé par	Médiateur
But	Ajouter des nouvelles relations à notre requête.
Description	Cette étape consiste à rechercher une nouvelle relation sémantique trouvée entre les composants de la requête.
Pré condition	L'établissement des mapping pour la requête globale
Post condition	Requête enrichie

Réécriture	
Réalisée par	Médiateur
But	Reformulation des sous requêtes en termes des attributs locaux pour chaque source pertinente.
Description	.Après avoir détecté les sources pertinentes les sous requêtes sont reformuler en termes des attributs des classes locales déjà classifiées.
Pré condition	Une sous requête adaptée.
Post condition	Des sous-requêtes réécrites en termes de classes locales déjà classifiées.
Connexion à l'adaptateur	
Réalisé par	Médiateur
But	Traduire les sous requêtes réécrites en langages de requêtes des sources mères
Description	Le médiateur doit envoyer chaque sous-requête réécrite à l'adaptateur appropriée
Pré condition	Le médiateur doit détecter l'adaptateur appropriée à la sous-requête afin qu'il puisse effectuer une traduction correcte pour que cette dernière pourra être interrogée par la source locale qui lui convient
Post condition	Sous-requête envoyée.

Tableau 8:Traitement de requête.

4.2. Recueil des besoins techniques

La capture des besoins techniques recense toutes les contraintes et les choix dimensionnant la conception du système.

Parmi les besoins techniques principaux, on recense :

-L'application doit être réalisée dans un environnement Open source, ce qui offre des outils de développement gratuits et avec la possibilité de modifier le code source de ces derniers.

- Les sources données sont créés à partir de deux éditeurs :
  - **Protégé** : pour la construction des sources au format (.owl).
  - **Oxygen XML Editor :** pour la construction des sources au format (.xml)
  - **Mysql SGBDr :** pour la construction des sources relationnelles.

- Pour gérer les sources (.owl) on va utiliser différentes **APIs :**
  - **Jena** pour exploitation,
  - **JWS** pour le calcul de similarité structurelle,
  - **JAWS** pour la spécification des différentes définitions (et synonymes) d'un concept donné,
- Pour gérer les sources (.xml) on va utiliser différentes **APIs** :
  - **Jdom2** pour la manipulation des documents (.xml).
  - **Saxon9** pour pouvoir exécuter des requêtes sur des documents (.xml)
  - L'utilisation de ces APIs sera expliquée en détail dans le chapitre « Implémentation ».

5. Analyse du système

Dans la phase d'analyse nous présenterons les diagrammes d'activité et les diagrammes de séquences :

5.1. Diagramme d'activités

Le diagramme d'activités n'est autre que la transcription dans UML de la représentation du processus telle qu'elle a été élaborée lors du travail qui a préparé la modélisation : il montre l'enchaînement des activités qui concourent au processus **[14].**

5.1.1. Diagrammes d'activités pour l'utilisateur

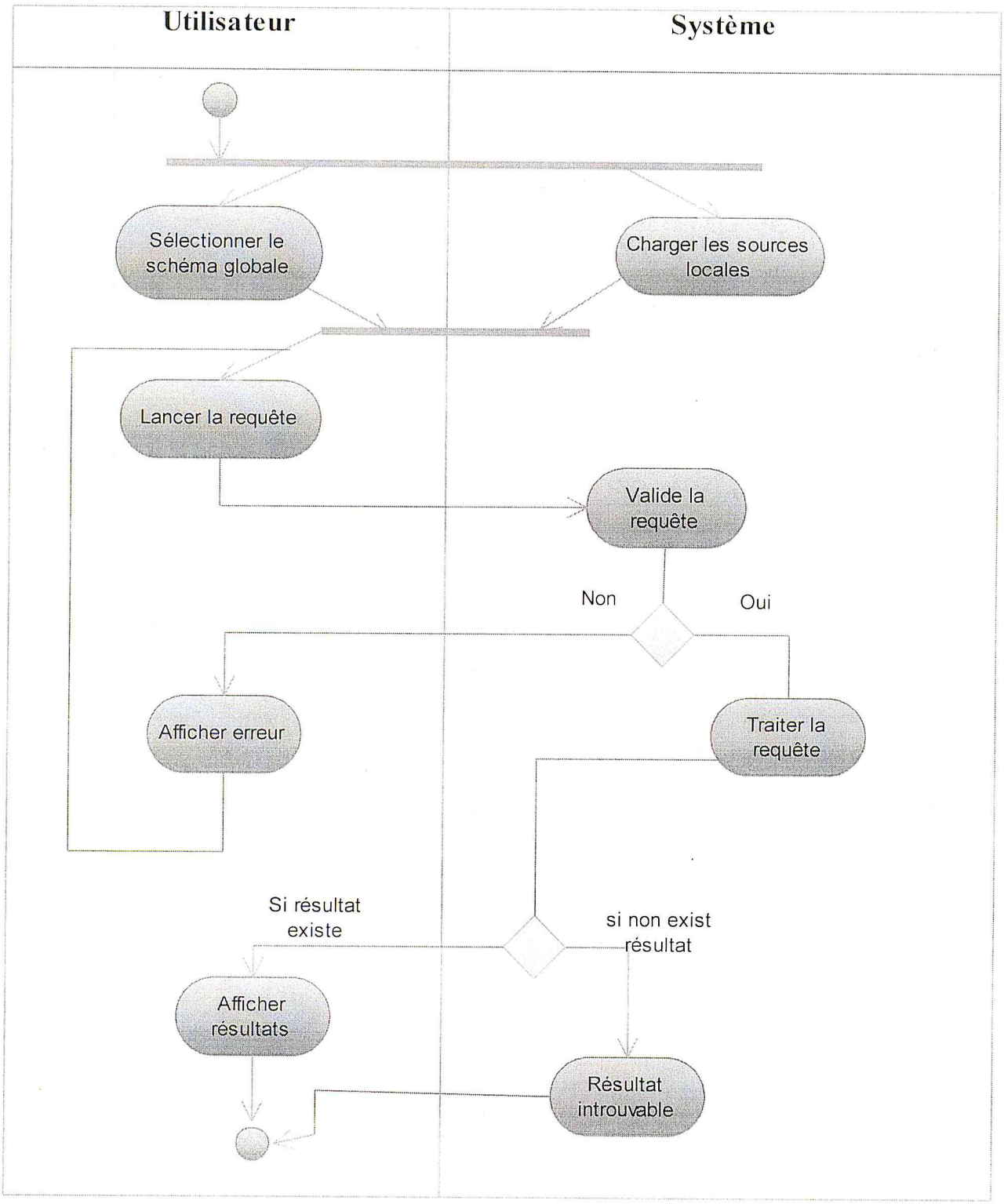


Figure 24:Diagramme d'activité pour l'utilisateur le processus de lancement d'une requête.

5.1.2. Diagrammes d'activités pour l'Expert

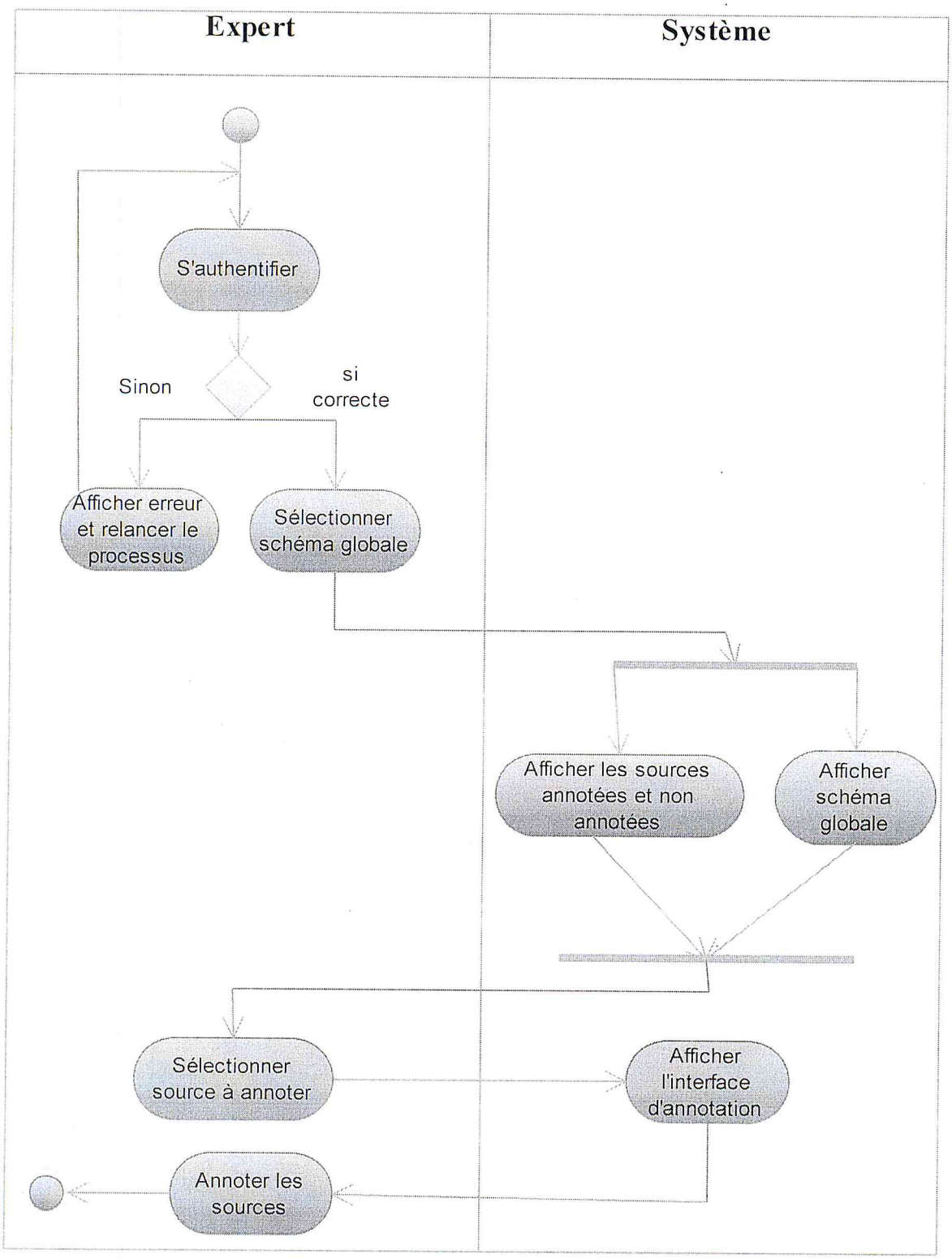


Figure 25:Diagramme d'activité pour l'annotation.

5.1.3. Diagramme d'activité pour le médiateur et l'adaptateur

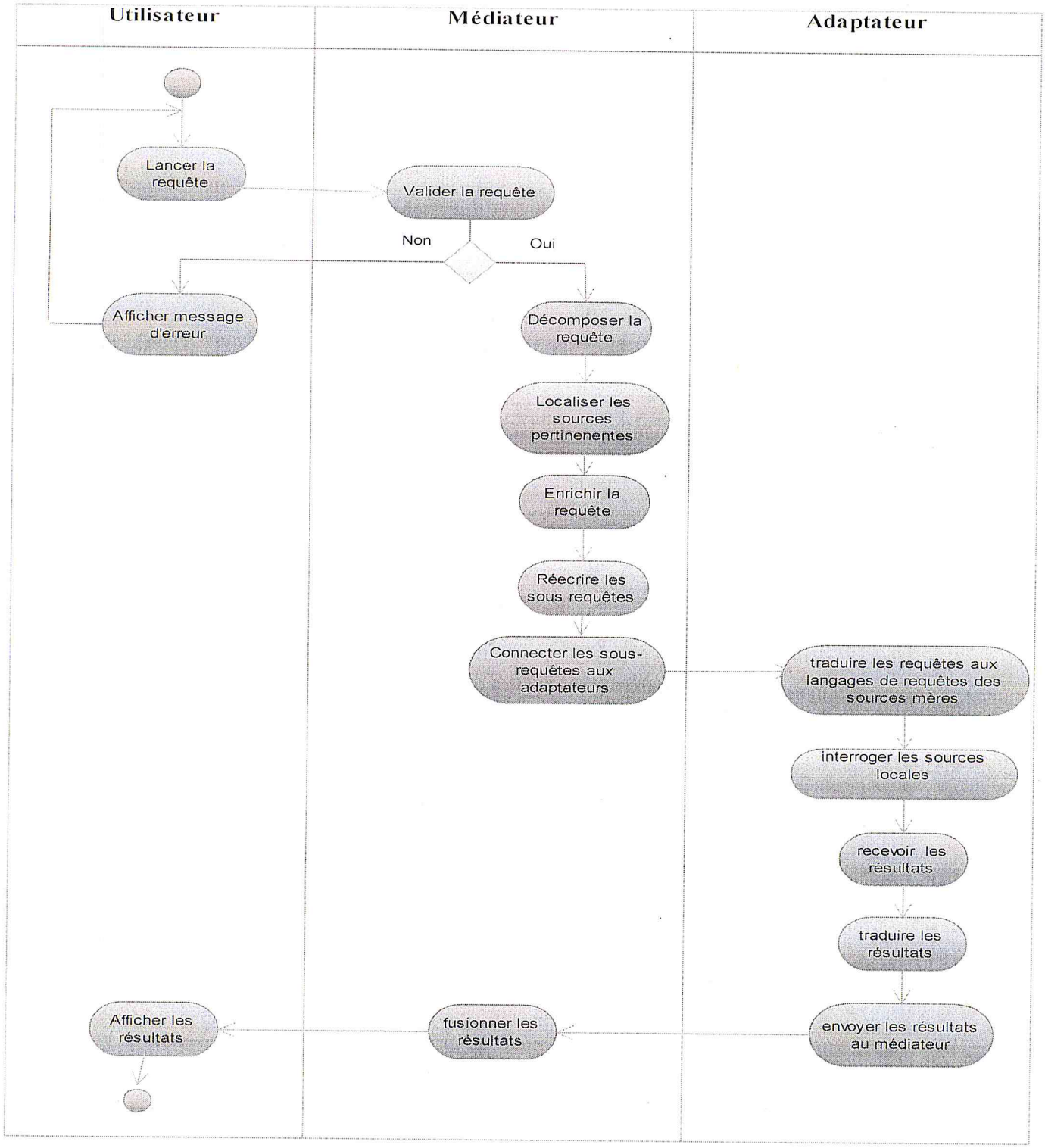


Figure26: Diagramme d'activité pour le traitement de la requête et l'affichage des résultats.

5.2. Scénarios et diagrammes de séquences

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre. On peut représenter les mêmes opérations par un graphe dont les nœuds sont des objets et les arcs (numérotés selon la chronologie) les échanges entre objets **[13].**

5.2.1. Authentification

- *Scénario et diagramme de séquences « Authentification pour l'expert » (le cas normal)*

1. L'expert se connecte au système.
2. Le système affiche la boite d'authentification.
3. L'expert saisi ses informations personnelles (nom d'utilisateur et mot e passe).
4. Le système vérifie l'identité de l'expert.
5. Le système confirme l'identité de l'expert.
6. Le système Attribue l'accès privilégié à l'expert.

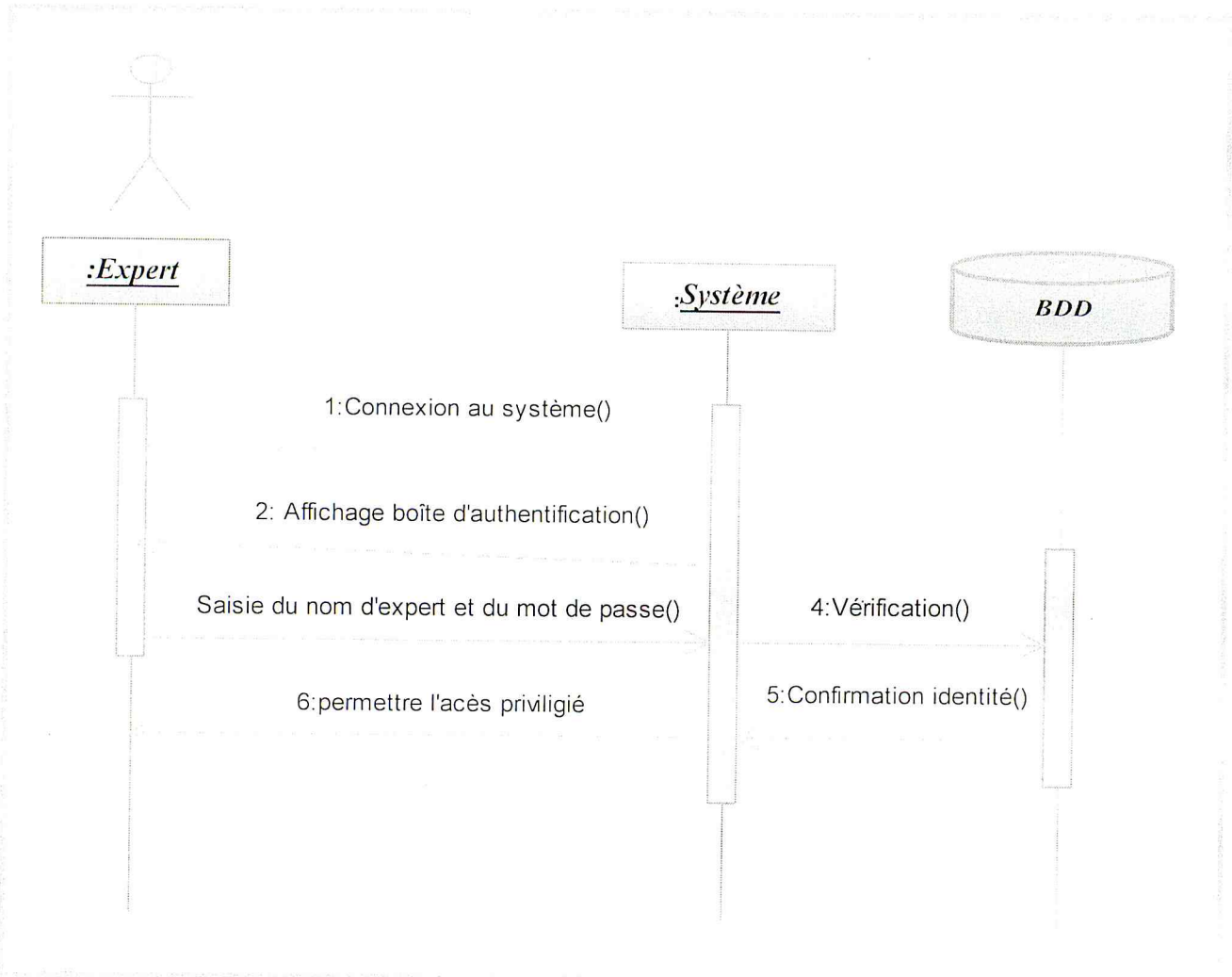


Figure 27:Diagramme de séquences "Authentification" (cas n°1).

- *Scénario et diagramme de séquences « Authentification » (le cas d'erreur)*

1. L'expert se connecte au système.
2. Le système affiche la boite d'authentification.
3. L'expert saisi ses informations personnelles (nom d'utilisateur et mot e passe).
4. Le système vérifie l'identité de l'expert.
5. Déni de l'identité de l'expert.
6. Afficher un message d'erreur.

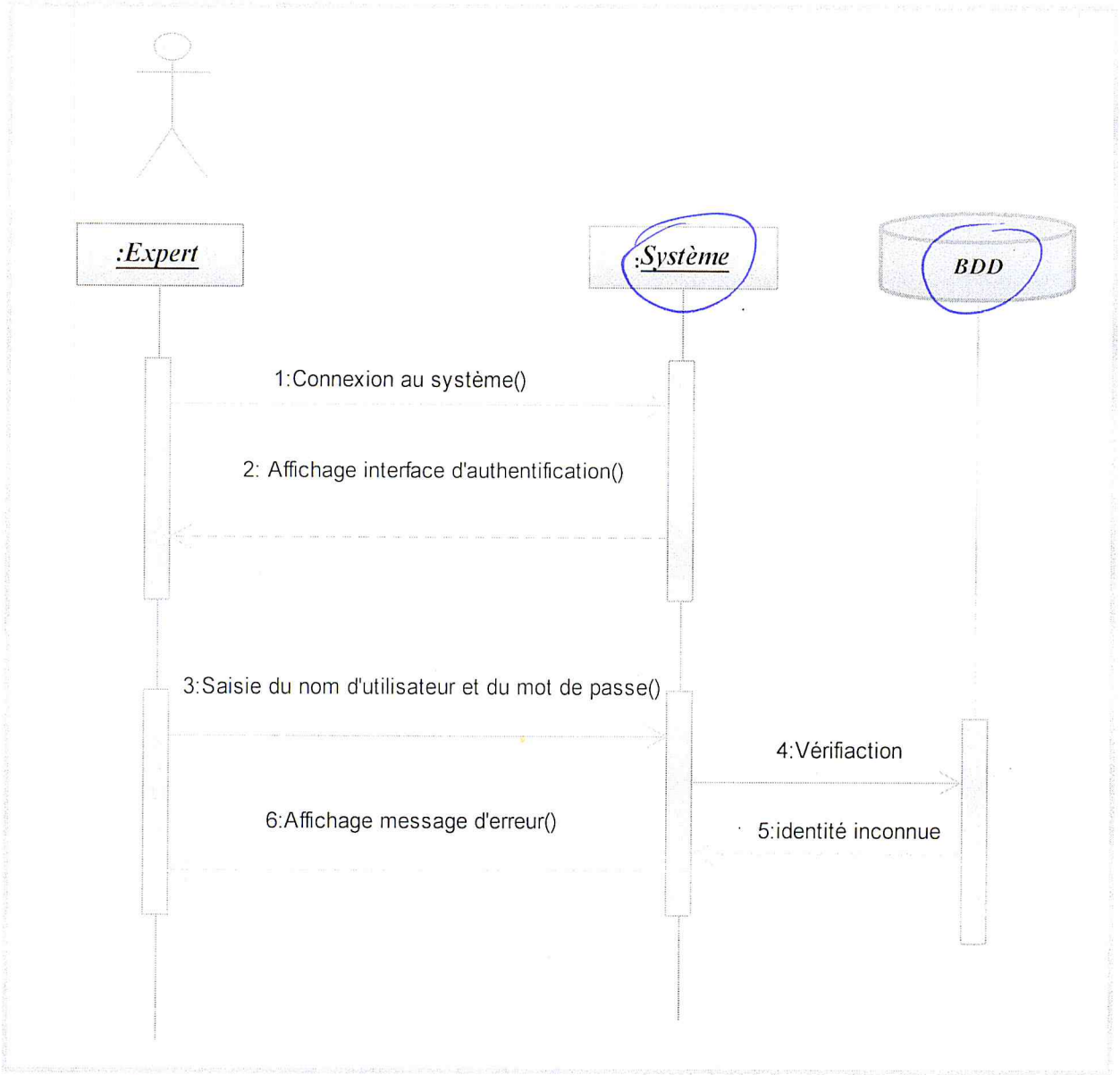


Figure28:Diagramme de séquences "Authentification" (cas n°2).

5.2.2. Traitement de la requête

- *Scénario et diagramme de séquences « Traitement de la requête »*

1. Le système affiche la page d'accueil.
2. L'utilisateur sélectionne un schéma global.
3. Le système affiche le schéma global et les sources locaux sélectionnées.
4. L'utilisateur saisi la requête.
5. Le système vérifie la requête.

6. Le système envoie la requête au médiateur.
7. Le médiateur décompose la requête en deux parties (partie pour les classes et une partie pour les propriétés.
8. Le médiateur localise les sources pertinentes en se basant sur la table de mapping.
9. Le médiateur effectue un enrichissement pour la requête.
10. Le médiateur réécrire les sous-requêtes en termes des schémas sources locaux
11. Le médiateur connecte les sous-requêtes aux adaptateurs (XML, OWL, Relationnel).
12. L'adaptateur traduit la sous-requête reçue.
13. L'adaptateur interroge la source locale qui lui correspond.
14. L'adaptateur reçoit les résultats.
15. L'adaptateur traduit les résultats.
16. Et renvoie les résultats au médiateur.
17. Le médiateur fusionne les résultats reçus.
18. Renvoie le résultat final.
19. Le système affiche le résultat.

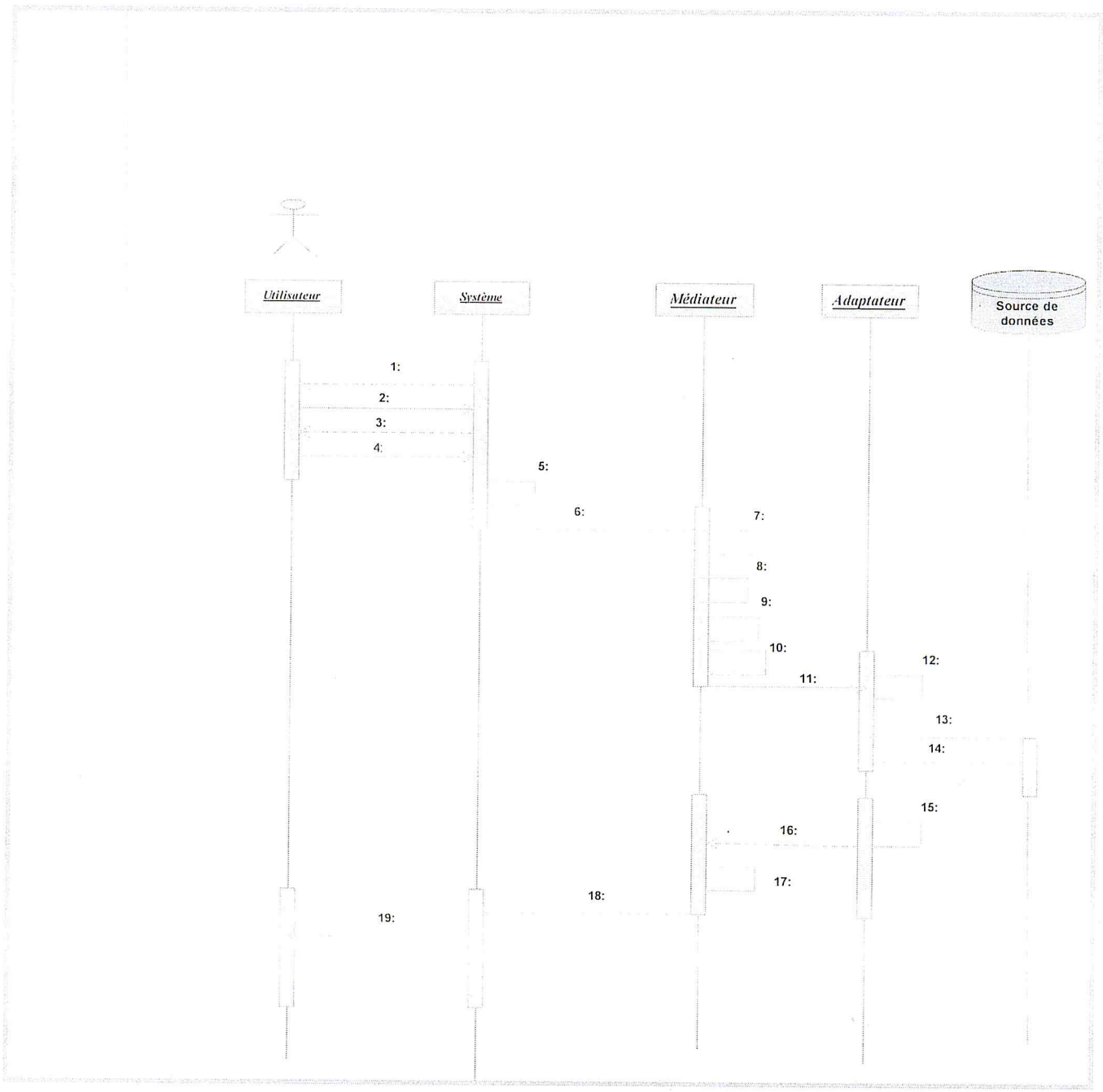


Figure29:Diagramme de séquences « Traitement de la requête ».

- *Scénario et diagramme de séquences « Ajouter une annotation ».*

1. L'expert s'authentifie.
2. Le système confirme l'authentification.

3. Le système affiche la page d'accueil.
4. L'expert sélectionne un schéma global.
5. Le système retourne le schéma global et les sources locales qui lui correspondent.
6. S'il existe des sources avec un statut non-annoté l'expert sélectionne une source non-annotée.
7. L'expert demande au système de lui permettre effectuer une annotation.
8. Le système affiche l'interface d'annotation.
9. L'expert ajoute une définition et l'associer à une classe ou propriété.
10. L'expert confirme la définition.
11. Le système ajoute la définition à la base lexicale.

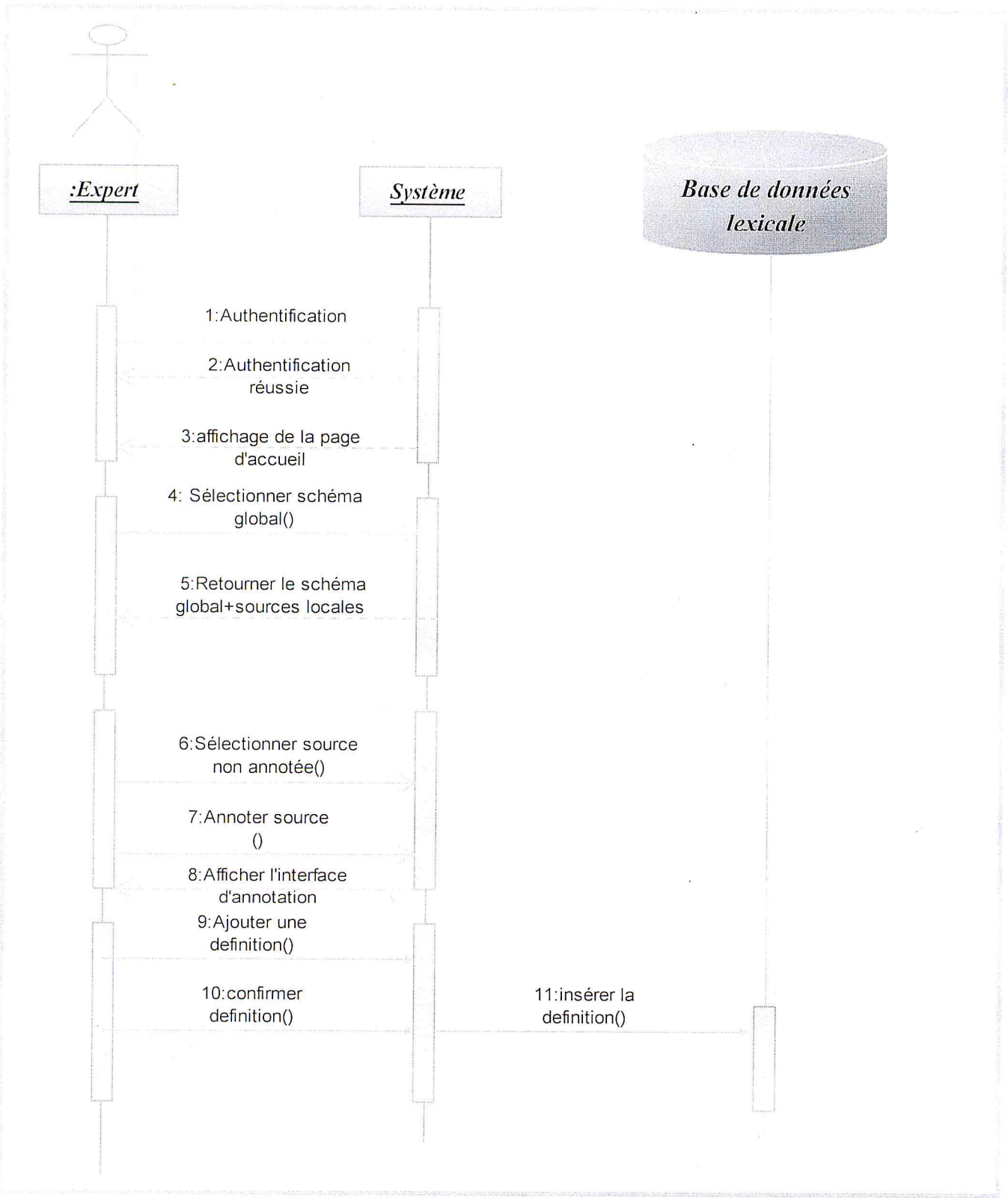


Figure30: Diagramme de séquences « Annotation ».

- *Scénario et diagramme de séquences « Génération de la table de mapping »*

1. L'expert lance le processus d'extraction des relations sémantiques.
2. Le système effectue une recherche dans la base lexicale.
3. La base lexicale retourne les concepts locaux associés avec leur propre définition.
4. Le système regroupe les concepts et propriétés similaires.
5. Le système crée un représentant pour chaque groupe.
6. Le système envoie un message à l'expert indiquant que le processus d'extraction terminé.
7. L'expert lance le processus de génération de la table de mapping.
8. Le système calcule les mesures de similarités entre les classes globales et les représentants.
9. Le système met en correspondance les classes globales et les classes locales qui leurs correspond .
10. Le système génère une table de mapping.
11. Le système envoie un message indiquant que la table de mapping est générée.

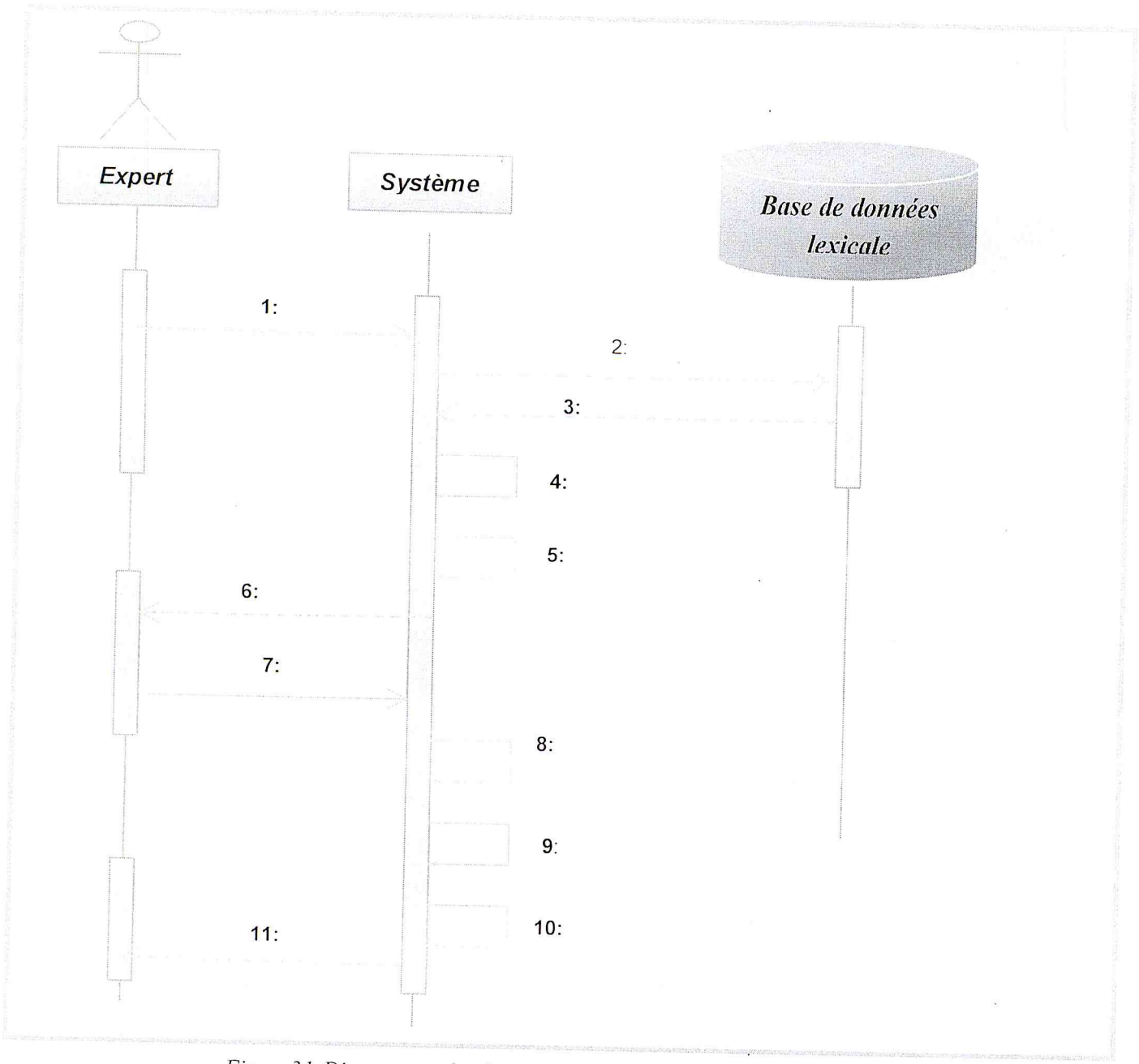
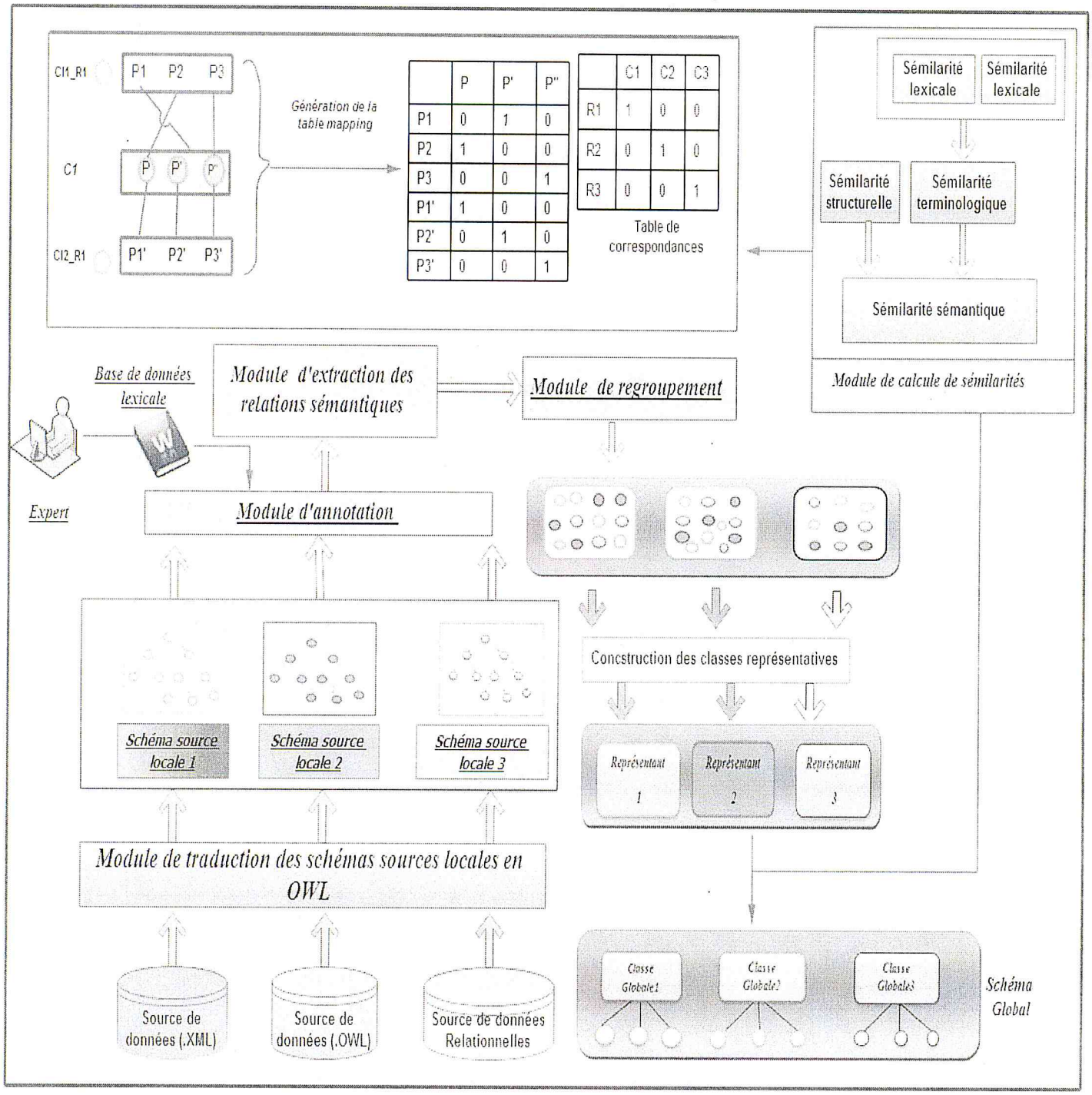


Figure31:Diagramme de séquences Génération des mappings».

6. Conception du système

6.1. Architecture du système



	P	P'	P''
P1	0	1	0
P2	1	0	0
P3	0	0	1
P1'	1	0	0
P2'	0	1	0
P3'	0	0	1

	C1	C2	C3
R1	1	0	0
R2	0	1	0
R3	0	0	1

Figure32: Architecture détaillée du système pour la génération des mappings.

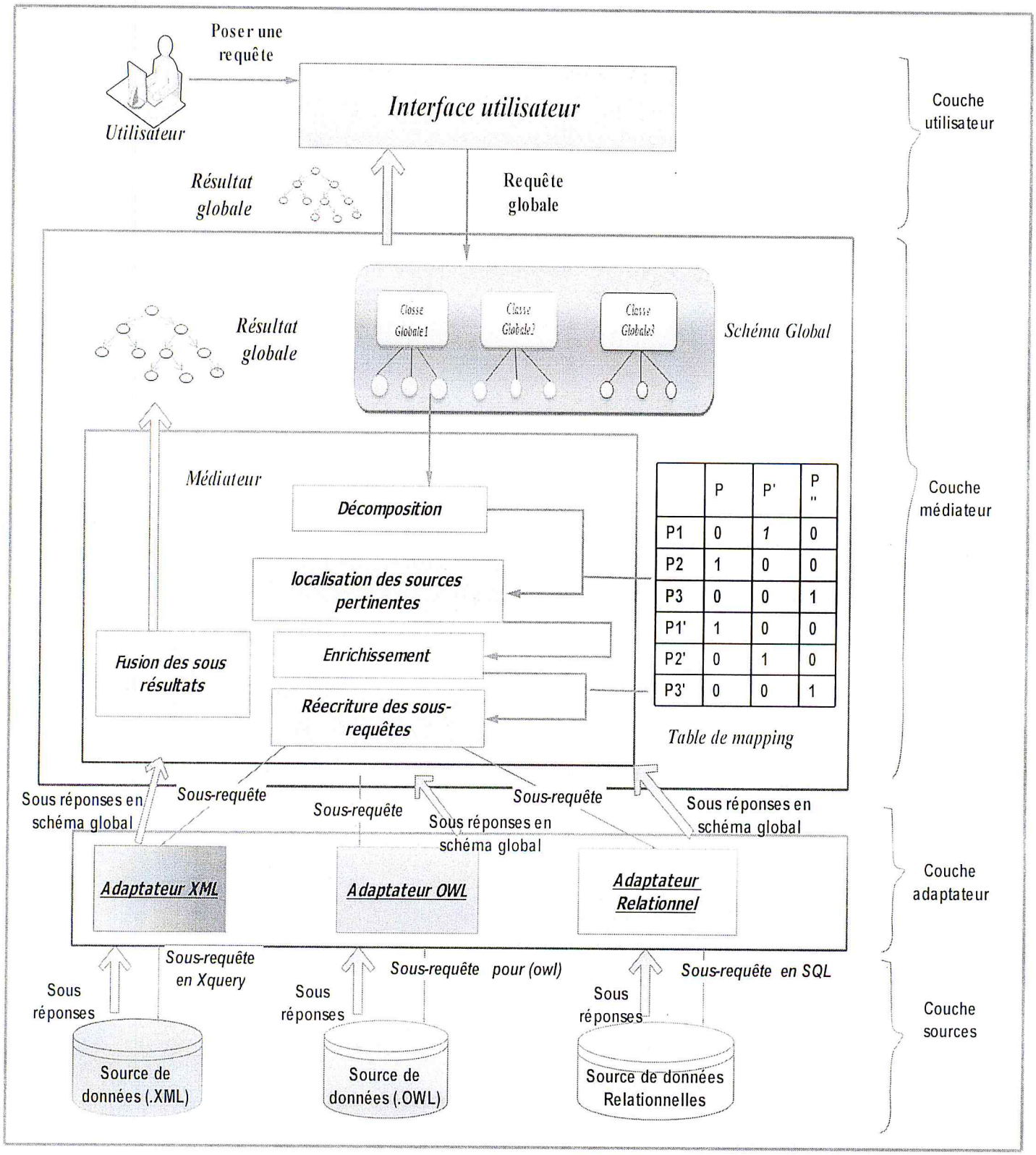


Figure33:Architécture détaillée du système.

Chapitre IV: Analyse des besoins et conception

Dans les sections suivantes de ce chapitre, nous allons nous intéresser à chaque module du système individuellement afin d'apporter plus amples explications.

- ***Architecture détaillée du système pour la génération des mappings***

Module1 : « Module de traduction des schémas sources locales en OWL »

Ce module est chargé à faire la conversion des schémas sources locales en (.OWL).

Pour ce faire le module est censé à faire l'extraction d'un schéma OWL pour chaque source locale. (Source relationnelle, source semi structurée (.XML), source de connaissances (.OWL)

L'algorithme traduction des schémas sources locales

- **Algorithme : Conversion des schémas sources locales en (.OWL) pour la source Relationnelle**

Entrée : Schéma local Relationnel
Sortie : Fichier RelationnelOwl.owl
Début
Connecter à la source de données relationnelles
Créer un fichier (RelationnelOwl.owl)
Si connexions établit **alors**
Début
Rechercher dans la base de données
Extraire les métadonnées et mettre les dans l'ensemble E
Pour chaque Element e de l'ensemble E
Si e = nom d'une table **Alors**
Ajouter e au fichier RelationnelOWL.owl autant que classe
Si e = clé étrangère **Alors**
Ajouter e au fichier RelationnelOWL.owl autant que objectProperty
Si e = Attribut **Alors**
Ajouter e au fichier RelationnelOWL.owl autant que DataTypeProperty
Ecrire fichier(RelationnelOWL.owl)
FinSi
Fin

- **Algorithme : Conversion des schémas sources locales en (.OWL) pour la source (.XML)**

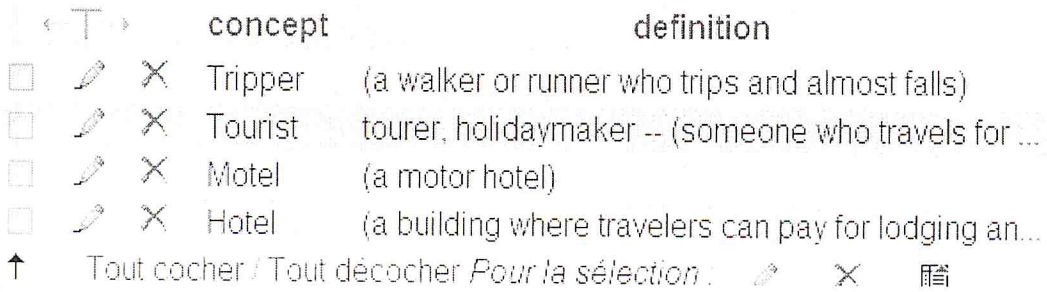
Entrée : schéma local XML
Sortie : Fichier (XML_OWL.owl)
Début
Lire Fichier(TouristData.xml)
Créer un fichier (TouristData.owl)

Rechercher dans le fichier (TouristData.xml)
Extraire les Fils direct de la racine
Extraire les Fils des Fils Direct de la racine
Pour chaque Element Fils des Fils Direct de la racine
Créer des classes (Fils des Fils Direct de la racine)
Ajouter les classes au fichier (TouristData.owl)
Pour chaque Element Fils des Fils Direct de la racine
Extraire ses petits-fils et leurs attributs s'ils existent
Si attribut (petits-fils) ==ForeignKey
Créer ObjectProperty (petits-fils)
Sinon
Créer dataTypeProperty (petits-fils)
Ecrire Fichier(RelationnelOWL.owl)

Module2 : « Module d'annotation des concepts localaux »

Le but de ce module est d'associer à chaque concept ou propriété appartenant aux schémas locals une défintion, donc l'expert séléctionne le mot qu'il désire annoter, ensuite il choisit la définition la plus apropriée en se basant sur un thésaurus nommé (WordNet).

Les informations envoyée par l'expert sont stockées dans une une table nommée « Annotation ».un exmple est illustré dans la figure suivante :



concept	definition
Tripper	(a walker or runner who trips and almost falls)
Tourist	tourer, holidaymaker -- (someone who travels for ...
Motel	(a motor hotel)
Hotel	(a building where travelers can pay for lodging an...

Tout cocher / Tout décocher *Pour la sélection* :

Figure 34:table annotation

Module3 : « Module d'extraction des relations sémantiques »

Le but de ce module est d'extraire tout les concepts et les propriétés annotées qui possèdent la même définitions .Pour ce faire le module doit établir une recherche dans la table annotation et extraire les definitions des concepts (classes et propriètés).

Module4 : « Module de regroupement »

Lors de cette étape notre système regroupe les concepts et propriétés qui sont similaires en plusieurs groupes ; puis le module doit stocker les relations extraites dans une base de données nommée « relation sémantique » et il insère les propriétés et les classe locales similaires dans une table crée dynamiquement puis, il choisit aléatoirement un représentant pour chaque groupe. Les figures suivantes montrent un exemple :

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
classesLocales	varchar(40)	latin1_swedish_ci		Non			
propL1	varchar(40)	latin1_swedish_ci		Non			
propL2	varchar(40)	latin1_swedish_ci		Oui	*NULL*		
propL3	varchar(40)	latin1_swedish_ci		Oui	*NULL*		
propL4	varchar(40)	latin1_swedish_ci		Oui	*NULL*		
propL5	varchar(40)	latin1_swedish_ci		Oui	*NULL*		
propL6	varchar(40)	latin1_swedish_ci		Oui	*NULL*		

Tout cocher / Tout décocher *Pour la sélection* :

Figure 35 : schéma relationnel des relations sémantiques

classesLocales	propL1	propL2	propL3	propL4	propL5	propL6
house	idhouse	name	room_number	maxperson	idisland	*NULL*
menage	idhouse	label	island	idHouse	maxperson	room_number

Tout cocher / Tout décocher *Pour la sélection* :

Figure 36: exemple1 des relations sémantiques

classes	propL1	propL2	propL3	propL4	propL5	propL6
booking	idHouse	idTourist	week	*NULL*	*NULL*	*NULL*
Allocation	id_room	idtripper	date_end	date_start	duration	price
reservation	id_house	id_tourist	week	*NULL*	*NULL*	*NULL*

Tout cocher / Tout décocher *Pour la sélection :*

Figure 37: exemple2 des relations sémantiques

Module5 : « Module de calcul de similarité »

Lors de cette étape, nous allons calculer les différentes mesures de similarité qui vont nous permettre de faire les correspondances entre les concepts et propriétés globales et les concepts et propriétés locales.

Ceci s'effectuera en trois étapes :

- ***Calcul de similarités terminologiques***

Les mesures à base de comparaison syntaxique et lexicale sont appliquées sur des couples de concepts afin de mesurer leur similarité terminologique.

Il existe plusieurs mesures de similarité terminologique, Dans notre système on va utiliser la distance de Levenshtein pour la similarité syntaxique et WordNet pour la similarité lexicale.

- *Similarité syntaxique:* **la distance de Levenshtein** est une distance qui compare deux chaines de caractères $s1$ et $s2$ en calculant le nombre de suppression, insertion et substitution requises pour transformer $s1$ à $s2$.

La distance de Levenshtein entre deux chaînes s1 et s2 est définie par :

$$\text{Sim}=1-\frac{LD(s1,s2)}{\max(|s1|,|s2|)}$$

Où LD (s1, s2) est la distance de *Levenshtein* entre deux chaines de caractères **[16]**.

Similarité lexicale: Les mesures de similarité lexicales ou aussi appelé les mesures de similarité sémantiques qui prennent en considération le sens du mot.

WordNet est une ressource lexicale de langue anglaise, qui regroupe des termes (noms, verbes, adjectifs et adverbes) en ensembles de synonymes appelés synsets. Un synset regroupe tous les termes dénotant un concept donné. Les synsets sont reliés entre eux par des relations sémantiques: relation de généralisation / spécialisation, relation composant/composé. Les techniques basées sur les chaînes de caractères ne sont pas suffisantes quand les concepts sont sémantiquement proches et quand leurs noms sont différents, l'interrogation d'une ressource linguistique telle que WordNet peut indiquer que les concepts sont similaires [17].

- ***Calcul de similarités structurelles***

Les techniques structurelles consistent à exploiter la structure de l'ontologie à comparer, souvent représentées sous forme de graphes, et la comparaison de similarité entre deux entités de deux ontologies peut être basée sur la position des entités dans leurs hiérarchies. Ces techniques sont basées sur l'hypothèse suivante « si deux entités de deux ontologies sont semblables, leurs voisins le sont également d'une certaine façon ».

Module5 : « Module de génération de la table de mapping »

En se basant sur le résultat obtenu par le module de calcul de mesures de similarité et sur la table des représentants, en premier temps une table de correspondances est générée dont les colonnes représentent les classes globales et les lignes représentent les représentants. La table de correspondances est remplie de la façon là où il y'a une similarité entre les concepts la valeur de la case est :

correspondance [Représentant][Classes Globales] == 1 , sinon
correspondance[Représentant][Classes Globales] == 0 .

Puis en se basant sur la table des représentants et la table de correspondance une table de mapping est générer pour chaque case correspondance [Représentant][Classes Globales]==1

Dont les lignes représentent les propriétés de la classe locales et les colonnes représentent les propriétés des classes globales associées aux représentants.

Après avoir calculé les mesures de similarité entre les propriétés de chaque classe globale et les propriétés de chaque classe locale la table de mapping peut être remplie.

S'il existe une similarité entre propriété globale et propriété locale alors la case de la table de mapping prend la valeur :

mapping [propriété locale][propriété globale] == 1 sinon la valeur de la case est égales à 0.

La figure suivante illustre les étapes qu'on vient de décrire :

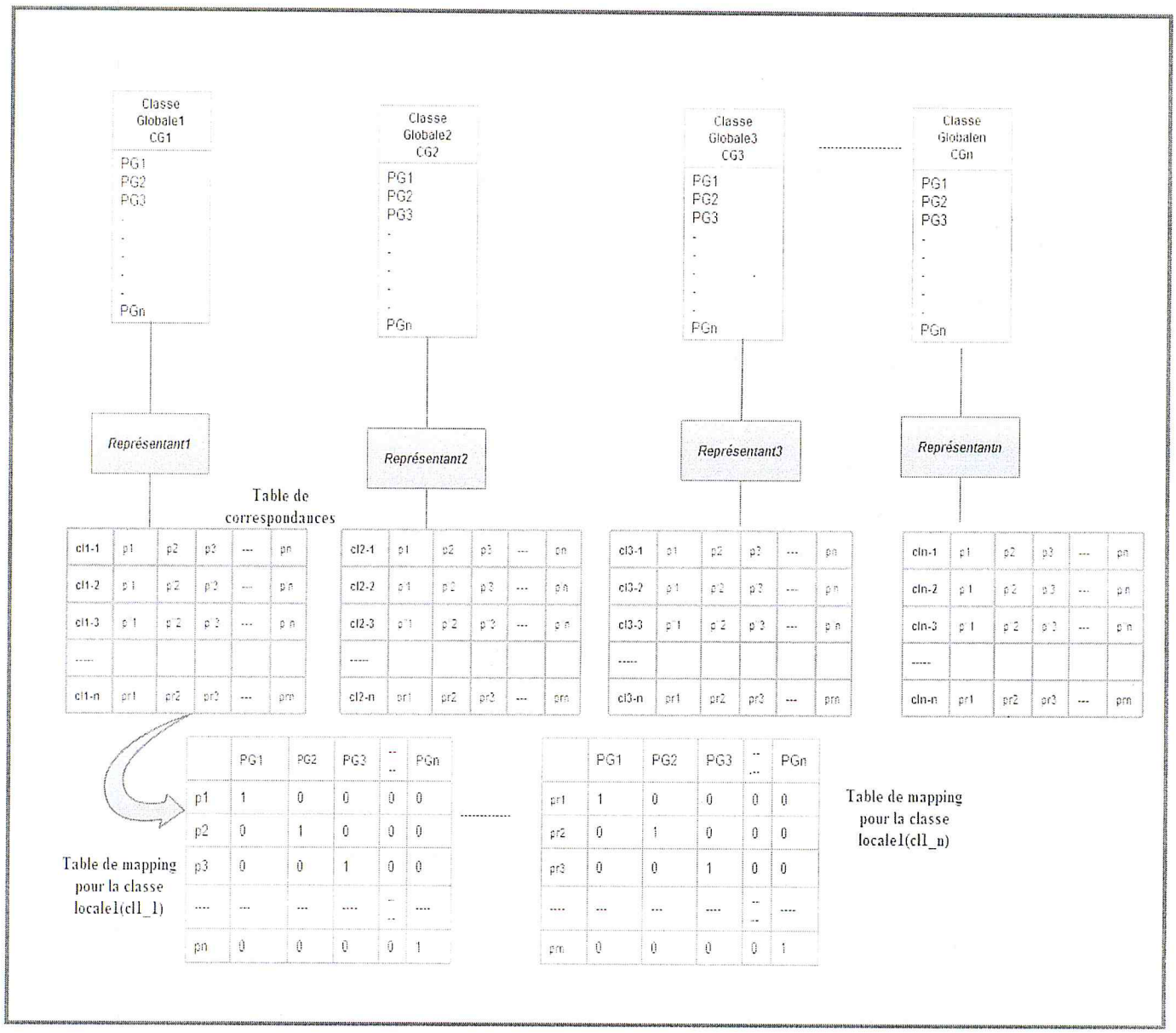


Figure 38 : Description du processus de génération de la table de mapping

Chapitre IV: Analyse des besoins et conception

Notre système à une architecture à 4 couches, de bas en haut commençons par :

❖ ***Couche sources de données***

Le domaine des sources de données qu'on a choisi s'inscrit dans le domaine de tourismes, nous disposerons de plusieurs sources hétérogènes de différentes natures et structure.

Les sources de données traitées sont :

➢ ***Les bases de données relationnelles :***

Dans ce modèle, les données sont représentées par des tables, sans préjuger de la façon dont les informations sont stockées dans la machine. Les tables constituent donc la structure logique du modèle relationnel. Au niveau physique, le système est libre d'utiliser n'importe quelle technique de stockage (fichiers séquentiels, indexage, adressage dispersé, séries de pointeurs, compression, ...) dès lors qu'il est possible de relier ces structures à des tables au niveau logique. Les tables ne représentent donc qu'une abstraction de l'enregistrement physique des données en mémoire.

Le succès du modèle relationnel auprès des chercheurs, concepteurs et utilisateurs est dû à la puissance et à la simplicité de ses concepts. En outre, contrairement à certains autres modèles, il repose sur des bases théoriques solides, notamment la théorie des ensembles et la logique des prédicats du premier ordre.

Les objectifs du modèle relationnel sont de:

- Proposer des schémas de données faciles à utiliser ;
- Améliorer l'indépendance logique et physique ;
- Mettre à la disposition des utilisateurs des langages de haut niveau ;
- Optimiser les accès à la base de données ;
- Améliorer l'intégrité et la confidentialité ;
- Fournir une approche méthodologique dans la construction des schémas.

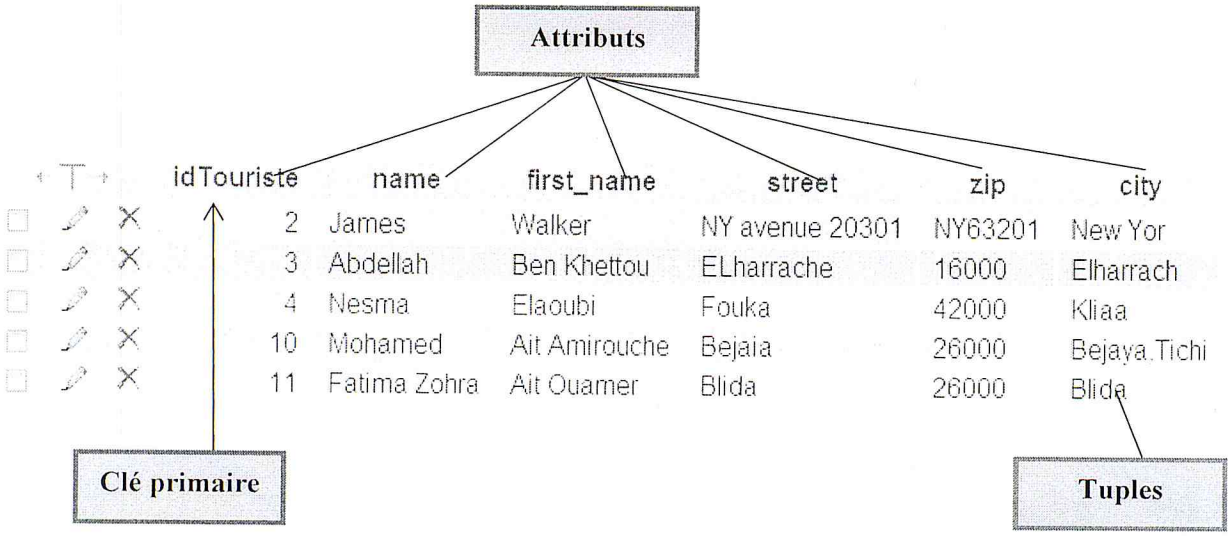


Figure39: table tourist

idhouse	name	room_number	maxperson	idisland
1	paradise House	110	3	1
2	hell house	20	3	1

Figure40: table house

- ***Les sources de données semi structurées :***

Les données semi structurées sont les données qui possèdent une structure flexible et qui n'ont pas un schéma à priori mais plutôt dont le schéma peut être extrait à partir de la données. La plupart du temps, un ensemble de données semi structurées est représenté sous la forme d'un graphe dont les feuilles contiennent les données et dont les nœuds et les liens représentent la structure de l'ensemble.

- *Le Modèle XML*

XML a été mis au point par le XML Working Group sous l'égide du World Wide Web Consortium (W3C) dès 1996. Depuis le 10 février 1998, les spécifications XML 1.0 ont été reconnues comme recommandations par le W3C, ce qui en fait un langage reconnu.

Chapitre IV: Analyse des besoins et conception

XML (entendez eXtensible Markup Language et traduisez Langage à balises étendu, ou Langage à balises extensible) est en quelque sorte un langage HTML amélioré permettant de définir de nouvelles balises. Il s'agit effectivement d'un langage permettant de mettre en forme des documents grâce à des balises.

XML peut être considéré comme un métalangage permettant de définir d'autres langages, c'est à- dire définir de nouvelles balises permettant de décrire la présentation d'un texte XML est à la fois un format de description des données et un conteneur de ses données. Il a trouvé deux utilisations principales: en tant que format de stockage et comme format de transmission entre différentes applications.

Note source de données (xml) est nommée (TouristData.xml).La figure suivante présente un bout du schéma de la source :

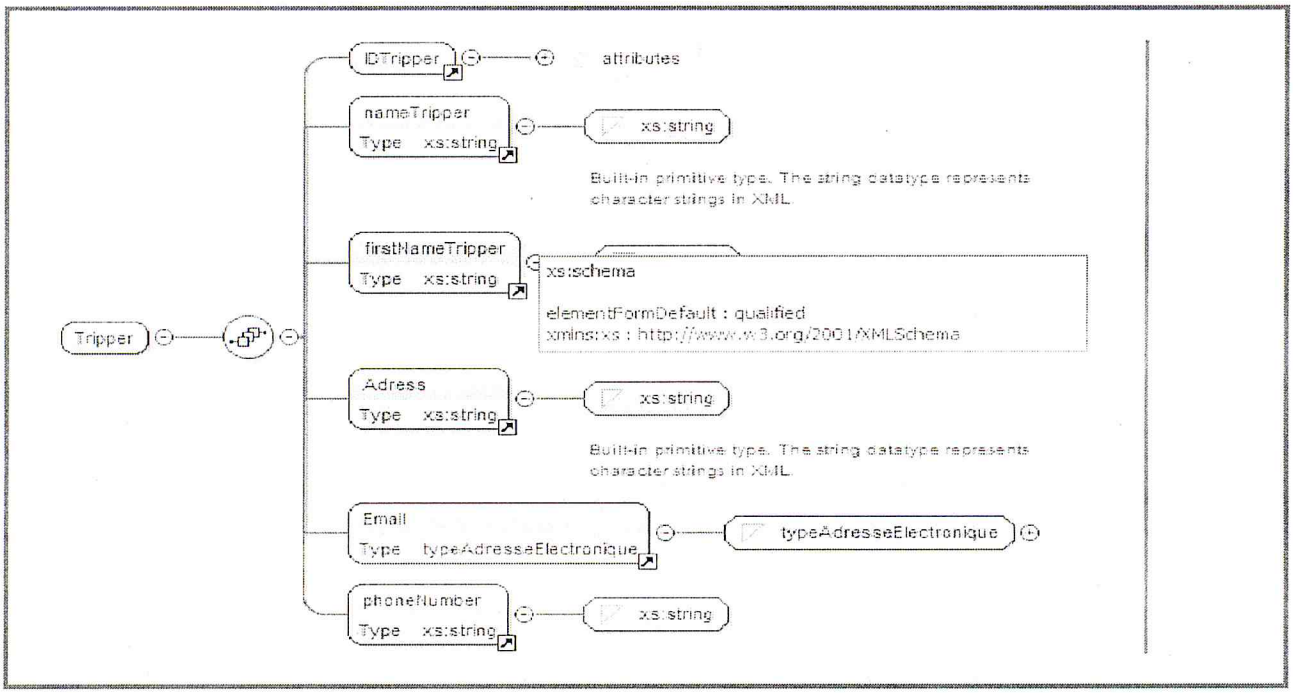
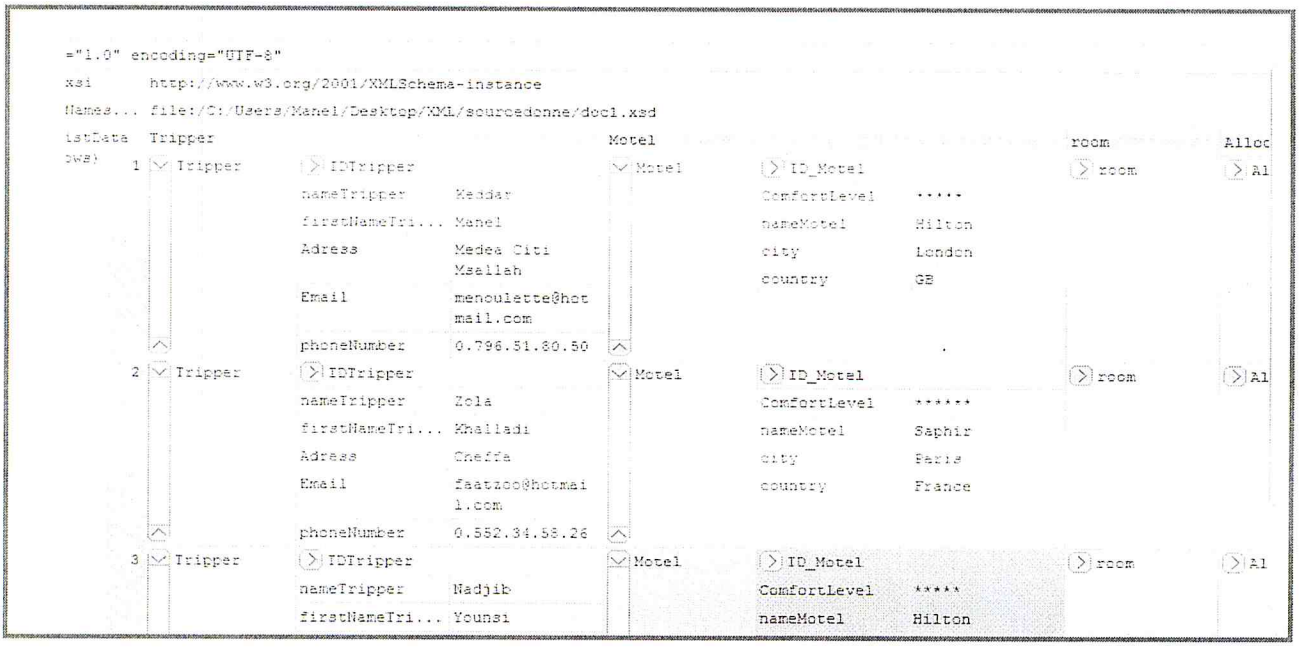


Figure 41 : XML schéma avec l'éditeur Oxygène



="1.0" encoding="UTF-8"

xsi http://www.w3.org/2001/XMLSchema-instance

Names... file:/C:/Users/Manel/Desktop/XML/sourcedonne/doc1.xsd

	istData	Tripper			Motel			room	Alloc
	1	Tripper	IDTripper		Motel	ID_Motel		room	Al
			nameTripper	Keddar		ComfortLevel	*****		
			firstNameTri...	Manel		nameMotel	Hilton		
			Adress	Medea Citi Msallah		city	London		
						country	GB		
			Email	menoulette@hotmail.com					
			phoneNumber	0.796.51.80.50					
	2	Tripper	IDTripper		Motel	ID_Motel		room	Al
			nameTripper	Zola		ComfortLevel	******		
			firstNameTri...	Khalladi		nameMotel	Saphir		
			Adress	Cheffa		city	Paris		
			Email	faatzoo@hotmail.com		country	France		
			phoneNumber	0.552.34.58.26					
	3	Tripper	IDTripper		Motel	ID_Motel		room	Al
			nameTripper	Nadjib		ComfortLevel	*****		
			firstNameTri...	Younsi		nameMotel	Hilton		

Figure 41 :Document XML avec l'éditeur Oxygène

- **Fragment de document XML**

```
<?xml version="1.0" encoding="UTF-8"?>
<TouristDataRoot xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation="file:/C:/Users/Manel/Desktop/XML/sourcedonn
e/doc1.xsd">
    <touristData>
        <Tripper>
            <IDTripper  nature="PrimaryKey" >1</IDTripper>
            <nameTripper>Keddar</nameTripper>
            <firstNameTripper>Manel</firstNameTripper>
            <Adress>Medea Citi Msallah</Adress>
            <Email>menoulette@hotmail.com</Email>
            <phoneNumber>0.796.51.80.50</phoneNumber>
        </Tripper>
        <Motel>
            <ID_Motel nature="PrimaryKey">H100</ID_Motel>
            <ComfortLevel>*****</ComfortLevel>
            <nameMotel>Hilton</nameMotel>
            <city>London</city>
            <country>GB</country>
        </Motel>
        <room>
            <ID_room nature="PrimaryKey">R1_1</ID_room>
            <ID_Motel nature="ForeignKey" Range="Motel">H100</ID_Motel>
        </room>
        <Allocation>
            <IDTripper nature="ForeignKey" Range="Tripper">1</IDTripper>
            <ID_room nature="ForeignKey" Range="Room">R1_1</ID_room>
            <ID_Motel nature="ForeignKey" Range="Motel">H100</ID_Motel>
            <duration>P5M</duration>
            <date_start>2011-03-11</date_start>
            <date_end>2011-08-11</date_end>
            <price unite="Euro">2000</price>
        </Allocation>
    </touristData>
```

Notre source de données est nommée (tourist.owl) voici un fragment de ce document :

```
<owl:ObjectProperty rdf:about="&tourist;idHouse">
    <rdfs:domain rdf:resource="&tourist;booking"/>
    <rdfs:range rdf:resource="&tourist;menage"/>
  </owl:ObjectProperty>
<owl:ObjectProperty rdf:about="&tourist;idTourist">
    <rdfs:domain rdf:resource="&tourist;booking"/>
    <rdfs:range rdf:resource="&tourist;tourist"/>
  </owl:ObjectProperty>
<owl:DatatypeProperty rdf:about="&tourist;Island">
    <rdfs:domain rdf:resource="&tourist;menage"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </owl:DatatypeProperty>
<owl:DatatypeProperty rdf:about="&tourist;city">
    <rdfs:domain rdf:resource="&tourist;tourist"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </owl:DatatypeProperty>
```

❖ *Couche adaptateur*

L'adaptateur cache l'hétérogénéité au médiateur. Il est associé à une seule source et joue le rôle d'intermédiaire entre cette source et le médiateur.

C'est un « Traducteur » qui fait :

- La Traduction de la sous requête envoyée par le médiateur en langage de requêtes des sources mères.
- L'extraction des résultats de la sous requête envoyé par le médiateur.
- L'envoie du résultat au médiateur.

➢ **Adaptateur Relationnel** : cet adaptateur est chargé de traduire la sous requête écrite en terme de schéma local relationnel en (SQL). Et de fournir un sous résultat.

➢ **Adaptateur XML** : cet adaptateur est chargé de traduire la sous requête écrite en terme de XMLs en (XQuery).

➢ **Adaptateur OWL :** est chargé d'interroger la source OWL pour extraire une réponse.

❖ *Couche Médiateur*

Le médiateur est chargé à effectuer le traitement de requêtes et la fusion des résultats pour ce faire il est décomposé en 5 modules :

- *Module1 : «Décomposition »*

Ce modulé est chargée à détecter les classes et les propriétés et décomposer la requête globale en deux parties une partie pour les classes globales et une partie pour les propriétés.

- *Module2 : « Localisation des sources pertinentes »*

En se basant sur la table de mapping ce module est censé à détecter les sources pertinentes pour chaque classe et propriétés.

- *Module3 : « Enrichissement »*

Ce module est censé à vérifier si l'utilisateur a sélectionné des classes qui ont des relations entre eux si ces relations existent le module les ajoutes.

- *Module4 : «Réécriture des sous requêtes »*

En se basant sur les sources pertinentes détectées ce module doit réécrire la requête globale en sous requête écrite en termes de schémas locaux de chaque source pertinente détectée.

- ***Module5 : «Fusion des sous résultats »***

Ce dernier est chargé à fusiner les sous résultats retournés par chaque adaptateur et retourne à l'utilisateur un résultat global.

➢ ***Schéma global***

La présence d'un schéma global est nécessaire puisqu'il fournit un vocabulaire unique servant à exprimer les requêtes des utilisateurs. Ce schéma unifie les schémas hétérogènes des sources à intégrer en se basant sur une description homogène, uniforme et abstraite du contenu des sources par des vues.

Notre schéma global est un fichier (. owl)

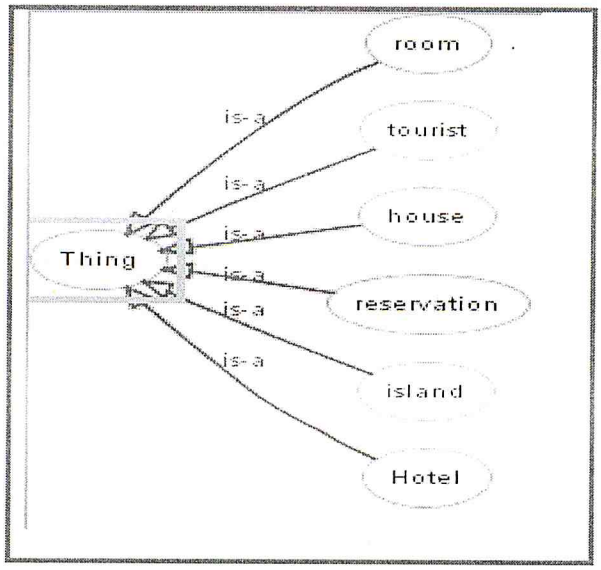


Figure 42: Classes globales

- **Fragment de document (owl) qui comporte le schema global**

```
//
    // Object Properties
    //
<!-- http://www.semanticweb.org/ontologies/2012/4/touristG.owl#idIsland -->


<owl:ObjectProperty rdf:about="#idIsland">
        <rdfs:domain rdf:resource="#house"/>
        <rdfs:range rdf:resource="#island"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/ontologies/2012/4/touristG.owl#idRoom --
>

    <owl:ObjectProperty rdf:about="#idRoom">
        <rdfs:domain rdf:resource="#reservation"/>
        <rdfs:range rdf:resource="#room"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/ontologies/2012/4/touristG.owl#idTourist
-->

    <owl:ObjectProperty rdf:about="#idTourist">
        <rdfs:domain rdf:resource="#reservation"/>
        <rdfs:range rdf:resource="#tourist"/>
    </owl:ObjectProperty>



    <!-- http://www.semanticweb.org/ontologies/2012/4/touristG.owl#idhotel -
->

    <owl:ObjectProperty rdf:about="#idhotel">
        <rdfs:range rdf:resource="#Hotel"/>
        <rdfs:domain rdf:resource="#room"/>
    </owl:ObjectProperty>
```

❖ *Couche utilisateur*

C'est une simple Interface de communication permet à un utilisateur de communiquer avec le système, Il envoie des requêtes au médiateur et reçoit des réponses. cette interface contient des champs de sélection qui aide l'utilisateur à sélectionner les éléments constituant la requête.

❖ *Exemple qui illustre les étapes de traitement d'une requête*

Etant donné le schéma global suivant :

La classe :Tourist

DatatypeProperty :idtourist

DatatypePropert :first_name

DatatypeProperty :name

DatatypeProperty :postCode

DatatypeProperty :street

La classe :reservation

DatatypeProperty :week

ObjectProperty :idRoom

ObjectProperty :idTourist

ObjectProperty :idHouse

La classe : Hotel

DatatypeProperty :confort

DatatypeProperty :city

DatatypeProperty :idHotel

La classe : room

DatatypeProperty :idroom

objectProperty :idHotel

Admettant que l'utilisateur veut connaitre les noms et la durée des touristes qui ont réservé dans des hôtels :

Pour ce faire l'utilisateur sélectionne la classe globale « tourist » puis la propriété « name » après la classe « hotel » et la propriété « idHotel » ensuite la classe « reservation » puis la propriété « week » et enfin il lance la requête globale.

Le système décompose la requête en deux parties :

Parties des classes contenant : « tourist », « hotel », « reservation ».

Parties des classes propriétés : « name », « idHotel », « week ».

Par la suite le système doit localiser les sources pertinentes en se basant sur la table de mapping =>

	Tourist
Tourer	1
Tourist	1
Tripper	1

	Hotel
Motel	1

	reservation
Reservation	1
Booking	1
Allocation	1

Puis le système doit vérifier pour toute combinaison de classe l'existences des relation entre ces dernières pour pouvoir rajouter des relations en cas ou l'utilisateur n'a pas sélectionner ces relations cette étape s'appelle l'enrichissement :pour notre cas nous distinguons l'existance d'une relation entre « réservation » et « tourist » donc le système rajoute à la partie des propriétés « idtourist ».

Ainsi qu'il y'a une relation entre la classe room et la classe Hôtel donc le système doit rajouter

La classe room dans la partie des classes et idroom dans la partie des propriétés

Puis en se basant sur la table de mapping le système doit réécrire la requête globale en sous requêtes écrite en termes de schéma locales

Classe :tourist

Classe global	*SourceRelationnelle*	*Source(owl)*	*Source(xml)*
idtourist	idTourer	idTourist	IDTripper
First_name	First_name	First_name	First_name
name	Name	name	nameTripper
postcode	Zip	postcode	?
Street	Street	street	Adress

Puis l'adaptateur doit traduire les sous requêtes écrites en terme locaux en langage d'interrogation de la source locale qui lui est appropriée.

Les sous requêtes générée sont :

Les sous requêtes (sql) :

Select week,name from reservation ,tourer join reservation on tourer.idTourer=reservation.idtourer

Les sous requêtes pour (.owl) :

[tourist,booking]

[idtourist]

Les sous requêtes (Xquery) :

```
Xquery version « 1.0 »,for $t in doc(' touristData.xml ')/TouristDataRoot

For $i in $t/touristData

For $j in $i/Tripper let $a :=$j/IDTripper  For $j in$i/room let $b :=$j/idroom for $j in $i/Allocation let $a :=$j/IDTripper  let $a :=$j/idroom

Return($a,$b)
```

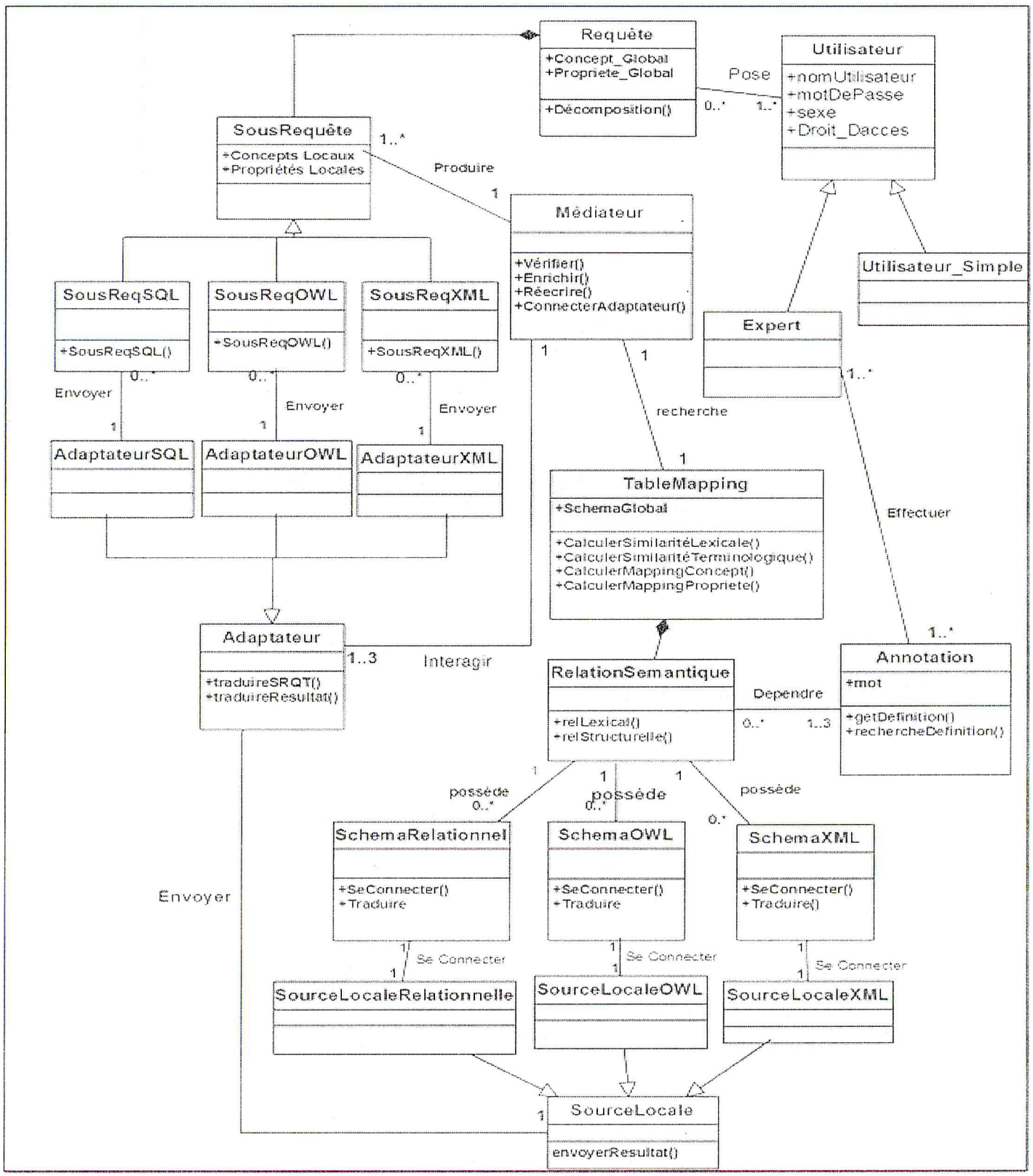


Figure 43:Diagramme de classe de notre système

- *La description des classes*

La classe utilisateur	
Les attributs	**La déscription**
• nomUtilisateur • mot_De_Passe • sexe • droit_D'acces	C'est la classe mère qui regroupe tous les type des utilisateurs.
La classe Expert	
Les attributs	**La description**
• nomUtilisateur • mot_De_Passe • sexe • droit_D'acces	C'est la classe qui contient les caractéristiques et taches appropriées à l'expert
La classe Utilisateur_Simple	
Les attributs	**La description**
• nomUtilisateur • mot_De_Passe • sexe • droit_D'acces	C'est la classe qui regroupe les autre utilisateurs qui veulent poser une requête.
Requête	
Les methodes	**La description**
Analyser()	Cette classe doit faire une analyse sur la requête Ensuite elle l'envoie au Mediateur.
SousRequêteSQL	
La description	Cette classe represente une sous requête avec les concepts et les propriétés du schéma relationnel.
SousRequêteOWL	
La description	Cette classe represente une sous requête avec les concepts et les propriétés du schéma OWL.
SousRequêteXML	
La description	Cette classe represente une sous requête avec les concepts et les propriétés du schéma XML.
AdaptateurSQL	

La description	Cette classe doit faire la traduction de la sous requête écrite e en terme du schéma relationnel en SQL.
AdaptateuOWL	
La description	Cette classe doit faire la traduction de la sous requête ecrite e en terme de la source OWL en SPARQL.
AdaptateurXML	
La description	Cette classe doit faire la traduction de la sous requête ecrite e en terme du schéma XML en XQUERY.
Mediateur	
Les méthodes	**La description**
Reecrire()	le médiateur doit réecrire la requête globale écrite en terme des concepts et des propiétés globales en utilisant les concepts et les propriétés locales extraites de la table de mapping
Enrichir()	Le médiateur doit rechercher une nouvelle relation sémantique trouvée entre les composants de la requête réécrite.
Verifier()	Le médiateur doit fai la validité de la requêter
connecterAdaptateur()	Le médiateur envoie la requête réécrite à l'adaptateur.
TableMapping	
Les attributs	**La description**
schemaGlobal	Pour établir les mapping il faut la présence du schéma global.
Les méthodes	**La description**
calculerSimilariteLexical()	Cette classe calcule une mesure de similarité lexicale entre le schéma global et les schéma locaux.
calculerSimilariteTreminologique()	Cette classe calcule une mesure de similarité terminologique entre le schéma global et les schéma locaux.
calculerMappingConcept()	Une fois trouvé deux concepts similaire, cette classe doit les inserer dans la table de mapping.
calculerMappingPropriete()	Une fois trouvé deux propriétés similaire, cette

	classe doit les inserer dans la table de mapping.
RelationSemantique	
Les méthodes	**La description**
relLexical()	Cette classe doit comparer toutes les definitions des concepts et des propriétés des schémas locaux et établir une relation sémantique entre ceux qui possédent la même défintion.
relStructurelle()	Cette classe doit parcourir le schéma relationnel pour vérifier s'il existe des relations entre les tables,et établie une relation structurelle entre les deux tables qui sont reliées.
Annotation	
les attributs	**La description**
Mot	Le mot selectionner par l'expert
Les méthodes	**La description**
getDefiniton()	La méthode qui recupere la definition
rechercherDefinition()	La méthode qui fait une recherche dans le dictionnaire wordnet pour avoir une définition.
SchemaRelationnel	
Les méthodes	**La description**
Se connceter()	Cette classe effectue une connection vers la base de données relationnelle locale.
Traduire()	Cette traduit la base de données en un schéma relationnel.
SchemaRelationnel	
Les méthodes	**La description**
Se connceter()	Cette classe effectue une connection vers la source locale.
Traduire()	Cette traduit la base de données en un schéma XML.

Tableau 9:Description du diagramme de classe.

7. Conclusion

Dans ce chapitre nous avons présenté une conception détaillée de notre système (GRHOM) générateur des requête d'un système de médiation en utilisant le langage UML, cette étude conceptuelle nous a permet de mettre en évidence les étapes nécessaires pour la création de ce dernier.

La prochaine étape consiste donc en la concrétisation du ce que nous avons proposé, en d'autres termes, la réalisation d'un système capable de générer automatiquement des requêtes de médiation avec la prise en compte de l'hétérogénéité des sources liée aux conflits sémantiques présentés dans ces sources.

Chapitre v:
Implémentation

1. Introduction

Après avoir effectué la conception de notre système, nous allons à présent entamer la réalisation de ce dernier.

2. Environnement de développement

2.1. Langages utilisés

2.1.1. Java

Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton de Sun Microsystems.

Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris).

Le langage Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage peut-être utilisé sur internet pour des petites applications intégrées aux pages web (applet) ou encore comme langage serveur (JSP).

2.1.2. MYSQL

MYSQL est un Système de gestion de bases de données relationnelles (SGBDR) sous licence GNU très utilisé pour mettre en ligne des bases de données.

Il permet d'entreposer des données de manière structurée (Base, Table, Champs, Enregistrements). Le noyau de ce système permet d'accéder à l'information entreposée via un langage spécifique le SQL.

2.2. OUTILS

2.2.1. Netbeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, JavaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java). Un environnement Java Development Kit JDK est requis pour les développements en Java.

NetBeans constitue par ailleurs une plate forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate forme.

2.2.2. Protégé

Protégé est un éditeur d'ontologie open source, disponible à l'adresse http://protege.standford.edu. Il a été développé au département d'Informatique Médicale de l'Université de Standford.

L'éditeur d'ontologie Protégé « version 3.2 beta » a été utilisé pour la création des sources (.owl) et le schéma global de notre système. L'objectif est de générer automatiquement le code OWL correspondant à la source. Il est à noter que « Protégé » offre beaucoup de fonctionnalités, qui n'ont pas été toutes utilisées dans le présent travail. Il est conçu principalement autour du concept des plugins.

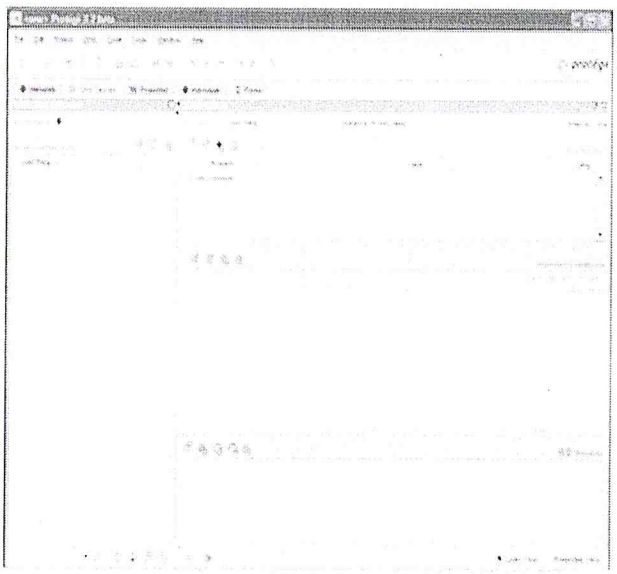


Figure 44:Interface Protégé

2.2.3.Oxygène

<oXygen> XML Editor 3.1. Il s'agit d'un logiciel offrant un certain nombre de fonctionnalités facilitant la saisie de code XML.

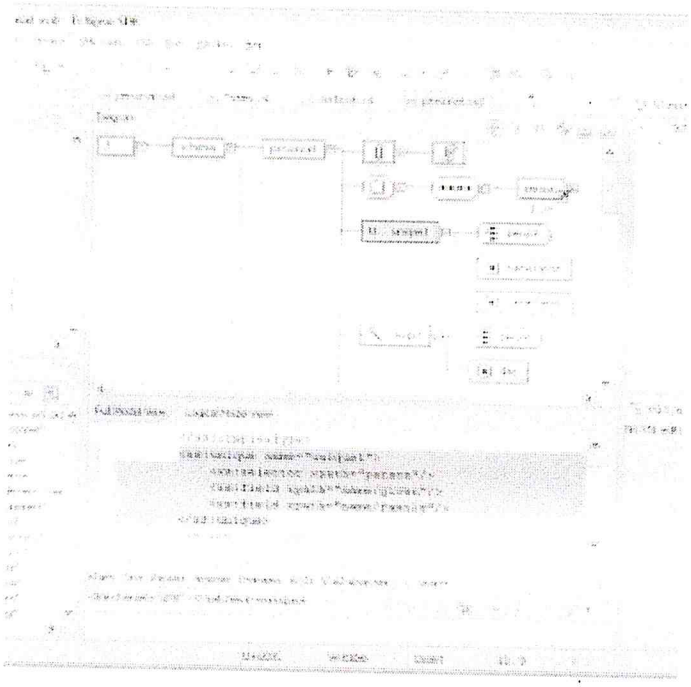


Figure45:Interface Oxygène

2.3. LES APIS

- Jena: JENA est un ensemble d'outils (une API) permettant de lire et de manipuler des ontologies décrites en OWL. Au cours de notre développement, on a utilisé la version Jena2.6 qui était la version la plus avancée. Et qui permettait entre autres : la création et l'extraction de concepts, de propriétés (relations et attributs) sur les concepts

- JTatoo : JTattoo est une librairie java distribuée en open source qui permet aux développeurs d'enjoliver leurs applications en leur offrant d'excellentes interfaces graphiques et ce à travers des thèmes visuels dits « Look and Feels » qui change du thème standard de Java. [8]

- JAWS : Comme son nom l'indique, JAWS (Java API for WordNetSearching) est une librairie Java qui permet de chercher et de trouver à travers la base dictionnaire WordNet (aussi bien la version 2.1 que la version 3.0) des définitions, des synonymes et antonymes … etc

JAWS a été créé par Brett Spell, adjoint membre du département CSE (computer Science and Engineering) à l'université américaine SouthernMethodist.

- JWS (Java WordNetSmilarity) : C'est une librairie Java distribué en open source qui permet de mesurer la similarité entre deux chaînes de caractère à l'aide de différentes

formules, dans notre application nous avons utilisé la distance de Wu & Palmer qui se basant sur la hiérarchie de WordNet.

3. Le système GRHOM

3.1. Diagramme d'accessibilité du système :

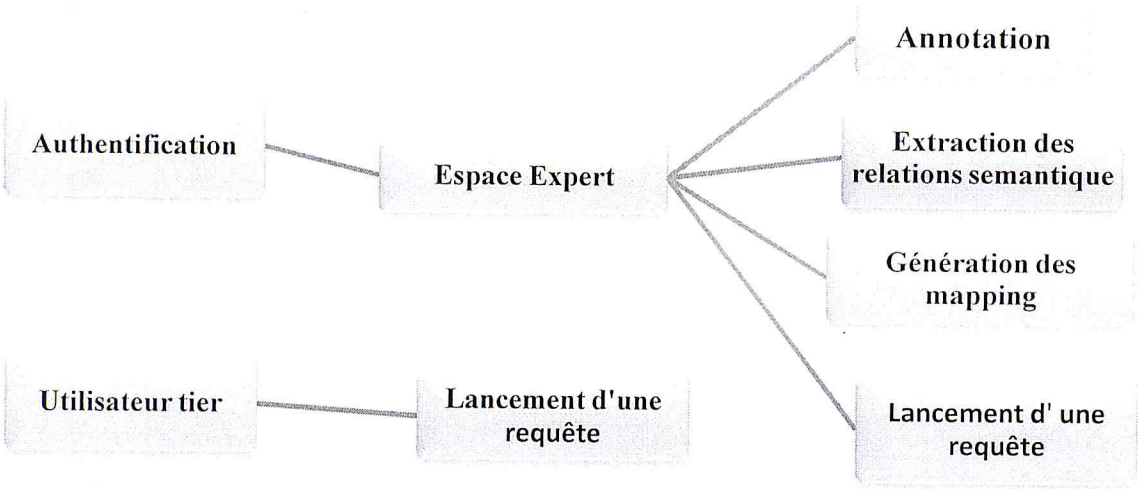


Figure46 : Diagramme d'accessibilité du système GRHOM.

3.2. Architecture de déploiement

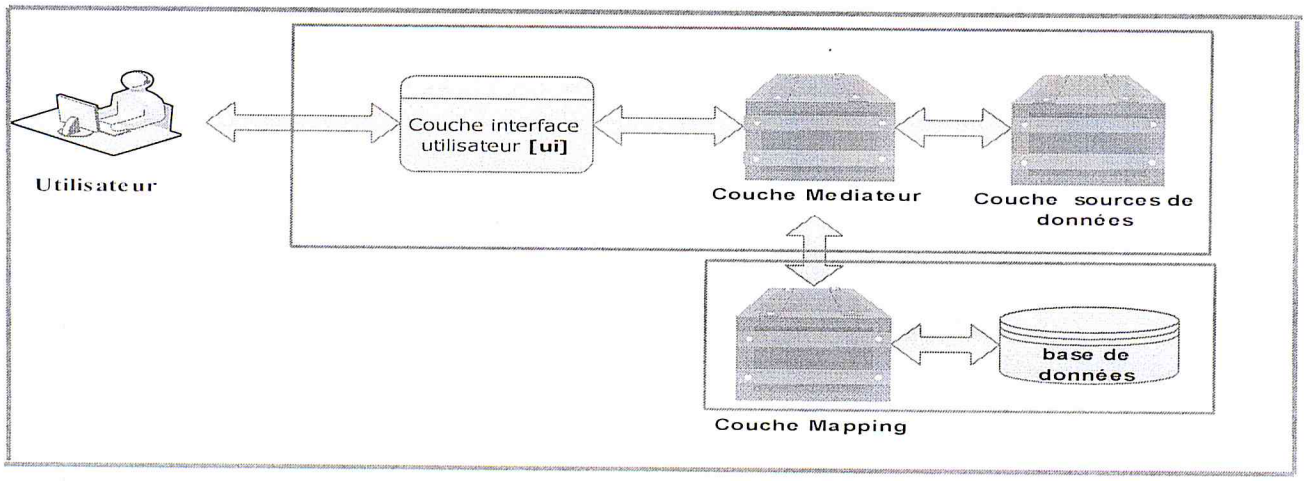


Figure47 : Architecture de déploiement

Pour la construction d'GRHOM nous avons utilisé l'architecture suivante :

- La couche [interface utilisateur] : s'occupe d'afficher l'interface graphique qui permet à l'utilisateur de piloter l'application et d'en recevoir des informations. Donc l'utilisateur envoie sa requête globale cette couche communique avec la couche médiateur.
- La couche [Médiateur] : cette couche s'occupe de recevoir la requête globale envoyée par l'utilisateur et d'effectuer le traitement de la requête et la génération des requêtes de médiations pour ce faire la couche médiateur doit communiquer avec la couche Mapping.
- La couche [Mapping] : cette couche s'occupe à générer des mapping, pour ce faire elle a besoin de stocker des informations sur les sources de données sur des bases de données crées dynamiquement.
- La couche [Sources de données] : cette couche s'occupe à traduire les sous requêtes envoyées par le médiateur et retourner les sous résultats.

La communication va de la gauche vers la droite :

- L'utilisateur fait une demande (envoie une requête globale) à la couche [interface utilisateur]
- Cette demande est mise en forme par la couche [interface utilisateur] et transmise à la couche [médiateur]
- Si pour traiter cette demande, la couche [médiateur] à besoin de communiquer avec la couche [Mapping] et au final des sous requête de médiation sont générée et envoyée à la couche [sources de données].
- La couche source de données retourne le résultat au médiateur ce dernier fusionne les résultats et les envoie à la couche [interface utilisateur] qui les visualise.
- Chaque couche interrogée rend sa réponse à la couche de gauche jusqu'à la réponse finale à l'utilisateur.

3.3.Modèle de déploiement

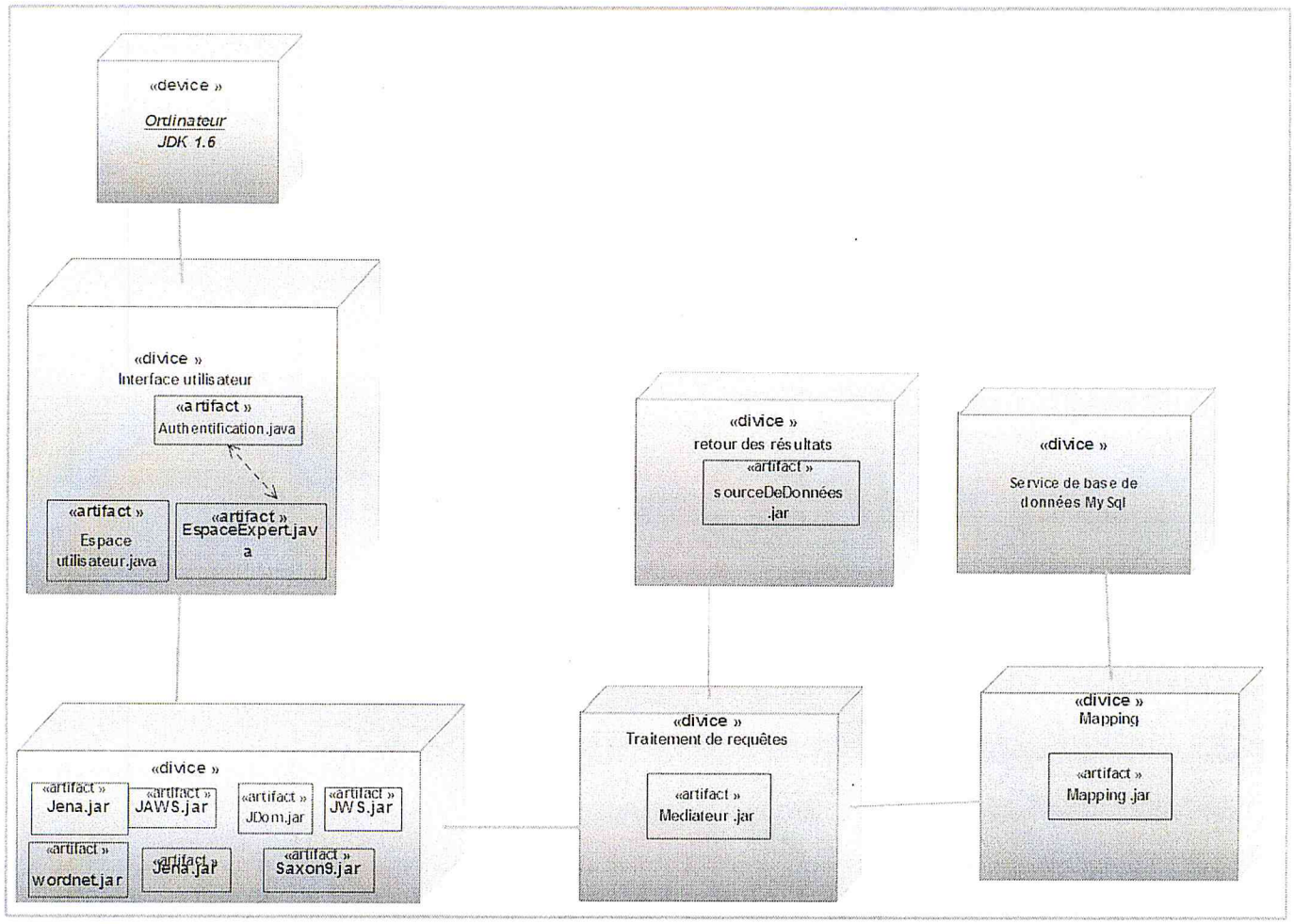


Figure48:Diagramme de déploiement pour le système GRMoH

4. Présentation de l'application

Dès qu'on lance le système, l'interface principale (Figure 49) s'affiche. Cette dernière permet à l'utilisateur de charger un schéma global avec ses sources locales et aussi de poser une requête. Si les sources n'ont pas été annotées par un expert, le système affiche un message d'erreur en lui demandant de contacter l'expert (Figure.51). Dans ce cas, l'expert doit s'authentifier pour accéder au système afin d'effectuer l'annotation(Figure 53), si l'expert ne possède pas de compte il doit s'inscrire (Figure 54).

Dés qu'un expert se connecte, une interface s'affiche (Figure 59) qui contient en plus un menu de traitement (Figure 60). Donc l'expert possède le droit de lancer le processus d'intégration et d'effectuer ou modifier une annotation (Figure 56).

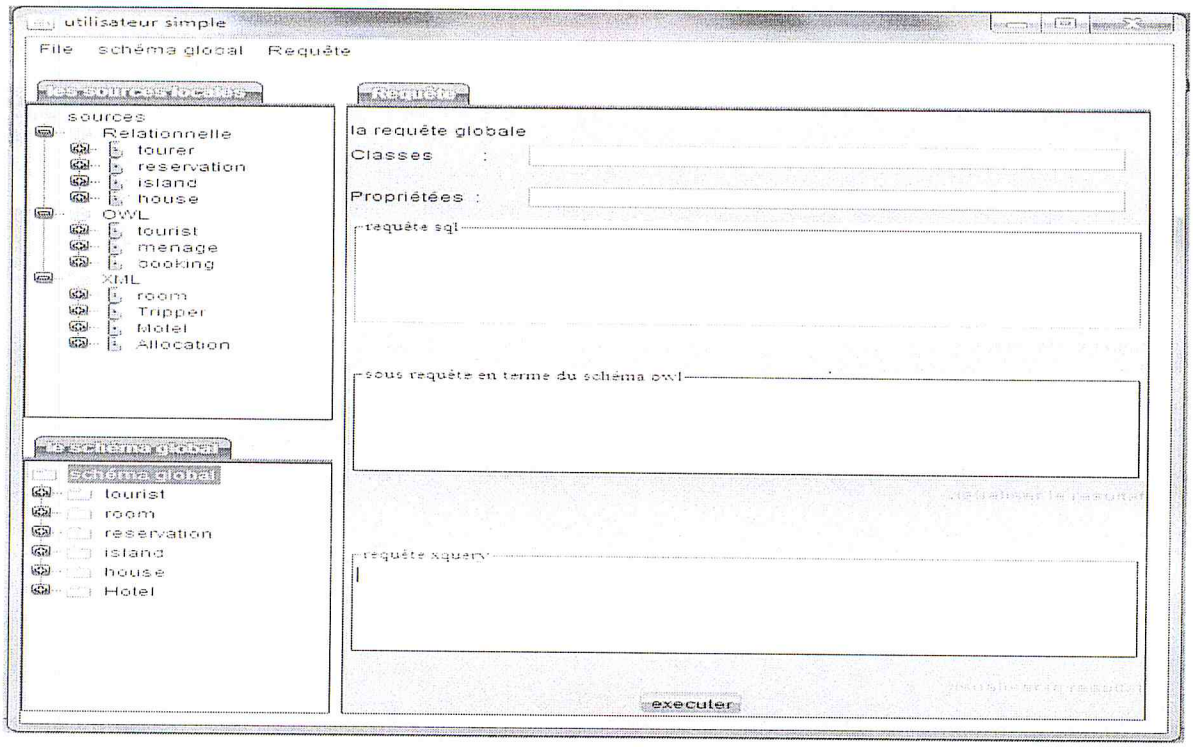


Figure49 : l'interface d'un utilisateur simple

Quand un utilisateur tente de charger un schéma global, il clique sur le menu suivant :

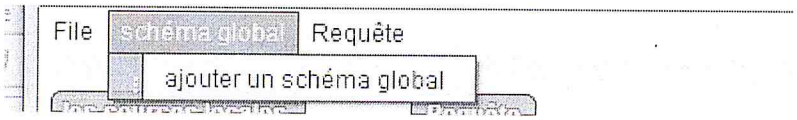


Figure50 : menu d'un utilisateur simlpe

Si les sources n'ont été annotées, le message d'erreur suivant s'affiche :

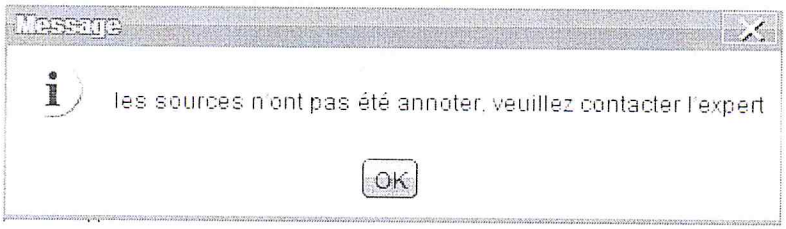


Figure51: message d'erreur pour les sources non annotées

L'expert s'authentification via le menu suivant :

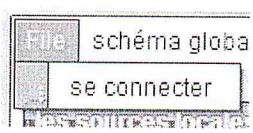


Figure52 : menu pour la connexion d'un expert

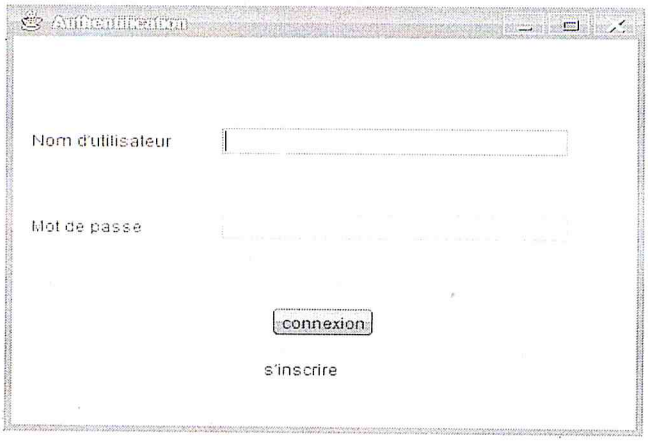


Figure53 : la boite d'authentification:

Lorsque l'utilisateur fait un click sur « s'inscrire » la fenêtre suivante s'affiche :

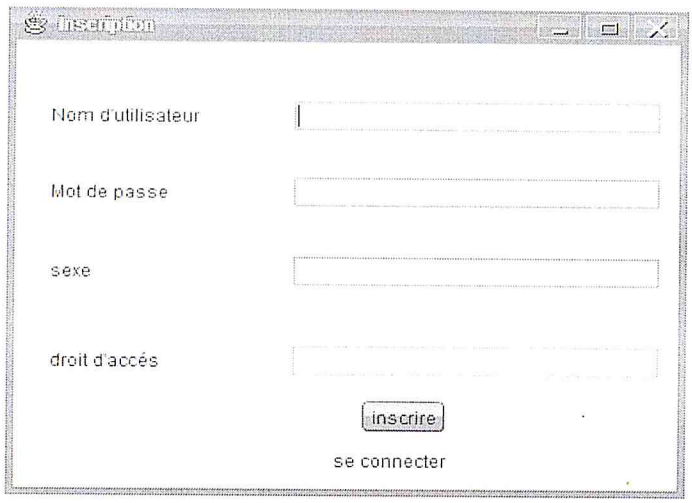


Figure 54 : l'interface d'inscription

Cette dernière permet à l'utilisateur de saisir ses informations personnelles et de valider l'inscription en cliquant sur « inscrire ».

Dans l'arbre ci-dessous, on distingue deux types de nœuds : les nœuds qui sont en bleu, cela signifie que le nœud n'a pas été annoté, et les nœuds en vert, cela signifie que le nœud a été annoté.

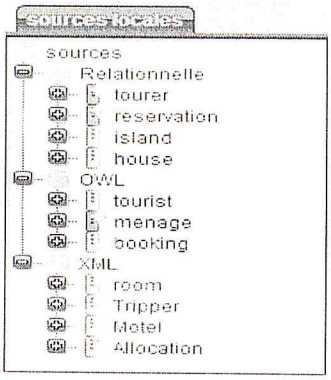


Figure55 : l'arbre des sources locales

L'expert a le choix soit d'effectuer une annotation soi de modifier une annotation en utilisant le menu suivant :

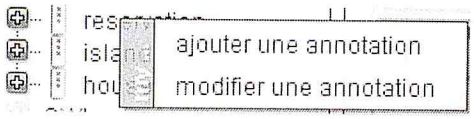


Figure 56 : menu d'annotation

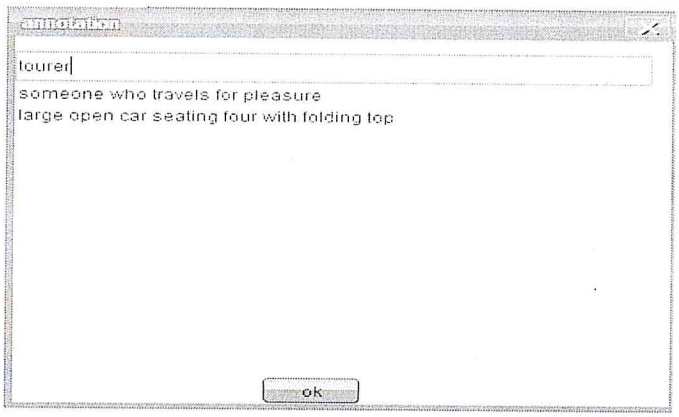


Figure 57: la boite de dialogue d'annotation

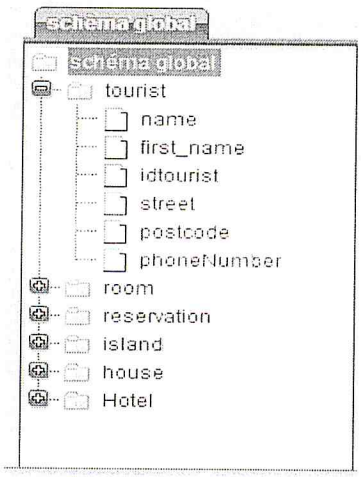


Figure 58: l'arbre du schéma global

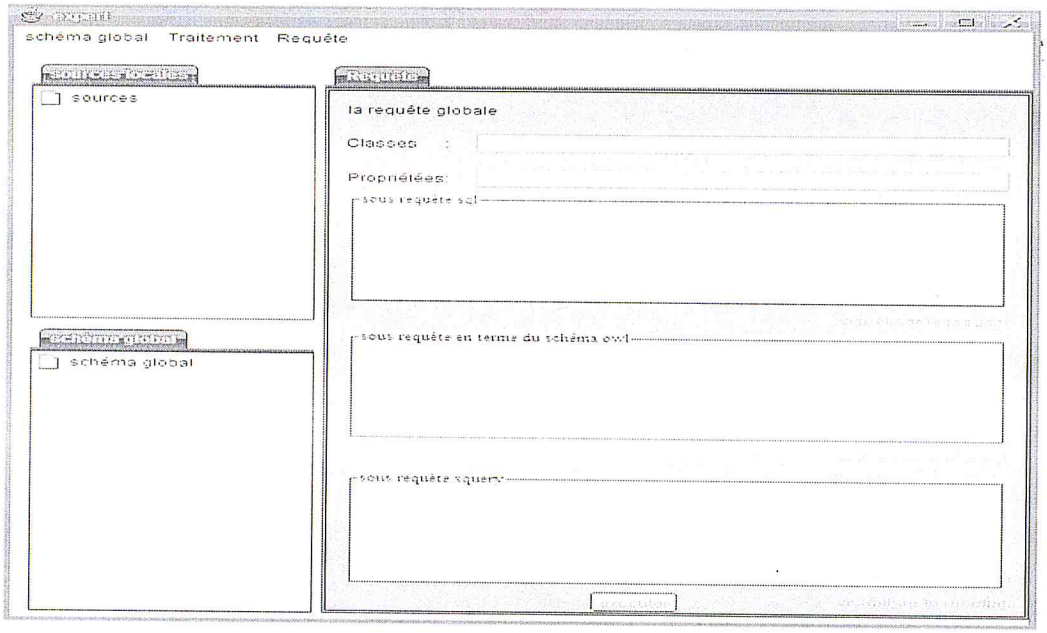


Figure59 : l'interface d'un expert

Un click sur le menu suivant permet de générer les relations sémantiques et la table de mapping.

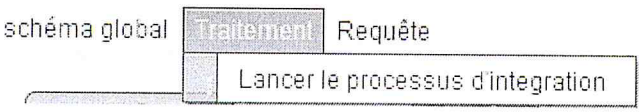


Figure60 : menu d'un expert

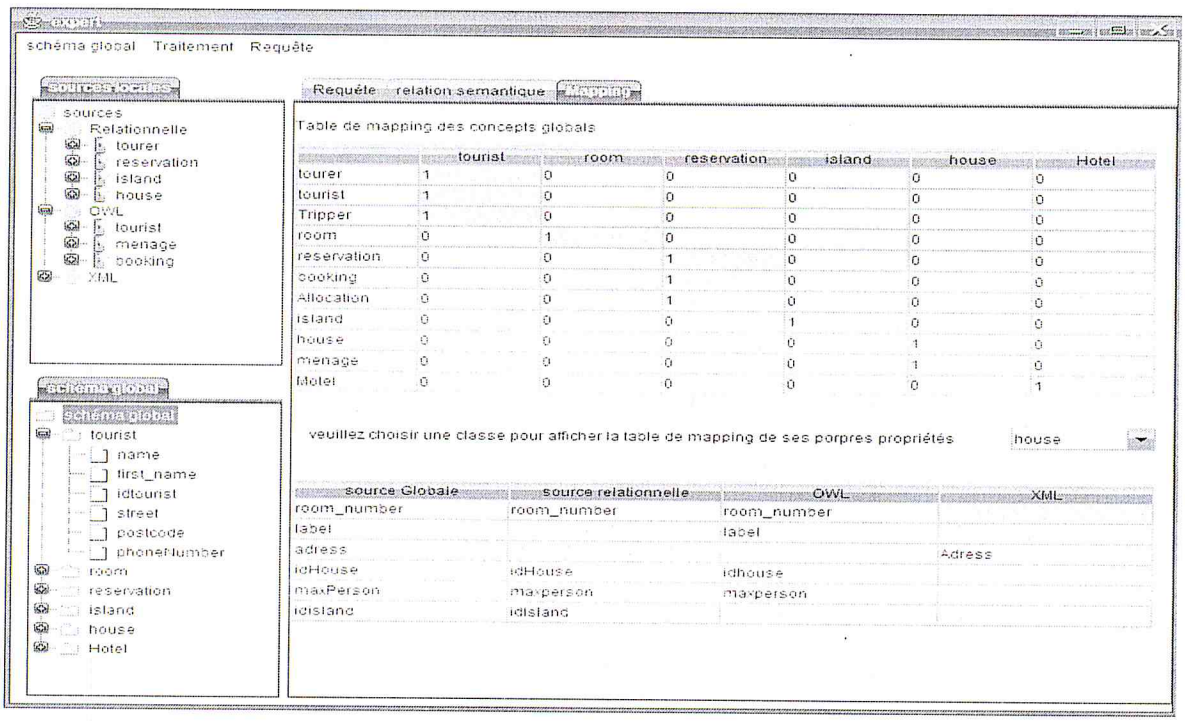


Figure 61 : l'interface qui comporte l'onglet de la table de mapping

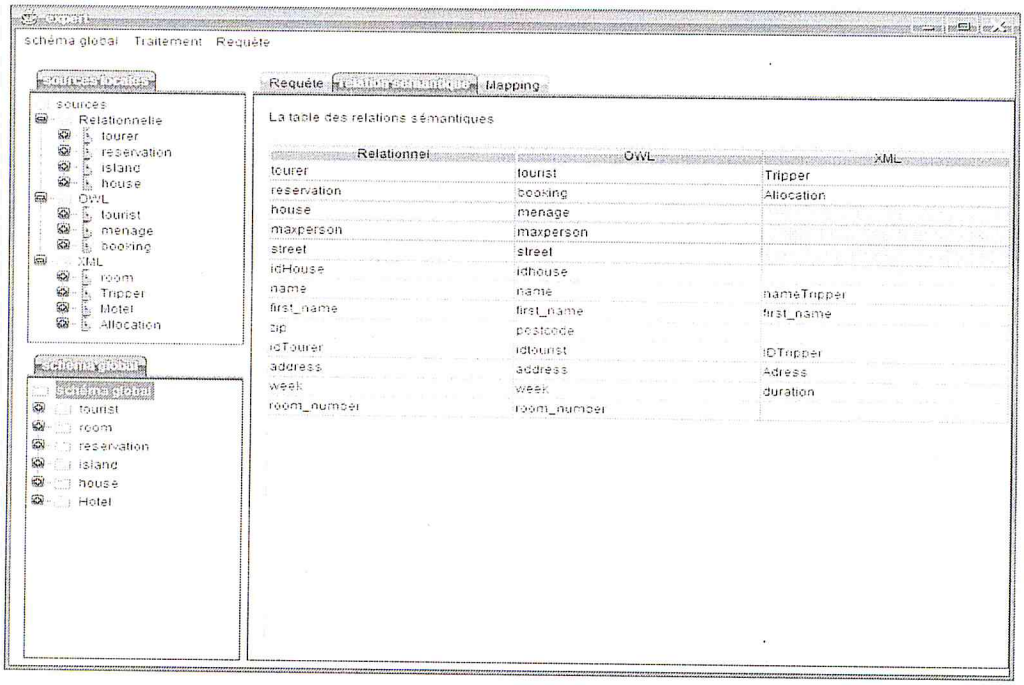


Figure 62 : l'interface qui comporte l'onglet de la table des relations sémantiques

Voici un exemple qui montre comment un utilisateur ou un expert peuvent poser une requête.

La requête globale est ensemble de concepts et propriétés globaux, ensuite notre système génère à partir de cet ensemble est sous requêtes locales.

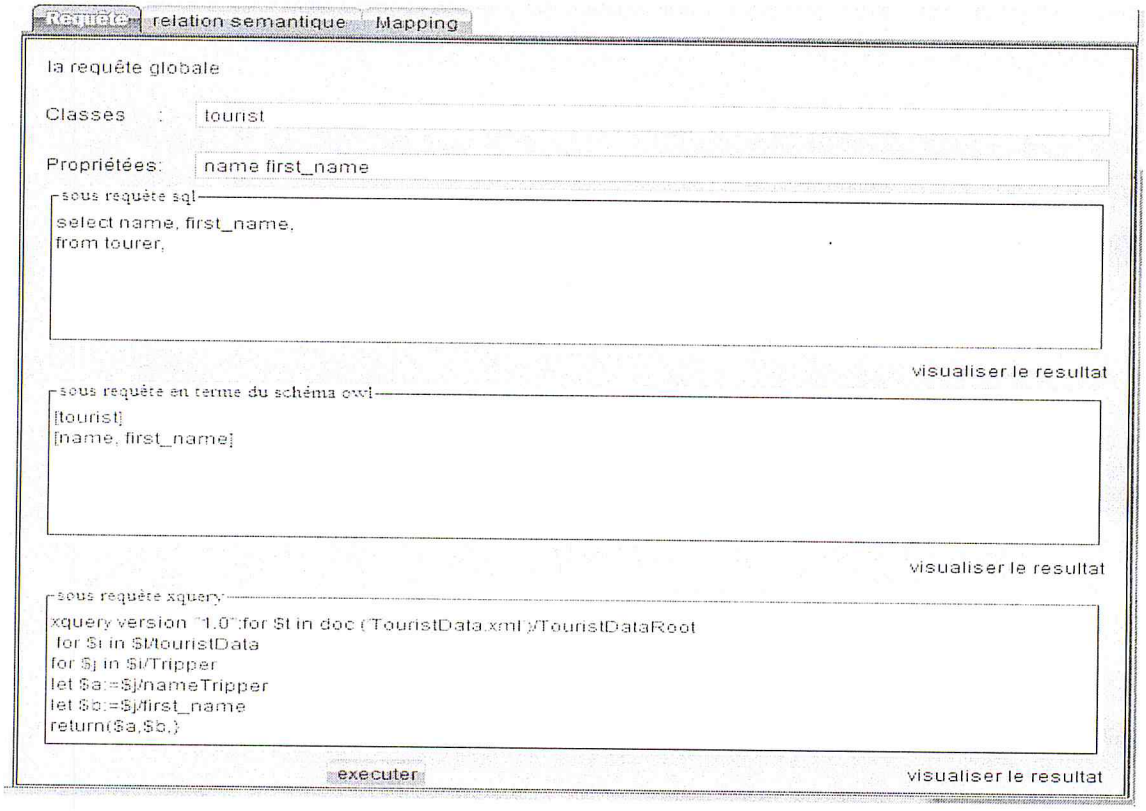


Figure63: la génération des sous requêtes locales

Chapitre VI:

Test et validation

Chapitre VI : Tests et validation

1. Introduction

D'après ce qui a été dit dans les chapitres précédant, notre système consiste à générer automatiquement des requêtes de médiation à partir d'une requête écrite en termes de schéma globale.Toute fois pour validernotre système, nous avons effectué un ensemble de requêtes manuelles générées par l'expert et un ensemble de requêtes automatiquement par notre système. Pour évaluer le degré de pertinence des requêtes obtenues, on doit procéder à leur évaluation. Et ce, en utilisant quelques mesures comme : précision, rappel, overall et f-mesure (dites mesures de performance du système).

2. Définition des mesures de performance utilisées

Les mesures utilisées pour *évaluer la qualité des requêtes produites par le système sont* principalement les mesures de calcul de la pertinence en recherche d'information, telles que la ***précision*** et le ***rappel***.

Le calcul de ces mesures**[18],** est basé sur la comparaison entre les requêtes produites par un système automatique qu'on appellera **S** et un ensemble de requêtes produites par un expert qu'on notera **H**.

- Les requêtes*correctes généré*par un système sont appelées « *the true positives (TP)* » et sont calculées ainsi :

$$TP = S \cap H$$

- Les correspondances *incorrectes trouvées* par un système sont appelées « the false positives (FP) » et sont calculées ainsi :

$$FP = S - S \cap H$$

- Les requêtes*correctes omises* par un système sont appelées « the false négatives (FN) » et sont calculées ainsi :

$$FN = H - S \cap H$$

- La *précision* est une mesure d'exactitude, elle varie entre [0,1] elle est calculée de la manière suivante :

$$Precision = \frac{|TP|}{|TP + FP|} = \frac{S \cap H}{S}$$

- Le rappel est une mesure de perfection, elle varie entre [0,1], elle est calculée de la manière suivante :

$$Rappel = \frac{|TP|}{|TP + FN|} = \frac{S \cap H}{H}$$

Les valeurs obtenues par le calcul des requêtes ne sont comparables par le biais de la précision et du rappel, en fait, *le rappel peut prendre des valeurs importantes aux dépens de la précision*, en retournant toutes les requêtes possibles.

En même temps, *la précision peut prendre des valeurs importantes aux dépens du rappel*, en retournant que les requêtes correctes cependant peu nombreuses.

- C'est pour ces raisons qu'il est préférable de prendre en considération les deux mesures simultanément via une mesure qui combine le rappel et la précision telles que : la *F-mesure* qui se calcule de la manière suivante :

$$F - Mesure = \frac{2 * (Rappel * Précision)}{Rappel + Précision}$$

La *F-mesure* est une mesure globale de la qualité des correspondances produites, elle varie entre [0,1], cette mesure alloue la même importance à la précision et au rappel.

- Une autre mesure de la qualité de l'alignement et qui combine le rappel et la précision : *l'overall*qui se calcule de la manière suivante :

$$OVERALL = Rappel(2 - \left(\frac{1}{Précision}\right))$$

L'overall peut prendre des valeurs négatives si le nombre de fausses requêtes (FP) trouvées par le système dépasse le nombre de requêtes correctes (TP) générées par le système.

Dans la plupart des cas ***l'overall*** est plus petit que le rappel et la précision, ce qui rend difficile d'atteindre un ***overall*** supérieur a**0.5**.

- Une mesure pour évaluer *le pourcentage d'erreurs* du système automatique est le *Fallout*qui se calcule de la manière suivante :

$$\boldsymbol{Fallout = \frac{FP}{FP + TP}}$$

3. Jeux de Données

Pour prouver la validité de notre système nous avons utilisé 10 requêtes puis nous allons comparer les sous requête générer manuellement avec les requête généré automatiquement.

3.1. Requêtes générer automatiquement :

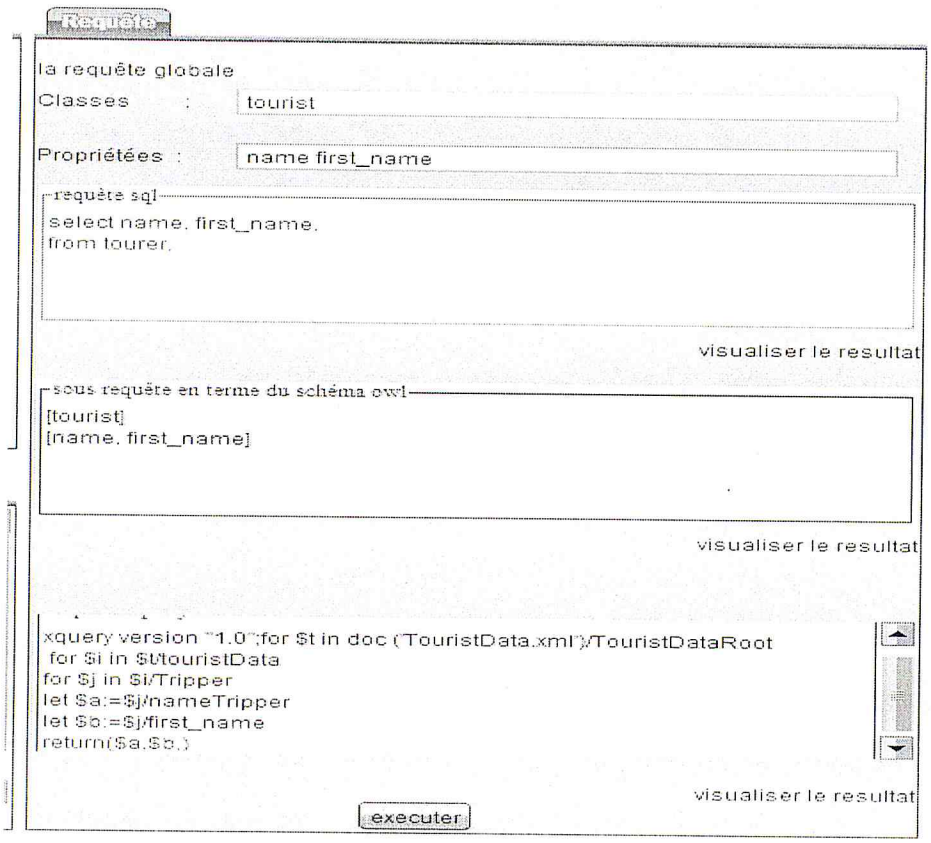


Figure 66: Requête1

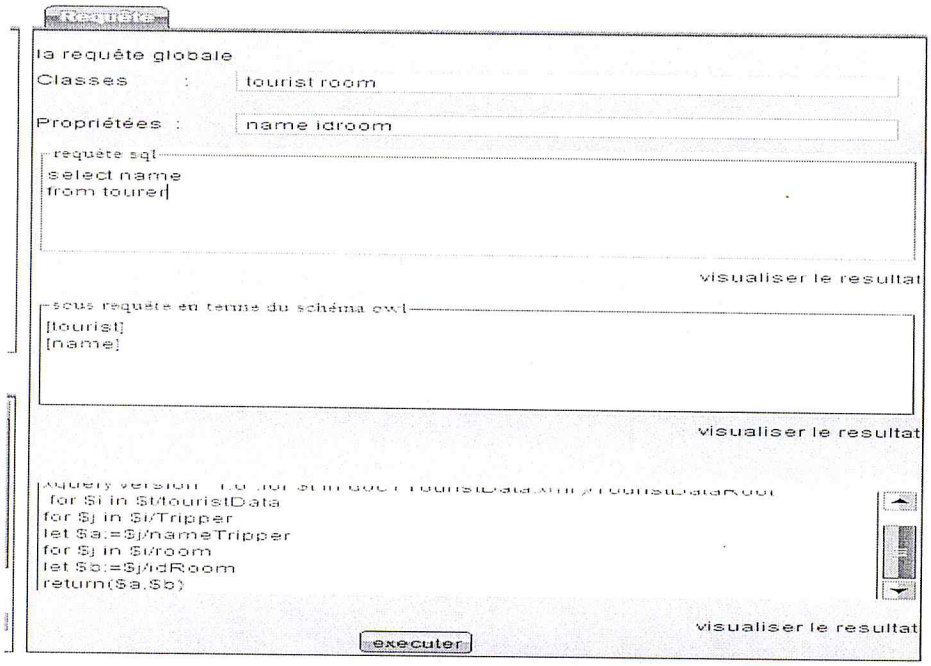


Figure 67: Requête2

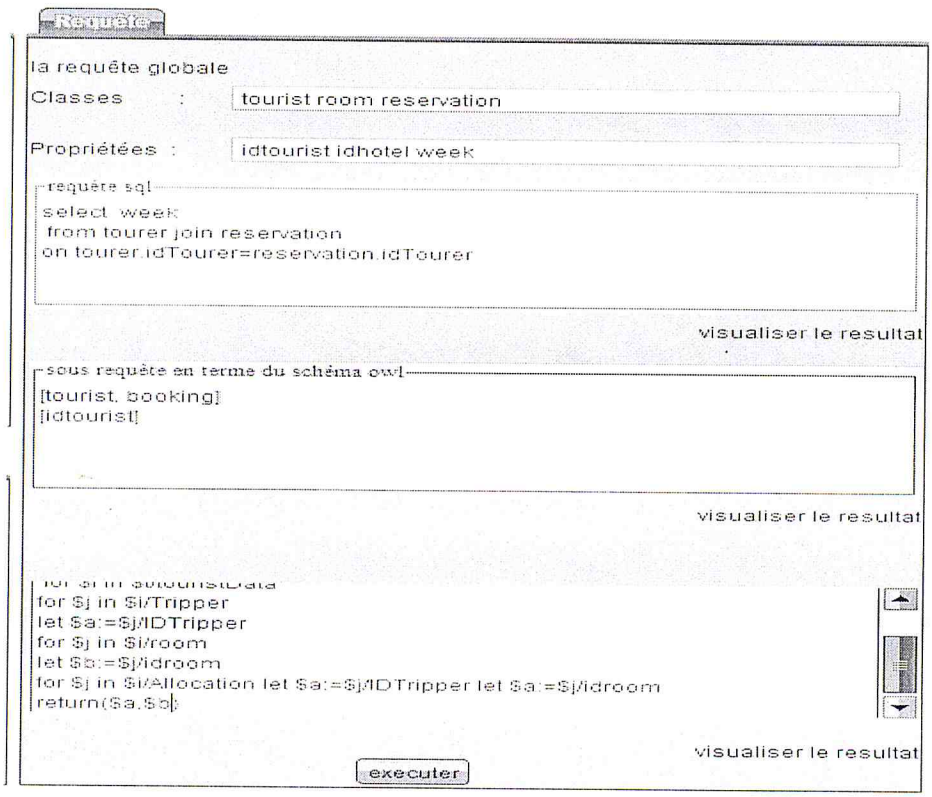


Figure 68: Requête3

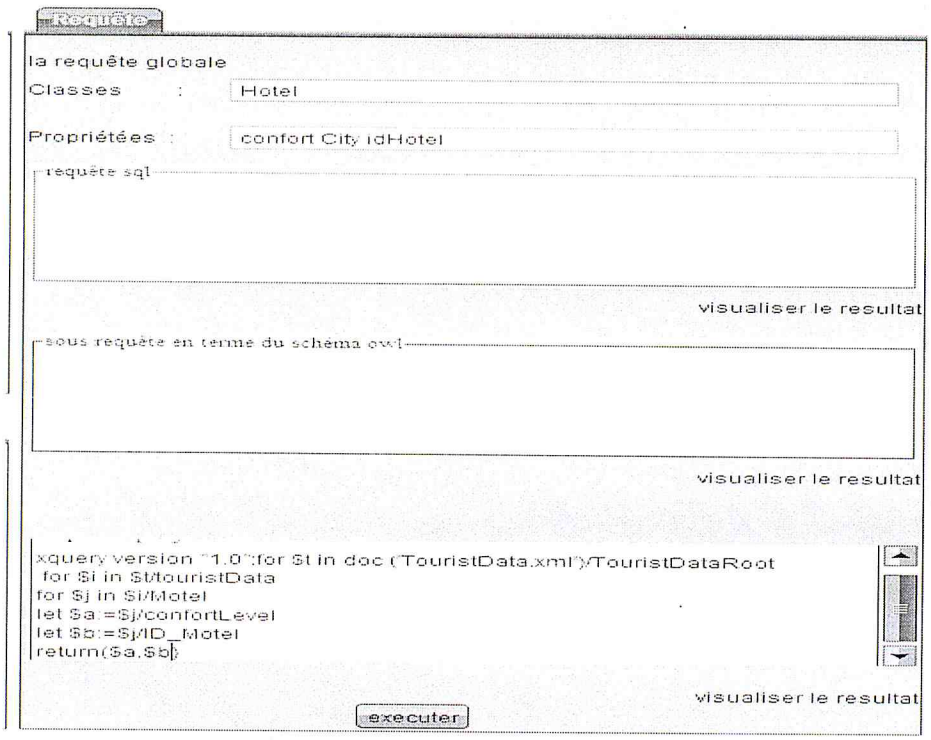


Figure 69: Requête4

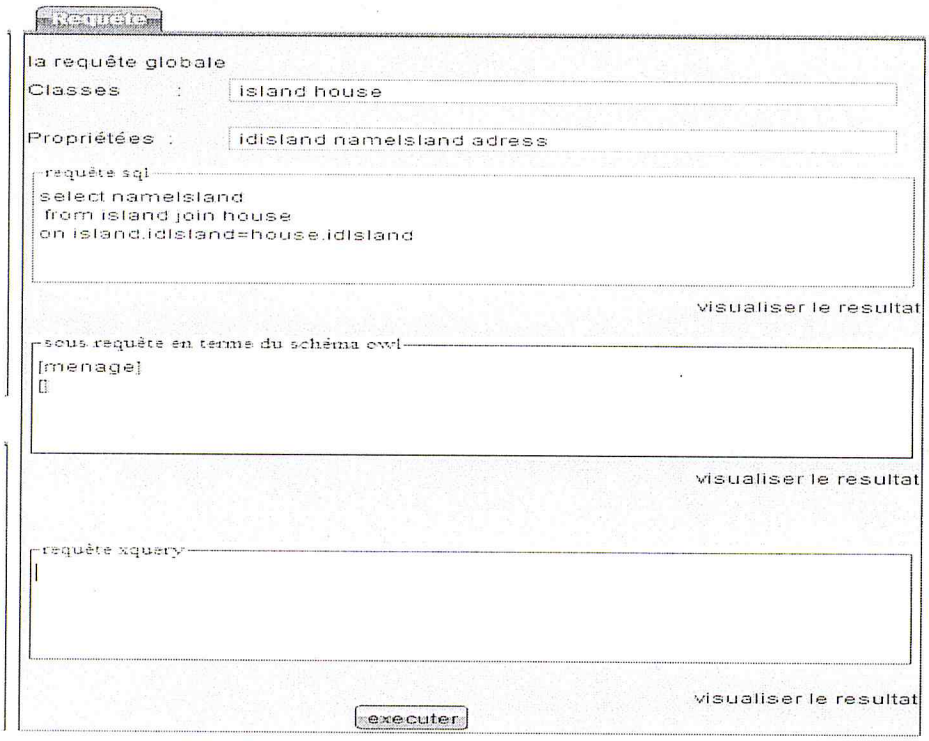


Figure 70: Requête5

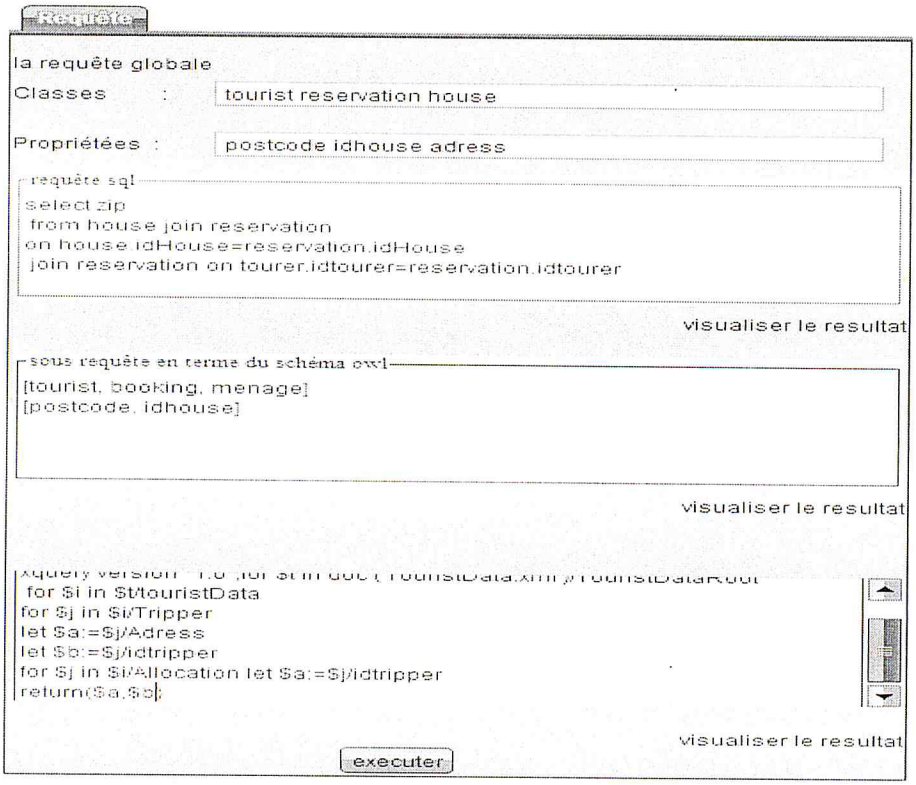


Figure 71: Requête6

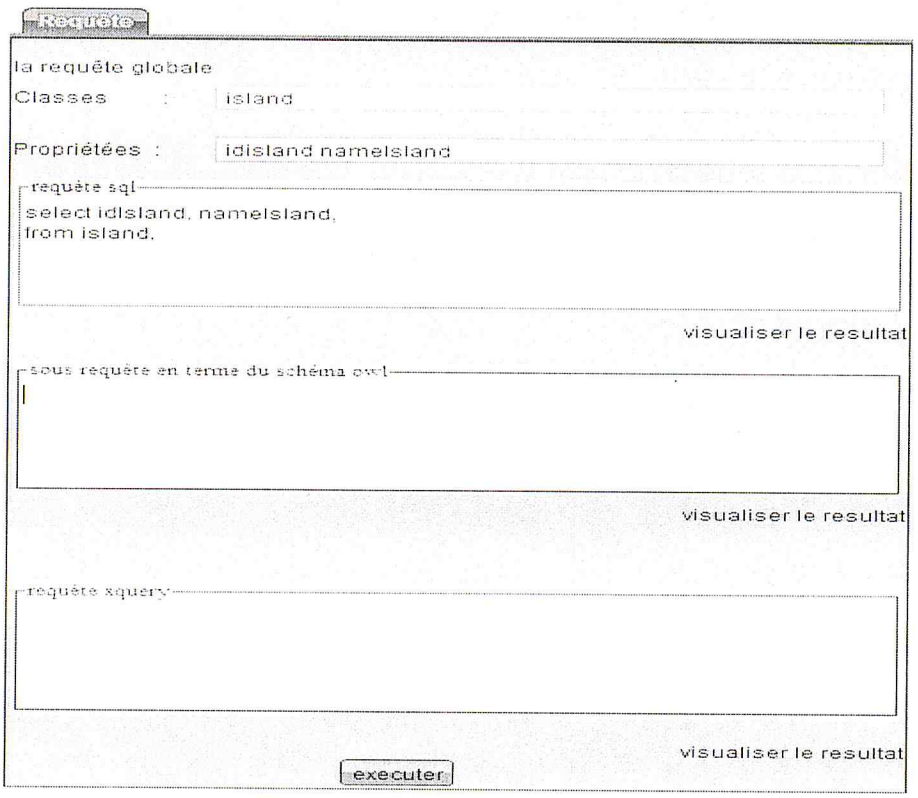


Figure 72: Requête7

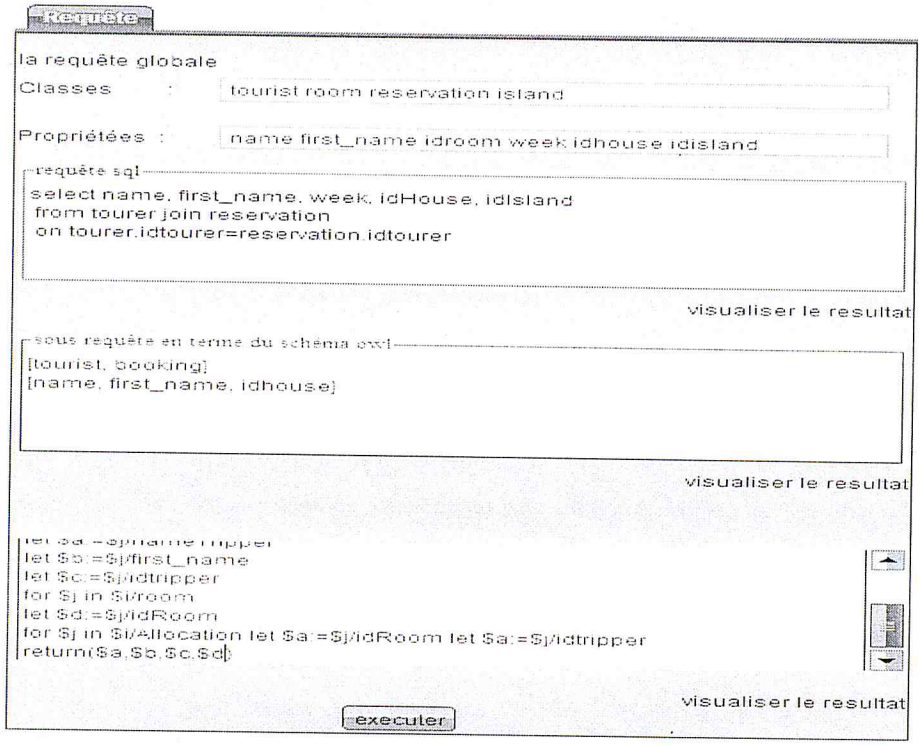


Figure 73: Requête8

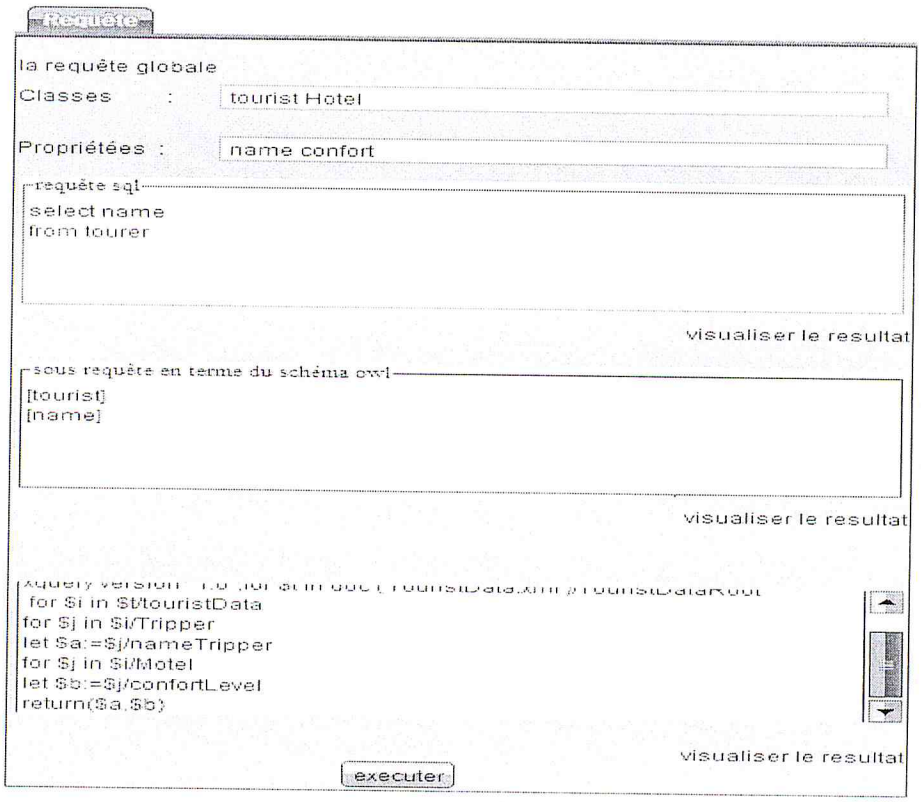


Figure 74: Requête9

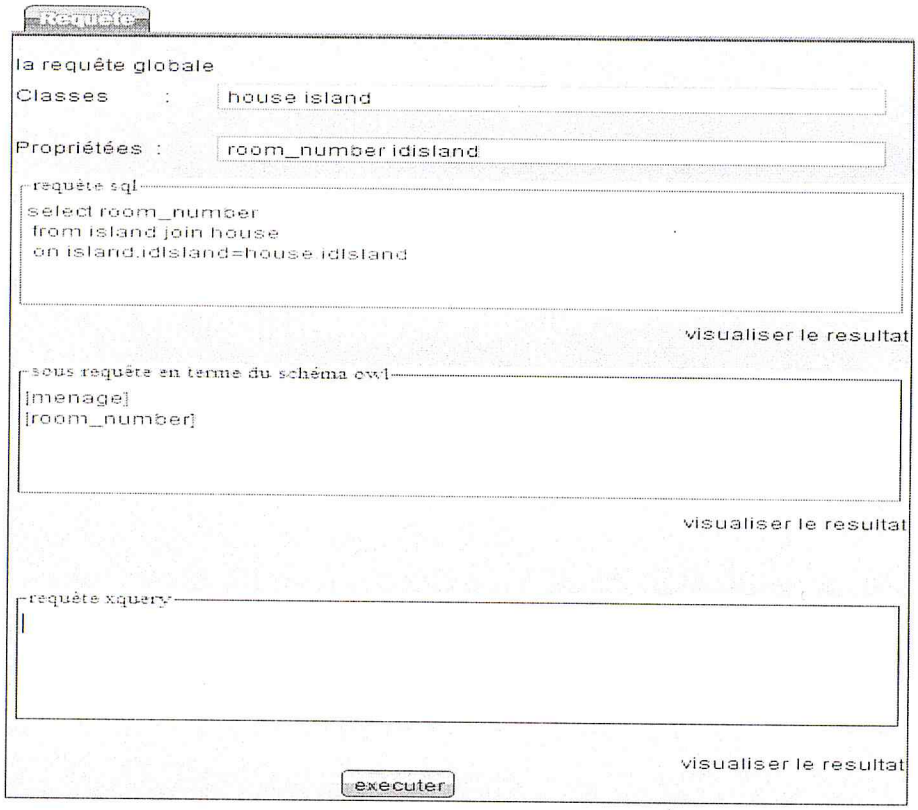


Figure 75: Requête10

3.2. Requête générer manuellement

	Sous requête sql	Sous requête owl	Sous requête xquery
Requête1	Selectname,first_name from tourer	[name,first_name] [tourist]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataR oot for $i in $t/touristData for $j in $i/Tripper let $a:=$j/nameTripper let $b:=$j/first_name return($a,$b)
Requête2	Select name fromtourer	[tourist] [name]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataR oot for $i in $t/touristData for $j in $i/Tripper let $a:=$j/nameTripper

			for $j in $i/room let $b:=$j/idRoom return($a,$b)
Requête3	select week from tourer join reservation on tourer.idTourer=reservation.idToure	[tourist,booking] [idtourist]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataRoot for $i in $t/touristData for $j in $i/Tripper let $a:=$j/IDTripper for $j in $i/room let $b:=$j/idroom for $j in $i/Allocation let $a:=$j/IDTripper let $a:=$j/idroom return($a,$b)
Requête4			xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataRoot for $i in $t/touristData for $j in $i/Motel let $a:=$j/confortLevel let $b:=$j/ID_Motel return($a,$b)
Requête5	select nameIsland, from island join house on island.idIsland=house.idIsland	[menage] []	
Requête6	select zip from house join reservation on house.idHouse=reservatio	[booking,tourist, menage] [idhouse,postcode]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataRoot for $i in $t/touristData for $j in $i/Allocation let

	n.idHouse join reservation on tourer.idtourer=reservation.idtourer		$a:=$j/idtripper for $j in $i/Tripper let $a:=$j/Adress let $b:=$j/idtripper return($a,$b)
Requête7	select idIsland, nameIsland from island		
Requête8	select name, first_name, week, idIsland, idHouse fromtourerjoinreservation on tourer.idtourer=reservation.idtourer	[tourist, booking] [name, first_name, idhouse]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataRoot for $i in $t/touristData for $j in $i/Tripper let $a:=$j/nameTripper let $b:=$j/first_name let $c:=$j/idtripper for $j in $i/room let $d:=$j/idRoom for $j in $i/Allocation let $a:=$j/idRoom let $a:=$j/idtripper return($a,$b,$c,$d)
Requête9	select name fromtourer	[tourist] [name]	xquery version "1.0";for $t in doc ('TouristData.xml')/TouristDataRoot for $i in $t/touristData for $j in $i/Tripper let $a:=$j/nameTripper for $j in $i/Motel let $b:=$j/confortLevel return($a,$b)
Requête10	select room_number,	[menage]	

	idIsland from island join house on island.idIsland=house.idIsland	[room_number]	

Tableau 10: Tableau des requêtes générées manuellement

3.3. La comparaison entre les requêtes générées automatiquement et les requête générées manuellement

Sous requêtemanuelle	Sous requêtemanuelle	Performance De la sous requête maneulle	Performance de la sousrequête de EMSH
Requête1	Requête1	✓	✓
Requête2	Requête2	✓	✓
Requête3	Requête3	✓	✓
Requête4	Requête4	✓	✓
Requête5	Requête5	✓	✓
Requête6	Requête6	✓	✓
Requête7	Requête7	✓	✓
Requête8	Requête8	✓	✓
Requête9	Requête9	✓	✓
Requête10	Requête10	✓	✓
Requêtesvalideproduites par le Système : $\|S\| = 10$ *Requêtevalideproduitesmanuellement : $\|H\| = 10$*			

Tableau 11 : tableau de comparaison

2. Mesures de performance de GRMoH

A travers le tableau 11, nous pouvons constater que mesures de performances de GRMoH ont été optimales (du moins par rapport à ce jeux de données).

TP	FP	Précision	Rappel	*F-mesure*	OVERALL	Fallout
1	1	1	1	1	1	0

Tableau 12: Mesures de performances de GRMoH.

Conclusion générale

Conclusion générale

1. Conclusion générale

Le problème de définition de requêtes de médiation est un problème complexe en raison de lagrande diversité des sources hétérogènes et distribuées qui peuvent intervenir dans un système de médiation et du grand volume de méta-données qui les décrivent, mais aussi en raison des conflits liés à l'hétérogénéité des données qui peuvent exister entre deux sources de données. Bien que les systèmes de médiation soient de nos jours des systèmes de plus en plus développés et connus, très peu de travaux se sont intéressés à la génération de requêtes de médiation et à la prise en compte des conflits liés à l'hétérogénéité des sources durant le processus de génération. La plupart se sont focalisés sur la détection et (ou) sur la résolution d'un conflit particulier dans le contexte de l'intégration de données. Ce manque de travaux méritant une exploration plus approfondie dans le domaine de la recherche, a motivé notre étude à explorer plus en détails la problématique liée à ces systèmes et d'apporter une solution originale, adaptée aux attentes des utilisateurs.

Face à cette problématique, nous avons proposé pour un environnement hétérogène, une approche de génération automatique des requêtes de médiation. Toute en tenant compte des conflits qui peuvent être présentés dans les diverses sources hétérogènes.

Les travaux menés dans ce mémoire nous ont permis d'approfondir nos connaissances dans plusieurs domaines :

- Notre projet de fin d'études nous a été d'un grand apport sur plusieurs plans principalement le plan conception, Nous avons appris l'aspect essentiel dans les applications d'intégration.
- Sur le plan technique nous avons acquis une certaine expérience dans le domaine de développement par les technologies suivantes : Java, JDOM et XQuery, Jena et plusieurs API.

Nos recherches et constatations, nous ont conduit à la mise au point d'une méthode originale pour la génération automatique des requêtes de médiations

Conclusion générale

L'originalité de notre méthode réside dans la combinaison des caractéristiques suivantes :

- Automatisation totale du processus de génération de requêtes, l'utilisateur n'aura qu'à saisir sa requête globale en sélectionnant les classes et propriétés à interroger.
- Intégration: le système s'intègre facilement dans n'importe quel environnement car il porte en son sein toutes les API essentielles, logiciels et ressources externes nécessaires à son bon fonctionnement.
- Souplesse dans le travail, l'utilisateur tiers sera soulagé de ne pas avoir à maitriser un langage de requête pour pouvoir poser des requêtes sur les sources qu'il veut les interroger.
- Evolutivité, de par sa construction en architecture simple, EMSH pourra être mis à jour, en effet peut être muni et fourni de nouvelles fonctionnalités ce qui n'aura pas d'incidence sur celles qu'il possède déjà.
- Interactivité homme-machine.
- Ergonomie : Son interface très malléable, rend EMSH très agréable d'utilisation. Lui offrant la possibilité de charger des sources locaux associées avec leur schéma global, de visionner leurs classes et propriétés leurs table de mapping.

En résumé, en dira que les objectifs initiaux ont été satisfaits.

2. Perspectives

Durant les quelques mois consacrés à la réalisation de notre projet de fin d'études nous nous somme obstinéesà atteindre les objectifs qui nous étaient fixés au départ en y ajoutant d'autres idées.

- Utilisation de la technologie SMA (Système Multi Agents) qui émane du contexte suivant : « l'intelligence d'un individu ne se forge que lorsqu'il est en contact avec d'autres individus de son espèce, cet individu que l'on appelle *agent* a donc besoin d'interagir, de collaborer, de rentrer en conflit, de s'adapter et de communiquer avec son environnement pour pouvoir accomplir des buts beaucoup plus complexes. ». Relevant du domaine de l'intelligence

artificielle, Les SMA, du fait de leur nature distribuée, facilitent énormément l'adaptabilité du système à toute évolution (aussi bien positive que négative), comme : changement des systèmes opératoires, changement de la configuration du système, mise à jour de des sources, ajout et suppression des fonctionnalités, etc.

- Intégrer un agent à la place d'un expert qui pourra effectuer l'annotation des concepts automatiquement.
- L'intégration de nouveaux sources telle que (les sources objets , fichier texte , d'autres applications).
- Rendre le système plus extensible en traitant le cas de la distribution des sources ;L'optimisation des requêtes et des résultats, et le temps de réponse.

L'innovation assure le cycle de vie d'un logiciel

Nous retiendrons les paroles de Mr Steve Jobs qui a dit :

« L'innovation, c'est une situation qu'on choisit parce qu'on a une passionbrûlante pour quelque chose. »

Annexes

1. Présentation de JAVA

Java est un langage de programmation compilé et interprété, orienté objet. Il a été créé en 1991 par Sun pour pallier aux contraintes que posait le C++, et cela dans le but de développer des logiciels pour l'électronique grand public (appareils ménagers). La syntaxe de ce langage est assez proche du C et est plus claire que le C++. L'objectif pour lequel JAVA a été conçu nécessite certaines caractéristiques comme : La robustesse, la compatibilité, la facilité de programmation et la petite taille du runtime ou des codes générés. JAVA présente beaucoup d'avantages grâce auxquels il a mérité son succès:

- Orienté objet : la brique de base du programme est donc l'objet, instance d'une classe.
- Indépendant des architectures.
- Portable : une fois le programme est compilé, il pourra fonctionner aussi bien sous des stations Unix, que sous Windows ou autre.
- Interprété : par la machine virtuelle de JAVA (JVM).
- Gère la mémoire automatiquement.
- Possède une API très riche : différents packages permettent d'accéder au réseau, aux entrées/sorties et aux différents composants graphiques.
- Distribué, simple, robuste, sécurisé, multithread et dynamique.

Sa portabilité et son indépendance des architectures ou plates-formes constituent son principal atout et grâce auquel il peut fonctionner sur différentes plates-formes et sous différents systèmes d'exploitation. Ces deux caractéristiques sont dues à la machine virtuelle Java (JVM). A coté de ses nombreux avantages, JAVA présente un seul inconvénient : la lenteur lors de la conversion des instructions de la JVM en instructions compréhensibles par la machine. Cependant deux solutions sont mises en place pour éviter cet inconvénient :

- Compiler le code pour une machine spécifique, à ce moment là le programme n'est plus portable
- Doter la JVM d'un JIT (Just In Time compiler) qui traduit le code JVM d'une classe dès sa première utilisation en code spécifique à la machine.

2. JAVA Virtual Machine (JVM)

La machine virtuelle appelée aussi interpréteur, joue le rôle d'un traducteur. Elle traduit en ByteCode le code compilé (ensemble de fichiers de classes) qui est sous forme de pseudo-code et non en code machine, et cela afin d'être exécuté sur n'importe quelle plate-forme matérielle et logicielle : que l'on soit sur un Pentium, un PowerPC, un Sparc ou sur un Alpha, sous Windows, MacOS, Solaris ou Linux, etc. Cependant, la présence de la JVM au niveau de la plate forme est nécessaire à l'exécution des programmes JAVA sur une plate-forme quelconque.

Les JVMs sont fournies soit par le JDK (Java Developpement Kit), soit par les navigateurs ou bien par les environnements de développement spécifiques tel Borland JBuilder. Des JVM capables d'exécuter du code Java, peuvent être fournit également par certaines solutions telle Java Plug-in de Sun. La JVM peut être obtenue à partir du site de Sun Microsystems.

3. Environnement de développement

Il existe plusieurs environnements de développement pour JAVA tel le JDK (Java Development Kit) qui est gratuit, mais la plus part sont payants. Ces environnements mettent à la disposition des développeurs : un éditeur intégré, un compilateur, un débuggeur sophistiqué et de nombreux générateurs de code (surtout concernant les interfaces graphiques).

3.1. Le Java Development Kit (JDK)

Le JDK est un environnement gratuit comportant un compilateur et une machine virtuelle (minimal et suffisant). Il est téléchargeable à partir du site de Sun. Il comporte l'ensemble des éléments qui ont pour but le développement, la mise au point et l'exécution des programmes Java. Il peut être considéré comme un ensemble d'outils plus un jeu de classes et de service plus un ensemble de spécifications. Il existe plusieurs versions de JDK, il est important de connaître la version employée car les classes disponibles peuvent être différentes d'une version à une autre.

4. Application Programming Interface (API)

Une API fournit aux programmeurs les outils de base nécessaires afin de leur faciliter le travail. Elle se compose d'un ensemble de fonctions, routines et méthodes. Par l'API de JAVA le programmeur va pouvoir accéder à toutes les ressources de la machine, et celles du réseau

Internet. Elle est divisée en packages. Plusieurs packages sont fournis en standard avec JAVA tels :

- java.awt et javax.swing pour les interfaces graphiques,
- java.applet pour la réalisation des applications qui peuvent s'exécuter dans un navigateur Internet.
- java.sql pour accéder aux bases de données.
- java.beans pour la programmation orientée composants (les Java Beans sont des composants logiciels réutilisable tel NetBeans ou JBuilder).
- java.rmi pour la mise en place des applications réparties telles les applications client/serveur.
- D'autres packages se trouvent dans d'autres applications et il faudra les importer tels : Les API PROTEGE et JENA pour la manipulation des ontologies.

5. Jena

Jena est un frame work écrit en Java, dont l'objectif est de fournir un environnement facilitant le développement d'applications dédiées au web sémantique. Jena permet de manipuler des documents RDF, RDFS et OWL, et fournit en plus un moteur d'inférences permettant des raisonnements sur les ontologies. Jena comportes les outils suivantes :

- Une API pour le langage RDF.
- Un module de lecture/ écriture sur les serveurs RDF : RDF/XML, N3et NTRIPLE.
- Une API pour le langage OWL.
- RDQL : un langage d'interrogation des ontologies Implémentées en RDF et OWL.

Le schéma ci-dessous montre la manière de création d'une ontologie dans Jena.

5.1. Le langage des ontologies et l'API JENA

Il existe plusieurs langages d'implémentation des ontologies, mais les langages les plus utilisés sont les langages RDF et OWL, ce dernier comporte trois variantes de complexité variable : OWL Full, OWL DL et le OWL Lite.

L'API JENA est neutre c'est-à-dire : les noms des classes java ne mentionnent pas le langage utilisé dans l'implémentation de l'ontologie i.e. : OntClass, ObjectProperty. Pour faire la distinction entre les différentes implémentations, chaque langage possède un profil qui permet d'identifier les constructeurs et les URI des classes et des propriétés autorisées. Chaque profil est lié à un modèle de l'ontologie qui est une version étendue de la classe du Modèle de JENA. Le Modèle général autorise l'accès aux déclarations dans une collection de données RDF, OntModel étend ceci en ajoutant le support pour les types d'objets attendus supposés être dans une ontologie: classes (dans une hiérarchie de classes), propriétés (dans une hiérarchie de propriétés) et les individus (instances).

6. Les sources

6.1. La sources OWL

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY tourist "http://www.semanticweb.org/zola/ontologies/2012/4/touris
]>


<rdf:RDF xmlns="http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#
     xml:base="http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:tourist="http://www.semanticweb.org/zola/ontologies/2012/4/tourist.
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about=""/>


    <!--
    ///////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////
     -->



    <!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#idHouse

    <owl:ObjectProperty rdf:about="#idHouse">
        <rdfs:domain rdf:resource="#booking"/>
        <rdfs:range rdf:resource="#menage"/>
    </owl:ObjectProperty>


    <!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#idTouri

    <owl:ObjectProperty rdf:about="#idTourist">
        <rdfs:domain rdf:resource="#booking"/>
        <rdfs:range rdf:resource="#tourist"/>
    </owl:ObjectProperty>
```

```
<!--
///////////////////////////////////////////////////////////////////////
//
// Data properties
//
///////////////////////////////////////////////////////////////////////
 -->



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#Island -->

<owl:DatatypeProperty rdf:about="#Island">
    <rdfs:domain rdf:resource="#menage"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#city -->

<owl:DatatypeProperty rdf:about="#city">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#first_n


<owl:DatatypeProperty rdf:about="#first_name">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#idhouse

<owl:DatatypeProperty rdf:about="#idhouse">
    <rdfs:domain rdf:resource="#menage"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>


<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#idtouri

<owl:DatatypeProperty rdf:about="#idtourist">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>


<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#label -

<owl:DatatypeProperty rdf:about="#label">
    <rdfs:domain rdf:resource="#menage"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

```
<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#maxpers

<owl:DatatypeProperty rdf:about="#maxperson">
    <rdfs:domain rdf:resource="#menage"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#name --

<owl:DatatypeProperty rdf:about="#name">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#postcod

<owl:DatatypeProperty rdf:about="#postcode">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>



<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#room_nu

<owl:DatatypeProperty rdf:about="#room_number">
    <rdfs:domain rdf:resource="#menage"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>


<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#street

<owl:DatatypeProperty rdf:about="#street">
    <rdfs:domain rdf:resource="#tourist"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#week --

<owl:DatatypeProperty rdf:about="#week">
    <rdfs:domain rdf:resource="#booking"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>
```

```
    -->

    <!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#booking -->

    <owl:Class rdf:about="#booking"/>

    <!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#menage -->

    <owl:Class rdf:about="#menage"/>

    <!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#tourist -->

    <owl:Class rdf:about="#tourist"/>

<!--
////////////////////////////////////////////////////////////////////////////////
//
// Individuals
//
////////////////////////////////////////////////////////////////////////////////
-->

<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#booking1 -->

<owl:Thing rdf:about="#booking1">
    <rdf:type rdf:resource="#booking"/>
    <week>03</week>
    <idHouse rdf:resource="#house1"/>
    <idTourist rdf:resource="#tourist1"/>
</owl:Thing>

<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#house1 -->

<menage rdf:about="#house1">
    <rdf:type rdf:resource="&owl;Thing"/>
    <room_number>3</room_number>
    <Island>UK</Island>
    <label>label1</label>
    <idhouse>01</idhouse>
    <maxperson>2</maxperson>
</menage>

<!-- http://www.semanticweb.org/zola/ontologies/2012/4/tourist.owl#tourist -->

<tourist rdf:about="#tourist1">
    <rdf:type rdf:resource="&owl;Thing"/>
    <first_name>khelladi</first_name>
    <city>blida</city>
    <street>rue d&#39;alger</street>
    <idtourist>01</idtourist>
    <postcode>02509</postcode>
    <name>fatma zohra</name>
</tourist>
</rdf:RDF>
```

6.2. La source XML

```
- <TouristDataRoot xsi:noNamespaceSchemaLocation="file:/C:/Users/Manel/Desktop/XML/sourcedonne/doc1.xsd">
  - <touristData>
    - <Tripper>
        <IDTripper nature="PrimaryKey">1</IDTripper>
        <nameTripper>Keddar</nameTripper>
        <firstNameTripper>Manel</firstNameTripper>
        <Adress>Medea Citi Msallah</Adress>
        <Email>menoulette@hotmail.com</Email>
        <phoneNumber>0.796.51.80.50</phoneNumber>
      </Tripper>
    - <Motel>
        <ID_Motel nature="PrimaryKey">H100</ID_Motel>
        <ComfortLevel>*****</ComfortLevel>
        <nameMotel>Hilton</nameMotel>
        <city>London</city>
        <country>GB</country>
      </Motel>
    - <room>
        <ID_room nature="PrimaryKey">R1_1</ID_room>
        <ID_Motel nature="ForeignKey">H100</ID_Motel>
      </room>

    - <Allocation>
        <IDTripper nature="ForeignKey">1</IDTripper>
        <ID_room nature="ForeignKey">R1_1</ID_room>
        <ID_Motel nature="ForeignKey">H100</ID_Motel>
        <duration>P5M</duration>
        <date_start>2011-03-11</date_start>
        <date_end>2011-08-11</date_end>
        <price unite="Euro">2000</price>
      </Allocation>
  </touristData>
- <touristData>
    - <Tripper>
        <IDTripper nature="PrimaryKey">2</IDTripper>
        <nameTripper>Zola</nameTripper>
        <firstNameTripper>Khalladi</firstNameTripper>
        <Adress>Cheffa</Adress>
        <Email>faatzoo@hotmail.com</Email>
        <phoneNumber>0.552.34.58.26</phoneNumber>
      </Tripper>
    - <Motel>
        <ID_Motel nature="PrimaryKey">H101</ID_Motel>
        <ComfortLevel>******</ComfortLevel>
        <nameMotel>Saphir</nameMotel>
        <city>Paris</city>
        <country>France</country>
      </Motel>
    - <room>
        <ID_room nature="PrimaryKey">R3</ID_room>
        <ID_Motel nature="ForeignKey">H101</ID_Motel>
      </room>
```

```
    - <Allocation>
        <IDTripper nature="ForeignKey">2</IDTripper>
        <ID_room nature="ForeignKey">R3</ID_room>
        <ID_Motel nature="ForeignKey">H101</ID_Motel>
        <duration>P5D</duration>
        <date_start>2012-02-02</date_start>
        <date_end>2012-02-07</date_end>
        <price unite="Euro">100</price>
      </Allocation>
  </touristData>
- <touristData>
  - <Tripper>
      <IDTripper nature=" PrimaryKey">3</IDTripper>
      <nameTripper>Nadjib</nameTripper>
      <firstNameTripper>Younsi</firstNameTripper>
      <Adress>Alger ALgeria</Adress>
      <Email>nadjibo@hotmail.com</Email>
      <phoneNumber>0.25.58.86.11</phoneNumber>
    </Tripper>
  - <Motel>
      <ID_Motel nature="PrimaryKey">H100</ID_Motel>
      <ComfortLevel>*****</ComfortLevel>
      <nameMotel>Hilton</nameMotel>
      <city>London</city>
      <country>GB</country>
    </Motel>
  - <room>
      <ID_room nature="PrimaryKey">R3_3</ID_room>
      <ID_Motel nature="ForeignKey">H100</ID_Motel>
    </room>

  - <Allocation>
      <IDTripper nature="ForeignKey">3</IDTripper>
      <ID_room nature="ForeignKey">R3_3</ID_room>
      <ID_Motel nature="ForeignKey">H100</ID_Motel>
      <duration>P5D</duration>
      <date_start>2012-02-02</date_start>
      <date_end>2012-02-07</date_end>
      <price unite="Euro">100</price>
    </Allocation>
  </touristData>
- <touristData>
  - <Tripper>
      <IDTripper nature="PrimaryKey">4</IDTripper>
      <nameTripper>James</nameTripper>
      <firstNameTripper>Bond</firstNameTripper>
      <Adress>Los Angeles Avenue 1020</Adress>
      <Email>james.bond@hotmail.com</Email>
      <phoneNumber>0.256.68.43.54</phoneNumber>
    </Tripper>
  - <Motel>
      <ID_Motel nature="PrimaryKey">H101</ID_Motel>
      <ComfortLevel>******</ComfortLevel>
      <nameMotel>Saphir</nameMotel>
      <city>Paris</city>
      <country>France</country>
    </Motel>
  - <room>
      <ID_room nature="PrimaryKey">R22</ID_room>
      <ID_Motel nature="ForeignKey">H101</ID_Motel>
    </room>
```

```
        - <Allocation>
            <IDTripper nature="ForeignKey">4</IDTripper>
            <ID_room nature="ForeignKey">R22</ID_room>
            <ID_Motel nature="ForeignKey">H101</ID_Motel>
            <duration>P16D</duration>
            <date_start>2007-05-15</date_start>
            <date_end>2007-05-31</date_end>
            <price unite="Euro">116</price>
          </Allocation>
        </touristData>
      - <touristData>
        - <Tripper>
            <IDTripper nature="PrimaryKey">5</IDTripper>
            <nameTripper>Nicola</nameTripper>
            <firstNameTripper>Sarckozy</firstNameTripper>
            <Adress>Paris sud avenue 2001</Adress>
            <Email>n.sarko@hotmail.com</Email>
            <phoneNumber>00.33.15.123.56.26</phoneNumber>
          </Tripper>
        - <Motel>
            <ID_Motel nature="PrimaryKey">H102</ID_Motel>
            <ComfortLevel>*********</ComfortLevel>
            <nameMotel>Le Méridien Etoile</nameMotel>
            <city>Paris</city>
            <country>France</country>
          </Motel>
        - <room>
            <ID_room nature="PrimaryKey">RP200</ID_room>
            <ID_Motel nature="ForeignKey">H102</ID_Motel>
          </room>

      - <Allocation>
          <IDTripper nature="ForeignKey">5</IDTripper>
          <ID_room nature="ForeignKey">RP200</ID_room>
          <ID_Motel nature="ForeignKey">H102</ID_Motel>
          <duration>P2M</duration>
          <date_start>2006-06-01</date_start>
          <date_end>2006-08-01</date_end>
          <price unite="Euro">5000</price>
        </Allocation>
      </touristData>
    </TouristDataRoot>
```

6.3. La source relationnelle

Table	Action	Enregistrements	Type	Interclassement	Taille	Perte
house		2	InnoDB	latin1_swedish_ci	16,0 Kio	-
island		2	InnoDB	latin1_swedish_ci	16,0 Kio	-
reservation		2	InnoDB	latin1_swedish_ci	16,0 Kio	-
tourer		5	InnoDB	latin1_swedish_ci	16,0 Kio	-
4 table(s)	Somme	11	MyISAM	latin1_swedish_ci	64,0 Kio	0 o

Tout cocher / Tout décocher Pour la sélection :

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
idhouse	int(11)			Non			
name	varchar(30)	latin1_swedish_ci		Non			
room_number	int(11)			Non			
maxperson	int(11)			Non			
idisland	int(11)			Non			

Tout cocher / Tout décocher *Pour la sélection :*

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
idIsland	varchar(30)	latin1_swedish_ci		Non			

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
idTouriste	int(11)			Non			
name	varchar(30)	latin1_swedish_ci		Non			
first_name	varchar(30)	latin1_swedish_ci		Non			
street	varchar(120)	latin1_swedish_ci		Non			
zip	varchar(30)	latin1_swedish_ci		Non			
city	varchar(30)	latin1_swedish_ci		Non			

Tout cocher / Tout décocher *Pour la sélection :*

Serveur: localhost ▸ Base de données: tourisme ▸ Table: reservation *"InnoDB free: 3072 kB"*

Afficher | Structure | SQL | Rechercher | Insérer | Exporter | Importer | Opérations | Vider | Supprimer

Base de données
tourisme (4)

tourisme (4)
- house
- island
- reservation
- tourer

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
id_house	int(11)			Non	0		
id_tourist	int(11)			Non	0		
week	int(11)			Oui	*NULL*		

Tout cocher / Tout décocher *Pour la sélection :*

Version imprimable | Gestion des relations | Suggérer des optimisations quant à la structure de la table
Ajouter 1 champ(s) En fin de table En début de table Après id_house Exécuter

Index:

Nom de la clé	Type	Cardinalité	Action	Champ
PRIMARY	PRIMARY	2		id_house
				id_tourist

Créer une clef sur 1 colonne(s) Exécuter

Espace utilisé			Statistiques	
Type	Espace		Information	Valeur
Données	16 384	o	format	Compact
Index	0	o	Interclassement	latin1_swedish_ci
Total	16 384	o	Création	Dimanche 03 Juin 2012 à 11:33

Ouvrir une nouvelle fenêtre phpMyAdmin

7. Les schémas des sources locales

7.1. Schéma de la source relationnelle

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY www "http://www.ontologie.fr/" >
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY touristDBowl "http://www.ontologie.fr/touristDBowl#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<rdf:RDF xmlns="&www;touristDBowl#"
     xml:base="&www;touristDBowl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:www="http://www.ontologie.fr/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:touristDBowl="&www;touristDBowl#">
    <owl:Ontology rdf:about=""/>

  <!--
  ///////////////////////////////////////////////////////////////////////////////////////
  //
  // Object Properties
  //
  ///////////////////////////////////////////////////////////////////////////////////////
   -->



  <!-- http://www.ontologie.fr/idhouse -->

  <owl:ObjectProperty rdf:about="&www;idhouse">
      <rdfs:range rdf:resource="&www;house"/>
      <rdfs:domain rdf:resource="&www;reservation"/>
  </owl:ObjectProperty>



  <!-- http://www.ontologie.fr/idisland -->

  <owl:ObjectProperty rdf:about="&www;idisland">
      <rdfs:domain rdf:resource="&www;house"/>
      <rdfs:range rdf:resource="&www;island"/>
  </owl:ObjectProperty>



  <!-- http://www.ontologie.fr/idtourer -->

  <owl:ObjectProperty rdf:about="&www;idtourer">
      <rdfs:domain rdf:resource="&www;reservation"/>
      <rdfs:range rdf:resource="&www;tourer"/>
  </owl:ObjectProperty>
```

```
<!--
///////////////////////////////////////////////////////////////////////////////////////
//
// Data properties
//
///////////////////////////////////////////////////////////////////////////////////////
 -->



<!-- http://www.ontologie.fr/city -->

<owl:DatatypeProperty rdf:about="&www;city">
    <rdfs:domain rdf:resource="&www;tourer"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>



<!-- http://www.ontologie.fr/first_name -->

<owl:DatatypeProperty rdf:about="&www;first_name">
    <rdfs:domain rdf:resource="&www;tourer"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>



<!-- http://www.ontologie.fr/idHouse -->

<owl:DatatypeProperty rdf:about="&www;idHouse">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="&www;house"/>
    <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>

 <!-- http://www.ontologie.fr/idIsland -->

 <owl:DatatypeProperty rdf:about="&www;idIsland">
     <rdf:type rdf:resource="&owl;FunctionalProperty"/>
     <rdfs:domain rdf:resource="&www;island"/>
     <rdfs:range rdf:resource="&xsd;int"/>
 </owl:DatatypeProperty>



 <!-- http://www.ontologie.fr/idTourer -->

 <owl:DatatypeProperty rdf:about="&www;idTourer">
     <rdf:type rdf:resource="&owl;FunctionalProperty"/>
     <rdfs:domain rdf:resource="&www;tourer"/>
     <rdfs:range rdf:resource="&xsd;int"/>
 </owl:DatatypeProperty>



<!-- http://www.ontologie.fr/maxperson -->

<owl:DatatypeProperty rdf:about="&www;maxperson">
    <rdfs:domain rdf:resource="&www;house"/>
    <rdfs:range rdf:resource="&xsd;int"/>
</owl:DatatypeProperty>



<!-- http://www.ontologie.fr/name -->

<owl:DatatypeProperty rdf:about="&www;name">
    <rdfs:domain rdf:resource="&www;tourer"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

```
    <!-- http://www.ontologie.fr/room_number -->

    <owl:DatatypeProperty rdf:about="&www;room_number">
        <rdfs:domain rdf:resource="&www;house"/>
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>



    <!-- http://www.ontologie.fr/street -->

    <owl:DatatypeProperty rdf:about="&www;street">
        <rdfs:domain rdf:resource="&www;tourer"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>



    <!-- http://www.ontologie.fr/touristOBowl#nameIsland -->

    <owl:DatatypeProperty rdf:about="#nameIsland">
        <rdfs:domain rdf:resource="&www;island"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>



    <!-- http://www.ontologie.fr/touristOBowl#week -->

    <owl:DatatypeProperty rdf:about="#week">
        <rdfs:domain rdf:resource="&www;reservation"/>
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>


    <!-- http://www.ontologie.fr/zip -->

    <owl:DatatypeProperty rdf:about="&www;zip">
        <rdfs:domain rdf:resource="&www;tourer" />
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>


    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Classes
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->


    <!-- http://www.ontologie.fr/house -->

    <owl:Class rdf:about="&www;house"/>


    <!-- http://www.ontologie.fr/island -->

    <owl:Class rdf:about="&www;island"/>


    <!-- http://www.ontologie.fr/reservation -->

    <owl:Class rdf:about="&www;reservation"/>


        <!-- http://www.ontologie.fr/tourer -->

        <owl:Class rdf:about="&www;tourer"/>
</rdf:RDF>
```

7.2. le schéma de la sources XML

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY www "http://www.ontologie.fr/" >
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY touristXML "http://www.ontologie.fr/touristXML#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<rdf:RDF xmlns="&www;touristXML#"
     xml:base="&www;touristXML"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:www="http://www.ontologie.fr/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:touristXML="&www;touristXML#">
    <owl:Ontology rdf:about=""/>



    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->


 <!-- http://www.ontologie.fr/idmotel -->

 <owl:ObjectProperty rdf:about="&www;idmotel">
     <rdfs:range rdf:resource="&www;Motel"/>
     <rdfs:domain rdf:resource="&www;room"/>
 </owl:ObjectProperty>


 <!-- http://www.ontologie.fr/idroom -->

 <owl:ObjectProperty rdf:about="&www;idroom">
     <rdfs:domain rdf:resource="&www;Allocation"/>
     <rdfs:range rdf:resource="&www;room"/>
 </owl:ObjectProperty>


 <!-- http://www.ontologie.fr/idtripper -->

 <owl:ObjectProperty rdf:about="&www;idtripper">
     <rdfs:domain rdf:resource="&www;Allocation"/>
     <rdfs:range rdf:resource="&www;Tripper"/>
 </owl:ObjectProperty>


 <!--
 ///////////////////////////////////////////////////////////////////////////////////////
 //
 // Data properties
 //
 ///////////////////////////////////////////////////////////////////////////////////////
  -->
```

```
<!-- http://www.ontologie.fr/Adress -->

<owl:DatatypeProperty rdf:about="&www;Adress">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/Email -->

<owl:DatatypeProperty rdf:about="&www;Email">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/IDTripper -->

<owl:DatatypeProperty rdf:about="&www;IDTripper">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/ID_Motel -->

<owl:DatatypeProperty rdf:about="&www;ID_Motel">
    <rdfs:domain rdf:resource="&www;Motel"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/city -->

<owl:DatatypeProperty rdf:about="&www;city">
    <rdfs:domain rdf:resource="&www;Motel"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/confortLevel -->

<owl:DatatypeProperty rdf:about="&www;confortLevel">
    <rdfs:domain rdf:resource="&www;Motel"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/country -->

<owl:DatatypeProperty rdf:about="&www;country">
    <rdfs:domain rdf:resource="&www;Motel"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/date_end -->

<owl:DatatypeProperty rdf:about="&www;date_end">
    <rdfs:domain rdf:resource="&www;Allocation"/>
    <rdfs:range rdf:resource="&xsd;date"/>
</owl:DatatypeProperty>
```

```
<!-- http://www.ontologie.fr/date_start -->

<owl:DatatypeProperty rdf:about="&www;date_start">
    <rdfs:domain rdf:resource="&www;Allocation"/>
    <rdfs:range rdf:resource="&xsd;date"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/duration -->

<owl:DatatypeProperty rdf:about="&www;duration">
    <rdfs:domain rdf:resource="&www;Allocation"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/first_name -->

<owl:DatatypeProperty rdf:about="&www;first_name">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/nameMotel -->

<owl:DatatypeProperty rdf:about="&www;nameMotel">
    <rdfs:domain rdf:resource="&www;Motel"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/nameTripper -->

<owl:DatatypeProperty rdf:about="&www;nameTripper">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/phoneNumber -->

<owl:DatatypeProperty rdf:about="&www;phoneNumber">
    <rdfs:domain rdf:resource="&www;Tripper"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/price -->

<owl:DatatypeProperty rdf:about="&www;price">
    <rdfs:domain rdf:resource="&www;Allocation"/>
    <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>


<!-- http://www.ontologie.fr/touristXML#idRoom -->

<owl:DatatypeProperty rdf:about="#idRoom">
    <rdfs:domain rdf:resource="&www;room"/>
    <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>
```

```
    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Classes
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->


    <!-- http://www.ontologie.fr/Allocation -->

    <owl:Class rdf:about="&www;Allocation"/>


    <!-- http://www.ontologie.fr/Motel -->

    <owl:Class rdf:about="&www;Motel"/>


    <!-- http://www.ontologie.fr/Tripper -->

    <owl:Class rdf:about="&www;Tripper"/>


    <!-- http://www.ontologie.fr/room -->

    <owl:Class rdf:about="&www;room"/>
</rdf:RDF>
```

Références bibliographiques

Référencesbibliographiques

[10]: Miller R.J., Haas L., Hernandez M.A., "Schema Mapping as query discovery", In proc of the Int Conf on Very Large Data Bases (VLDB), Cairo, Egypt, p.77-88, ,2000.

[11] : Popa L., Yu C., "Constraint-based XML query rewriting for data integration",Proc of Int Conf ACM SIGMOD (SIGMOD'04), Paris, France, p.371-382, 2004.

[12] :IXIA (IndeX-basedIntegrationApproach) A HybridApproch to Data Integration.Shokoh Kerman-Shahani.University Joseph Fourier Grenope I .2009.

[13] :L'intégration des données par médiation basée sur les ontologies pour l'analyse en ligne (OLAP) à la demande, Université Lion, Laboratoire ERIC,*(3 juillet 2007).*

[14] : Intégration des sources de métadonnées hétérogènes : XML, RDF et RuleML par l'approche médiateu .KRELIFA Halla.Université Saad Dehleb Blida, *(juin 2011).*

[16] : Enzenat, J., Shvaiko, P. : Ontologymatching. Springer-Verlag,heidelberg (2007)

[17] : Ted Pedersen, S.p,Michelizzi, J. : Wordnet ::similarity :mesuring the relatedness of concepts. In : DemonstrationPapersat the HumanLanguageTechnologyConference of the North American Chapter of the Associatior for ComputationalLinguistics. pp.38-41(2004).

[18] : Mme Ghomari Leïla (Née Zemmouchi), Alignement d'ontologies de domaine : Etude comparative syntaxique versus sémantique, cas d'application COMA++ VS OWL-CTXMatch, Mémoire pour l'obtention du diplôme de Magister en Informatique, à l'Ecole Supérieur d'Informatique (E.S.I) Alger, Algérie(2008-2009).