République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida
N° D'ordre :.....

Faculté des sciences

Département d'informatique

Mémoire Présenté par :

BOUKHIT Amina BOUHDJEUR Nesrine

En vue d'obtenir le diplôme de master

Domaine : Mathématique et informatique

Filière : Informatique
Spécialité : Informatique
Option : Ingénierie de logiciel

Thème : Optimisation des Requêtes du Médiateur en Utilisant le Data Mining

Soutenu le :

M. Bala	Président
M. Kamèche	Examinateur
M. Derrar	Examinateur
Mme. Fareh	Promotrice

Promotion 2014 / 2015

REMERCIEMENTS

Tout d'abord, nous tenons à remercier DIEU le tout puissant de nous avoir donné la force, le courage et la patience pour accomplir notre travail.

Nous tenons à remercier nos très chers parents pour leur tendresse, leur encouragement et leur soutien moral et matériel dans le but d'assurer notre réussite.

Nous tenons à remercier vivement notre promotrice madame ***Fareh*** *pour sa précieuse aide, sa patience, sa générosité et disponibilité et surtout son soutien morale.*

Nous tenons à remercier aussi monsieur ***CHikhi*** *pour son précieuse aide et nous sommes reconnaissantes à Tous nos enseignants qui nous ont facilité la compréhension et la maitrise.*

Nous remercions le membre de jury pour nous avoir fait l'honneur de juger notre travail.

Enfin un grand merci à nos amis et collègues et tous ceux qui ont contribué de près ou de loin à la réalisation de ce mémoire.

Amina & Nesrine

Je dédie ce modeste travail

A mes très chers parents qui ont tout fait pour que je réussisse dans ma vie et mes études, je leur souhaite le bonheur et la bonne santé et J'espère que vous seriez toujours fiers de moi

A mes chers frères Abdelhak, Amine, Abdellah et Abdenour

A mes chères sœurs

A ma chère grande sœur, son épouse Aissa et leurs enfants Yasser et Hibatallah

A mon grand-père, mes grandes mères, à tous mes oncles et mes tantes, cousines et cousins

A toute ma famille sans exception

A tous mes amies en particulier Nesrine, Fatma, Kheira, Mimi, Khadidja, Hadjer, Amina, Sihem, Imen, Lamia …

A tous ceux qui me sont chères

Amina

DEDICACES

Je dédie ce travail:

A mes très chers parents qui m'ont toujours soutenu et encouragé.

Mon oncle Abdellah et sa femme Zahia.

A mon chère grand frère Ahmed, sa femme Fatma Zohra et son enfant Adem.

A mes chers frères Ben Youcef, Mohamed et Hamza.

A ma chère grande sœur Houria et son épouse Hichem.

A mes chers Sœurs Salima, Bochra, Fatima, Zoubida et Rima.

A mes grandes mères.

A mes très chères tantes.

A mes cousins et cousines.

A mon binôme Amina.

A tous mes amis sans exception.

A tous ceux qui m'ont aidé et soutenu pendant tout mon cursus universitaire.

Toute personne qui m'aime et que j'aime.

Nesrine

ملخص

أنظمة الوساطة هي أنظمة جد معروفة و متطورة, تحاول ان تقدم للمستخدم طريقة وصول بسيطة و قوية للمعلومات فهي قادرة على دمج مصادر البيانات المختلفة لتوفير رؤية مركزية وموحدة للبيانات عن طريق إخفاء تفاصيل **عدم التجانس**.

إعداد وتنفيذ أنظمة الوساطة يطرح عددا من المشاكل، لا سيما من حيث مشكلة زمن الاستجابة للاستعلام . بشكل عام، هناك العديد من الخطط لتشغيل نفس الاستعلام، غالبا ما يكون بعضها أسرع بكثير من البعض الآخر, المشكلة التي تنشأ هي كيفية اختيار **خطة التنفيذ المثلى** والفعالة.

الهدف الرئيسي من هذا المشروع هو تحسين وقت تنفيذ الاستعلامات في أنظمة الوساطة وذلك باستخدام **بيانات التعدين** .

الكلمات المفتاحية: أنظمة الوساطة, عدم التجانس, استعلام, خطة التنفيذ المثلى, بيانات التعدين.

Résumé

Les **systèmes de médiation** sont aujourd'hui très développés et connus et tentent de proposer à l'utilisateur un accès simple et puissant à l'information, ils sont capables d'intégrer diverses sources de données pour offrir une vue centralisée et uniforme des données, en masquant les détailles de l'**hétérogénéité**.

La mise on œuvre des systèmes de médiation pose un certain nombre de problèmes, particulièrement le problème de performances en terme de temps de réponse de **requête**. En général, il existe de nombreux plans pour exécuter la même requête, parmi lesquelles certaines sont souvent beaucoup plus rapides que les autres, la problématique qui se pose c'est comment choisir le **plan d'exécution optimal** et efficace.

L'objectif principal de ce projet est l'optimisation de temps d'exécution des requêtes sur les systèmes de médiation, en utilisant le **Data Mining**.

Mots Clés: systèmes de médiation, hétérogénéité, requête, plan d'exécution optimal, Data Mining.

Abstract

Mediation systems are today highly developed, known and try to offer the user a simple and powerful access to information, they are able to integrate various data sources to provide a centralized and uniform view of data by masking the **heterogeneity** details.

The implementing of mediation systems poses a number of problems, particularly the problem in terms of performance of **query** response time. In general, there are many plans to run the same query, some of which are often much faster than the others, the issue which arises is how to choose the efficient and **optimal execution plan**. The main objective of this project is to optimize the execution time of queries on mediation systems, using the **Data Mining**.

Keywords: Mediation systems, heterogeneity, query, optimal execution plan, Data Mining.

Sommaire

Chapitre III : *Conception du Système.*

Chapitre IV : *Implémentation.*

Chapitre V : *Tests et Validation.*

Liste des Tableaux

Liste des Figures

Introduction Générale

L'énorme quantité d'informations disponibles sur des sources de données autonomes, distribuées et hétérogènes, nécessite une stratégie de recherche, de sélection et d'analyse, de plus en plus performants, pour localiser et extraire précisément l'information désirée. Les systèmes d'intégration d'informations offre un accès à de multiples sources de données à travers des requêtes posés selon un schéma global. L'intégration des données est le processus par lequel plusieurs sources de données autonomes, réparties et hétérogènes telle que chaque source est associée à un schéma local sont intégrées sous forme de source unique représentée par un schéma global.

Un système d'intégration est, soit un entrepôt de données (approche matérialisée), soit un médiateur (approche virtuelle). Pour notre projet nous intéressons aux systèmes de médiations, ces systèmes qui reposent sur l'architecture médiateurs/adaptateurs tentent de proposer à l'utilisateur un accès simple et puissant à l'information.

Problématique

L'explosion du nombre de sources d'information multiplie les besoins techniques d'intégration des sources autonomes et hétérogènes et demande aux chercheurs à développer beaucoup plus ces systèmes. La problématique posée concernant les systèmes de médiation est qu'une requête globale doit être divisée et envoyée à toutes les bases contenant les attributs figurant dans cette requête sachant qu'il y'a certaines bases qui ne contiennent aucune réponse, ainsi que pour une seule requête globale il existe plusieurs plans pour l'exécuter alors qu'il y'a certain plans beaucoup plus rapide que les autres et donc il faut déterminer le meilleur plan d'exécution pour chaque requête.

Objectif

L'objectif de notre projet rentre dans la problématique générale de la médiation des données hétérogènes, parmi ces données hétérogènes on s'intéresse aux données structurés qui sont représentés par le modèle relationnel et le modèle relationnel objet, et semi structuré représentées par XML. Notre but est d'optimiser l'exécution de requête d'une manière qu'elle ne sera envoyée qu'aux sources contenant sûrement des réponses et qu'elle sera exécuter selon le plan le plus optimal.

Pour réaliser notre objectif et arriver à *concevoir un système d'optimisation* de requêtes dans les systèmes de médiation, nous avons passé par plusieurs étapes. Ces étapes sont présentées dans les chapitres ci-dessous :

Chapitre I: Médiation des Données Qui permet d'obtenir une bonne compréhension des aspects caractéristiques de la médiation de données ainsi que les différents composants de ces systèmes.

Chapitre II: Data Mining Dans ce chapitre nous présentons quelques définitions concernant le domaine de Data Mining, ainsi que ces domaines d'applications.

Chapitre III: Conception du Système Ce *chapitre comporte la conception de notre* système qui se divise en deux étapes principales, la première c'est la conception du système de médiation et la deuxième la conception de système d'optimisation de requêtes dans les systèmes de médiation en utilisant le Data Mining. Pour modéliser notre système nous avons utilisé le langage de modélisation UML.

Chapitre IV: Implémentation du Système Dans ce chapitre nous décrirons les différents outils utilisés pour la réalisation de notre projet, ensuite nous présentons une démonstration globale des différentes fonctionnalités de notre application.

Chapitre V: Tests et Validation du Système Ce chapitre vise à démontrer que notre système répond à l'objectif souhaité par notre projet.

CHAPITRE I

Médiation des Données

1. Introduction

L'accès aux données est un sujet très important dans le monde informatique. Beaucoup de travaux ont été réalisés pour faciliter l'accès aux diverses données réparties d'une manière efficace et stable. Les entreprises, les établissements et les différentes sociétés ont développé des systèmes d'informations pour stocker, interroger et mettre à jour les données. Chacun de ces systèmes dispose d'une façon pour définir le format de données et surtout un mode d'accès aux données permettant aux utilisateurs d'interagir avec ces données. Ce type de système d'information est appelé source de données. Comment répondre à une requête qui nécessite d'accéder à plusieurs sources de données est une problématique importante depuis les années précédentes, car la diversité des sources de données entraine beaucoup de difficultés notamment en termes de performance **[1]**. L'objectif de l'intégration de données est de combiner les données résidant sur différentes sources et de fournir aux utilisateurs une vue unifiée et centralisée de ces données en masquant les détails sur l'hétérogénéité de ces sources, sachant qu'une source peut être décrite par sa localisation, le type de données qu'elle gère, le mode d'interrogation et le format de résultats. Pour notre thème on cherche à optimiser les requêtes du médiateur en utilisant les techniques du datamining.

2. Les Systèmes de Médiation

Les systèmes de médiation tentent de proposer à l'utilisateur un accès simple et puissant à l'information. Ils sont capables d'intégrer des sources de données diverses pour offrir à l'utilisateur une vue centralisée et uniforme des données, en masquant la diversité de localisations, de formats et de modes d'accès. Ils reposent sur l'architecture médiateurs/adaptateurs (Figure 1). Le médiateur procure à l'utilisateur ou à d'autres médiateurs (intermédiaires) une vue centralisée des données. Il y a un adaptateur par source de données, qui est chargé d'encapsuler la source. Il fournit pour les médiateurs un modèle de données et un mode d'accès uniformes. L'architecture médiateurs/adaptateurs est devenue une solution largement acceptée par la plupart des systèmes de fédération de données hétérogènes réparties. **[2] [3]**

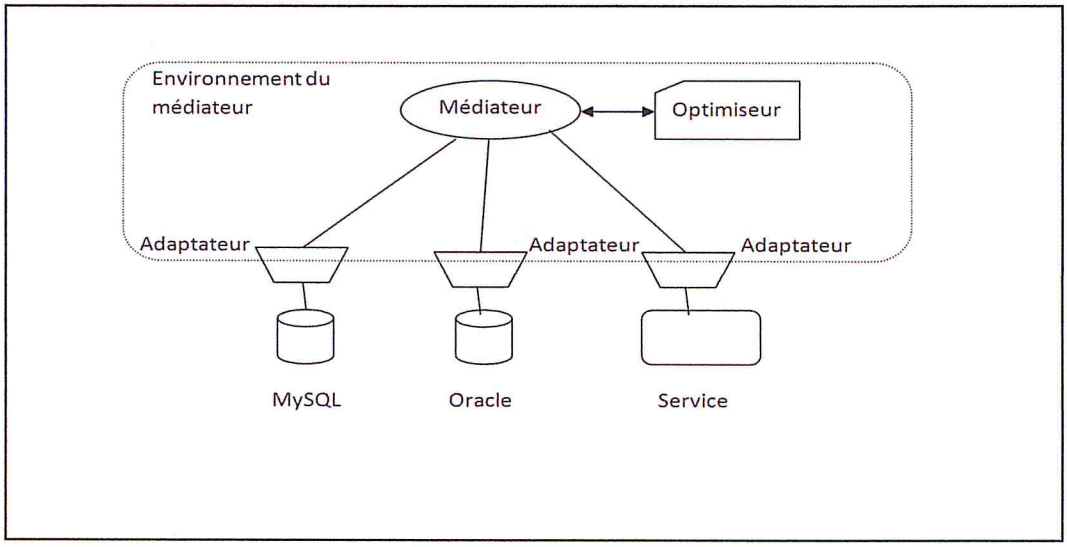


Figure 1 : Architecture d'un système de médiation. **[4]**

2.1. Le Médiateur

Le médiateur est un module logiciel recevant directement la requête d'un usager et devant la traiter. Celui-ci doit localiser l'information nécessaire pour répondre à la requête, résoudre les conflits schématiques et sémantiques, interroge les différentes sources et intègre les résultats partiels dans une réponse homogène et cohérente. Il doit: **[4] [5]**

- Accepter les requêtes des utilisateurs.
- Localiser les sources pertinentes.
- Décomposer les requêtes.
- Envoyer les plans d'exécution à faire exécuter par les adaptateurs des différentes sources.
- Combiner (recomposer) les résultats des adaptateurs.

2.2. L'Adaptateur (Wrapper)

Un Wrapper de données est un logiciel qui convertit les requêtes exprimées dans le modèle de médiation provenant d'un ou de plusieurs médiateurs vers le modèle des bases de données locales. Il convertit les données et les résultats d'une requête, du modèle des bases de données locales vers le modèle de médiation. Un Wrapper de données offre une interface homogène locale d'accès aux données sources (résolution des conflits syntaxiques). **[6]**

3. Traitement des Requêtes du Médiateur

L'utilisation d'un système de médiation permet d'exécuter des requêtes simples et complexes. Une requête posée en termes du schéma global est réécrite au niveau du médiateur pour donner naissance à plusieurs sous-requêtes. Chaque sous requête est transmise à la source correspondante via l'adaptateur associé. Les réponses livrées par les adaptateurs sont ensuite combinées, au niveau du médiateur, pour former la réponse à la requête initiale.

En conséquence, le traitement de ces requêtes nécessite une grande puissance de calcul du côté système. Le médiateur global est obligé d'optimiser le processus de traitement de requêtes et de répondre le plus rapidement possible aux utilisateurs. Les systèmes d'information contiennent toujours un module chargé à l'optimisation de requêtes, appelé optimiseur. Ce dernier analyse les requêtes pour déterminer un moyen de les traiter. En général, il existe de nombreuses solutions pour exécuter la même requête, parmi lesquelles certaines sont souvent beaucoup plus rapides que les autres. Comment choisir la solution la plus efficace devient la problématique essentielle d'un optimiseur. **[1]**

3.1. Le Processus Général pour le Traitement d'une Requête

Une requête est une expression décrivant les informations que l'utilisateur recherche parmi les sources de données. Cette expression est écrite en langage de requêtes défini par le système de médiation. Le rôle d'un système de médiation consiste à reformuler la requête d'un utilisateur en termes des schémas sources, puis à la décomposer en sous requêtes, qui seront envoyées aux différentes sources. Les résultats sont ensuite renvoyés au médiateur, qui les intègre avant de les transmettre à l'utilisateur.

Les tâches principales dans le traitement des requêtes sont :

La décomposition d'une requête : il s'agit à partir d'une requête posée sur une vue intégrée, de localiser les données intervenant dans sa résolution, de produire des sous-requêtes spécifiques à chacune des sources et de les ordonner. La localisation des sous-requêtes nécessite des structures spécifiques de gestion de méta-données.

La recomposition des résultats : une fois les sous-requêtes soumises à chacune des sources, il s'agit de savoir recomposer les différents résultats entre eux.

La requête utilisateur exprimée dans les termes du schéma global doit être reformulée en une requête exprimée dans les termes des schémas des sources de données.

La requête ne décrit que synthétiquement les informations recherchées, sans préciser la façon d'y accéder. Le système doit donc traduire l'expression automatiquement, via son module d'analyse de requêtes, en une séquence d'opérations élémentaires, appelée plan d'exécution, dont l'exécution pourra retourner les résultats envisagés. Mais la traduction n'est pas unique. En effet, plusieurs plans d'exécution peuvent être équivalents, c'est à dire donner le même résultat pour la requête. L'ensemble de ces plans d'exécution forme un espace de recherche. L'optimiseur doit employer un modèle de coût pour prévoir le coût d'exécution de chaque plan d'exécution afin d'en choisir celui avec un coût minimal, malgré que l'exploration de l'espace de recherche est parfois difficile pour les requêtes complexes car il existe tellement de solutions possibles que l'optimiseur doit utiliser une stratégie de recherche afin trouver la solution idéale en un temps acceptable, en évitant d'explorer tout l'espace de recherche. **[7]**

3.1.1. La Reformulation d'une Requête

Une des différences principales entre un système d'intégration de données à base d'un médiateur et un système traditionnel de base de données est que dans les systèmes à base du médiateur les utilisateurs posent leurs requêtes en termes du schéma global. Les données sont stockées dans les sources, organisées sous des schémas sources. Par conséquent, pour répondre aux requêtes, il est nécessaire d'établir une correspondance entre le schéma global et les schémas des sources de données (schémas locaux).

Le processeur de requête doit être capable de reformuler une requête écrite sur le schéma global en une requête écrite sur les schémas des sources de données.

On distingue trois approches pour établir la correspondance entre le schéma global et les schémas des sources de données à intégrer. **[4]**

3.1.1.1. L'Approche GAV

Un schéma global est considéré comme une vue sur des schémas sources. GAV (Global As View), a été la première à être proposée pour l'intégration de données. Dans l'approche GAV, pour chaque relation utilisée dans le schéma global, on définit une vue composée des termes des relations des schémas sources. La transformation d'une requête écrite en utilisant le schéma global en une requête écrite en utilisant des schémas sources

est une opération simple, cette simplicité est considérée comme avantage de l'approche GAV.

Les inconvénients de cette approche est qu'il suppose que les sources à intégrer soient connues à l'avance, ainsi une modification sur l'ensemble des sources ou sur leurs schémas entraîne une reconsidération complète du schéma global. [7]

3.1.1.2. L'Approche LAV

LAV (Local As View) Chaque relation d'un schéma source est définie comme vue sur le schéma global. Dans le cas d'une approche LAV, la requête sur le schéma global doit être reformulée suivant les schémas sources, telle que chaque source est spécifiée de manière indépendante.

Cette approche a l'avantage qu'elle est très flexible par rapport à l'ajout (ou la suppression) de sources de données à intégrer, cela n'a aucun effet sur le schéma global, seules des vues doivent être ajoutées (ou supprimées). De l'autre coté, le prix à payer pour cette flexibilité et cette simplicité de mise à jour est la complexité de la construction des réponses car la reformulation de requêtes en termes de vues est la seule possibilité pour obtenir des requêtes exprimées en termes des vues, que l'on doit ensuite exécuter pour interroger les sources de données. La reformulation de la requête dans l'approche LAV est plus complexe que GAV, parce qu'il faut reformuler la requête par un mécanisme d'inférence qui réécrit les relations du schéma global en relations des schémas sources. [7]

3.1.1.3. L'approche GLAV

GLAV (Global and Local As View) est une combinaison entre GAV et LAV prend les avantages des deux approches prétendantes. Elle offre plus de flexibilité à la mise à jour par les utilisateurs ou les sources locales. GLAV associe a une requête conjonctive écrite sur le schéma global à une requête conjonctive écrite sur des schémas sources. [7]

4. Le Modèle du Coût

Le but d'un modèle de coût est d'estimer le coût d'exécution des plans. Il permet ainsi de choisir le meilleur plan d'exécution ayant le moindre coût d'exécution. Le modèle de coût contient, d'une part, des statistiques sur les données et sur le système SGBD et

d'autre part, des formules pour évaluer la taille des résultats intermédiaires pour déterminer le coût des plans.

Ces formules reposent en général sur un certain nombre de paramètres et d'hypothèses simplificatrices. L'unité de mesure du coût dépend de l'objectif d'optimisation. Le coût peut être mesuré :

- En unité de temps si l'objectif est de réduire le temps de réponse du plan.
- En unité de travail si l'objectif est de réduire la consommation de ressources du système.
- En nombre de connexions si l'objectif est de réduire les appels aux sources de données.
- En unité monétaire si l'objectif est de réduire le prix d'exécution de la requête, dans le cas certains accès aux sources sont coûteux.
- En unité de flux de données si l'objectif est de réduire la taille de données circulées dans le réseau à un moment donné. **[1] [8]**

Le modèle de coût est une fonction qui accepte en paramètre un plan d'exécution et retourne son coût. Le coût du plan est calculé en cumulant le coût des opérateurs élémentaires, de proche en proche selon l'ordre défini par le plan d'exécution lui-même, jusqu'à obtenir le coût total du plan. Pour réaliser cette estimation de coût, une formule de coût est associée à chaque opérateur. La formule de coût dépend des caractéristiques de l'opérateur et aussi de son algorithme d'implémentation.

Dans un système de médiation, les sources de données hétérogènes sont normalement autonomes et ne divulguent pas les informations concernant leurs traitements des opérations. L'estimation de coût des plans pose deux nouveaux problèmes :

- Les opérateurs dans le plan sont répartis sur les sources.
- Les médiateurs ne connaissent pas l'implémentation des opérateurs traités sur les sources, et donc ne possèdent pas les informations de coût précisées.

Cependant, de nombreuses sources de données fournissent des informations partielles de coût des opérateurs qu'elles sont capables traiter. Ces informations permettent au système d'avoir une meilleure estimation de coût pour l'optimiseur, qui pourra donc bien choisir un plan d'exécution performant.

La performance d'un optimiseur dépend de deux aspects essentiels, la précision du modèle de coût appliqué et la stratégie d'optimisation choisie. Le modèle de coût peut être considéré comme une fonction générique qui accepte en paramètre un plan d'exécution physique et retourne, après le calcul, le coût d'exécution total du plan. **[8] [9]**

4.1. Estimation des Coûts [1]

Pour le calcul du coût total, il faut prendre en considération le coût de communication, le coût de traitement du médiateur et le coût des différents traitements des adaptateurs.

Coût –total = coût –communication + coût –médiateur + coût –adaptateur

4.1.1. Coût –communication

C'est le temps de communication entre le médiateur et les adaptateurs. Ce coût est lié à la taille des données transférées (taille de la requête, cardinalité du résultat et la taille d'un résultat) et même au paramétrage du réseau (débit de communication, protocole utilisé). Ce coût peut être estimé come suit :

Coût –communication = temps-d'initialisation + temps-transmission

$$= \text{temps-d'initialisation} + \frac{\text{nombre–d'octets–a–transmettre}}{\text{debit–en–octets/secondes}}$$

Ou le temps d'initialisation est le temps d'établissement de la connexion auquel on ajoute le temps de transmission des données.

4.1.2. Coût-médiateur

C'est le temps que le médiateur met à traiter la requête et à traiter les différentes réponses renvoyées par les adaptateurs avant de donner le résultat final, ce coût dépend des caractéristiques de la machine sur la quelle tourne le médiateur (CPU, mémoire) et a la façon d'implémentation des différents calculs sur les données (opérateur de jointure, sélection, projection et la reconstruction du résultat) au sein du médiateur.

Coût-médiateur = temps-CPU

= temps-par-instruction X nombre-d'instructions

4.1.3. Coût-adaptateurs

C'est le temps que les adaptateurs mettent à répondre a une sous requête qui leur a été envoyée par le médiateur. Ce coût est lié au traitement de la requête par l'adaptateur (conversion de la requête et des résultats), mais surtout au coût de traitement de la source (temps d'exécution d'une requête). Le coût d'un adaptateur pour une requête donnée peut être estimé comme suit :

Coût-adaptateur = temps-CPU + temps-entrees-sorties

= temps-par-instruction X nombre-d'instructions

+ temps-d'une-entree / sortie X nombre-d'entrees-sorties

5. Conclusion

Dans ce chapitre, nous avons élaboré un ensemble de définitions qui nous permettre de mieux comprendre les déférentes approches, aspects et composants des systèmes de médiations, ainsi du problématique posée concernant l'exécution des requêtes afin d'augmenter le niveau de performance et d'exécuter les requêtes d'une manière optimale.

CHAPITRE II

Data Mining

1. Introduction

Le Data Mining est un domaine pluridisciplinaire regroupe des techniques et des méthodes de plusieurs domaines permettant, à partir d'une très importante quantité de données brutes, d'en extraire de façon automatique ou semi-automatique des informations cachées, pertinentes et inconnues auparavant en vue d'une utilisation industrielle ou opérationnelle de ce savoir. Il peut également mettre en avant les associations et les tendances et donc servir d'outil de prévisions au service de l'organe décisionnel. On distingue le Data Mining supervisé qui sert essentiellement à la classification des données et le Data Mining non supervisé qui est utilisé dans la recherche d'associations ou de groupes d'individus.

2. Définition du Data Mining

Le data mining est un ensemble des techniques et de méthodes du domaine des statistiques, des mathématiques et de l'informatique permettant l'extraction, à partir d'un important volume de données brutes, de connaissances originales auparavant inconnues. Il vise à découvrir de l'information cachée que les données renferment et que l'on découvre à la recherche d'associations. Il permet donc de mieux comprendre les liens entre des phénomènes qui apparaissent distincts. **[14]**

3. Les tâches du Data Mining [12] [13]

Lors de la fouille automatique des données le Datamining va utiliser des méthodes/algorithmes plus ou moins complexes, établir des corrélations entre ces données, définir des comportements types et exécuter différentes tâches. Dans ce qui suit une présentation de l'ensemble des tâches du Data Mining :

3.1. Classification

La classification est un processus qui permet d'organiser un ensemble de données en classe cohérente et homogène tel que les éléments appartenant a une classe partagent de nombreuses caractéristiques communes et donc se rassemblent fortement, le but de la classification est de formé des classes qui sont bien isolés tel que les éléments appartenant a deux classes différentes ne se rassembles pas, ils ne partagent pas les même caractéristiques. Les techniques les plus appropriées à la classification sont: les

arbres de décision, le raisonnement basé sur la mémoire ainsi que l'analyse de liens. On distingue deux types de classification :

3.1.1. Classification Supervisé

Dans cette approche les classes existent a priori, c'est-à-dire on dispose d'un ensemble de classes d'objets. Le but de cette classification est de pouvoir allouer une classe à un nouvel objet.

3.1.2. Classification Non Supervisé

Dans cette approche les classes ne sont pas connues a priori et les objets sont non étiquetés. Le but de cette classification est de regrouper dans un même groupe, classe, cluster ou bien segment les objets considérés comme similaires.

3.2. Estimation

Recherche de la valeur d'un ou de plusieurs paramètres, d'une loi statistique à partir d'observation ou de sondages. La technique la plus appropriée à l'estimation est le réseau de neurones.

3.3. Prédiction

Action de prédire ca veut dire annoncer d'avance ce qui doit se produire soit par intuition ou divination, soit par certaines règles, soit par conjecture ou raisonnement, elle se base sur les caractéristiques du passé et présent et le résultat se trouve dans le future. Les techniques les plus appropriées à la prédiction sont: l'analyse du panier de la ménagère (ou règles d'association), le raisonnement basé sur la mémoire, les arbres de décision et les réseaux de neurones.

3.4. Association

Action d'associer quelqu'un a quelque chose, des choses diverses entre eux. Il s'agit de trouver des similitudes ou des associations. Le sequencing est le terme anglais utilise pour préciser que l'association se fera dans le temps. La technique la plus appropriée à l'association est l'analyse du panier de la ménagère.

Havasu utilise l'approche LAV et suppose que les descriptions de source en termes de schéma de médiateur sont fournies dans le cadre d'un processus d'enregistrement de source.

La requête d'utilisateur est posée sur le schéma de médiateur. Elle est envoyée ensuite avec la métrique d'utilité à Multi-R, Le module d'optimisation de la requête. Multi-R utilise les statistiques fournies par le StatMiner pour soutenir une optimisation multi-objective de requête fondée sur les coûts. **[15] [16]**

6. La Similarité

La similarité est la quantité qui reflète la force du rapport entre deux objets ou deux caractéristiques. **[17]**

6.1. Définitions Relatives à la Similarité

Dans ce qui suit nous présentant des définitions relatifs a la similarité :

6.1.1. Similarité des Concepts [17]

La similarité S entre les concepts, est une fonction d'une paire d'entités à un nombre réel exprimant la similarité entre ces deux entités telle que:

Soit C l'ensemble des concepts :

- $\forall c1, c2 \in C, S(c1, c2) \geq 0$ (positivité)
- $\forall c1, c2, c3 \in C, S(c1, c1) \geq S(c2, c3)$ et $S(c1, c1) = S(c1, c2) \Leftrightarrow c1 = c2$ (auto similarité ou maximalité)
- $\forall c1, c2 \in C, S(c1, c2) = S(c2, c1)$ (symétrie)
- $\forall c1, c2, c3 \in C, S(c1, c2) = S(c2, c3) \Rightarrow S(c1, c2) = S(c1, c3)$ (transitivité)
- $\forall c1, c2 \in C, S(c1, c2) \leq \infty$ (finitude)

6.1.2. Dissimilarité [17]

La dissimilarité est parfois utilisée au lieu de la similarité. Elle est définie de manière analogue à la similarité, sauf qu'elle n'est pas transitive. Soit C l'ensemble des concepts :

- $\forall c1, c2 \in C, DS(c1, c2) \geq 0$ (positivité)
- $\forall c1, c2, c3 \in C, DS(c1, c1) \leq DS(c2, c3)$ et $DS(c1, c1) = 0$ (minimalité)
- $\forall c1, c2 \in C, DS(c1, c2) = DS(c2, c1)$ (symétrie)
- $\forall c1, c2 \in C, DS(c1, c2) \leq \infty$ (finitude)

6.1.3. Distance [18] [17]

La distance est une mesure utilisée aussi souvent que les mesures de similarité. Elle mesure la dissimilarité des deux entités, si la valeur de la fonction de similarité de deux entités est élevée, la distance entre elles est petite et vice-versa sa.

La distance D est une fonction de la dissimilarité satisfaisant la définitivité et l'inégalité triangulaire, soit C l'ensemble des concepts :

- $\forall c1, c2 \in C, D(c1, c2) = 0 \Leftrightarrow c1 = c2$ (définitivité)
- $\forall c1, c2, c3 \in C, D(c1, c2) + D(c2, c3) \geq D(c1, c3)$ (inégalité triangulaire)

6.1.4. Normalisation

Les valeurs de similarité sont souvent normalisées pour pouvoir être combinées dans des formules plus complexes. Si la valeur de similarité et la valeur de dissimilarité entre deux entités sont normalisées, notées S et DS, alors on a $S + DS = 1$.

Une mesure est une mesure normalisée si les valeurs calculées par cette mesure ne peuvent varier que dans un intervalle de 0 à 1. Ces valeurs calculées sont appelées valeurs normalisées. Les fonctions du calcul sont appelées fonctions de normalisation. [17]

6.2. Mesures de Similarité [17]

Les mesures de similarité, de dissimilarité et la distance peuvent être classées selon la nature des entités que l'on veut comparer : des termes, des chaînes de caractères, des structures, des instances (des individus des classes), des modèles théoriques…etc.

Il existe différentes mesures de similarité, catégorisées et classifiées selon les techniques utilisées :

6.2.1. Mesures simples

Les mesures simples sont divisées à leur tour en quatre types de mesures :

6.2.1.1. Mesures terminologiques

Elles sont employées pour calculer la valeur de similitude des entités textuelles, telles que des noms, des métadonnées sur les noms, des étiquettes, des commentaires,… **[22] [18]**

Nous trouvons dans cette méthode deux approches : l'approche syntaxique et l'approche lexicale, appelée aussi linguistique **:**

- **L'approche syntaxique** : effectue la correspondance à travers les mesures de dis-similarité des chaines de caractères (par exemple, la distance de Hemming qui permet de calculer le nombre de positions dans lesquelles deux chaînes de caractères se différent). **[22]**
- **L'approche lexicale ou linguistique** :

Les méthodes linguistiques utilisant des ressources externes (dictionnaires, taxonomies,…) la similarité entre deux entités représentées par des termes est calculée à partir des liens sémantiques déjà existants dans les ressources externes.

les informations exploitées peuvent être celles intrinsèques (des propriétés linguistiques internes des termes telles que des propriétés morphologiques ou syntaxiques) ou celles extrinsèques en effectuant la correspondance à travers les relations lexicales (par exemple, synonymie, hyponymie, etc.). **[22]**

6.2.1.2. Mesures structurelles

Ce sont des méthodes qui déduisent la similarité de deux entités en exploitant des informations structurelles lorsque les entités en question sont reliées aux autres par des liens sémantiques ou syntaxiques, formant ainsi une hiérarchie ou un graphe des entités.

Nous appelons méthodes structurelles internes les méthodes qui calculent la similarité entre deux concepts en exploitant les informations relatives à leur structure interne (restrictions et cardinalités sur les attributs, valeurs des instances,…) et méthodes structurelles externes, les méthodes qui se servent de la structure hiérarchique et se basent sur des techniques de comptage d'arcs pour déterminer la similarité sémantique entre deux entités. **[18] [19]**

6.2.1.3. Mesures extensionnelles

Elles déduisent la similarité entre deux entités qui sont notamment des concepts ou des classes en analysant leurs extensions (leurs ensembles d'instances).

Chaque instance peut être représentée par un vecteur de noms et/ou de valeurs. Des calculs de similarités entre vecteurs permettent de comparer les instances. **[19]**

6.2.1.4. Mesures sémantiques

La similarité sémantique est une évaluation du lien sémantique entre deux concepts dont le but est d'estimer le degré par lequel les concepts sont proches dans leur sens.

La définition de la similarité sémantique repose sur trois suppositions. La similarité entre deux concepts est liée aux caractéristiques qu'ils ont en commun (plus ils ont de caractéristiques communes, plus les concepts sont similaires) et à leurs différences (plus deux concepts sont différents, moins ils sont similaires). La similarité maximale est obtenue lorsque deux concepts sont identiques. **[20]**

Les mesures de similarité sont classées par rapport aux caractéristiques des concepts pertinents et reposent soit sur la distance entre les concepts à travers leurs liens, soit sur l'information contenue par les concepts, soit sur les deux.

6.2.2. Mesures combinées

Les différentes techniques citées auparavant peuvent ensuite être utilisées ensemble dans un algorithme « hybride » (deux ou plusieurs techniques dans un même algorithme) ou en un paramétrage d'algorithmes exécutés en parallèle (composite). **[21]**

6.2.2.1. Combinaison séquentielle (hybride)

Elles combinent plusieurs mesures lorsqu'une seule est insuffisante. La méthode la plus simple pour combiner les mesures est l'utilisation séquentielle de ces dernières en choisissant un ordre d'exécution. Par exemple, nous choisissons de lancer une mesure terminologique avant de lancer une autre mesure structurelle ou sémantique. **[19] [21]**

6.2.2.2. Combinaison parallèle (composite)

Une autre manière de combiner les résultats des différentes mesures consiste tout d'abord à lancer parallèlement plusieurs mesures, puis par la suite à combiner leurs résultats. **[21]**

7. Conclusion

Dans ce chapitre, nous avons élaboré un ensemble de définition concernant le domaine de Data Mining, avec les approches existantes pour l'optimisation des requêtes du médiateur qui utilisent les techniques de ce domaine et se basant sur le calcul de mesure de similarité afin de trouver le meilleur plan d'exécution de requête et pour obtenir la réponse le plus rapidement possible.

CHAPITRE III

Conception du Système

1. Introduction

Après avoir élaboré les chapitres précédents respectivement l'état de l'art sur les systèmes de médiation et les algorithmes du Data Mining utilisés pour le traitement des requêtes. Nous passons dans ce chapitre à la conception et la modélisation du système, commençant par la spécification des besoins, puis la conception du système de médiation et a la fin la conception du système d'optimisation des requêtes dans les systèmes de médiation en utilisant le Data Mining.

2. Le Cycle de vie d'un logiciel [24] [25]

Le cycle de vie d'un logiciel, désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, ainsi la conformité du celui ci avec les besoins exprimés. Il existe plusieurs modèles de cycle de vie, parmi ces modèles le modèle de cycle de vie en cascade.

Dans ce modèle le principe est très simple : chaque phase se termine à une date précise par la production de certains documents ou logiciels. Les résultats sont définis sur la base des interactions entre étapes, ils sont soumis à une revue approfondie et on ne passe à la phase suivante que s'ils sont jugés satisfaisants.

Le cycle de vie d'un logiciel comprend généralement au minimum les étapes suivantes :

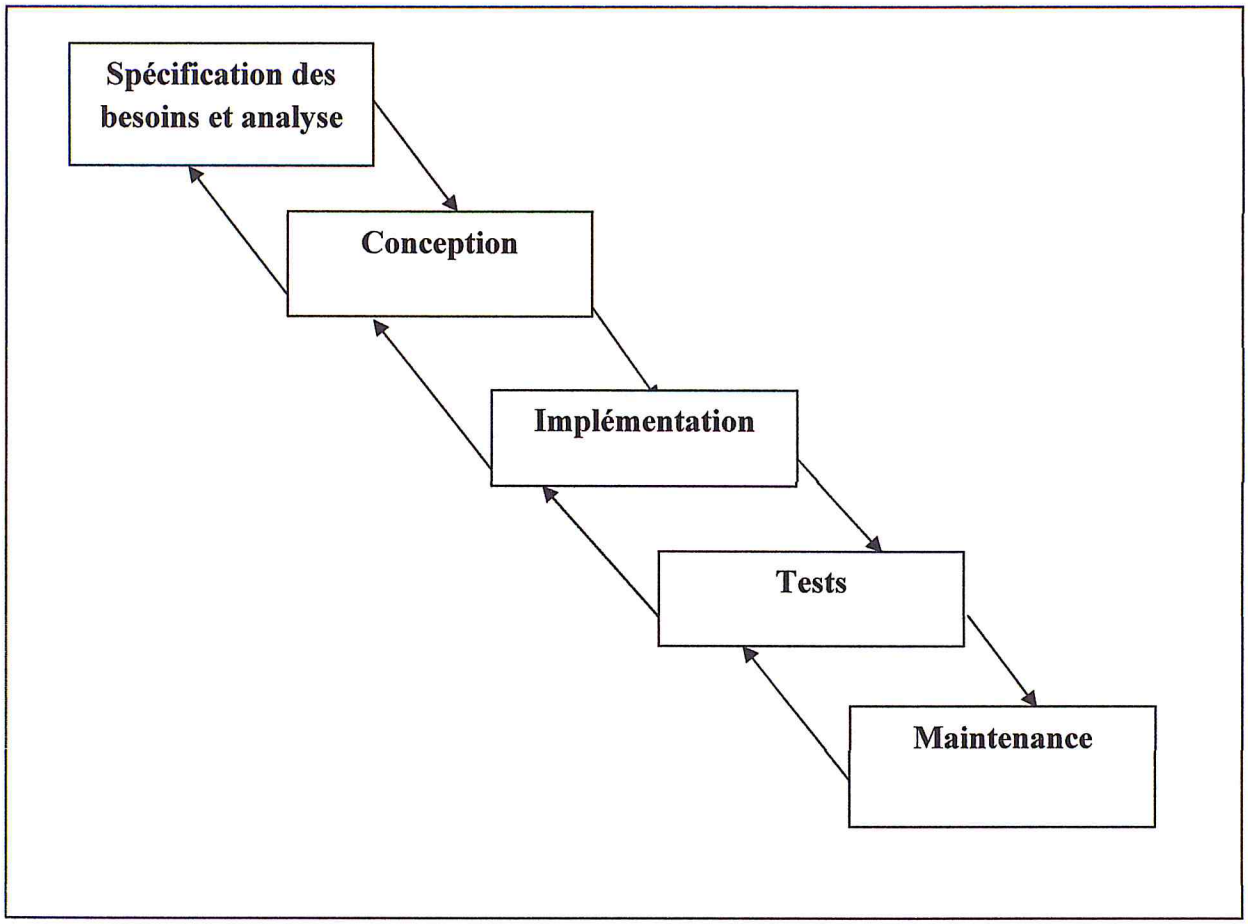


Figure 2: Le cycle de vie de modèle en cascade. **[24]**

2.1. Spécification des besoins et analyse

- **Spécification 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 (technique) 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.

- **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.

2.2. 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 et constitue un point de départ à l'implémentation,elle décompose le travail d'implémentation en sous-système et crée une abstraction transparente de l'implémentation.

2.3. 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.

2.4. 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.

2.5. Maintenance

Elle comprend toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel.

3. Langage de Modélisation 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, BOOCH1 (GRADY BOOCH le concepteur de la méthode) et OOSE. 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.

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. **[22]**

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.

UML 2.0 comporte ainsi treize types de diagrammes représentant autant de vues distinctes pour représenter des concepts particuliers du système d'information, pour la conception et la modélisation de notre système nous avons choisi quelques diagrammes de ce langage. **[22] [23]**

4. Modélisation du Système

Pour modéliser notre système il faut d'abord passer par les étapes suivantes :

4.1. Spécification des Besoins et Analyse

Il s'agit de recenser les fonctionnalités du système. Qui sont les besoins spécifiant un comportement d'entrée / sortie du système. Les besoins fonctionnels déduits à partir de notre étude concernant le système de médiation se résument ci-dessous :

- Lors de sa connexion à l'interface de l'application l'utilisateur lance sa requête dans un langage commun.
- Le médiateur est chargé à la décomposition de la requête globale, puis il envoi les sous requêtes au adaptateur attaché a la source appropriée.
- L'adaptateur est chargé de traduire les requêtes du langage global au langage local associé à la base de données.
- L'adaptateur est chargé de traduire le résultat fournit par les sources de données du langage local au langage global.
- Le médiateur est chargé de fusionner les résultats fournit par les adaptateurs a l'utilisateur.

De ce qui concerne les besoins fonctionnels d'optimisation de requêtes, elles se résument ci-dessous :

- L'administrateur doit effectuer l'étape d'apprentissage et construire des classes homogènes de requêtes.
- Le médiateur est chargé au traitement de la requête globale posée par l'utilisateur et de retourner la réponse en un temps optimal on appliquant les techniques du Data Mining.

Après la spécification des besoins, il faut maintenant 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 **[22]**. Le tableau ci-dessous identifie les acteurs de notre système et décrit la mission de chacun.

Acteurs	Mission
Utilisateur	L'utilisateur s'occupe du lancement de la requête mais avant il doit s'authentifier.
Administrateur	L'application doit permettre aux administrateurs de : • S'authentifier : L'administrateur doit s'authentifier avec un nom et un mot de passe. • L'administrateur peut jouer le rôle d'un utilisateur. • L'administrateur peut consulter les différents schémas du système. • L'administrateur peut lancer l'étape d'apprentissage. • L'administrateur gère les comptes qui peuvent accéder au système.

Tableau 1: Les acteurs du système.

4.1.1. Diagramme de Cas d'Utilisation

L'identification des cas d'utilisation donne un aperçu des fonctionnalités futures que doit implémenter le système. Ce diagramme représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système et assure la relation entre l'utilisateur et les objets que le système met en œuvre, la figure suivante représente le diagramme de cas d'utilisation global du système. **[22]**

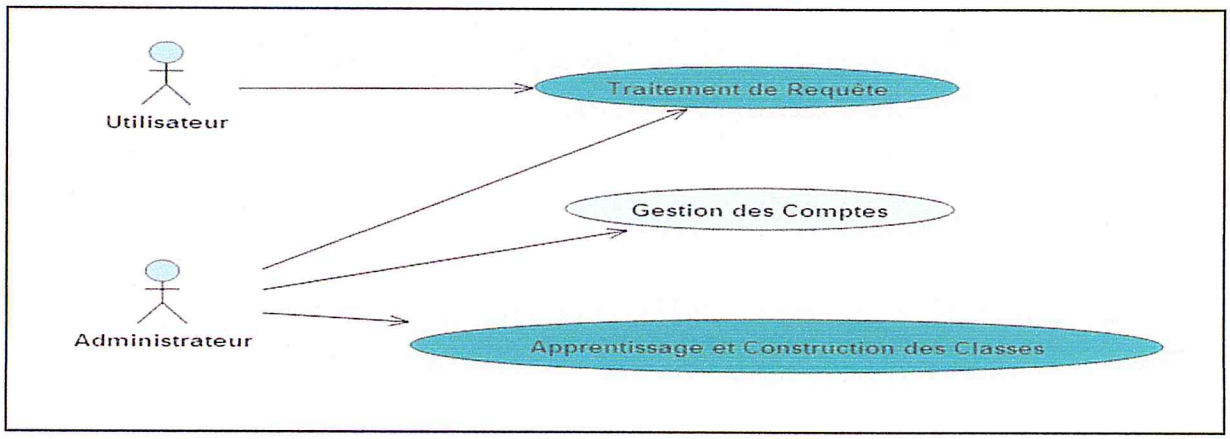


Figure 3 : Diagramme de Cas d'utilisation Global du Système.

La figure suivante représente le cas d'utilisation « Apprentissage et Construction des Clases » d'une manière plus détaillée :

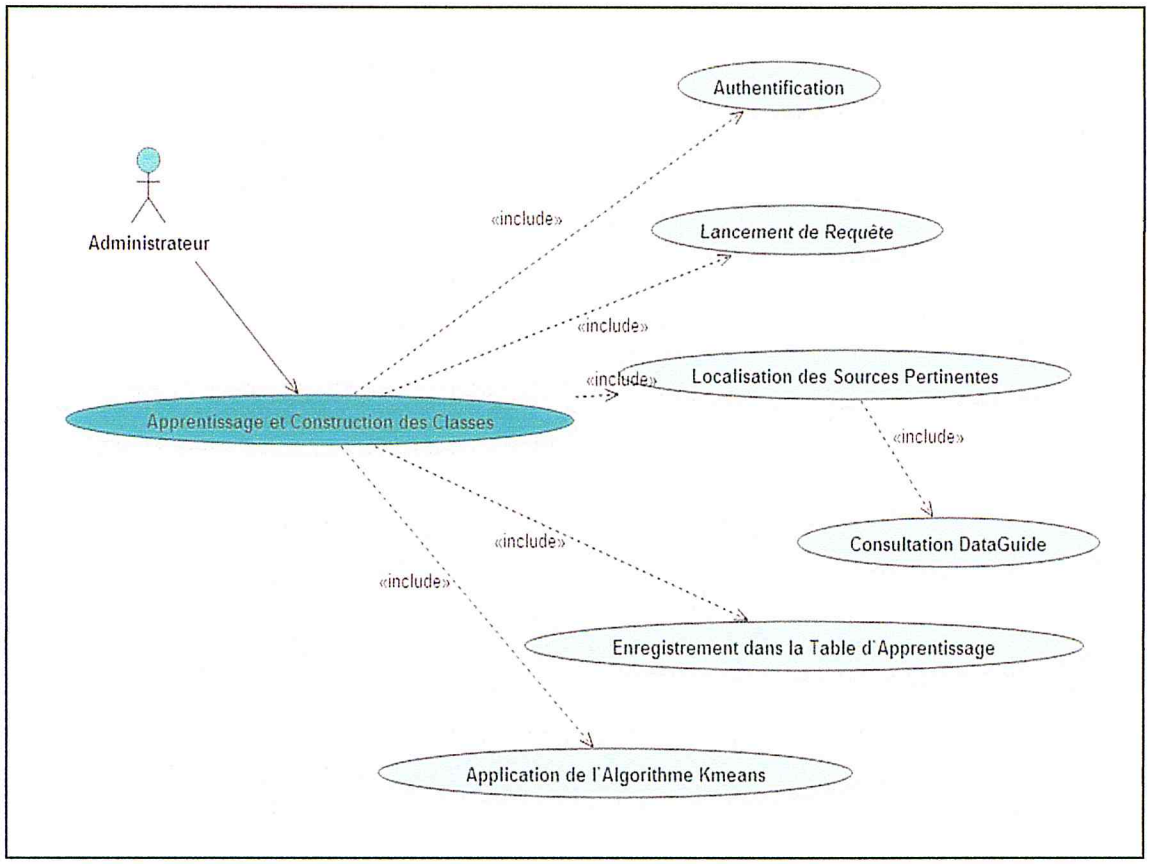


Figure 4: Diagramme de Cas d'utilisation « Apprentissage et Construction des Classes ».

De la même manière le cas d'utilisation « Traitement de Requête » peut aussi être détaillé comme il est montré ci-dessous :

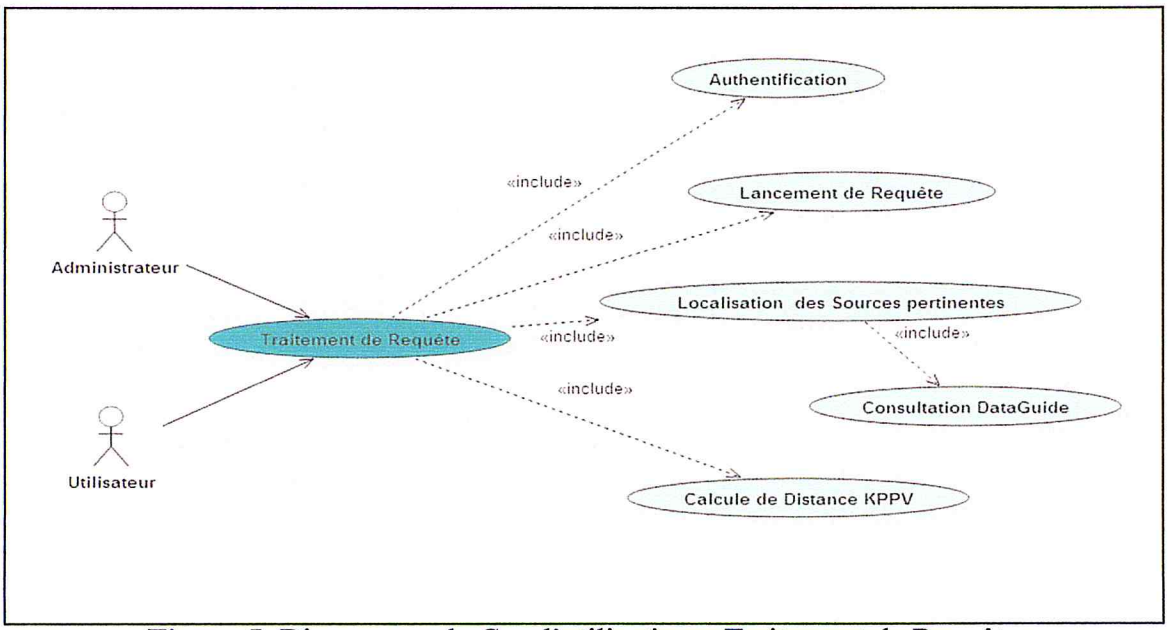


Figure 5: Diagramme de Cas d'utilisation « Traitement de Requête ».

4.1.2. Diagramme de Séquence

Le diagramme de séquence est un diagramme d'UML qui sert à modéliser l'aspect dynamique du système. Il permet de visualiser les messages échangés entre les objets du système représentés sur l'axe horizontal du diagramme. L'axe vertical représente le temps. **[22]**

Les figures suivantes représentent respectivement les diagrammes de séquences « Localisation des Sources Pertinentes », « Enregistrement dans la Table Apprentissage » et « Application de Kmeans ».

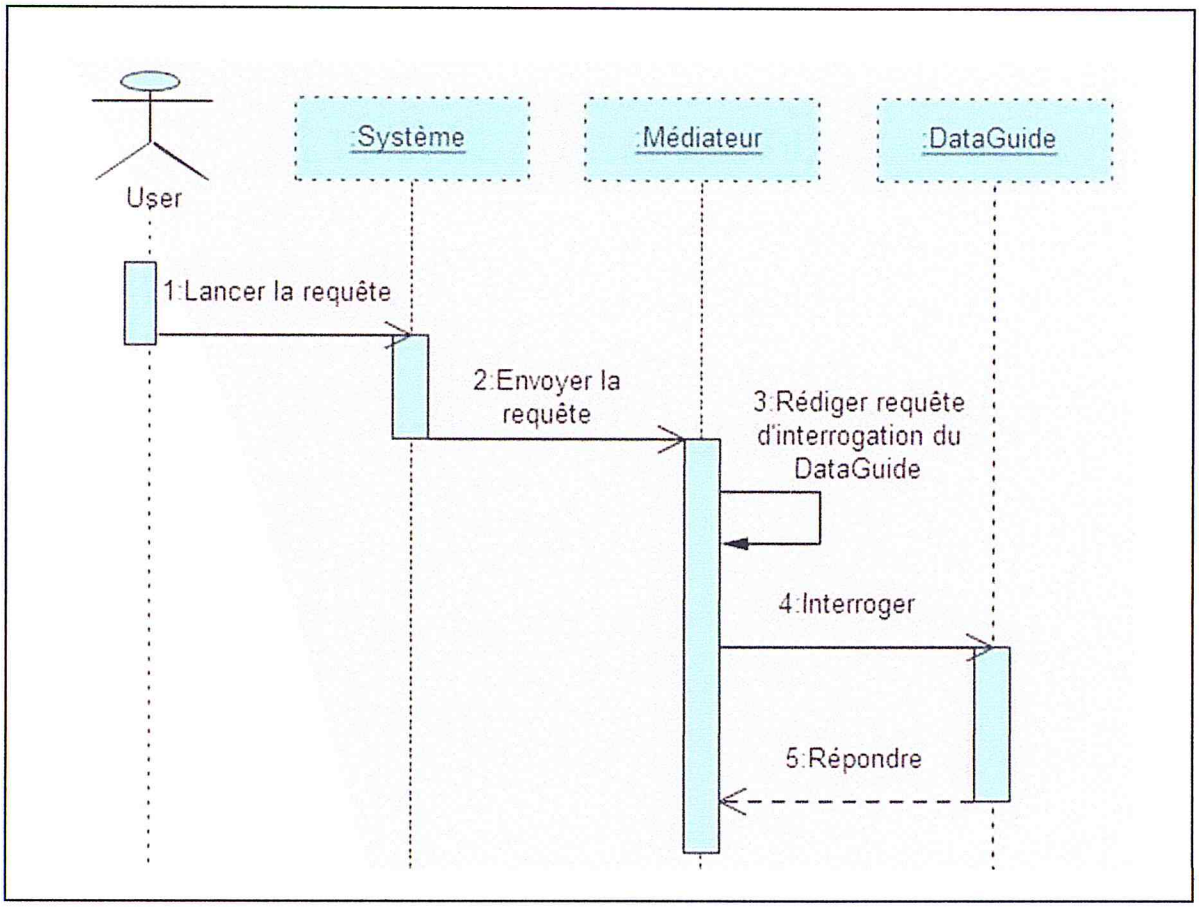


Figure 6: Diagramme de Séquence « Localisation des Sources Pertinentes».

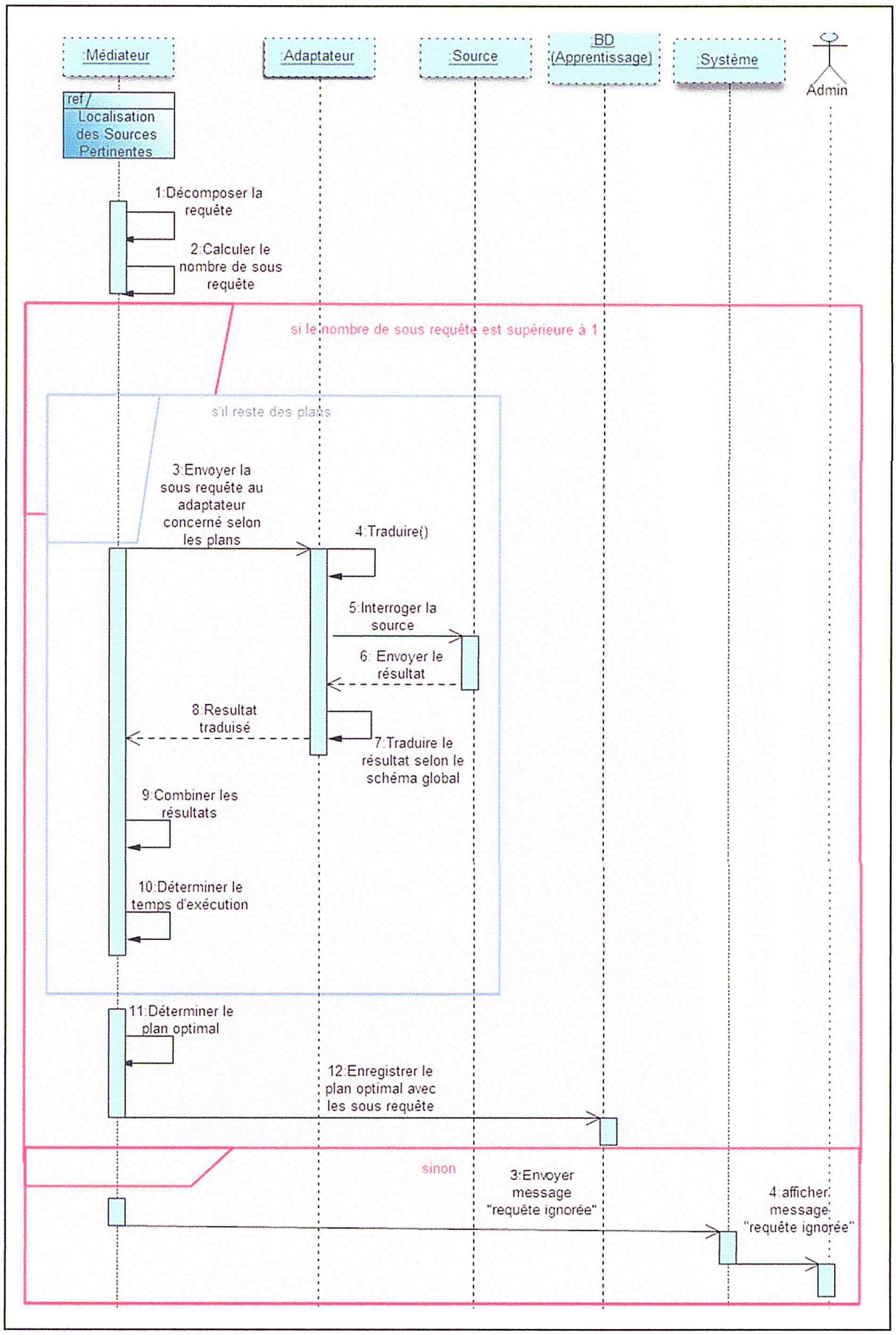


Figure 7 : Diagramme de Séquence « Enregistrement dans la Table Apprentissage».

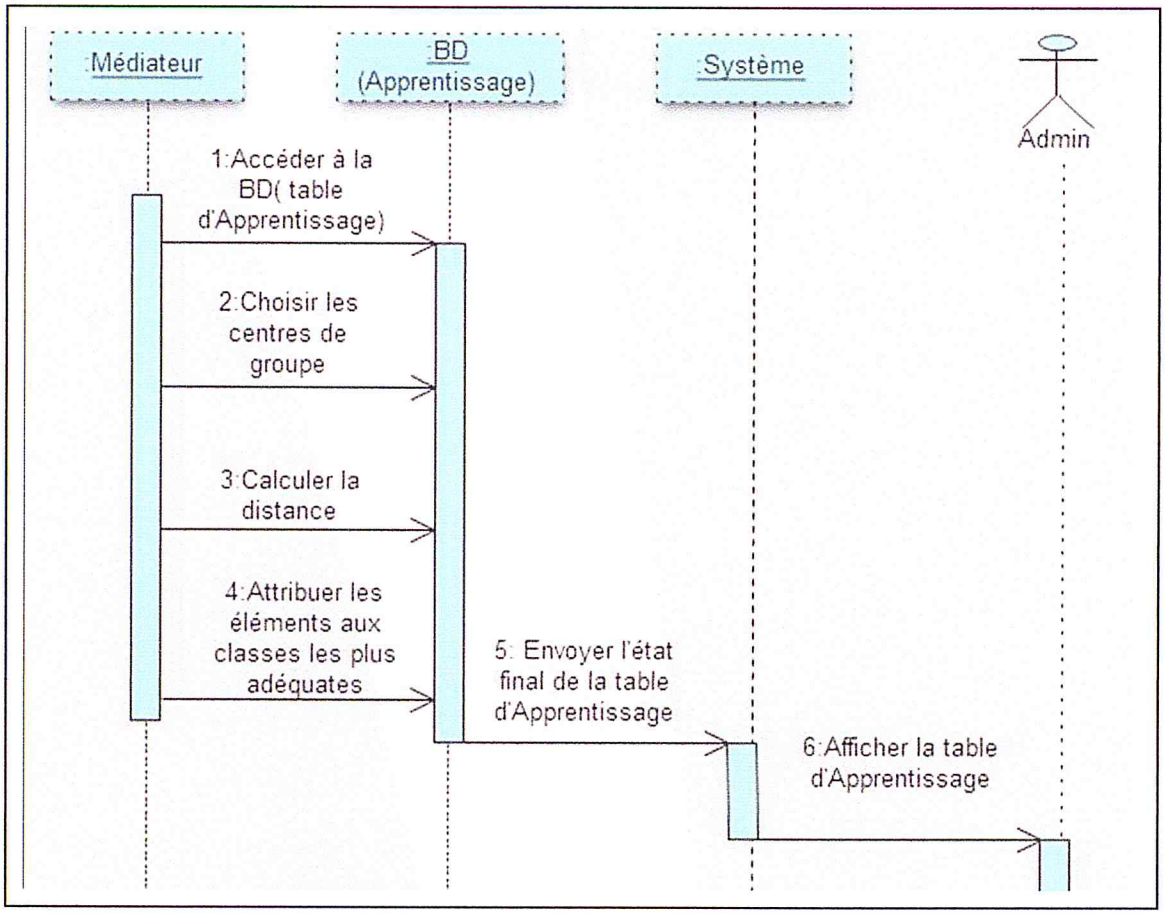


Figure 8: Diagramme de Séquence « Application de Kmeans».

4.1.3. Diagramme d'Activité

Le digramme d'activité est un diagramme dynamique d'UML, permettant de présenter l'enchainement détaillé des opérations, dans ce qui suit nous avons présenté deux diagramme d'activité le premier pour « l'Apprentissage et la Construction des Classes » et le deuxième pour « le Traitement de requête ». **[22]**

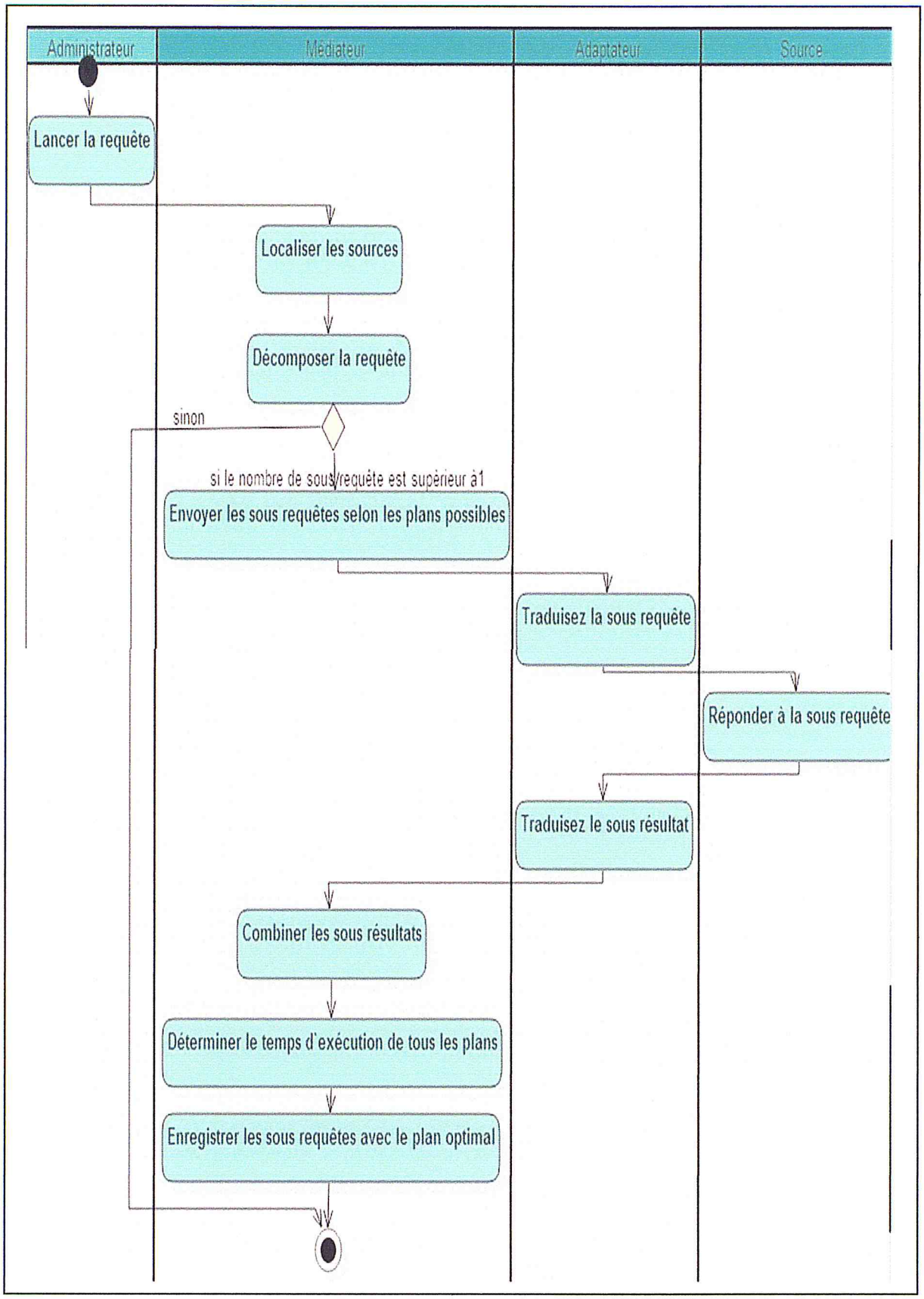


Figure 9: Diagramme d'Activité « Apprentissage et Construction des Classes ».

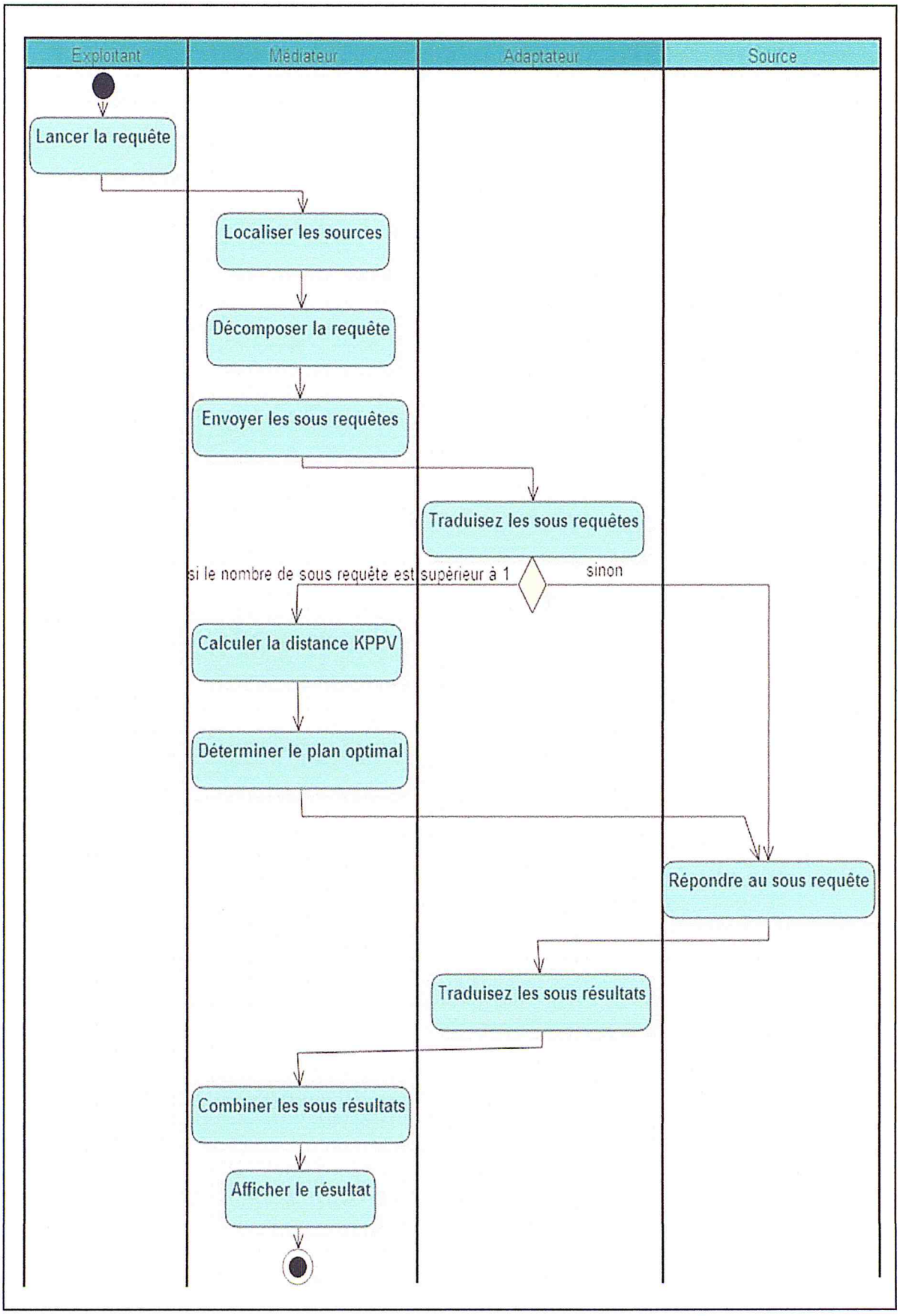


Figure 10: Diagramme d'Activité « Traitement de Requête ».

4.2. Conception du Système

La conception constitue le point de départ de l'implémentation, elle permet d'accueillir une compréhension approfondie du système.

4.2.1. Modélisation du contexte

La figure suivante représente le contexte du notre système :

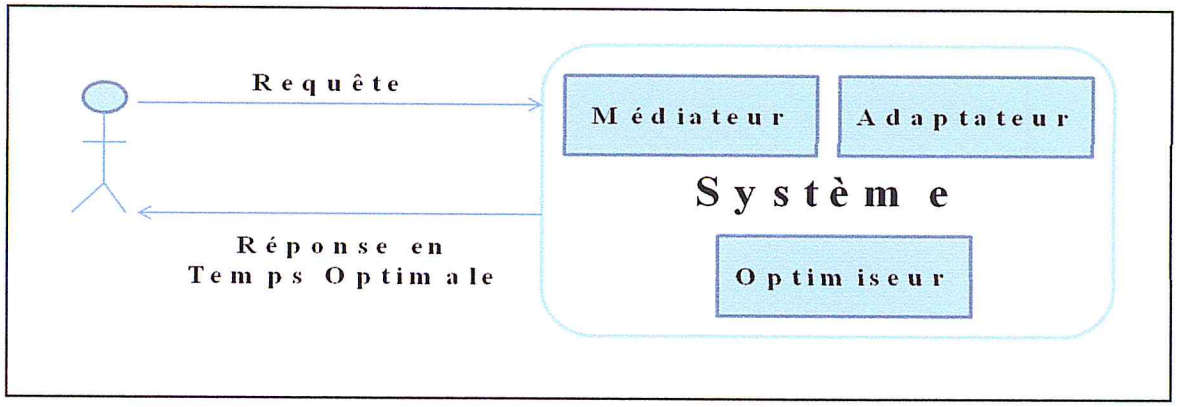


Figure 11: Diagramme de contexte du système.

4.2.2. Diagramme de Paquetage

Le diagramme de Paquetage permet le regroupement des classes logicielles en sous-systèmes, leur objectif est d'encapsuler et décomposer la complexité, faciliter le travail en équipes, faciliter la réutilisation et l'évolutivité, la (figure 13) représente notre diagramme de paquetage. **[23]**

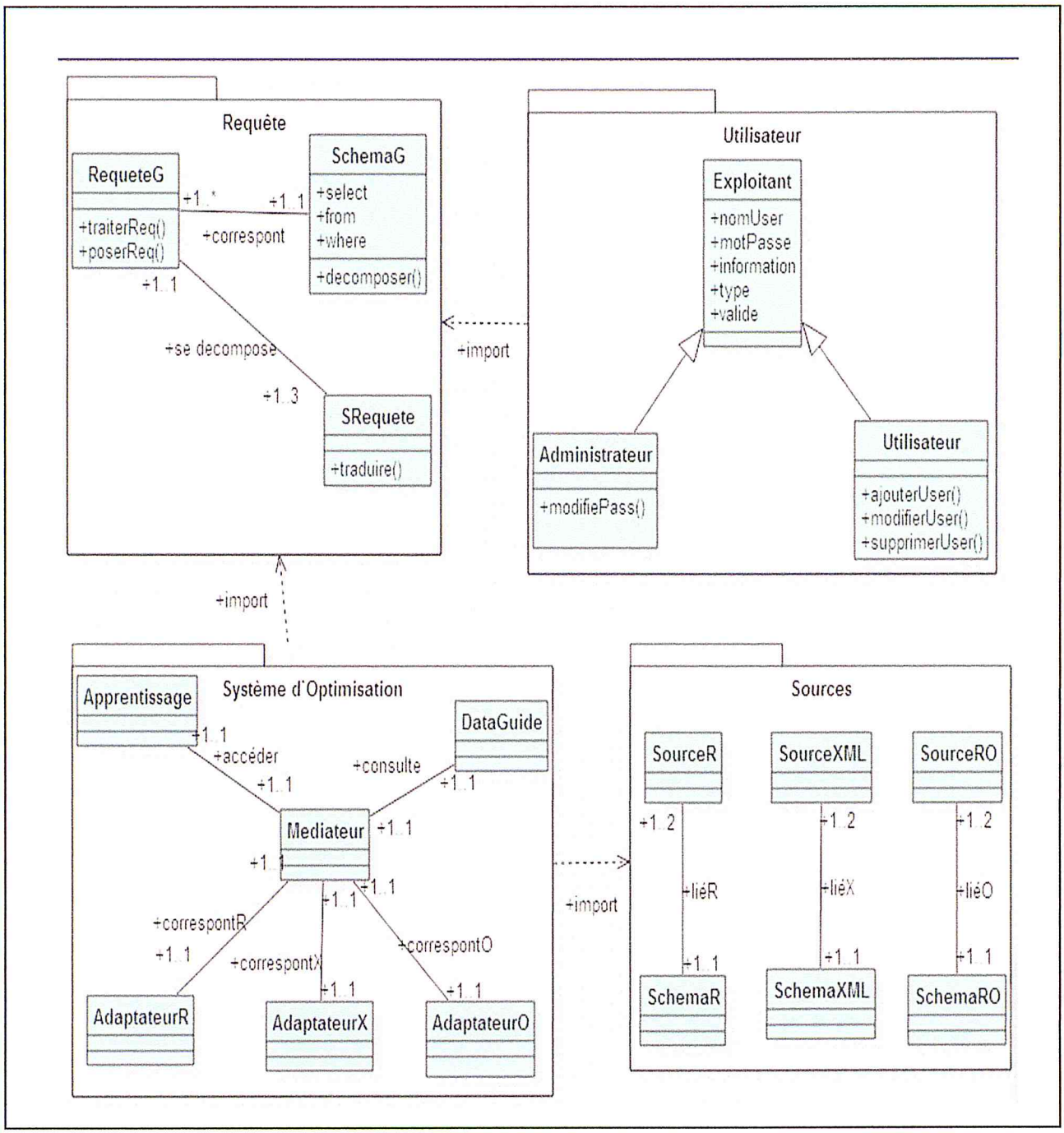


Figure 12: Diagramme de Paquetage du Système.

4.2.3. Architecture du Système

Pour la conception du système de médiation nous avons suivi l'approche GLAV avec une architecture du système à quatre couches montrée dans la figure suivante suivi d'une description de chaque étape :

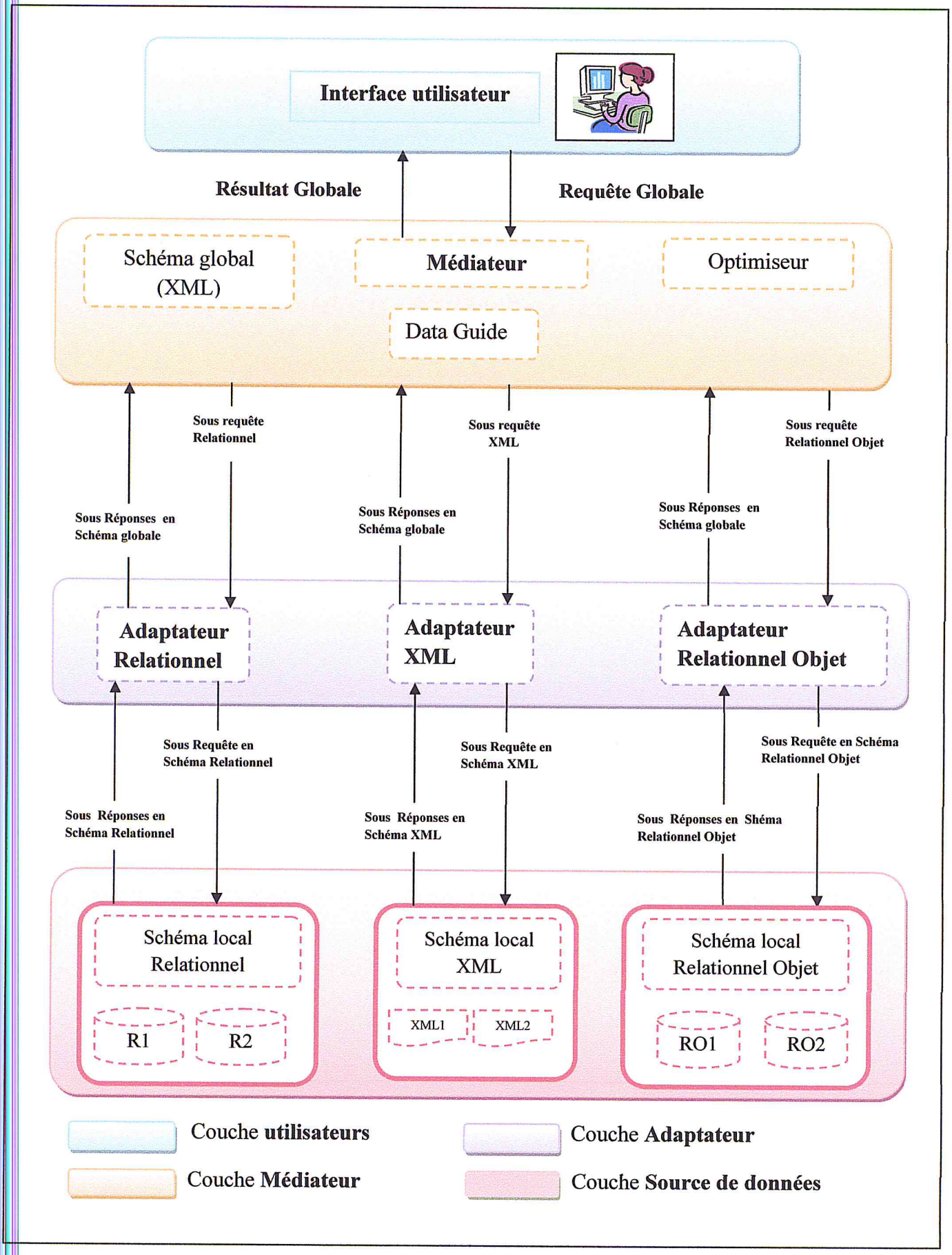


Figure 13: Architecture du système de Médiation.

4.2.3.1. Couche utilisateurs

C'est une simple couche de communication permet à un utilisateur de communiquer avec le système, Il envoie des requêtes au médiateur et reçoit des réponses.

4.2.3.2. Couche Médiateur

La couche la plus importante c'est la couche médiateur. Elle est décomposée en modules reliées entre eux. Cette couche contient :

- Un fichier XML nommé schéma global qui représente la structure globale des données (concepts et attributs globaux) voici son MLD[1].

Malade (NSSMalade[2], NomMalade, PrenomMalade, StatutMalade, SexeMalade, AdresseMalade, TelephoneMalade, DateNaissMalade).
Medecin (NSSMedecin, NomMedecin, PrenomMedecin, SpecialiteMedecin,AdresseMedecin, TelephoneMedecin, DateNaissMedecin).
Maladie (NumeroMaladie, NomMaladie, Description).
Atteint (NSSMalade*[3], NumeroMaladie*) .
Visite (NumVisite, DateVisite, PrixVisite, NSSMalade*, NSSMedecin*).
Diplome (NumDiplome, NomDiplome, DateObtention, Etablissement, NSSMedecin*).
Experience (IdExperience, Organisme, DateDebut, DateFin, NSSMedecin*).

- Le DataGuide qui est une base de données utilisée par le médiateur pour savoir les règles de mapping nécessaire à chaque requête globale et voir les correspondances entre les concepts globaux et locaux et entre les attributs globaux et locaux afin de trouver la bonne décomposition de la requête globale en sous requêtes, la figure suivante représente le diagramme de classe de DataGuide.

[1] **M**odèle **L**ogique de **D**onnées.
[2] Attribut souligné signifié une clé primaire.
[3] * signifié une clé étrangère.

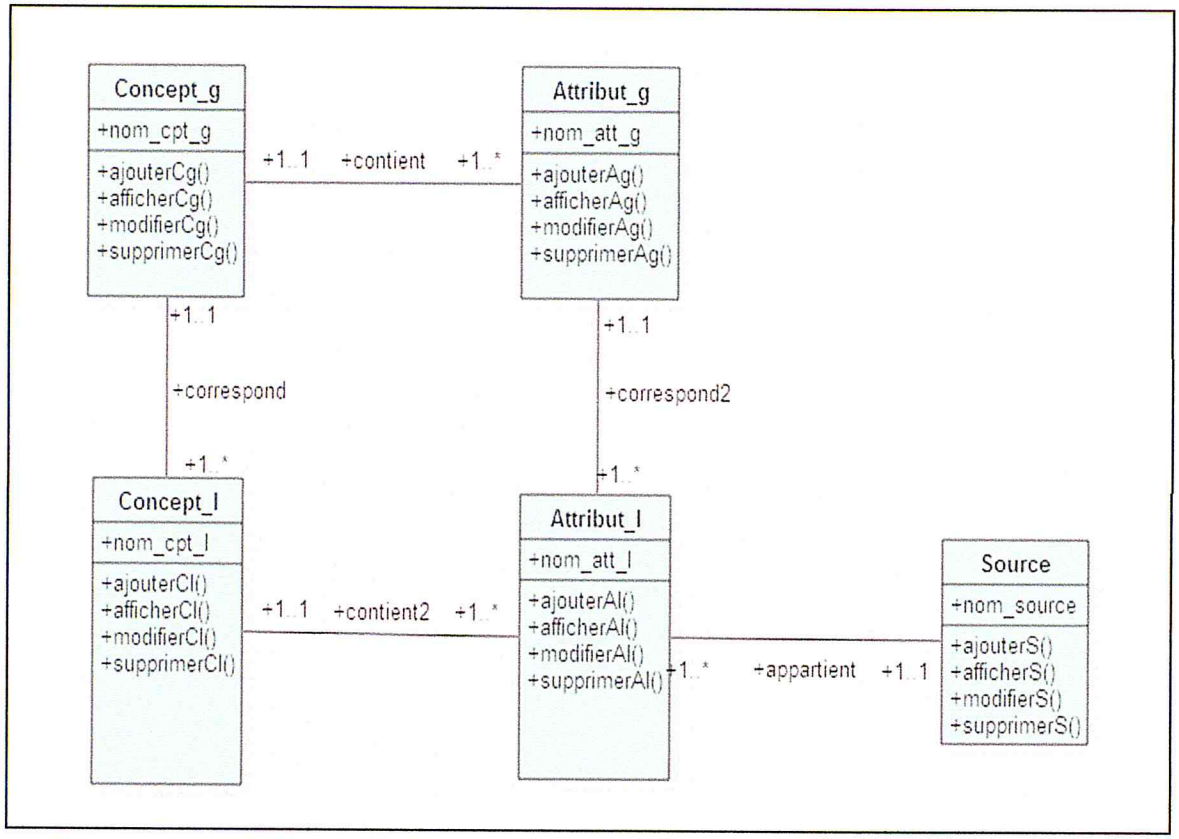
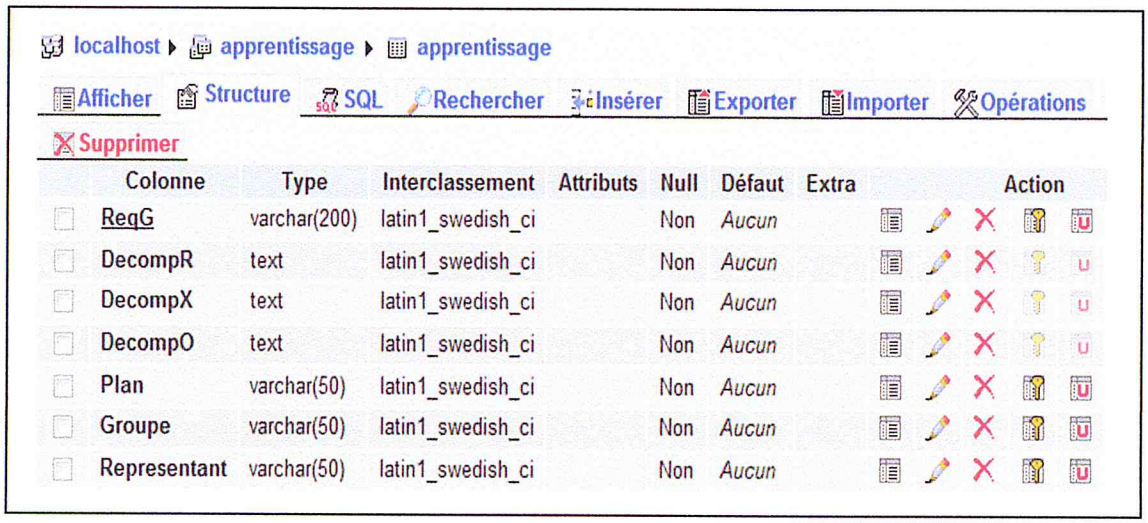


Figure 14: Diagramme de Classe de DataGuide.

- L'optimiseur est un module logiciel destiné à l'optimisation des requêtes en utilisant le Data Mining, tous d'abord à la construction des classes de requêtes puis au traitement des nouvelles requêtes. Pour bien accomplir son travail l'optimiseur à besoin d'une table de base de données nommé « Apprentissage » sa structure est la suivante :



	Colonne	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
☐	ReqG	varchar(200)	latin1_swedish_ci		Non	*Aucun*		
☐	DecompR	text	latin1_swedish_ci		Non	*Aucun*		
☐	DecompX	text	latin1_swedish_ci		Non	*Aucun*		
☐	DecompO	text	latin1_swedish_ci		Non	*Aucun*		
☐	Plan	varchar(50)	latin1_swedish_ci		Non	*Aucun*		
☐	Groupe	varchar(50)	latin1_swedish_ci		Non	*Aucun*		
☐	Representant	varchar(50)	latin1_swedish_ci		Non	*Aucun*		

Figure 15: Structure de la table apprentissage.

4.2.3.3. Couche Adaptateur

Le rôle de la couche adaptateur est de cacher l'hétérogénéité des différentes sources de données au médiateur, telle que chaque source est associé à un adaptateur qui joue le rôle d'intermédiaire entre cette source et le médiateur.

4.2.3.4. Couche Source de Données

La couche source de données contient les différentes sources à interroger. Sachant qu'une source peut être décrite par sa localisation, le type de données qu'elle gère, le mode d'interrogation ainsi que le format des résultats. Pour notre système nous avons choisit de limiter le cadre de notre étude sur trois sources de données hétérogènes.

- **Source de données relationnelles**

Les sources de données relationnelles sont des sources structurées tel que dans ce modèle, les données sont représentées par des tables. Le succès du ce modèle 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: **[7] [26]**

- 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.

Notre source de données relationnelles est une base de données à cinq tables, la MLD de son schéma local est le suivant :

Patient (NumPatient, NomPatient, PrenomPatient, DateNaissPatient).

Docteur (NSSDocteur, NomDocteur, PrenomDocteur, SpecialiteDocteur).

Mal (NumMal, NomMal).

Touche (NumPatient*, NumMal*).

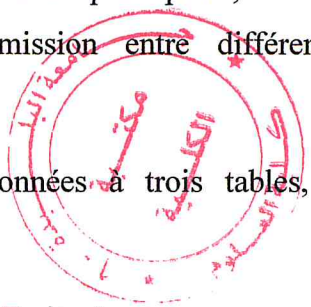
Consultation (NumConsultation, DateConsultation, PrixConsultation, NumPatient*, NSSDocteur*).

- **Source de Données Semi structurée**

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.

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 pour 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. **[10]**



Notre source de données XML est une base de données à trois tables, le MLD de son schéma local est le suivant :

MedecinTraitant (NSSMedecinTraitant, NomMedecinTraitant, PrenomMedecinTraitant, AdresseMedecinTraitant, TelephoneMedecinTraitant, DateNaissMedecinTraitant, SpecialiteMedecinTraitant).
Certificat (NumCertificat, NomCertificat, DateObtentionCertificat, EtablissementCertificat, NSSMedecinTraitant*).
Conscience (IdConscience, OrganismeConscience, DateDebutConscience, DateFinConscience, NSSMedecinTraitant*).

- **Source de Données Relationnel Objet**

Le modèle objet permet de traiter des objets complexes et donne de la flexibilité au niveau des types de données possibles. Tandis que le modèle relationnel est associé à un langage d'interrogation de haut niveau et permet l'exécution efficace de requêtes. Combiner les deux modèles permettrait de bénéficier des avantages des deux approches.**[27]**

Notre source de données Relationnel Objet est une base de données à cinq tables, le MLD de son schéma local est le suivant :

Malade (NSSMalade, NomMalade, PrenomMalade, StatutMalade, SexeMalade, AdresseMalade, TelephoneMalade, DateNaissMalade).
Praticien (NSSPraticien, NomPraticien, PrenomPraticien, SpecialitePraticien, AdressePraticien, TelephonePraticien, DateNaissPraticien).
Trouble (NumeroTrouble, NomTrouble, DescriptionTrouble).
Endommage (NSSMalade*, NumeroTrouble*) .
Controle (NumControle, DateControle, PrixControle, NSSMalade*, NSSPraticien*).

5. Optimisation des Requêtes

Comme on avait dit précédemment l'optimiseur est le module logiciel responsable à l'optimisation des requêtes, tous d'abord à la construction des classes de requêtes puis a l'exécution des nouvelles requêtes posées de façon optimale.

5.1. La Construction des Classes de Requêtes

La construction des classes de requêtes c'est l'étape d'apprentissage, qui doit être lancer par l'administrateur. Nous avons choisi pour cette phase l'algorithme Kmeans, il s'agit d'un ensemble de requêtes chacune avec son plan d'exécution optimal (le plan d'exécution optimal est obtenue après le calcul de temps d'exécution des différentes combinaisons des sous requêtes(les plans possibles)) en entrée et des classes de requêtes en sortie.

Algorithme : K-means **[14]**
Entrée
Ensemble de N requêtes, noté par x
Nombre de groupes souhaité, noté par k
Sortie
Une partition de K groupes $\{C_1, C_2, ..., C_K\}$
Début
1) Initialisation aléatoire des centres C*k* ;
Répéter
2) Affectation : générer une nouvelle partition en assignant chaque objet au groupe dont le centre est le plus proche ;
Avec μk le centre de la classe K ;
3) Représentation : Calculer les centres associe à la nouvelle partition ;

Jusqu'à convergence de l'algorithme vers une partition stable ;

Fin.

La figure suivante représente l'étape d'apprentissage d'une manière schématisé suivi d'une description des différentes étapes.

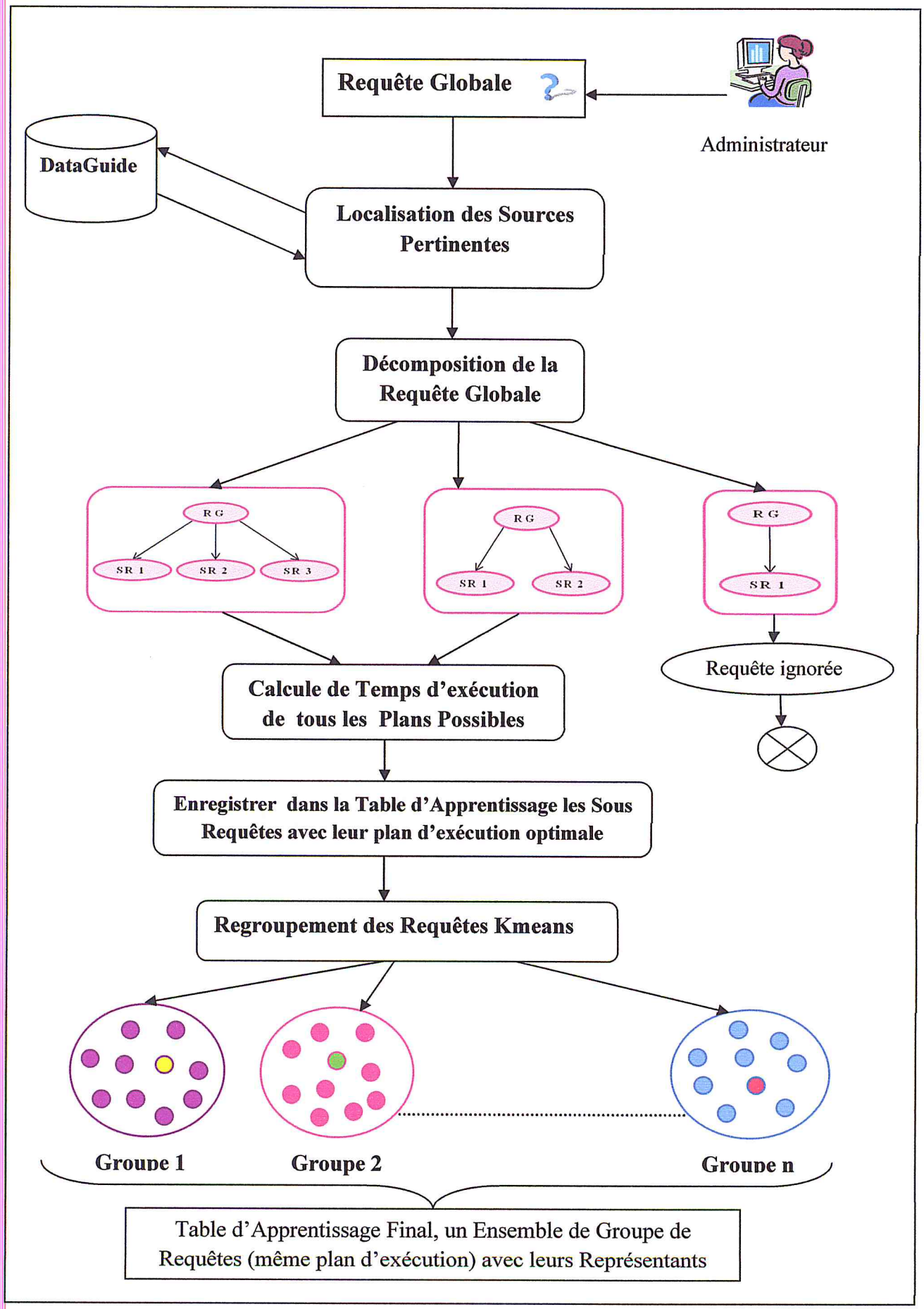


Figure 16: Etape d'Apprentissage.

Etape 1 : Réception de la Requête Globale

Le médiateur reçoit la requête globale posé par l'administrateur.

Etape 2 : Localisation des Sources Pertinentes

Le médiateur localise les sources pertinentes à l'aide de l'interrogation de DataGuide pour savoir les correspondances entre les concepts et les attributs globaux avec ceux locaux.

Etape 3 : Décomposition

Le médiateur décompose la requête globale en sous requêtes, si juste une seule sous requête est différente de null la requête sera ignoré, elle n'est pas utile à l'étape d'apprentissage car si une nouvelle requête sera posé sur le système et sa décomposition donne une seule sous requête, celle-ci elle sera exécuter directement sans l'utilisation des techniques de Data Mining. Sinon si les trois sous requêtes ou bien deux entre eux sont différentes de null, on passera à l'étape suivante

Etape 4 : Calcul des Temps d'Exécution

Dans cette étape le médiateur envoi les sous requêtes aux adaptateurs concernés on calculant le temps d'exécution des différentes combinaisons des sous requêtes pour garder a la fin le plan le plus optimal en terme de temps d'exécution.

Quand les adaptateurs reçoivent les sous requêtes envoyées par le médiateur ils font la traduction des sous requêtes pour les envoyées aux sources correspondantes, puis ils reçoivent les résultats envoyés par les sources afin de les traduire et les renvoyés au médiateur.

Etape 5 : L'Enregistrement

Dans cette étape un enregistrement de la requête globale, ces sous requêtes et son plan d'exécution optimal sera effectué.

Etape 6 : Regroupement des Requêtes

Pour regrouper les requêtes préparées dans l'étape précédente on va appliquer l'algorithme K Means afin de construire des classes homogènes de requêtes. Pour ce faire nous avons définit la distance suivante basée sur les plans d'exécution :

Distance (requête1, requête2) = 0 **Si** Plan (requête1) == Plan (requête2)

Distance (requête1, requête2) = 1 **Sinon**

Selon le type de distance que nous avons choisi et puisque notre travail concerne le traitement des requêtes non pas des données numérique, on va appliquer l'algorithme K Means pour une seule itération, car on ne peut pas recalculer les centres des groupes.

Algorithme : K-means **//version appliquée**
Entrée
Ensemble de N requêtes avec les plan d'exécution, noté par x
Nombre de groupes souhaité, noté par k
// le K choisi c'est le nombre possible de groupe
Sortie
Une partition de K groupes $\{C_1, C_2, ..., C_K\}$
Début
1) Initialisation aléatoire des centres C*k* ;
2) Affectation : générer une nouvelle partition en assignant chaque objet au groupe dont le centre est le plus proche; // même plan
Avec μk le centre de la classe K ;
Fin.

Dans ce qui suit nous avons choisit quelques requêtes globales comme des exemples de déroulement :

Exemple 1 :

Étape 1: Requête Globale

```
SELECT NomMedecin, MESpecialite FROM Medecin WHERE MESpecialite = ' Cardiologie '
```

Étape 2: Localisation des sources pertinentes

```
SELECT nom_att_l, nom_cpt_l, nom_source FROM attribut_l WHERE nom_att_g='NomMedecin'

OR nom_att_g='MESpecialite' AND nom_cpt_l IN ( SELECT nom_cpt_l FROM concept_l WHERE nom_cpt_g='Medecin')
```

nom_att_l	nom_cpt_l	nom_source
SpecialiteDocteur	Docteur	BDRelationnelle
SpecialitePraticien	Praticien	BDRO
SpecialiteMedecinTraitant	MedecinTraitant	BDXML
NomDocteur	Docteur	BDRelationnelle
NomMedecinTraitant	MedecinTraitant	BDXML
NomPraticien	Praticien	BDRO

Figure 17: Résultat de Requête (Exemple 1).

Étape 3: Décomposition

Selon le résultat de l'étape précédente, les sous requêtes envoyées sont (**R** désigne la sous requête **R**elationnel, **X** la sous requête **X**ML et **O** la sous requête relationnel **O**bjet):

R : **SELECT** NomDocteur, SpecialiteDocteur **FROM** Docteur **WHERE** SpecialiteDocteur = ' Cardiologie '

X : **SELECT** NomMedecinTraitant, SpecialiteMedecinTraitant **FROM** MedecinTraitant **WHERE** SpecialiteMedecinTraitant = 'Cardiologie '

O : **SELECT** NomPraticien, SpecialitePraticien **FROM** Praticien **WHERE** SpecialitePraticien = ' Cardiologie '

Sachant que les trois sous requêtes sont différentes de null, et que chaque sous requête sera envoyée aux deux sources qui correspondre a son schéma local, on obtiendra plusieurs possibilités des plans d'exécution avec un ordonnancement différent des jointures.

Pour des raisons d'optimisation et pour minimiser le nombre des plans d'exécution, nous avons décidé d'exécuter les requêtes d'un même schéma local en parallèle, Dans ce cas on obtiendra six possibilités des plans d'exécution :

1. $[(R1 \,//\, R2) \bowtie (X1 \,//\, X2)] \bowtie (O1 \,//\, O2)$
2. $[(X1 \,//\, X2) \bowtie (R1 \,//\, R2)] \bowtie (O1 \,//\, O2)$
3. $[(R1 \,//\, R2) \bowtie (O1 \,//\, O2)] \bowtie (X1 \,//\, X2)$
4. $[(O1 \,//\, O2) \bowtie (R1 \,//\, R2)] \bowtie (X1 \,//\, X2)$

5. [(X1 // X2) ⋈ (O1 // O2)] ⋈ (R1 // R2)
6. [(O1 // O2) ⋈ (X1 // X2)] ⋈ (R1 // R2)

Étape 4: Calcul de Temps d'exécution, Traduction et Interrogation

Le médiateur exécute par la suite les sous requêtes selon les six plans avec le calcul de temps d'exécution de chaque plan.

Les adaptateurs reçoivent les sous requêtes envoyées par le médiateur, chaque adaptateur fait la traduction de sous requête au langage de la source appropriée puis il interroge la source. De ce qui concerne la sous requête relationnelle elle reste elle-même avec le langage SQL, pour la sous requête relationnel objet un petit changement sera effectué sur la syntaxe du requête SQL (l'ajout des quottes") et pour la sous requête XML une traduction complète sera établit pour aller d'une requête en SQL vers une requête en XQuery (langage de requêtes pour les documents XML). Après la traduction des sous requêtes on obtient :

```
R :   SELECT NomDocteur, SpecialiteDocteur FROM Docteur WHERE
SpecialiteDocteur = ' Cardiologie '

X :   FOR $a IN document ("C:\xmlfiles\bdXml.xml")//MedecinTraitant
WHERE $a/ SpecialiteMedecinTraitant = "Cardiologie"
RETURN  <resultat>
                {$a/ NomMedecinTraitant }
                {$b/ SpecialiteMedecinTraitant }
          </resultat>

O :   SELECT "NomPraticien", "SpecialitePraticien" FROM "Praticien"
WHERE   "SpecialitePraticien" = ' Cardiologie '
```

Pour simplifier le traitement des sous requêtes XML et pour faciliter la traduction en langage XQuery nous avons choisi d'utiliser StelsXML qui est un type JDBC 4 driver (Java DataBase Connectivity) qui permet d'exécuter des requêtes et d'autres opérations JDBC sur des fichiers XML. Avec l'API StelsXML on peut facilement extraire et traiter les données contenues dans des documents XML en utilisant la syntaxe standard de SQL. **[28]**

A la fin de traduction, les sous requêtes seront envoyées aux sources appropriées puis les adaptateurs reçoivent les résultats envoyées par les sources ils les traduire et les envois aux médiateur.

Étape 5: Enregistrement

Le plan qui correspond au temps minimal sera enregistré avec la décomposition de requête dans la table d'apprentissage. Dans cet exemple (Exemple 1) le plan optimal c'est le premier plan.

Étape 6: Regroupement

Dans cette étape un regroupement des requêtes préparées dans l'étape précédente sera effectué on appliquant l'algorithme K Means.

Exemple 2 :

Étape 1: Requête Globale

```
SELECT NSSMalade, DateVisite FROM Visite WHERE
DateVisite='16 /09/2015'
```

Étape 2: Localisation des sources pertinentes

```
SELECT nom_cpt_l, nom_att_l, nom_source FROM attribut_l WHERE
nom_att_g= 'NSSMalade' OR nom_att_g='DateVisite' AND nom_cpt_l IN (select
nom_cpt_l FROM concept_l WHERE nom_cpt_g='Visite')
```

Le résultat de cette requête :

nom_att_l	nom_cpt_l	nom_source
DateConsultation	Consultation	BDRelationnelle
DateControle	Controle	BDRO
NumPatient	Patient	BDRelationnelle
NomPatient	Touche	BDRelationnelle
NumPatient	Consultation	BDRelationnelle
NSSMalade	Malade	BDRO
NSSMalade	Endommage	BDRO
NSSMalade	Controle	BDRO

Figure 18 : Résultat de Requête (Exemple 2).

Étape 3: Décomposition

Il n'ya pas de résultat pour la source XML car le concept global Visite n'a pas un correspondant dans cette source et donc la sous requête XML sera égale a null et elle ne sera pas envoyée. Dans ce cas les sous requêtes envoyées sont :

```
R : SELECT NumPatient, DateConsultation FROM Consultation WHERE
DateConsultation = '16 /09/2015'

O : SELECT "NSSMalade", "DateControle" FFROM "Controle" WHERE
"DateControle" = '16 /09/2015'
```

Sachant qu'on a deux sous requêtes différentes de null, on obtiendra deux possibilités des plans d'exécution :

1. (R1 // R2) $\bowtie$ (O1 // O2)
2. (O1 // O2) $\bowtie$ (R1 // R2)

Étape 4: Calcul de Temps d'exécution, Traduction et Interrogation

Le médiateur exécute par la suite les sous requêtes selon les deux plans avec le calcul de temps d'exécution de chaque plan, par la suite le plan qui correspond au temps minimal sera enregistré avec la décomposition de requête dans la table d'apprentissage. Dans cet exemple le plan optimal c'est le deuxième plan.

Étape 5: Enregistrement : Pareille a l'exemple1.

Étape 6: Regroupement : Pareille a l'exemple1.

Exemple 3 :

Étape 1: Requête Globale

```
SELECT * FROM Experience
```

Étape 2: Localisation des sources pertinentes

```
SELECT nom_att_l, nom_cpt_l, nom_source FROM attribut_l WHERE
nom_cpt_l IN (

SELECT nom_cpt_l FROM concept_l WHERE nom_cpt_g='Experience')
```

Le résultat de cette requête :

	nom_att_l	nom_cpt_l	nom_source
	IdConscience	Conscience	BDXML
	OrganismeConscience	Conscience	BDXML
	DateDebutConscience	Conscience	BDXML
	DateFinConscience	Conscience	BDXML
	NSSMedecinTraitant	Conscience	BDXML

Figure 19 : Résultat de Requête (Exemple 3).

Étape 3: Décomposition

Pour cet exemple le résultat retourné concerne

juste la source XML car le concept global Experience n'a pas des correspondants dans les deux autres sources et donc la sous requête XML c'est la seul qui sera envoyée :

```
X :   SELECT * FROM Consience
```

Et donc on aura un seul plan d'exécution :

1. (X1 // X2)

Dans cet exemple la sous requête sera ignorée à ce stade, elle ne sera pas enregistré dans la table d'apprentissage, on la besoin pas car si une nouvelle requête arrive et le médiateur trouve que le résultat existe sur une seule source, la requête sera exécuter directement sans l'utilisation des techniques du Data Mining.

Étape 4: Calcul de Temps d'exécution, Traduction et Interrogation /

Étape 5: Enregistrement /

Étape 6: Regroupement /

Après une étude approfondie, nous avons limité le nombre de groupe de requêtes à dix (10) selon les plans d'exécution possibles :

1. $[(R1 // R2) \bowtie (X1 // X2)] \bowtie (O1 // O2)$
2. $[(X1 // X2) \bowtie (R1 // R2)] \bowtie (O1 // O2)$
3. $[(R1 // R2) \bowtie (O1 // O2)] \bowtie (X1 // X2)$

4. [(O1 // O2) ⋈ (R1 // R2)] ⋈ (X1 // X2)
5. [(X1 // X2) ⋈ (O1 // O2)] ⋈ (R1 // R2)
6. [(O1 // O2) ⋈ (X1 // X2)] ⋈ (R1 // R2)
7. (R1 // R2) ⋈ (O1 // O2)
8. (O1 // O2) ⋈ (R1 // R2)
9. (X1 // X2) ⋈ (O1 // O2)
10. (O1 // O2) ⋈ (X1 // X2)

5.2. Attribution des Nouvelles Requêtes au Classes les Plus Adéquates

Lorsqu'une nouvelle requête est posée sur le système, un calcul de similarité syntaxique sera établit entre cette requête et les requêtes existantes a préalable (étape d'apprentissage) exactement entre la décomposition c'est-à-dire les sous requêtes on utilisant la distance de Hamming, la (figure 14) représente une description schématisé de l'étape de traitement d'une nouvelle requête.

La distance de Hamming est une notion mathématique, définie par **Richard Hamming**, et utilisée en informatique, en traitement du signal et dans les télécommunications. Elle joue un rôle important en théorie algébrique des codes correcteurs. Elle permet de quantifier la différence entre deux séquences de symboles. C'est une distance au sens mathématique du terme. À deux suites de symboles de même longueur, elle associe le nombre de positions où les deux suites diffèrent. **[29]**

Le calcul de distance est utilisé dans l'application de l'algorithme KPPV, afin de trouver la classe la plus approprié à la requête pour exécuter cette dernière selon le plan d'exécution optimale de sa classe, puis le médiateur reçoit les résultats envoyés par les différents adaptateurs, il les combine en résultat global et retourner a la fin ce résultat a l'utilisateur.

L'algorithme K plus proche voisin est l'un des algorithmes les plus simple de la classification supervisé, c'est a partir d'un échantillon fini d'objets classés on désir construire une fonction capable de classer au mieux de nouveaux objets ne fessent pas partie de l'échantillon initiale.

Algorithme : KPPV **[12]**
Paramètre : Le nombre K de voisin
Données : Un échantillon de m exemple et leurs classes
Entrée : Un enregistrement y
Début : Pour chaque exemple x

Faire Calculer la distance (x, y)

Fait

Pour chaque x qui appartient au KPPV(y)

Faire Compter le nombre d'occurrence de chaque classe

Fait

Attribuer a y la classe la plus fréquente

Fin.

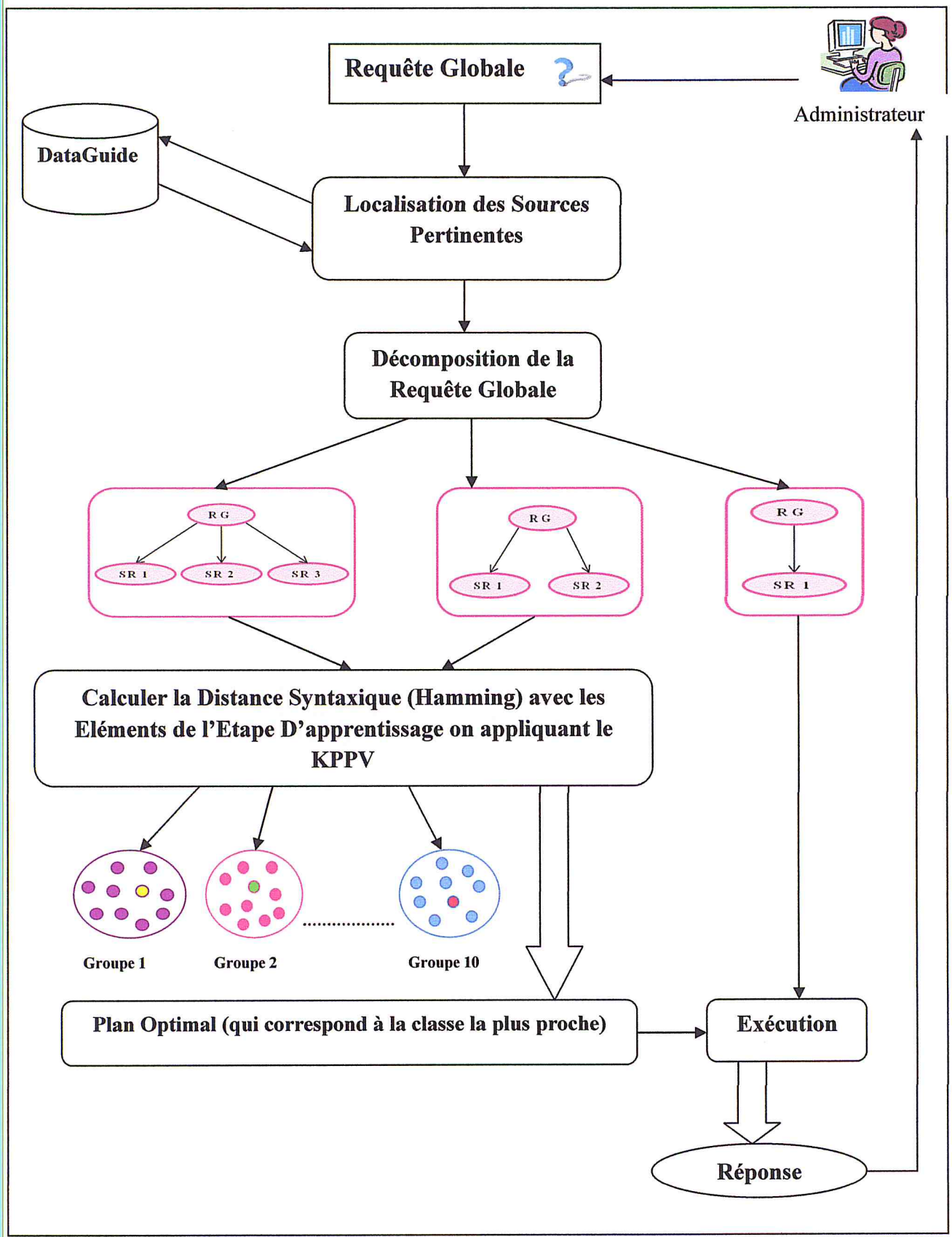


Figure 20 : Etape de Traitement d'une Nouvelle Requête.

6. Conclusion

La conception de la solution est une étape primordiale est importante dans la construction du nouveau système. Elle prépare à la phase de réalisation, nous dans ce chapitre on a montré notre solution en utilisant différents diagrammes d'UML ainsi qu'une description textuelle bien détaillée de notre solution. Le chapitre suivant porte sur la phase implémentation, celle qui donne lieu au système conçu précédemment.

CHAPITRE IV

Implémentation du Système

1. Introduction

Dans ce chapitre consacré à l'implémentation, nous allons présenter la démarche de réalisation des parties du système introduites précédemment dans le chapitre de conception. Nous allons présenter en premier lieu les outils qui nous ont servis à construire nos sources de données, puis les outils de développement de notre système et a la fin les principales interfaces de notre application.

2. Les Outils de Mise en Œuvre des Sources

2.1. Plateforme WampServer

Afin de mettre en œuvre notre application,

nous avons utilisé la plateforme WAMPSERVER (WAMP signifiant Windows Apache Mysql PHP). C'est une plate-forme de développement Web sous Windows pour des applications Web dynamiques à l'aide du serveur Apache2, du langage de scripts PHP et d'un système de gestion de base de données MySQL, le plus connus et le plus utilisé. C'est un logiciel (ou un ensemble de logiciels) permettant de manipuler les données d'une base des données, i.e. Sélectionner et afficher des informations tirées de cette base, modifier des données, ajouter ou bien supprimer. Il utilise pour cela le langage SQL. Il possède également PHPMyAdmin pour gérer plus facilement les bases de données. **[30]**

Nous avons manipulé nos bases de données relationnelles par l'outil WampServer qui est un environnement de développement intégré gratuit et qui simplifie le développement et la gestion de base de données.

2. 2. Oxygen XML Editor

L'éditeur XML <oXygen/> est une application multi plateformes pour le développement de document en utilisant des langages de formatage tels que XML, XSD, XSL, DTD.

Basé sur une technologie Java éprouvée, l'interface graphique utilisateur de l'éditeur XML <oXygen/> est facile à utiliser et offre des fonctionnalités robustes pour l'édition, la gestion de projet et la validation de sources structurées.

<oXygen/>supporte la sortie vers de multiples formats cibles, dont : PDF, PS, TXT, HTML et XML. **[31]**

Nous avons manipulé notre base de données XML ainsi nos schémas avec cet éditeur pour les raisons suivantes :

- Multi plateformes : Windows, Mac OS X, Linux,Solaris.
- Valide les XML Schémas, et les documents XML par rapport au schéma associé.
- Intègre une interface d'édition graphique qui facilite la création des documents.

2. 3. PostgreSql

PostgreSQL est un moteur de bases de données relationnelles objet, permet de manipuler des bases de données relationnelles ou relationnelles objet. C'est un moteur adapté à des bases métier, [donc riche en fonctionnalités et puissant. **[32**

Nous avons choisi cet outil pour la manipulation de nos bases de données relationnelles objet à cause des raisons suivants :

- Moteur transactionnel.
- Respect des normes SQL.
- MVCC (mécanisme permettant une concurrence efficace sans verrouiller les enregistrements pour assurer l'isolation des transactions).
- Procédures stockées dans de nombreux langages.
- Triggers.

3. Les Outils de développement

3.1. Langage de Programmation Java

Java est le nom d'une technologie mise au point par Sun Microsystems qui permet de produire des logiciels indépendants de toute architecture matérielle. Java est à la fois un langage de programmation orientée objet, et une plateforme d'exécution. Le langage Java a la particularité principale d'être portable sur plusieurs systèmes d'exploitation tels que Windows, Mac OS ou Linux. C'est la plateforme qui garantit la portabilité des applications développées en Java. **[33]**

Nous avons choisi le langage JAVA pour les raisons suivantes :

- L'application peut s'exécuter sur n'importe quel système d'exploitation à condition d'avoir la machine virtuelle java installée sur la machine (portabilité);

- La disponibilité de la documentation et de l'assistance (forums).
- Il est gratuité.

3.2. L'EDI 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. **[34]**

4. Présentation de l'Application

Cette section a pour objectif de présenter des copies d'écran des interfaces de notre application. Cette dernière consiste en un système d'optimisation de requêtes dans les systèmes de médiations offrant un ensemble de fonctionnalités. Dans ce qui suit, nous présentons les principales interfaces de avec une brève explication de chacune.

4.1. Authentification

Le lancement de notre application s'effectue à travers le lancement de sa page d'authentification (Figure 21). L'authentification permet de protéger le système et de garantir que l'utilisateur est bien celui qu'il prétend être. Le processus d'authentification consiste donc à comparer l'identité annoncée avec celle mémorisée dans la base de données.

Dans notre application, nous distinguons deux types de compte :

- Compte d'utilisateur ordinaire qui peut poser des requêtes au système.

- Compte d'administrateur. L'administrateur peut accéder, entre autre aux fonctionnalités d'un utilisateur ordinaire et aux fonctionnalités avancées permettant la gestion des comptes (consultation de la liste des comptes, suppression d'un compte), le lancement de l'étape d'apprentissage, la consultation des schémas ainsi de suite.

Cette interface permet aussi la récupération de mot de passe en cas d'oublie ainsi que l'inscription en tant qu'utilisateur.

Figure 21: Interfaces Authentification, Mot de Passe Oublié et Inscription.

Dans ce qui suit on va prendre des capteurs d'écran de notre application pour le compte administrateur car il est plus riche.

4.2. Gestion des Comptes

Dans cette interface l'administrateur bénéfice de plusieurs fonctionnalités, il peut modifier son mot de passe, consulter la liste des comptes, la liste des inscriptions et effectivement il peut établir des validations ou bien des suppressions.

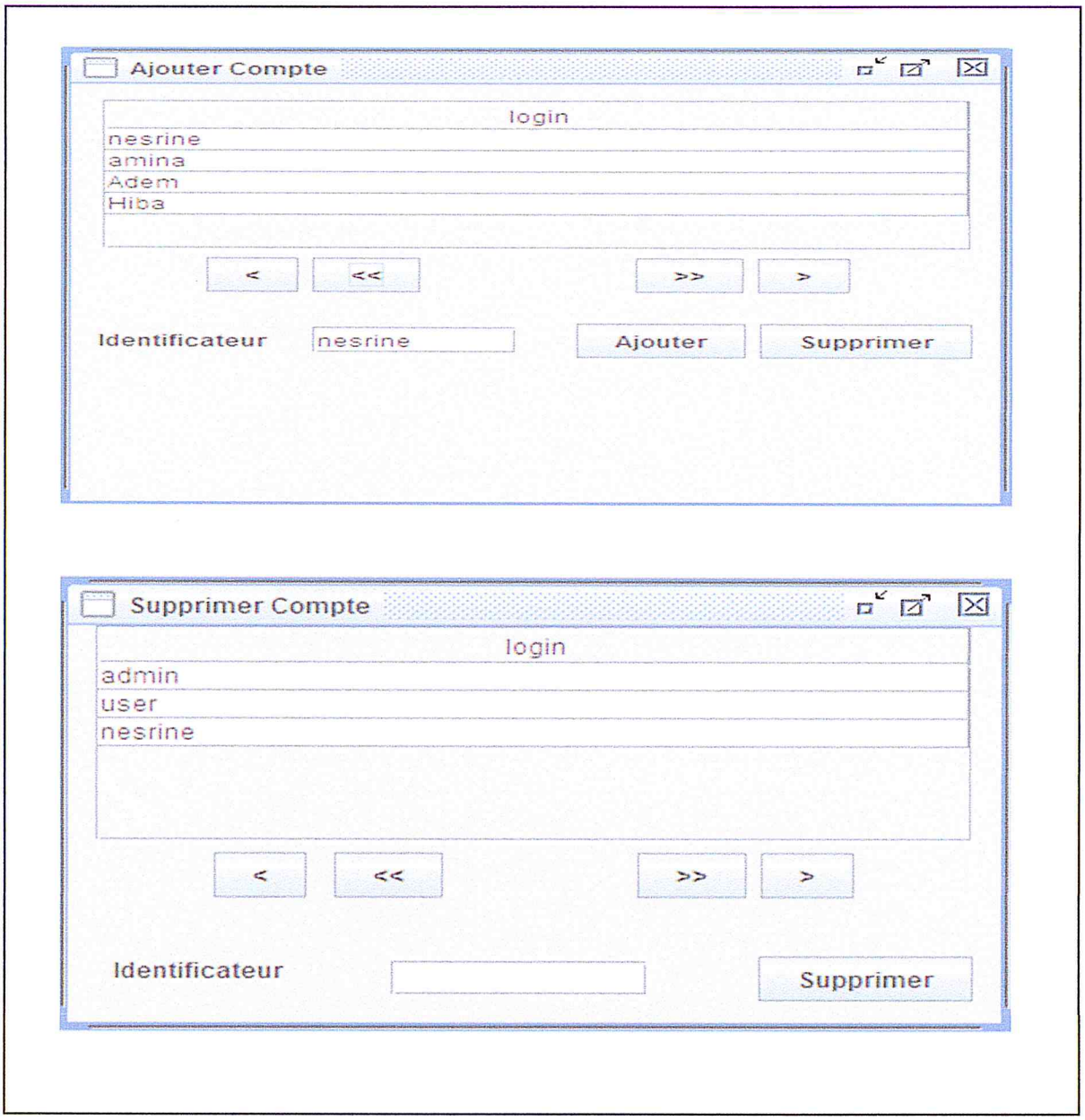


Figure 22: Interfaces Gestion des Comptes.

4.3. Consultation des Schémas

Cette interface permet à l'administrateur de consulter le schéma global ainsi que les différents schémas locaux.

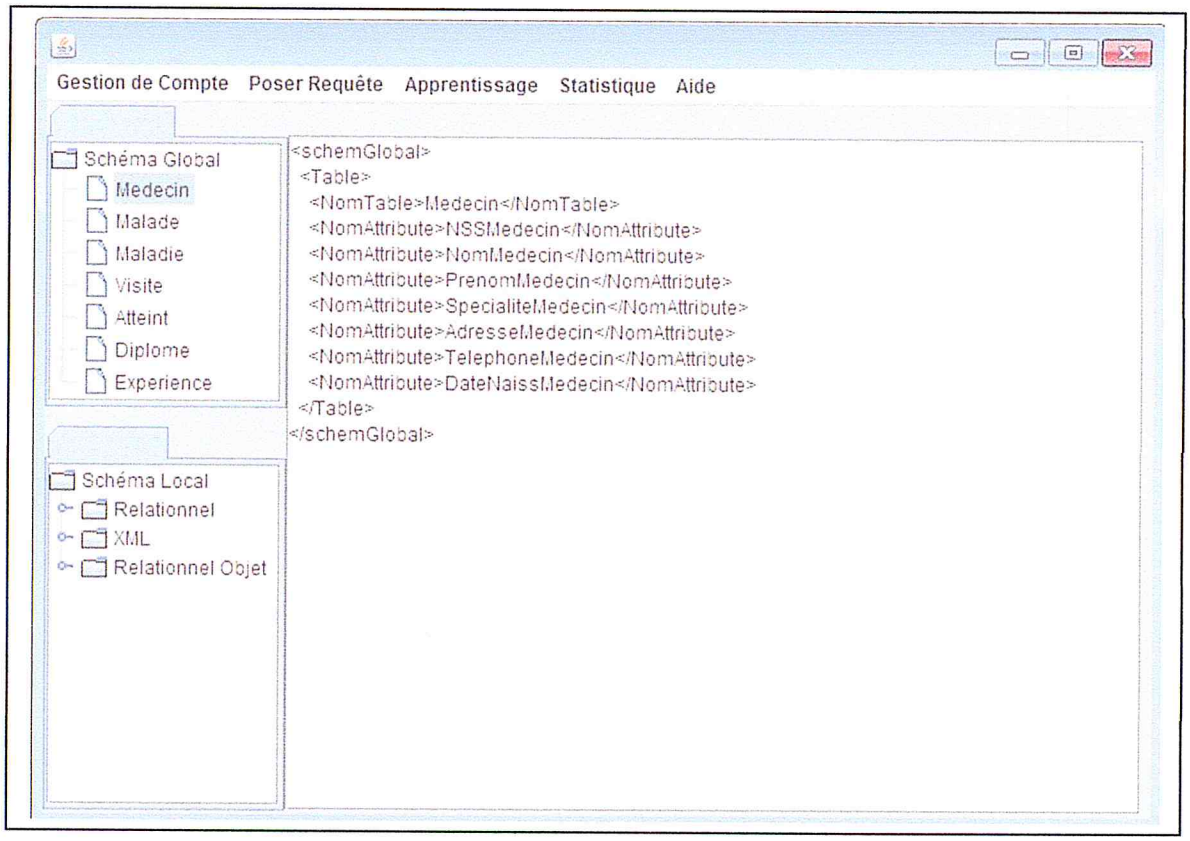


Figure 23: Interface Consultation des Schémas.

4.4. Apprentissage

Dans la fenêtre d'apprentissage l'administrateur pose des requêtes, il peut consulter la table d'apprentissage durant toutes les étapes avant le choix des centres, après… il peut aussi voir le résultat de regroupement ainsi que les différents groupes.

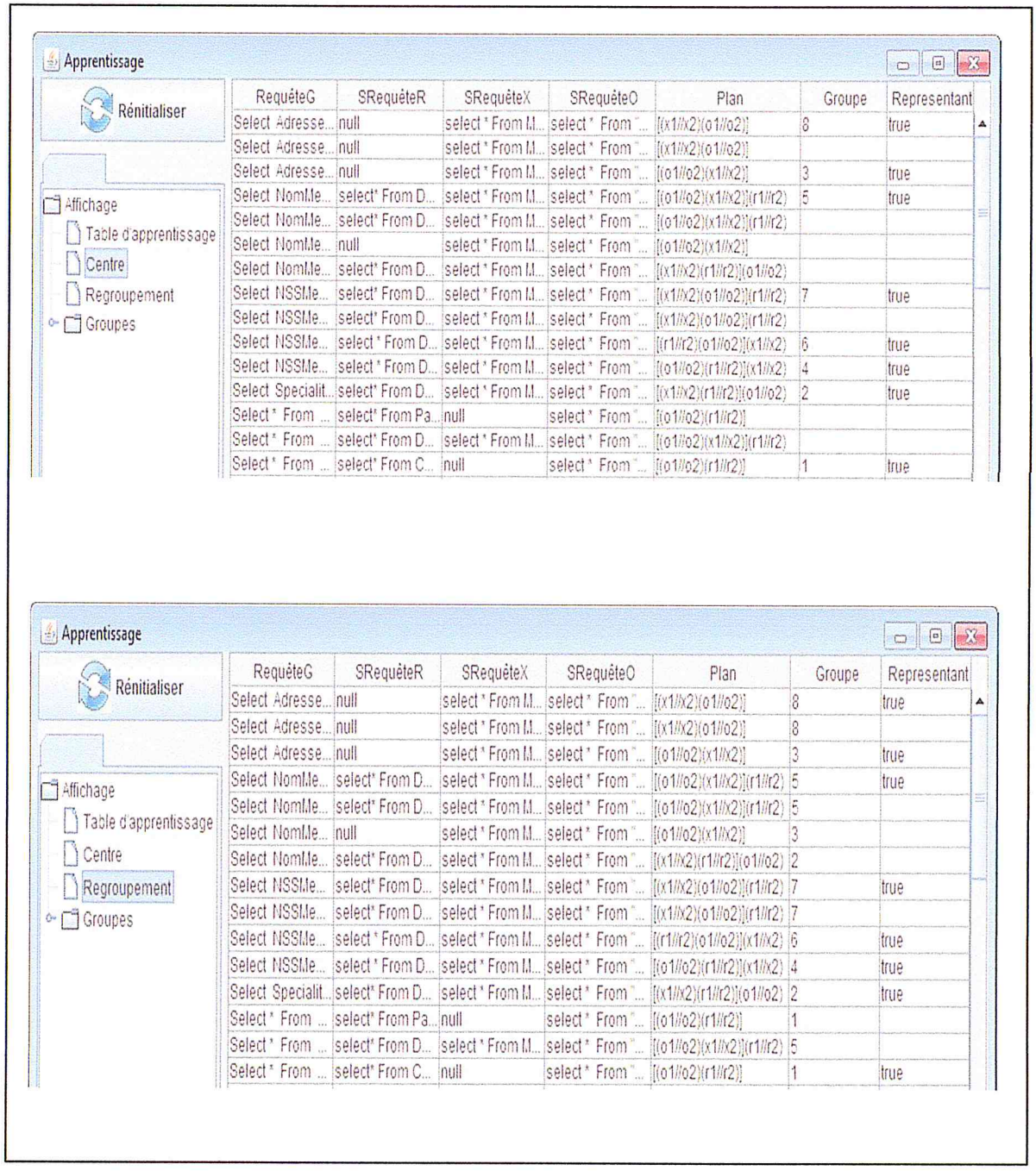


Figure 24: Interfaces montrant les Etapes d'Apprentissage.

4.5. Poser Requête

Cette interface permet à l'utilisateur d'avoir une réponse de requête en un temps optimal a cause des techniques de Data Mining utilisées.

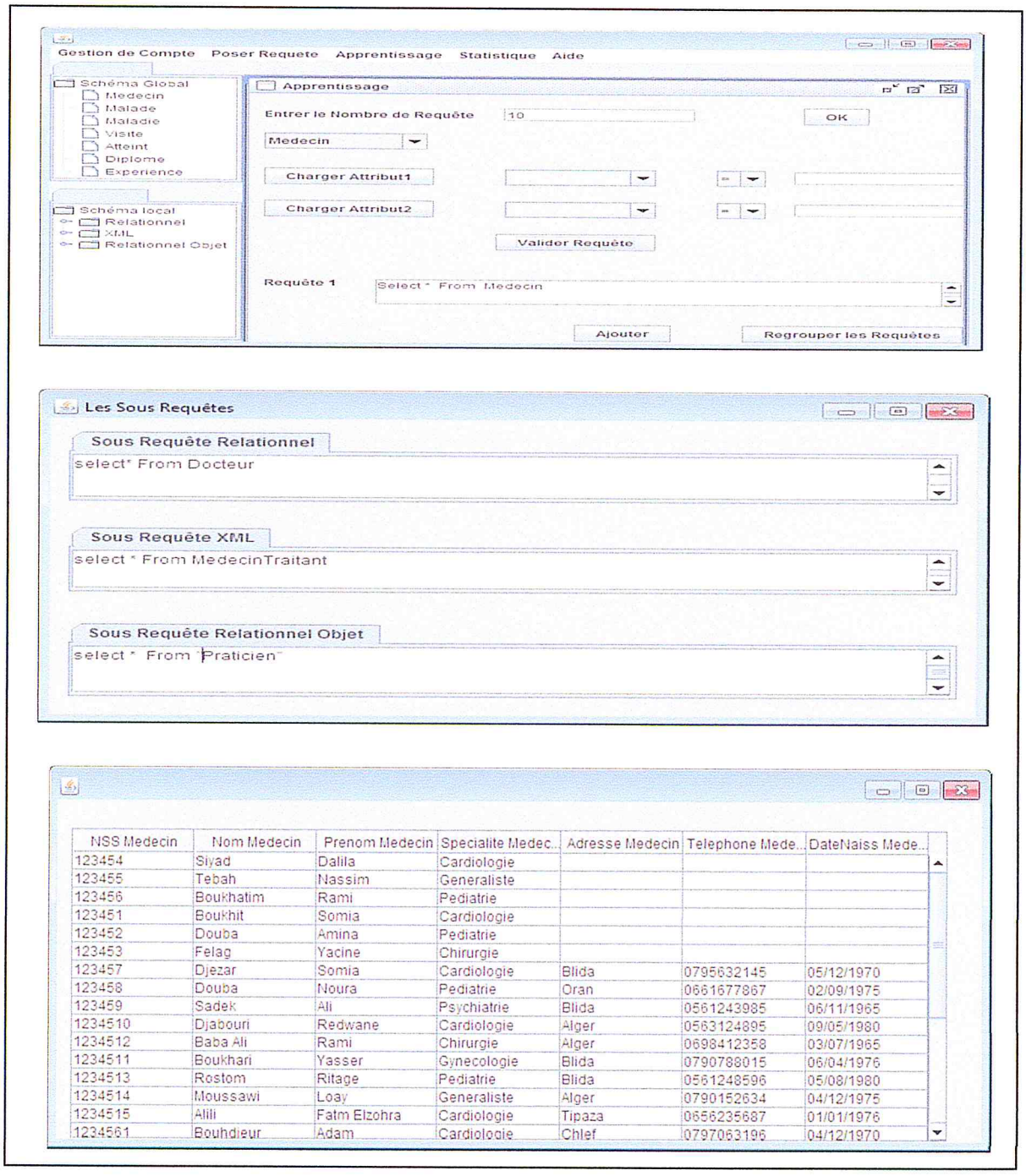


Figure 25: Interfaces Poser Requête.

4.6. Statistique

Cette fenêtre permet a l'administrateur de voir la différence en temps d'exécution en (m/s) d'une même requête exécutée une fois sans l'utilisation des techniques du Data Mining pour l'optimisation et une autre fois avec.

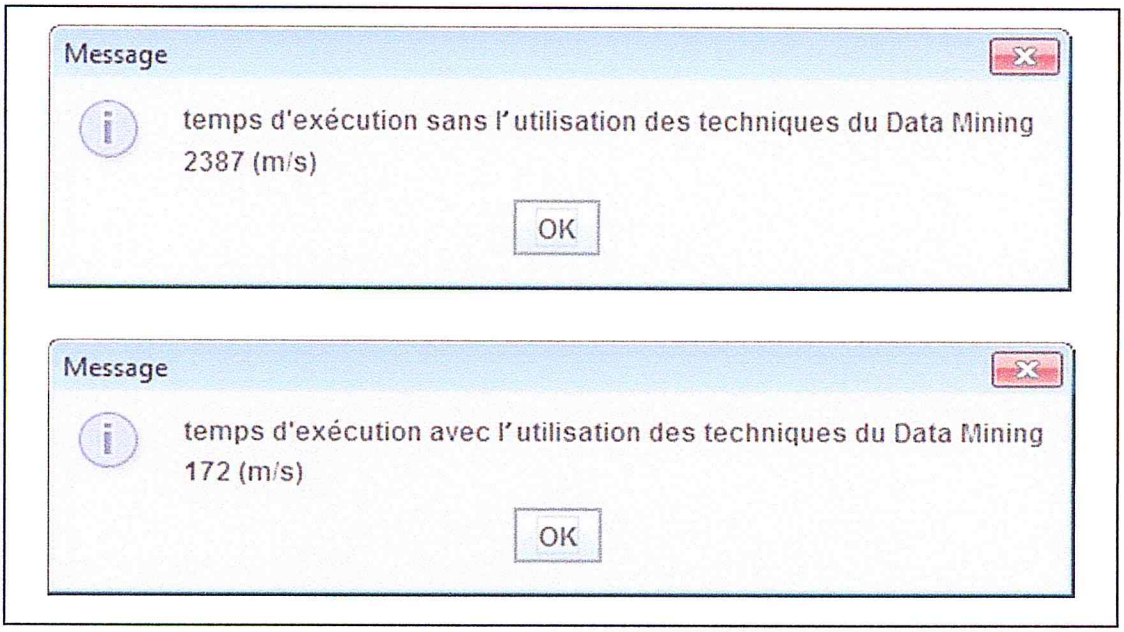


Figure 26: Interfaces Statistiques.

5. Conclusion

Dans ce chapitre, nous avons présenté les outils utilisés pour la mise en œuvre de notre application concernant les sources de données et le développement avec une présentation de celle-ci à l'aide des captures d'écran montrant les principales fonctionnalités.

CHAPITRE V

Tests et Validation du Système

1. Introduction

D'après ce qui a été dit dans les chapitres précédant, notre système consiste à l'optimisation des requêtes du médiateur en utilisant le Data Mining. Dans ce chapitre nous avons choisi deux types de validation de notre système, le premier concerne le résultat de l'application de KPPV et le deuxième concerne le temps d'exécution des requêtes en utilisant le Data Mining.

2. Première Validation

Pour valider il faut toujours faire plusieurs tests, c'est ce qu'on avait fait avec un ensemble de requêtes afin de comparer leur plan d'exécution optimal une fois en lançant les requêtes dans l'étape d'apprentissage et une deuxième fois en utilisant le KPPV. Dans les deux cas nous avons obtenue les même plans ce qui valide notre approche d'optimisation. La figure suivante représente les plans d'exécution de trois requêtes dans la table d'apprentissage puis avec le KPPV.

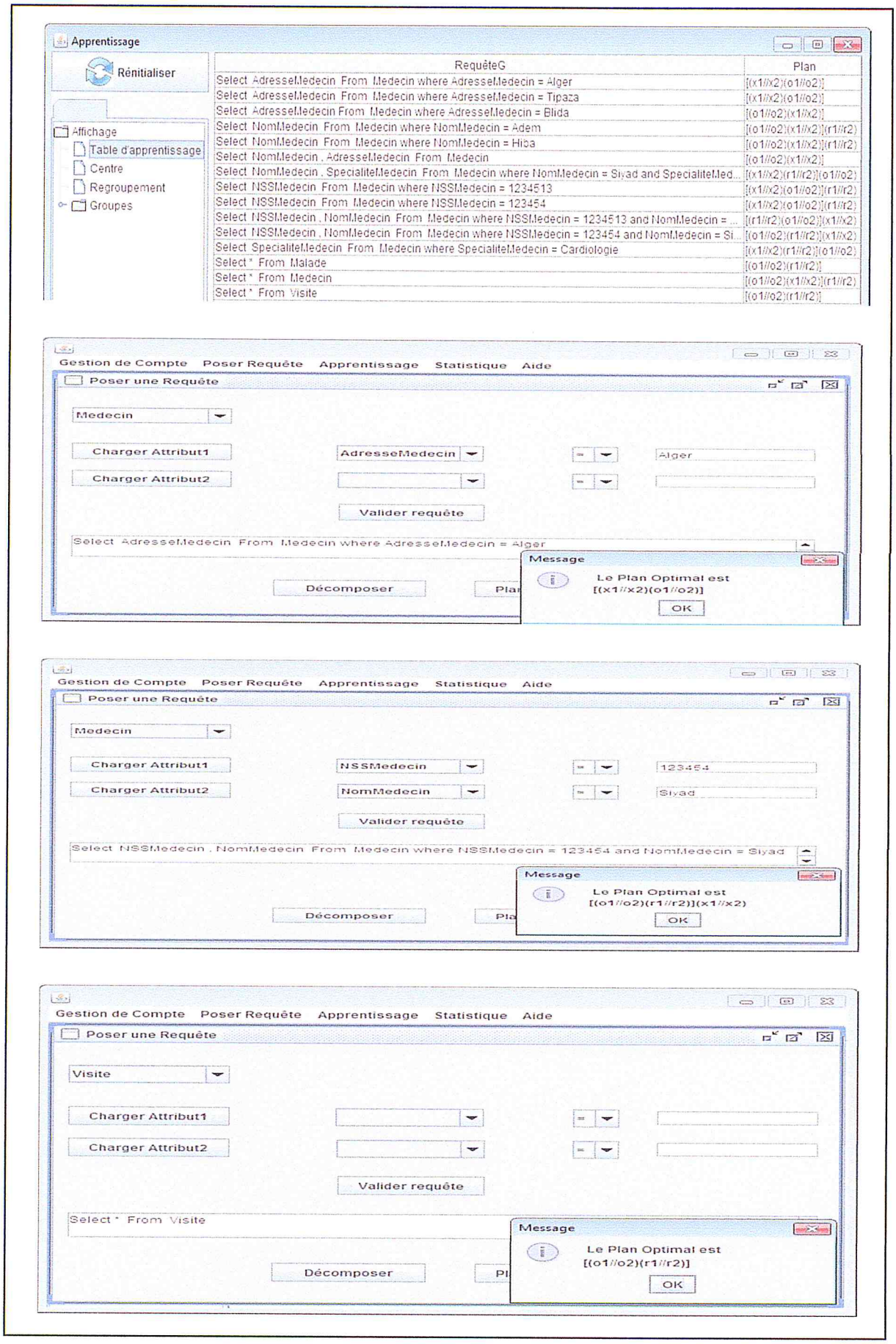


Figure 27 : Interface Montrant la Première Validation.

2. Deuxième Validation

De même comme la première validation il faut toujours faire des tests, c'est ce qu'on avait fait avec un ensemble de requêtes exécutés une fois avec Data Mining et une autre fois sans Data Mining. Les requêtes exécutées sont les suivantes :

R1 : Select * from Medecin

R2 : Select NomMedecin, SpecialiteMedecin from Medecin where NomMedecin='Siyad' and SpecialiteMedecin='Cardiologie'

R3 : Select AdresseMedecin from Medecin where AdresseMedecin='Alger'

R4 : Select StatutMalade from Malade where StatutMalade ='coma'

R5 : Select * from Maladie

R6 : Select NSSMalade, DateVisite from Visite

R7 : Select SpecialiteMedecin from Medecin where SpecialiteMedecin ='Pediatrie'

R8 : Select DateVisite from Visite where DateVisite='10/06/2015'

R9 : Select NomMedecin, PrenomMedecin from Medecin where NomMedecin='Douba' and PrenomMedecin='Amina'

R10 : Select NumeroMaladie from Atteint where NumerMaladie='11'

Le tableau suivant représente les temps d'exécution de chaque requête :

Requête	Temps d'exécution Avec Data Mining (m/s)	Temps d'exécution Sans Data Mining (m/s)
R1	62	1420
R2	47	1685
R3	47	1295
R4	265	390
R5	62	593
R6	62	842
R7	47	437
R8	140	1466
R9	125	1528
R10	78	452

Tableau 2: Temps d'Exécutions des Requêtes Avec et Sans Data Mining.

La figure suivante est une courbe représentant le tableau ci-dessus :

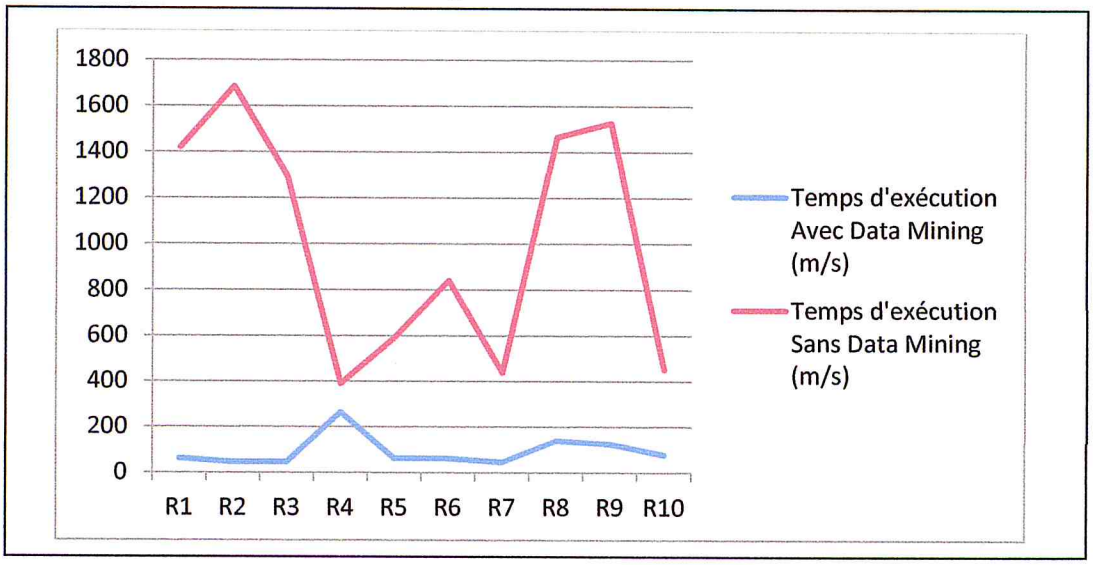


Figure 28: Courbe Représentant le temps d'Exécutions des Requêtes Avec et Sans Data Mining.

Après les résultats obtenues concernant le temps d'exécution des requêtes sans et avec l'utilisation du Data Mining, on peut bien voir la différence entre les deux comme on peut conclure que le Data Mining est utile dans le domaine d'optimisation des requêtes ce qui valide notre travail.

3. Conclusion

Après les tests et les deux types de validation que nous avons faits qui permettent de vérifier les résultats de l'implémentation en testant la construction de notre système, on peut conclure que notre système répond aux besoins spécifiés dans les chapitres précédents.

Conclusion Générale

De nos jours, de larges volumes de données sont disponibles publiquement, les types de données sont divers, et les ressources sont très nombreuses. L'architecture d'un système d'intégration de données doit permettre leur intégration donnant ainsi l'impression à l'utilisateur qu'il utilise un système homogène. Dans ce contexte, un problème essentiel de l'intégration est de gérer l'hétérogénéité des différentes sources de données. Parmi les systèmes d'intégration nous avons focalisé sur les systèmes de médiation qui sont aujourd'hui un axe de recherche captivant non seulement pour les chercheurs, mais aussi pour les différentes entreprises.

Le projet qui nous a été confié, porte sur la réalisation d'un système d'optimisation des requêtes par l'approche médiateur en utilisant le Data Mining. Pour ce faire nous avons passé par plusieurs étapes notamment la collecte d'information nécessaire à notre étude, la conception du système, l'implémentation et a la fin tests et validation de notre application qui répond aux objectifs fixés dans la phase de démarrage.

La solution que nous avons présenté et le projet que nous avons réalisé dans ce mémoire pour la résolution de problème d'optimisation des requêtes de systèmes de médiations n'est en réalité qu'une ouverture vers d'autre travaux car notre système peu encore évoluer et se voir améliorer. Comme perspectives nous envisageons d'implémenter un système de médiation dynamique permettant facilement l'ajout des nouvelles sources ainsi que la modification des schémas, ce dernier nécessite un temps supérieur au temps consacré pour la réalisation de notre mémoire.

L'achèvement de notre mémoire nous a permis d'avoir des notions avancées sur les systèmes de médiation et d'approfondir nos connaissances sur les systèmes de gestion de base de données ainsi sur les outils de développement. Au fait ce projet nous a permis d'intégrer dans la recherche scientifique, de plonger dans la pratique et d'activer nos connaissances acquises durant nos cinq années universitaires.

Références Bibliographique

[1] Proposition d'un Cadre Générique d'Optimisation de Requêtes dans les Environnements Hétérogènes Répartis, Tianxiao Liu, Université de Cergy-Pontoise, 2011, France.

[2] Localisation de sources de données et optimisation de requêtes réparties en environnement pair-à-pair, M. Raddad AL KING, l'Université Toulouse III, 2010, France.

[3] Médiation de données sémantique dans SenPeer, un système pair-à-pair de gestion de données, David Célestin Faye, 2007, Université de Nantes.

[4] Conception des systèmes d'information des observatoires environnementaux : Une architecture de médiation, Patricia Dzeakou, 1998, France.

[5] Interopérabilité Sémantique des Systèmes d'Information Distribués, BALA Mahfoud. Institut National de Formation en Informatique, INI, 2007.

[6] Etude et proposition d'une architecture de médiation entre sources de données hétérogènes. BAKHTOUCHI Abdelghani, Institut national de formation en informatique « Oued Smar – Alger », 2006.

[7] Coopération multi agents pour le traitement des requêtes sur des sources de données hétérogènes et distribuées, Benharzallah Saber, Université Mentouri, 2005, Constantine.

[8] Validating Mediator Cost Models with Disco, Hubert Naackey, Anthony Tomasic, PatrickValduriezy.

[9] Modèle de coût pour médiateurs de bases de données hétérogènes, Hubert Naack, 1999, France.

[10] Fédération de données semi structurées avec XML, Tuyet Trâm, Dang Ngoc, Université de versailles, 2003, France.

[11] Data mining, article disponible sur : http://www.ultra-fluide.com/ressources/datamining/presentation.htm , visité le : 03/03/15.

[12] DELCAMBRE Rudy, Les algorithmes de fouille de données, Probatoire, Février 2005.

[13] Data Mining, Diday, Université Paris Dauphine, UFR Informatique de Gestion, 2005.

[14] Proposition d'une solution au problème d'initialisation cas du K-means, Z.Guellil et L.Zaoui, Université des sciences et de la technologie d'Oran MB, Université Mohamed Boudiaf USTO -BP 1505 El Mnaouer, ORAN, Algérie.

[15] Subbarao Kambhampati, Ullas Nambiar, Zaiqing Nie, Sreelakshmi Vaddi, Havasu: AMulti-Objective, Adaptive Query Processing Frame work for web Data Integration, Department of Computer Science and Engineering, Arizona State University.

[16] Zaiqing Nie, Subbarao Kambhampati, Ullas Nambiar, Effectively Mining and Using Coverage and Overlap Statistics for Data Integration, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 17, NO. 5, MAY 2005.

[17] Sami Zghal, Contributions à l'alignement d'ontologies OWL par agrégation de similarités, Université d'Artois, Tunisie, 2010.

[18] Giorgos Stoilos, Giorgos Stamou, and Stefanos Kollias, A String Metric for Ontology Alignment, Department of Electrical and Computer Engineering, National Technical University of Athens, Zographou 15780, Greece.

[19] Catherine ROUSSEY, Jean-Pierre CHEVALLET, Recherche d'Information Semantique, Second Atelier Recherche d'Information SEmantique RISE, Marseille 25 mai 2010.

[20] Haïfa Zargayouna, Sylvie Salotti, Mesure de similarité dans une ontologie pour l'indexation sémantique de documents XML, Université Paris 11, France, 2004.

[21] M Abdeltif ELBYED, ROMIE, une approche d'alignement d'ontologies à base d'instances, Thèse présentée pour l'obtention du grade de Docteur de l'INSTITUT NATIONAL DES TELECOMMUNICATIONS, 16 Octobre 2009.

[22] Modélisation UML, Christine Solnon, INSA de Lyon - 3IF, 2013 – 2014.

[23] Introduction à UML 2.0, F.-Y. Villemin, CNAM, MAI NFE103, 2013-2014.

[24] Cycle de vie, Renaud Marlet, LaBRI / INRIA, 2007.

[25] Processus de développement Cycles de vie, Lydie du Bousquet.

[26] Base de Données Relationnelles, article disponible sur: http://codes-sources.commentcamarche.net/faq/1144-les-bases-de-donnees-relationnelles, visité le 16/04/2015.

[27] Base de Données Relationnelles Objet, article disponible sur : https://technet.microsoft.com/fr-fr/library/ms365368%28v=sql.105%29.aspx, visité le 16/04/2015.

[28] stelsXML JDBC, article disponible sur: http://www.csv-jdbc.com/stels_xml_jdbc.htm, visité le 10/05/2015.

[29] Distance entre mots, Thierry Lecroq, Université de Rouen, FRANCE

[30] WampServer, article disponible sur: http://www.wampserver.com/, visité le 20/05/2015.

[31] XML, article disponible sur: http://openclassrooms.com/courses/structurez-vos-donnees-avec-xml/les-bons-outils-2, visité le 21/04/2015.

[32] Postgre, article disponible sur: http://fr.wikipedia.org/wiki/PostgreSQL, visité le 21/04/2015.

[33] Java Que-ce que JAVA, article disponible sur : https://www.java.com/fr/download/faq/whatis_java.xml, visité le 20/05/2015.

[34] Netb, article disponible sur: http://fr.wikipedia.org/wiki/NetBeans, visité le 20/05/2015.