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

Université Mentouri de Constantine
Faculté des Sciences de l'Ingénieur
Département d'Informatique

N° Ordre : 117/TS/2011
Série : 17/INF/2011

THESE

Présentée à l'Université Mentouri de Constantine
Pour l'Obtention du diplôme de DOCTORAT en SCIENCES
Spécialité : Informatique

Une approche Hybride d'intégration de sources de données hétérogènes dans les datawarehouses

Par

Lahmar Fatima épouse Boulçane

Soutenue le 11 Décembre 2011
devant le Jury composé de:

Mohammed Benmohammed, Professeur, Université de Constantine..Président de jury
Mahmoud Boufaida, Professeur, Université de Constantine……Directeur de Thèse
Nacer eddine Zarour, Professeur, Université de Constantine……....Examinateur
Nadir Farah, Professeur, Université de Annaba …..……...……..Examinateur
Abdelmadjid Zidani, M.C.A, Université de Batna……..……..Examinateur

Remerciements

Mes vifs remerciements vont tout d'abord au professeur Mahmoud Boufaida pour m'avoir encadrée tout au long de ces années. Je le remercie pour toute la confiance qu'il m'a témoignée, pour son entière disponibilité, sa patience et pour tous ses conseils avisés.

Je remercie très sincèrement Mohammed Benmohammed, professeur au département d'Informatique de l'université Mentouri de Constantine, pour l'honneur qu'il m'a fait en présidant le jury de thèse.

Je remercie également Nacer eddine Zarour, Professeur, Université de Constantine, Nadir Farah, Professeur, Université de Annaba et Abdelmadjid Zidani, M.C.A, Université de Batna d'avoir accepté d'être examinateurs de mon travail. Je leur exprime ma gratitude pour l'honneur qu'ils m'ont fait en participant à ce jury, pour évaluer ce travail.

Je remercie tous ceux qui m'ont accompagnée au cours de toutes ces années, famille, proches et amis pour m'avoir apportée chacun à sa manière affection, amour, soutien et encouragements.

– Enfin, je remercie tous ceux qui ont contribué directement ou indirectement à l'aboutissement de cette thèse.

Une très tendre pensée va à mes défunts frères et à tous ceux qui ne sont plus parmi nous.

Table des matières

Introduction Générale 9
1. Contexte et problématique de l'intégration de données 10
2. Motivation 10
3. Objectifs 11
4. Principales contributions 12
5. Organisation de la thèse 13
Partie1 : Etat de l'art 16
Introduction 17
Chapitre 1 : Intégration de données hétérogènes 18
1. Introduction 18
2. Problématique de l'intégration de données 18
2.1 Hétérogénéité des données 18
2.1.1 Conflits de représentation 21
2.1.2 Conflits de nom (termes) 21
2.1.3 Conflits de contexte 21
3. Systèmes d'intégration 21
4. Evaluation de requêtes 23
5. Classification des systèmes d'intégration 24
5.1 Degré de matérialisation 24
5.1.1 Architecture virtuelle 25
5.1.2 Architecture matérialisée 26
5.2 Approches d'intégration de données 26
5.2.1 Approche GAV 27
5.2.2 Approche LAV 28
5.2.3 Approches d'intégration hybrides 29
5.3 Nature du processus d'intégration 30
5.3.1 Les approches manuelles 30
5.3.2 Approches semi-automatiques 30
5.3.3 Approches automatiques 30
5.4 Niveaux d'abstraction 31
6. Conclusion 31
Chapitre 2 : Les systèmes de médiation 32
1. Introduction 32
2. Présentation générale d'un système de médiation 32
3. Architecture de médiation 33
3.1 Adaptateur (Wrapper): 34
3.2 Médiateur (Mediator)**:** 35
4. Problèmes étudiés 35
5. Intégration dans les systèmes de médiation 36
6. Evaluation de requêtes 36
7. Prototypes 36
7.1 TSIMMIS 37
7.2 GARLIC et MIX 38
7.3 STRUDEL 39

7.4 YAT 41
7.5 AGORA / LeSelect 42
8. Conclusion 43
Chapitre 3 : Les entrepôts de données (Datawarehouses) 44
1. Introduction 44
2. Principe général 44
2.1 Définition d'un entrepôt de données 44
2.2 Approche entrepôt de données 45
3. Processus d'intégration de données dans les entrepôts de données 47
3.1 Etapes d'intégration 47
3.2. Types d'intégration 49
4. Alimentation d'un entrepôt de données 50
4.1 Définitions 50
5. Prototypes 51
5.1 WHIPS 51
5.2 SIRIUS 52
5.3 DWQ 52
5.4 Projet EVOLUTION 53
6. Conclusion 54
Bilan 55
PARTIE 2 : 57
Introduction 58
Chapitre 4 : Présentation de l'approche HAV pour l'intégration de sources de données hétérogènes dans les 60
1. Introduction 60
2. Mapping 60
2.1 Global-as-View (GAV) 61
2.2 Local-as-View (LAV) 62
2.3 Bilan sur les approches GAV et LAV 64
3. Présentation de l'approche HAV 64
3.1 Intégration de sources de données dans HAV 65
3.1.1 Transformation de schémas dans HAV 66
3.1.2 Transformation de langages dans HAV 66
3.2 Traitement de requêtes dans HAV 67
4. Une architecture pour HAV 69
4.1 Description des différents composants du système 71
4.2 Comparaison des différentes approches avec HAV 73
4.3 Synthèse des approches 75
5. Cadre formel pour HAV 76
5.1 Définition Formelle d'un système d'intégration basé sur HAV 76
5.2 Définition formelle du mapping dans HAV 79
6. Conclusion 79
Chapitre 5: Validation de l'approche HAV 81
1. Introduction 81
2. Validation qualitative 81
2.1 Etude de cas 81
2.1.1 Le mapping MSp,S: .(Mapping LAV, sources-schémas partiels) 82

2.1.2 Le mapping: *Mg,p* : (Mapping GAV, schéma global- schémas partiels) 83
2.1.3 Traitement de requêtes dans HAV 83
2.1.4 Ajout d'une nouvelle source 85
2.1.5 Résumé 86
3. La Validation quantitative 88
3.1 Surcoût induit par l'architecture de médiateurs empilés 88
3.2 Matérialisation d'un entrepôt de données 89
3.3 Validation par comparaison 92
3.3.1 Méthode d'implémentation: 92
3.4 Intégration de sources de données de même modèle 102
4. Conclusion 103
Conclusion et perspectives 104
Références 107

Liste des figures

PARTIE 1 ……. 16

Figure 1.1 Exemples de catalogues électroniques hétérogènes …….. 19
Figure 1.2 Système d'intégration……. 22
Figure 1.3 Approche GAV …….. 27
Figure 1.4 Approche LAV …….. 28
Figure 1.5 Approche GLAV …….. 29
Figure 2.1 Architecture de médiation ……. 34
Figure 2.2 Architecture de TSIMMIS …… 38
Figure 2.3 Architecture de GARLIC ……39
Figure 2.4 Architecture de STRUDEL ……40
Figure 2.5 Architecture de YAT……... 41
Figure 2.6 Architecture d'AGORA ……. 43
Figure 3.1 Architecture fonctionnelle d'un data warehouse [Gardarin 2001]…….. 45
Figure 3.2 Etapes de traitement du premier niveau de construction ……47
d'un entrepôt de données
Figure.3.3 Vue opérationnelle des composants utilisés pour la construction…….... 48
d'entrepôts de données
Figure 3.4 Architecture d'entreposage virtuel ……..49

PARTIE 2 …….. 57

Figure 4.1 Exemple de définition du schéma global dans GAV……. 61
Figure 4.2 Exemple d'ajout d'une nouvelle source dans GAV …….. 62
Figure 4.3 Exemple de définition du schéma global dans LAV ……..63
Figure 4.4 Exemple d'ajout d'une nouvelle source dans LAV ……63
Figure 4.5 Exemple dans HAV ……... 66
Figure 4.6 Principe de traitement de requêtes dans un système …….. 69
d'intégration construit selon HAV.
Figure 4.7 Une architecture à trois niveaux ……. 71
Figure 5.1 Temps de réponse suivant l'architecture Mo et M1 [NGOG 2003] ….. 89
Figure 5.2 Prise d'écran de l'implémentation de l'exemple dans l'ETL SQL ….... 93
Server Integration Services 2008

ملخص :

المشاكل الرئيسية لأنظمة إدماج المعطيات بما في ذلك مستودعات المعطيات أو أنظمة الوساطة ترتبط ارتباطا وثيقا وذلك لعدم تجانس المعطيات و عدم قابلية تطور أنظمة المعطيات ومدى تعقيد معالجة الاستعلام وتغيرات في الخطة الشاملة. العمل المقدم في هذه الأطروحة هو أساسا بحث ذو توجه مزدوج يستهدف من ناحية ملخص أنظمة الإدماج والاقتراحات التي يمكن أن تنشأ عن مثل هذا الملخص.
نقترح إذا نهجا لإدماج المعطيات الغير متناسقة والتي عرفناها بـ HAV(Hybrid As View). مساهمة هذا النهج هو عن موضوعين متكاملين:
أ- اقتراح هيكل وسطاء متعددة التي تتكون أساسا من نوعين من العناصر: وسطاء متخصصة ووسيط شامل. الوسطاء المتخصصة يقدم كل واحد منها نظرة شاملة لمصادر المعطيات المتجانسة. الوسيط الشامل يدمج المخططات الجزئية التي تقدمها مجموعة الوسطاء المتخصصة لتوفير الوصول إلى رؤية موحدة ممثلة بمخطط شامل
ب- نمثل بنموذج العلاقة بين المخطط الشامل والمصادر من خلال المخططات الجزئية بالجمع بين أفضل ما في النهج GAV (Global As View)وLAV(Local As View)

كلمات المفتاح :
مستودع المعطيات، الوسيط، نهج GAV ، نهج LAV، إدماج المعطيات الغير متناسقة.

Résumé:

Les problèmes majeurs des systèmes d'intégration de données notamment les entrepôts de données ou les systèmes de médiation, sont étroitement liés à l'hétérogénéité des données, l'évolutivité de l'intégration de données, la complexité du traitement des requêtes et l'évolution du schéma global. Le travail présenté dans cette thèse s'inscrit essentiellement dans une double orientation de recherche dont les objectifs sont d'une part la synthèse des approches d'intégration de données et d'autre part les propositions qui peuvent naître d'une telle synthèse. Nous proposons une approche hybride de l'intégration de données hétérogènes baptisée : Hybrid As View (HAV). La contribution de cette approche est sur deux axes complémentaires: (i) proposer une architecture multi-médiateurs essentiellement constitué de deux types de composants: les médiateurs spécialisés et un médiateur global. Les médiateurs spécialisés offrent chacun une vue intégrée des sources de même modèle. Le médiateur global intègre les schémas partiels fournis par l'ensemble des médiateurs spécialisés pour fournir un accès sur une vue uniforme représentée par un schéma global. (ii) modéliser la relation entre le schéma global et les sources à travers les schémas partiels en combinant le meilleur des deux approches Global As View (GAV) et Local As View (LAV).

Mots-clef : entrepôt de données, médiateur, approche GAV, approche LAV, intégration de données hétérogènes.

Abstract:

The major problems of data integration systems are closely related to the heterogeneity of the data, the scalability of data integration, the query processing complexity and the global schema evolution. The work presented in this thesis is essentially a double orientation of research whose objectives are, on one hand the synthesis of data integration approaches and on the other hand the propositions which can arise from such a synthesis. We propose a hybrid approach of the

integration of heterogeneous data baptized: Hybrid As View (HAV). The contribution of this approach is on two complementary axes: (i) to propose a multi-mediators architecture essentially made up of two types of components: specialized mediators and a global mediator. The specialized mediators provide each one an integrated view of sources with the same model. The global mediator integrates the partial schemas provided by the set of the specialized mediators to provide an access on a uniform view represented by a global schema. (ii) to model the relation between the global schema and the sources through the partial schemas by combining the best of the two approaches Global As View (GAV) and Local As View (LAV).

Key-words: Datawarehouse, mediator, GAV approach, LAV approach, heterogeneous data integration.

Introduction Générale

Les entrepôts[1] de données ou 'datawarehouse' [INMON et al 1994] étendent les concepts et la technologie traditionnelle des bases de données pour créer des systèmes d'aide à la décision. Ils permettent, d'unifier les données de production issues de sources hétérogènes de manière à les rendre exploitables par une analyse décisionnelle. En effet, Ils intègrent des informations provenant des sources de données qui sont souvent hétérogènes et distribuées, qui ont pour objectif de fournir une vue globale de l'information aux analystes et aux décideurs. En conséquence, la conception et la mise en œuvre d'un entrepôt est une tâche complexe. Elle se compose de trois processus : extraction-intégration, organisation et interrogation. Nous trouvons le processus d'extraction-intégration entre les sources de données et l'entrepôt. Ce processus est responsable de l'identification des données dans les diverses sources internes et externes, de l'extraction de l'information qui nous intéresse, de la préparation et de la transformation (nettoyage, filtrage,...) des données.

A l'intérieur de l'entrepôt, nous trouvons le processus d'organisation, il est responsable de structurer les données par rapport à leur niveau de granularité (agrégats).
Finalement, le troisième processus correspond à l'interrogation qui se place entre l'entrepôt et les différents outils pour arriver à l'analyse des données, pour les différents utilisateurs de l'entreprise.

Chaque processus présente des problématiques spécifiques. Ainsi, par exemple, lors du premier processus, la difficulté principale consiste en l'intégration des données, de manière à ce qu'elles soient de qualité pour leur stockage. Pour l'organisation, ils existent plusieurs problèmes comme: la sélection des vues à matérialiser, le rafraîchissement de l'entrepôt, la gestion de l'ensemble de données (courantes et historisées), entre autres. En ce qui concerne le processus d'interrogation, nous avons besoin d'outils performants et conviviaux pour l'accès et l'analyse de l'information.

Notre proposition se place à l'intérieur des processus d'extraction-intégration. En effet, le thème de cette thèse est d'étudier comment combiner au mieux des approches d'intégration de données. Un système d'intégration de données peut être caractérisé par son architecture et son modèle d'intégration. Nous pouvons distinguer entre deux architectures fondamentales pour l'intégration de données. L'approche *médiateur* [Wiederhold 1992, 1995] est fondée sur la définition de *mapping* permettant la traduction de requêtes : une requête formulée par l'utilisateur dans les termes du schéma global est traduite en une ou plusieurs sous-requêtes qui sont évaluées sur les données sources. Les réponses sont combinées et transformées afin d'être compatibles avec le schéma global et conformes à la requête posée par l'utilisateur. L'approche *entrepôt* [Widom 1995] applique le principe des *vues matérialisées* et intègre les données en accord avec le schéma global. Le résultat est un entrepôt de données qui peut directement être interrogé à travers un langage adapté.

[1] Nous utiliserons dans cette thèse aussi bien 'entrepôt de données' que 'datawarehouse' pour signifier la même chose.

Chacune de ces deux approches a des avantages et des inconvénients en termes de traitement des données intégrées [Doucet et al 2001]. L'approche virtuelle favorise l'intégration de sources qui sont toujours disponibles avec de mises-à-jour fréquentes. L'approche matérialisée facilite l'intégration de sources qui ont des capacités d'interrogation limitées et/ou ne sont pas facilement accessibles. Elle pose le problème de rafraîchissement des données, mais permet des traitements de données indépendamment des capacités des sources.

Il existe également des approches hybrides qui essaient de combiner les avantages de ces deux approches en matérialisant les données dans un entrepôt et en utilisant l'approche virtuelle pour leur intégration. Un exemple d'un tel système est Xylème où les données sources sont stockées dans le format XML (sans être transformées) et intégrées à travers un mécanisme de vues entre les DTD concrètes des sources et des DTD abstraites montrées à l'utilisateur [Cluet et al 2001].

Cette thèse a pour cadre l'intégration de sources hétérogènes dans les entrepôts de données. Ainsi, dans ce chapitre nous présentons la motivation et les objectifs de notre travail, les questions et les contributions de recherche, ainsi que l'organisation de cette thèse.

1. Contexte et problématique de l'intégration de données

Les entrepôts de données ont été conçus pour l'aide à la décision. Ils intègrent les informations en provenance des différents systèmes transactionnels de l'entreprise. L'ensemble des données, y compris leur historique, est utilisé pour faire des calculs prévisionnels, des statistiques ou pour établir des stratégies de développement et d'analyses des tendances.

L'intégration de bases de données de ces systèmes est un problème complexe [Hacid et al 2005]. C'est pourtant une tâche que les entreprises peuvent difficilement éviter si elles veulent mettre en route de nouvelles applications ou réorganiser le système d'information existant pour une meilleure productivité. En effet, les sources sont souvent hétérogènes car elles ont été définies indépendamment les unes des autres. Le processus d'intégration doit donc permettre de traiter des sources qui ont des modèles de données et/ou des schémas différents. Dans un tel contexte, il est souvent nécessaire pour une application d'accéder simultanément à plusieurs sources, du fait qu'elles contiennent des informations pertinentes et complémentaires. Pour ce faire, la solution des systèmes d'intégration tels que les systèmes de médiation et les entrepôts de données a été proposée. Cette solution émerge dans une variété de situations commerciales (quand deux compagnies semblables doivent fusionner leurs bases de données) et scientifiques (intégration des résultats de recherche de différents laboratoires en bioinformatique). Le problème général abordé par notre travail de thèse concerne l'intégration de données hétérogènes dans les entrepôts de données avec l'objectif d'offrir une vue homogène de l'information d'une manière satisfaisante.

2. Motivation

L'environnement informationnel actuel se caractérise par des données fortement distribuées. Ces données surabondantes sont généralement éparpillées, puisqu'il existe souvent de multiples systèmes conçus chacun pour être efficace pour les fonctions pour lesquelles il est spécialisé. Ces

données sont également hétérogènes. En effet, avec l'apparition de l'internet et le développement des différentes représentations et formats des documents, les données peuvent être de plusieurs types : structurées (données relationnelles, données objet), semi-structurées (HTML, XML, graphes) ou même non structurées (texte, images, son). Dans un tel contexte, le besoin d'intégration se fait de plus en plus sentir. Cependant, pour répondre à ce besoin, le développement des applications d'intégration (telles que pour un traitement élaboré de données, pour la construction des systèmes de médiation ou des entrepôts de données) se voit contraint de composer avec la répartition des sources et l'hétérogénéité de leurs structures.

La problématique de l'intégration de données est générale à la constitution de tout entrepôt mais nous devons ici tenir compte de la nature hétérogène des sources car l'intégration de données hétérogènes de différents types de structuration est un volet de recherche intéressant. En effet, la plupart des travaux sur l'intégration en particulier dans les entrepôts de données traitant des données homogènes, l'intégration des données de différents types structurées, semi-structurées et non structurées reste un sujet récent et assez peu exploré. De plus, aucune des approches d'intégration proposées dans la littérature n'est pleinement satisfaisante.

3. Objectifs

Le but de notre travail est de contribuer à l'intégration de données hétérogènes dans les entrepôts de données. Ces derniers peuvent être définis comme étant des collections de données provenant de différentes bases de données, *intégrées*, orientées sujet, variant dans le temps, stockées et organisées pour alimenter une opération d'aide à la décision. En effet, notre objectif est l'intégration de données dans un entrepôt de données se basant sur plusieurs sources de données hétérogènes structurées, semi-structurées et non structurées. Certaines données de ces sources doivent être extraites, *intégrées*, transformées et fusionnées afin d'être présentées dans l'entrepôt. L'intégration nécessite alors l'accès aux données des sources et la capacité de les fusionner conformément à un schéma global.
L'objectif majeur est de proposer une approche d'intégration de données hétérogènes dans les entrepôts de données en utilisant l'approche virtuelle pour leur intégration.

Ce travail consiste dans un premier temps en la présentation de cette approche d'intégration de données hétérogènes dans le contexte des systèmes de médiation que nous adapterons dans un deuxième temps à la matérialisation d'un datawarehouse.

Nous tenons à préciser la définition de l'hétérogénéité, car elle est assez ambiguë dans la littérature. En effet, certains travaux traitant des données de différents types parlent de l'hétérogénéité. Alors que d'autres, traitant des données de même type mais avec des modélisations différentes utilisent aussi le terme d'hétérogénéité. Nous retrouvons même des travaux traitant des données de même type avec la même modélisation qui parlent de données hétérogènes.

Nous retiendrons pour notre thèse le concept de données hétérogènes : « toutes les données vérifiant l'une des deux propriétés suivantes :

1. Elles appartiennent au même type de données mais elles ont des modélisations différentes. Ainsi le traitement d'une base de données relationnelle et d'une base de données objet revient à traiter des données hétérogènes.

2. Elles appartiennent à de différents types de données. Ainsi le traitement d'une base de données relationnelle et d'une base de documents XML entre dans le cadre du traitement de données hétérogènes. »

Par ailleurs, il n'y a pas de modèle de représentation unique pour les données intégrées. Plusieurs modèles, selon les caractéristiques des sources et les manipulations à effectuer peuvent être retenus.

En résumé et ayant cette définition de l'hétérogénéité, nous visons à intégrer des sources de données hétérogènes en matérialisant les données dans un entrepôt et en utilisant l'approche virtuelle pour leur intégration. Cette intégration se fait en établissant une approche hybride d'intégration que nous avons baptisée Hybrid-As-View (HAV). L'objet de nos travaux est donc de présenter HAV ainsi que l'architecture qui la supporte, de proposer un cadre formel pour HAV et de la valider.

4. Principales contributions

Nous avons souligné précédemment que notre travail se situe principalement dans l'étape d'extraction-intégration. Pour cela, nous nous sommes focalisés sur les problématiques de la définition du schéma d'intégration de l'entrepôt, sur l'évolutivité de ses sources et sur sa matérialisation proprement dite. L'approche HAV que nous proposons pour la résolution de ces problèmes se base sur la combinaison des deux principales approches d'intégration GAV et LAV [Boulçane.F et al 2008] [Boulçane.F 2007] [Boulçane.F 2006], pour pallier les inconvénients de l'une et de l'autre quand elles sont utilisées séparément et répondre ainsi au besoin d'intégrer des sources de données hétérogènes d'une manière satisfaisant et efficace.
Dans ce qui suit, nous résumons, les principales contributions concernant HAV:

Première contribution

Les recherches sur les systèmes d'intégration ont fourni un ensemble riche d'approches d'intégration sur lesquelles peuvent être construits ces systèmes notamment les entrepôts de données. Les deux approches communément utilisées sont Global-As-View (GAV) et Local-As-View (LAV). Dans GAV le schéma médiateur est défini comme un ensemble de vues sur les sources de données et dans LAV les contenus des sources sont décrits comme des vues sur le schéma médiateur. Il existe par ailleurs des approches hybrides pour l'intégration de données.
L'approche HAV que nous proposons dans cette thèse est une autre alternative de combinaison des deux approches de base GAV et LAV. HAV combine GAV et LAV au vu de leurs caractéristiques duales comme nous le verrons. La contribution majeure dans notre travail est la combinaison des points forts de GAV et LAV qui permet ainsi de profiter pleinement de leurs avantages [Boulçane.F 2006].

L'originalité de notre approche réside dans la façon dont est réalisée la combinaison. En effet, l'architecture qui supporte notre approche HAV est une architecture multi-médiateurs composée essentiellement de deux types de composants: un ensemble de médiateurs spécialisés définis selon l'approche LAV qui correspond au niveau inférieur de l'architecture et un médiateur global défini selon l'approche GAV qui correspond au niveau supérieur de l'architecture. Les médiateurs spécialisés offrent chacun une vue intégrée de sources ayant le même modèle. Le médiateur global intègre les schémas partiels fournis par les médiateurs spécialisés pour donner un accès sur une vision uniforme représenté par un schéma global.

HAV est une approche qui vise à combiner le meilleur de GAV et LAV, elle réduit leurs limitations afin d'améliorer l'efficacité d'intégration de données en assurant la flexibilité et l'évolutivité. En effet, il est connu que la complexité de la vue globale croît avec le nombre de sources et l'approche GAV est plus efficace et réalisable si l'ensemble des sources de données à intégrer est petit et stable. Ces deux qualités sont réalisées dans notre approche comme nous le verrons dans la deuxième partie de cette thèse [Boulçane.F 2007]. En outre, l'inconvénient majeur de l'approche GAV est l'ajout d'une nouvelle source. Dans notre architecture, l'ajout d'une nouvelle source n'affecte pas le schéma global parce que cette tâche est traitée par le médiateur spécialisé correspondant qui est construit selon l'approche LAV. En d'autres termes, cette source est construite comme une vue sur le schéma partiel correspondant. Cela n'a aucun effet sur le schéma global parce que l'approche LAV est très flexible à l'ajout (ou la suppression) des sources de données à intégrer.

Deuxième contribution

Les requêtes dans HAV seront exprimées en termes du schéma global et leur exécution se fait en atteignant chaque source concernée par l'intermédiaire du médiateur spécialisé [Boulçane.F et al 2008]. Il est connu que le prix à payer pour la souplesse et la simplicité de mise à jour dans un médiateur conçu selon l'approche LAV est la complexité de la construction des réponses à une requête. Dans HAV cette complexité sera considérablement réduite parce que chaque médiateur spécialisé a un nombre réduit de sources à intégrer, et ces sources sont de même modèle. Ainsi, la réécriture de requêtes en termes de vue sera plus facile.

Cette thèse offre un cadre formel [Boulçane.F et al 2008] pour la définition de notre approche d'intégration de données. Ce cadre formel permettra de générer des vues virtuelles dans le domaine de l'intégration de données et de définir la correspondance entre le schéma global et les schémas sources selon un ensemble prédéfini de règles, en introduisant une couche intermédiaire de sources non matérialisées représentées par des schémas partiels entre le schéma global et les sources de données réelles.

5. Organisation de la thèse

Pour présenter nos travaux et le domaine dans lequel ils s'inscrivent, nous avons retenu pour ce mémoire de thèse une organisation en deux parties principales. La première comporte trois chapitres sur l'état de l'art en rapport avec les thèmes qu'aborde notre travail. La seconde partie décrit notre apport concernant l'intégration de données hétérogènes dans les entrepôts de données.

Nous donnons ci-dessous une vue d'ensemble suivi d'un bref résumé de chaque chapitre.

Première partie

La partie 1 est un état de l'art portant sur le problème de l'intégration de sources de données distribuées, autonomes, et hétérogènes. La contribution principale de cette synthèse est de proposer une classification des systèmes d'intégration existants. Cette classification se base principalement sur les critères de la représentation de données intégrées (virtuelle ou matérialisée), sur la nature hétérogène des données et sur le sens de la mise en correspondance entre schéma global et schémas locaux. Enfin, nous positionnons nos travaux et nous définissons notre problématique de recherche.

Dans le premier chapitre nous précisons le cadre de cette thèse en définissant les concepts de base des systèmes d'intégration de données. Nous décrivons les principaux problèmes d'hétérogénéité rencontrés et traités en général par la communauté des bases de données. Nous présenterons également les principales solutions ou approches qui sont proposées actuellement pour intégrer des sources de données hétérogènes dans les systèmes de médiation ou dans les entrepôts de données.

Dans le deuxième chapitre, nous précisons la place de l'intégration de données dans les systèmes de médiation, nous effectuons un état de l'art concernant la recherche actuelle dans les systèmes de médiation et nous décrivons quelques systèmes de médiation représentatifs.

Le troisième chapitre présente les caractéristiques et les fonctionnalités principales d'un entrepôt de données ainsi que les différentes approches d'intégration de sources hétérogènes dans les datawarehouses, après quoi nous décrivons quelques systèmes d'intégration représentatifs.

Deuxième partie

La seconde partie sera centrée sur l'approche d'intégration des données proposée à savoir HAV dont l'architecture, les composants et les fonctionnalités seront présentés et détaillés. Cette partie est composée de deux chapitres et concerne notre contribution pour l'intégration de données hétérogènes en général et les entrepôts de données en particulier.

Le chapitre 4 concerne la présentation de l'approche que nous proposons pour l'intégration de données hétérogènes. Nous présentons son architecture globale et nous détaillons ses différents composants. Un point important de l'approche proposée pour l'intégration de sources hétérogènes est le cadre formel proposé. Nous utilisons le formalisme proposé par M.Lenzerini [Lenzerini 2002] pour les systèmes d'intégration basés sur un schéma global et que nous adaptons à notre architecture multi-médiateurs.

Le chapitre 5 est consacré à la validation de notre approche dans lequel nous commencerons avec une comparaison de HAV avec les approches de base GAV et LAV pour ensuite la valider qualitativement (en utilisant une étude de cas) et quantitativement.

En conclusion, nous dressons le bilan de ce travail réalisé dans le cadre de l'intégration de données hétérogènes dans les entrepôts de données avec un résumé des principales contributions et limitations. Nous dégagerons ensuite les perspectives envisageables pour ces travaux.

Partie1 : Etat de l’art

Introduction

Les besoins en intégration de données sont cruciaux surtout avec l'essor du Web et le nombre croissant de sources et leur diversité. L'intégration de données est le processus qui consiste à combiner plusieurs sources de données de telle sorte qu'elles puissent être interrogées et mises à jour à travers une interface commune.

Un système d'intégration peut être réalisé grâce à des vues matérialisées, comme c'est le cas dans les entrepôts de données, ou grâce à des vues virtuelles dans le cas des systèmes de médiation. La communauté bases de données a beaucoup investi dans le développement d'architectures et d'outils pour l'intégration des données [Wiederhold 92]. Des approches de type entrepôts de données, fédération de bases de données ou médiateur/wrapper ont été proposées pour faciliter l'intégration des données [Cody et al 1995] [Roth et al 1997] [Cluet et al 1998] [Bright et al 1999] [Lacroix 2000] pour accéder aux sources de données, en extraire et traduire les résultats pour les représenter dans un modèle commun. Dans la littérature, nous trouvons différentes méthodes qui sont proposées pour l'intégration et la transformation des données et des schémas de données. Nous pouvons également trouver diverses classifications de ces méthodes.

Les systèmes d'intégration doivent permettre à l'utilisateur d'accéder, via une interface d'accès unique, à des données stockées dans plusieurs sources de données [Halevy et al, 2005], [Xu et al, 2004], [Lawrence et al 2002]. Ces sources de données diverses ont été conçues indépendamment par des concepteurs différents. En effet, la diversité des sources d'information distribuées et leur hétérogénéité sont une des principales difficultés rencontrées lors de leur intégration. Cette hétérogénéité peut provenir du format ou de la structure des sources (sources structurées : bases de données relationnelles, sources semi-structurées : documents XML, ou non structurées : textes), du mode d'accès et de requêtes.

Dans cette thèse, notre attention est portée sur les méthodes de base d'intégration virtuelle de données qui nous permettent d'intégrer les schémas, puis de matérialiser les données d'un entrepôt sur la base du schéma intégré. Pour ce faire, Nous donnerons dans cette première partie une image claire de l'intégration de données hétérogènes et distribuées en général, des systèmes de médiation et des entrepôts de données en particulier. Après quoi nous montrerons ce que les solutions actuelles peuvent réaliser et ainsi cerner le travail restant à faire pour mieux situer notre travail.

Chapitre 1 : Intégration de données hétérogènes

1. Introduction

Le problème de combiner des sources de données hétérogènes et de les interroger via une seule interface de requête ne date pas d'aujourd'hui. Depuis l'arrivée et l'adoption des bases de données, leur partage et leur combinaison sont naturellement devenus indispensables. Cette combinaison peut être effectuée de différentes façons et à différents niveaux de l'architecture du système. Intégrer les sources de données dans le but de fournir aux utilisateurs une interface d'accès uniforme est une tâche difficile. Cette difficulté concerne les aspects d'hétérogénéité des données, d'autonomie des sources, de la correspondance entre le schéma global et les schémas globaux et de la nature du processus d'intégration.
Ce chapitre fournit une description générale des systèmes d'intégration et un aperçu plus détaillé de plusieurs méthodes spécifiques qui sont utilisées pour l'intégration de données.

2. Problématique de l'intégration de données

Avant de faire un état de l'art des systèmes d'intégration de données, il est important de définir la nature du problème à résoudre.

Plusieurs problèmes doivent être pris en compte pendant la conception de systèmes d'intégration. Ces problèmes résultent de l'hétérogénéité des données. Cette hétérogénéité est dûe au fait que les sources de données ont été conçues indépendamment par des concepteurs différents, ceci explique le fait que les données relatives à un même sujet sont représentées différemment sur des systèmes d'information distincts. Cette hétérogénéité provient des choix différents qui sont faits pour représenter des faits du monde réel dans un format informatique. En effet, les données des sources sont structurellement indépendantes mais sont toujours supposées relever de domaines similaires.

L'intégration de données hétérogènes a conduit à de nombreux problèmes qui peuvent se classer en deux catégories : l'intégration de données hétérogènes et l'évaluation des requêtes [Ngog 2003], [Fuxman et al 2005].

2.1 Hétérogénéité des données

La question de l'hétérogénéité de données a longtemps été étudiée dans la communauté des bases de données [Lenzerini 2002], [Manolescu 2001], [Levy et al 1996], [Leonidas et al 2003]. En général, le problème de l'hétérogénéité peut concerner deux catégories [Hao et al 2004]:

- Hétérogénéité structurelle ou schématique: elle provient quand les sources adoptent différents modèles de données, structures de données ou schémas, par exemple les modèles de bases de données relationnelles ou orientées objets. De nombreux travaux concernant ce type d'hétérogénéité ont été proposés dans les contextes des bases de données fédérées et des multi-bases de données.

- Hétérogénéité sémantique : elle est dûe aux conflits sémantiques dans les termes, les expressions, etc., qui sont adoptés par différents schémas de données mais exprimés de diverses manières. Autrement dit, elle est dûe aux différentes interprétations pour les objets du monde réel. En effet, les sources de données ont été conçues indépendamment par des concepteurs différents ayant des objectifs applicatifs différents. Chacun peut donc avoir un point de vue différent sur le même concept. L'interopérabilité sémantique de données présente un défi majeur dans le processus d'´elaboration des systèmes d'intégration.

Avant de faire un état de l'art des systèmes d'intégration de données, il est important de définir la nature du problème à résoudre.

Les systèmes d'intégration doivent permettre à l'utilisateur d'accéder, via une interface d'accès unique, à des données stockées dans plusieurs sources de données. Ces sources de données ont été conçues indépendamment par des concepteurs différents. Cela entraîne l'hétérogénéité de données, c-`a-dire que les données relatives à un même sujet sont représentées différemment sur des systèmes d'information distincts. Cette hétérogénéité provient des choix différents qui sont faits pour représenter des faits du monde réel dans un format informatique. En effet, les données des sources sont structurellement indépendantes mais sont toujours supposées relever de domaines similaires.

Exemple 1 :

Pour illustrer ce problème d'hétérogénéité de sources, nous étudions l'exemple dans la figure 1.1 [Xuan 2006].

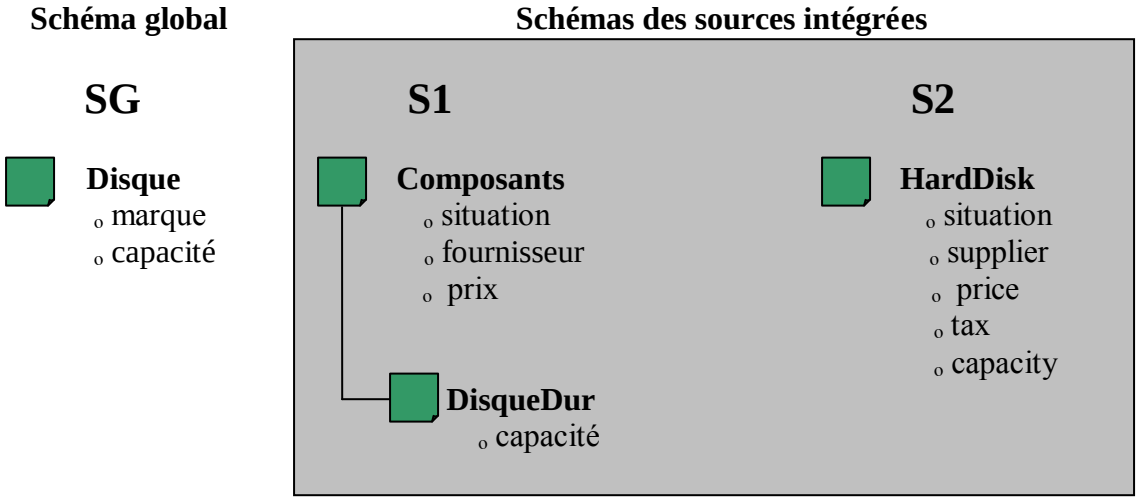


Figure. 1.1 : Exemple de catalogues électroniques hétérogènes

Supposons qu'une entreprise de commerce électronique vend des disques durs sur l'Internet. Les disques durs mis en vente sont fournis par des fournisseurs différents, où chacun organise ses produits dans son catalogue selon ses propres critères décidés localement. Cette entreprise doit

donc traduire les catalogues de différents fournisseurs à son format, appelé schéma global. Pour des raisons de simplicité, nous considérons deux catalogues S1 et S2 de deux fournisseurs et le catalogue de l'entreprise SG (figure 1.1).
Remarquons que les trois catalogues décrivent un disque dur différemment. Cette différence concerne le nombre de concepts utilisés pour définir chaque source, ainsi que l'aspect sémantique de chaque concept. Le tableau 1.1 décrit les différentes propriétés de SG, S1 et S2.

	SG	S1	S2
Classes	Disque dur : disque dur sous garantie	Composants : compo - sant informatique	HardDisk : disque dur
		Disque dur : disque dur	
Propriétés		Situation (domaine : boolean) : neuf ou occasion	Situation (domaine : boolean) : disponible ou non
	Marque (domaine : string) : marque de fabrique	Fournisseur (domaine : string) : marque de fabrique	Supplier (domaine : string) : marque de fabrique
		Prix (domaine : number) : le prix total (qui inclut la TVA) d'un matériel neuf	Price (domaine : number) : le prix hors taxe d'un disque dur
			Tax (domaine : number) : la TVA d'un disque dur
	Capacité (domaine : integer) : total de méga octets	Capacité (domaine : integer) : total de méga octets	Capacity (domaine : number) : total de giga octets

TAB 1.1 : Sémantique de données de trois catalogues

La question fondamentale lorsque nous voulons faire interopérer des bases de données hétérogènes est d'une part, l'identification de conflits entre les concepts dans des sources différentes qui ont des liens sémantiques, d'autre part, la résolution des différences entre les concepts sémantiquement liés.

Une taxonomie des conflits sémantiques a été proposée dans [Goh et al 1999] : (1) conflits de représentation, (2) conflits de nom, (3) conflits de contexte, et (4) conflits de mesure de valeur que nous allons détailler dans les sections suivantes.

2.1.1 Conflits de représentation

Ces conflits se trouvent dans le cas où nous utilisons des propriétés différentes ou des schémas différents pour décrire le même concept. Cela signifie que le nombre de classes et propriétés et les classes et propriétés représentant un concept *C* dans les sources ne sont pas égaux.

Exemple 2 :

Reprenons la Figure 1.1, où le fournisseur S1 utilise deux classes : Composants et DisqueDur et **4** propriétés : situation, fournisseur, prix, et capacité pour décrire un disque dur. Tandis que le fournisseur S2 utilise une seule classe : HardDisk et **5** propriétés : situation, supplier, price, tax, et capacity.
Un autre exemple de conflit de représentation entre les deux fournisseurs à mettre en évidence, c'est le cas o`u le fournisseur S2 utilise deux propriétés : price et tax pour calculer le prix d'un disque dur, tandis que S1 n'en utilise qu'une seule, à savoir prix.

2.1.2 Conflits de nom (termes)

Ces conflits se trouvent dans le cas où nous utilisons soit des noms différents pour le même concept ou propriété (*synonyme*), soit des noms identiques pour des concepts (et des propriétés) différents (*homonyme*).

Exemple 3 :

Le même concept de disque dur est nommé par DisqueDur dans la S1, et par HardDisk dans la S2. La propriété Situation se trouve dans les deux sources, mais avec deux significations différentes (voir Table 1.1).

2.1.3 Conflits de contexte

Le contexte est une notion très importante dans les systèmes d'information répartis. En effet, un même objet du monde réel peut être représenté dans les sources de données par plusieurs représentations selon un *contexte local à* chaque source. Ces conflits de contexte se trouvent dans le cas où les concepts semblent avoir la même signification, mais ils sont évalués dans différents contextes.

Exemple 4 :

La propriété prix de disque dur ne s'applique que pour un disque dur neuf dans S1, mais peut l'être pour tous les disques soit neufs ou d'occasions dans S2.
Par exemple, l'unité de mesure de la capacité dans la *S*1 est le méga octets tandis que celle dans la *S*2 est le giga octets.

3. Systèmes d'intégration

Au cours des vingt dernières années, trois architectures complémentaires de systèmes d'intégration d'informations se sont imposées : les entrepôts de données, les

médiateurs et les systèmes pair-à-pair. Les entrepôts de données stockent localement une partie des données des sources distantes. L'avantage est d'avoir un temps de réponse faible puisque les requêtes posées sont évaluées localement. Toutefois, l'inconvénient majeur est qu'il faut trouver une bonne fréquence de rafraichissement des données de l'entrepôt afin qu'elles ne se périment pas.
Contrairement aux entrepôts, les médiateurs ne stockent pas de données localement. Ceci les rend moins efficaces que les entrepôts lors de l'évaluation de requêtes, mais permet de garantir que les réponses sont calculées sur des données non périmées : les données des sources.
Enfin, les systèmes pair-à-pair permettent le passage à l'échelle des applications d'intégration d'informations, lorsque le nombre de sources à intégrer n'est plus gérable efficacement via une architecture centralisée. Dans notre travail nous nous intéressons aux deux premières architectures.

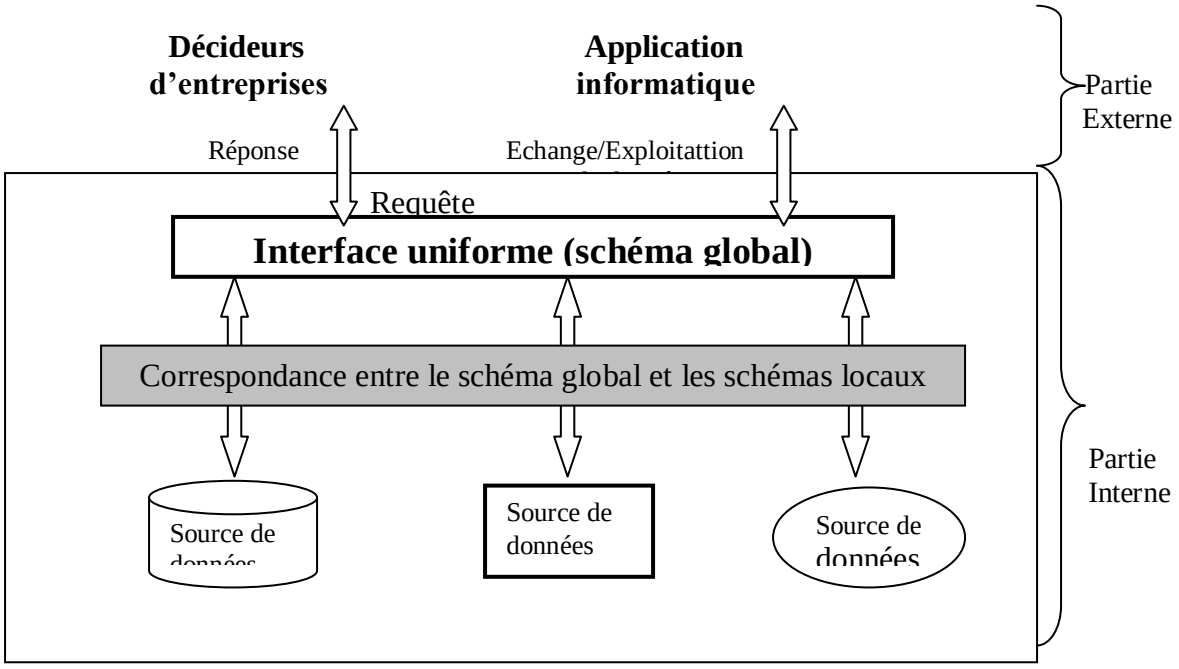


Figure 1.2 Système d'intégration

Dans les systèmes de médiation et les entrepôts de données, l'intégration de données consiste à éliminer d'abord les conflits entre les données causées par leur hétérogénéité (tel que cela a été présenté dans la section ci-dessus) et ensuite les représenter dans un seul schéma cohérent. Tout système d'intégration doit fournir les solutions aux problèmes suivants : (i) l'identification et la spécification des correspondances entre des données sémantiquement liées, (ii) fournir une vue globale intégrée des données représentées à travers des conceptualisations différentes, (iii) la gestion des mises à jour des données de différentes sources et leurs répercussions sur le schéma global [Xuan 2006].
Généralement, un système d'intégration comporte deux parties principales une *partie externe* et une *partie interne* (voir la figure 1.2):

- **La partie externe** correspond aux utilisateurs du système intégré comme, par exemple, les décideurs d'une entreprise, ou d'autres systèmes.

- **La partie interne** comprend d'une part les sources d'informations participant dans le processus d'intégration et d'autre part une interface (schéma global) permettant aux éléments de la partie externe d'accéder d'une manière transparente aux sources de données. Cette interface peut être une couche sans données propres (*approche médiateur*) ou une couche contenant sous une forme qui lui est propre une duplication des données pertinentes des sources (*approche entrepôt*). Les éléments externes au système ne voient pas les sources internes. Celles-ci sont *encapsulées, cachées* par l'interface. Les éléments externes doivent pouvoir agir sur le système d'intégration d'une manière transparente via un *schéma global*.

4. Evaluation de requêtes

Le traitement de requêtes est un mécanisme absolument obligatoire dans l'intégration des bases de données. Il commence par une reformulation de celle-ci à partir du schéma global aux schémas à l'exportation des sources de données. Les algorithmes pour la reformulation des requêtes de changement, dépendent de la manière dont la relation entre le schéma global et les schémas locaux a été définie (comme GAV ou LAV). Dans l'approche GAV, les requêtes sont reformulées simplement par le remplacement des éléments changeant [Garcia et al 2002]. Dans l'approche LAV, la reformulation des requêtes est beaucoup plus compliquée, et est connue comme étant la réponse aux requêtes formulées par l'utilisateur de vues [Levy et al 2000]. Il s'agit d'un problème NP-complet, et il ne peut être résolu facilement.

Lorsqu'une requête est reformulée, son optimisation a lieu, parce que les sources de données peuvent avoir des capacités limitées pour satisfaire les requêtes.
Certaines sources de données sont en mesure de s'acquitter de requêtes SQL, d'autres sources peuvent être assez limitées dans leur capacité à satisfaire les requêtes. La vitesse du réseau de transmission de données peut changer, l'optimiseur peut ne pas reconnaitre les statistiques de source de données qui sont généralement nécessaires dans l'optimisation de requêtes.

Compte tenu de ces problèmes, il est difficile d'utiliser les mêmes algorithmes d'optimisation de requêtes dans les applications d'intégration de données. C'est pourquoi les spécialistes ont proposé divers algorithmes adaptatifs [Ives et al 2002], où l'exécution et l'optimisation d'une requête peuvent se produire en parallèle.

Une fois l'optimisation terminée et un plan d'exécution préparé, la requête est exécutée. Les requêtes sont soumises aux adaptateurs[2] (wrappers) qui les reformule dans la syntaxe de la source de données et pose ainsi une requête à la source de données concrète. Les adaptateurs sont généralement assez compliqués et ils sont souvent basés sur les méthodes d'intelligence artificielle (comme l'apprentissage par ordinateur) [Ashish et al 1997]. Par la suite, les résultats sont reformulés dans la syntaxe et la sémantique du système middleware et sont intégrés.

2 Nous utiliserons dans cette thèse aussi bien 'Adaptateur' que 'Wrapper' pour signifier la même chose.

Quand une requête est formulée au médiateur, elle est posée indépendamment de la localisation des différentes données intervenant pour calculer le résultat. Cela introduit les difficultés suivantes :

- 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, de les ordonner et éventuellement d'introduire des opérateurs au niveau du composant de médiation afin de compléter cet ensemble de sous-requêtes. 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. Les résultats de chacune des sous-requêtes peuvent éventuellement faire l'objet d'un traitement additionnel soit parce que l'évaluateur de sous-requêtes n'a pas la capacité nécessaire pour traiter entièrement cette dernière, soit parce que les sous-requêtes comportent des dépendances entre elles.

- Au niveau de l'optimisation : le médiateur a rarement une vision sur la façon dont sont traitées les sous-requêtes au niveau des sources (placement des données, type de stockage, indexation, stratégie d'évaluation). De plus la distribution des données sur des sources disjointes ne permet pas d'utiliser directement les algorithmes employés normalement dans le cas d'un SGBD centralisé.

5. Classification des systèmes d'intégration

Pour faciliter la compréhension des caractéristiques, des avantages et des faiblesses des systèmes existants, nous citons deux critères de classification parmi ceux cités dans la littérature qui sont en relation avec notre travail, à savoir : (1) la représentation de données intégrées : virtuelle ou matérialisée, (2) le sens de la mise en correspondance entre schéma global et les schémas locaux et (3) la nature du processus d'intégration. Cette classification va nous permettre de mieux positionner notre travail par rapport à l'existant.

Comme cela a été mentionné plus haut, il existe essentiellement deux principales approches pour intégrer des données provenant de différentes sources: les systèmes de médiation et les entrepôts de données. Les systèmes de médiation permettent d'accéder par une même requête à des données se trouvant dans des sources différentes. Les entrepôts de données intègrent au sein d'une même base des données provenant de sources différentes, permettant ainsi d'effectuer des analyses et des fouilles de données sur la nouvelle base. Ceci constitue le premier point de classification :

5.1 Degré de matérialisation

La conception de nouvelles applications de traitement de données se fait plus dans un contexte où la plupart des données nécessaires sont déjà stockées dans des bases de données ou dans des fichiers construits de façon autonome pour les besoins des applications existantes. Pour faciliter leur réutilisation, les données à réutiliser peuvent être redéfinies sous forme d'une base de

données virtuelle (système de médiation) ou matérialisée (entrepôt de données), assurant l'intégration logique des données sous-jacentes.
Ce critère permet donc d'identifier le type de stockage de données du système d'intégration. Les deux prochains chapitres fournissent plus de détails respectivement sur les systèmes de médiation et les entrepôts de données.

5.1.1 Architecture virtuelle

Cette architecture consiste à développer une application chargée de jouer le rôle d'interface entre les sources de données locales et les applications d'utilisateurs. Ce type d'approche a été utilisé par un nombre important de projets [Wiederhold 1992], [Chawathe et al 1994], [Mena et al 1996], [Visser et al 1999].
Il repose sur deux composants essentiels : le médiateur et l'adaptateur.

- Le médiateur : chargé de la localisation des sources de données et des données pertinentes par rapport à une requête, il résoud de manière transparente les conflits de données. Un ensemble de connaissances sur les sources permet au médiateur de générer un plan d'exécution pour traiter les requêtes d'utilisateurs.

- L'adaptateur : c'est un outil permettant à un (ou plusieurs) médiateur(s) d'accéder au contenu des sources d'informations dans un langage uniforme. Il fait le lien entre la représentation locale des informations et leur représentation dans le modèle de médiation.

Dans cette approche, les données ne sont pas stockées au niveau du médiateur. Elles restent dans les sources de données et ne sont accessibles qu'à ce niveau. L'intégration d'information est fondée sur l'exploitation de vues abstraites décrivant de façon homogène et uniforme le contenu des sources d'information dans les termes du médiateur. Les sources d'information pertinentes, pour répondre à une requête, sont calculées en fonction de la définition de ces vues. Le problème consiste à trouver une requête qui est équivalente à la requête de l'utilisateur. Les réponses à la requête posée sont ensuite obtenues en évaluant cette requête sur les extensions des vues. Le problème d'hétérogénéité sémantique entre les différentes sources d'information ne se pose pas pour l'utilisateur puisqu'il est résolu par le médiateur. L'approche médiateur présente l'intérêt de pouvoir construire un système d'interrogation de sources de données sans toucher aux données qui restent stockées dans leur source d'origine. Par contre le médiateur ne peut pas évaluer directement les requêtes qui lui sont posées car il ne contient pas de données, ces dernières étant stockées de façon distribuée dans des sources indépendantes. Dans cette approche, deux problèmes apparaissent : comment construire le schéma intégré et comment effectuer la mise en correspondance entre les termes utilisés dans ce schéma et les structures des schémas des sources qui contiennent les données.

Un système de médiation est donc constitué d'un système global de gestion le médiateur, d'un ensemble d'adaptateurs et d'un ensemble de sources locales. Les sources locales sont intégrées dans le sens qu'un seul schéma global est construit à partir des schémas exportés par les sources locales. Des requêtes sont ainsi posées sur le schéma global et traitées par le médiateur. Les schémas des sources sont construits par des adaptateurs qui établissent la correspondance entre la source et le médiateur.

La plupart des systèmes de gestion de données hétérogènes et distribuées adoptent l'architecture à base de médiateurs et d'adaptateurs telle que définie par [Wiederhold 92].

5.1.2 Architecture matérialisée

Cette architecture consiste à centraliser physiquement, dans un entrepôt de données, l'ensemble de données consolidées à partir de diverses sources (catalogues électroniques, bases de données relationnelles, Web, etc.).

Le processus de construction d'un système d'intégration matérialisé se compose en quatre étapes principales [Hacid et al 2005] :

- l'extraction des données des sources de données,
- la transformation des données au niveau structurel,
- l'intégration sémantique des données,
- le stockage des données intégrées dans le système cible.

Ces étapes correspondent aux outils d'ETL (extract, transform and load) [Vassiladis 2005]. Les deux premières étapes sont habituellement prises en charge par le même composant logiciel, appelé adaptateur. Comme dans l'approche médiateur, l'objectif d'un adaptateur est d'aboutir à des données représentées dans un même format. Chaque adaptateur fournit ainsi une interface d'accès et de requêtes sur la source. Les données extraites sont ensuite intégrées en éliminant les conflits, puis stockées dans l'entrepôt.

L'avantage principal d'une telle approche est que l'interrogation d'un entrepôt de données se fait directement sur les données de l'entrepôt et non sur les sources originales. Nous pouvons donc utiliser les techniques d'interrogation et d'optimisation des bases de données traditionnelles. Par contre cette approche exige un coût de stockage supplémentaire et surtout un coût de maintenance causé par les opérations de mises à jour au niveau de sources de données (toute modification dans les sources locales doit être répercutée sur l'entrepôt de données) [Chawathe 1994].

Dans les entrepôts de données, les données sont extraites à partir de sources de données disparates et matérialisées dans l'entrepôt de données. Le principe de la matérialisation est l'intégration réelle de données conformément au schéma global.

5.2 Approches d'intégration de données

Un autre critère de classification est le mapping dans l'intégration de données qui sert à définir la relation entre le modèle global et celui des sources [Alexe et al 2008, Lawrence 2002, Idrissi 2008]. Pour ce faire, plusieurs approches et systèmes d'intégration ont été proposés dans la littérature.

L'intégration de données nécessite que chaque schéma local de chaque source de données subisse une transformation (mapping) en termes du schéma global de l'interface commune [Rousset

2002], [Lenzerini 2002], [Carey 1995], [Boglan 2008]. De ce fait, [Doan 2000] et bien d'autres ont classifié les systèmes d'intégration de données suivant la relation entre les schémas des sources locales et le schéma global du médiateur c-à-dire le mapping. Nous distinguons principalement deux types de systèmes : un schéma global défini comme vue sur les schémas locaux GAV (Global As View) ou des vues locales définies comme vues du schéma global LAV (Local As View) [Carey 1995, Calvanese 2004].

5.2.1 Approche GAV

Dans l'approche GAV (voir la figure 1.3), le mapping fait correspondre à chaque concept **Concept$_G$** du schéma global G une vue **Vue$_L$** sur le schéma local [Mesrobian et al 1996] [Koffina et al 2005]. Le traitement de la requête exprimée en termes du schéma global est direct. Nous procédons à un dépliement de requêtes. Dans cette approche, la transformation d'une requête sur le schéma global en requêtes sur les schémas locaux est une simple opération faite par le gestionnaire de vues. D'un autre côté, dans une architecture GAV, une modification sur les sources locales ou sur leur schéma entraîne une reconsidération du schéma global. De ce fait, dans l'approche GAV le passage à l'échelle est très mal supporté [Halevy 2005, Xu et al 2004].

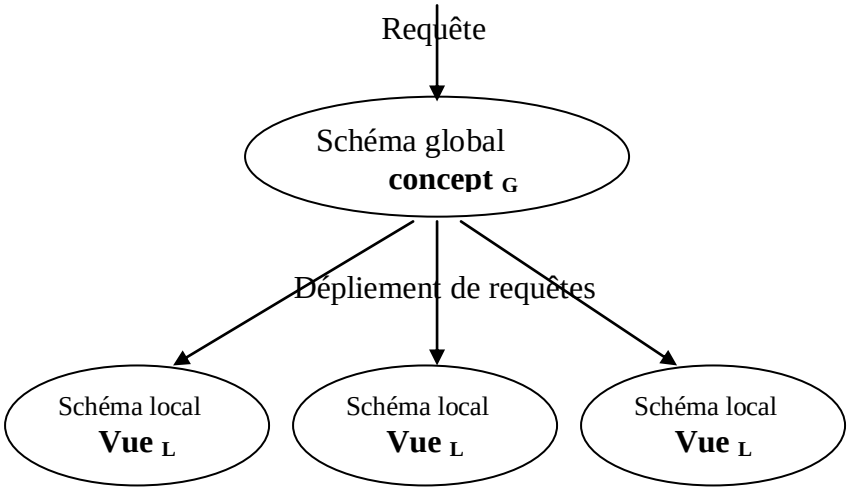


Figure 1.3 : Approche GAV

L'approche GAV a été la première à être proposée pour intégrer des informations. Elle consiste à définir à la main (ou de façon semi-automatique) le schéma global en fonction des schémas des sources de données à intégrer puis à le connecter aux différentes sources.
Pour cela, les prédicats du schéma global, aussi appelés relations globales, sont définis comme des vues sur les prédicats des schémas des sources à intégrer.
Comme les requêtes d'un utilisateur s'expriment en termes des prédicats du schéma global, nous obtenons facilement une requête en termes du schéma des sources de données intégrées, en remplaçant les prédicats du schéma global par leurs définitions.
Parmi les systèmes utilisant GAV, nous pouvons citer HERMES [Subrahmanian et al 1995], TSIMMIS [Chawathe 1994] et MOMIS [Beneventano et al 2000]. En effet, dans TSIMMIS, les

médiateurs sont conceptuellement représentés par des vues définies sur une ou plusieurs sources. Dans le cas d'ajout d'une nouvelle source, TSIMMIS reconstruit les médiateurs.

5.2.2 Approche LAV

L'approche LAV [Mesrobian et al 1996] [Koffina et al 2005] fait correspondre à chaque concept **Concept$_L$** du schéma local une vue **Vue$_G$** sur le schéma global (voir figure 1. 4). Le traitement de la requête exprimée en termes du schéma global est plus complexe car il se fait par une réécriture en termes de vues, autrement dit, la requête sur le schéma global doit être reformulée suivant les schémas des sources locales. De ce fait, LAV a une basse performance quand l'utilisateur pose des requêtes complexes [Pottinger 2000, Halevy 2005]. Dans cette approche, chaque source est spécifiée de manière indépendante. Un changement local de schéma est considéré en mettant à jour la vue locale.
L'approche LAV est l'approche duale de GAV, elle suppose l'existence d'un schéma global et elle consiste à définir les schémas des sources de données à intégrer comme des vues du schéma global.

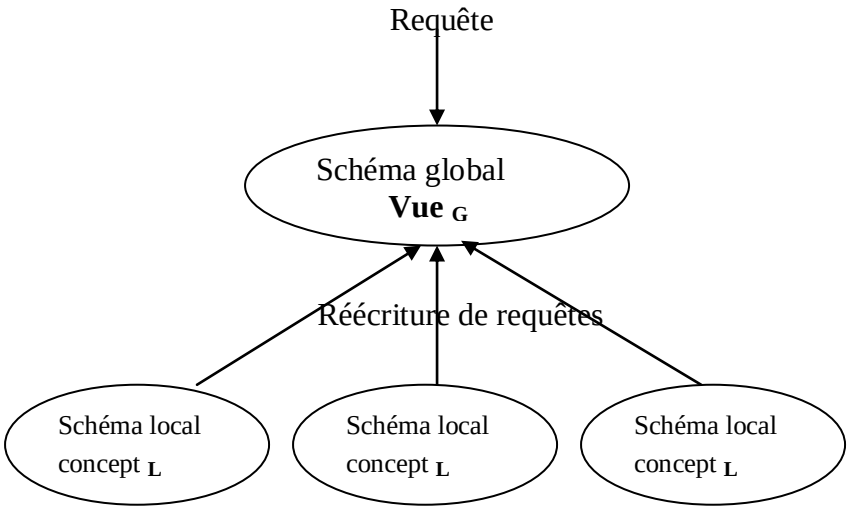


Figure 1.4: Approche LAV

Les principaux systèmes développés autour de cette approche sont : Infomaster [Genesereth et al 1997], PICSEL [Gomez et al 2005], Information Manifold [Levy et al 1996], Razor [Friedman et al 1997], Internet Softbot [Etzioni et al 1994], OBSERVER [Mena et al 1996].

InfoMaster, par exemple, est un entrepôt virtuel de catalogues développé par l'Université de Stanford. Il présente à l'utilisateur un catalogue virtuel, appelé schéma référencé, sur lequel il formule ses requêtes. Ce système utilise le langage KIF (Knowledge Interchange Format 4) pour définir une liste de règles qui associe les schémas des sources intégrées aux vues du schéma référencé. Ces règles permettent aussi de convertir l'information afin de la mettre sous format adéquat et de l'adapter aux sources. Lors de l'ajout d'une nouvelle source, le système d'intégration ajoute de nouvelles règles représentant la vue de la nouvelle source.

5.2.3 Approches d'intégration hybrides

Il y a eu des tentatives de fusionner les approches LAV et GAV de sorte qu'il serait facile de reformuler la requête, comme c'est le cas dans l'approche GAV, ainsi que pour modifier les schémas locaux, comme c'est le cas dans l'approche LAV. Les approches GAV et LAV ont toutes les deux des inconvénients. Ainsi, d'autres s'approchent hybrides ont été proposées [Koffina et al 2005]. En effet, la littérature décrit plusieurs méthodes (Both As View [McBrien et al 2003], GLAV (Global Local As View) [Cali 2003] [Friedman et al 1999] [Kapitskaia et al 1997]. and BGLAV (Both Global Local As View) [Xu et al 2004].

- **Approche GLAV**

Dans GLAV (Generalized-Local-as-View) [Friedman et al, 1999] figure 1.5, nous disposons de vues au niveau global et local. Une correspondance entre les vues au niveau global et local est requise. Ces correspondances n'ont pas de direction et s'appliquent dans les deux sens puisque les concepts dans l'ontologie globale sont considérés comme des vues. En fait, l'approche hybride permet d'avoir la correspondance entre les ontologies locales via le vocabulaire partagé. Donc la correspondance peut être calculée via l'ontologie globale.
Dans l'approche GLAV, l'indépendance du schéma global (permet l'ajout et la suppression de sources), l'adaptation de nouvelles sources et la complexité pour reformuler des requêtes sont les mêmes que dans l'approche LAV [Xu et al 2004].

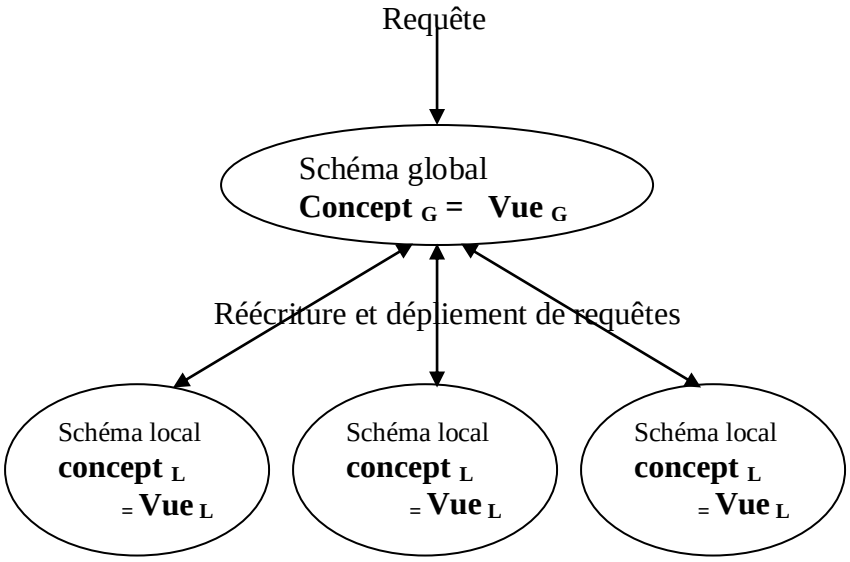


Figure 1.5: Approche GLAV

- **Approche BAV**

Both As View (BAV) est une autre approche d'intégration de données. C'est un cadre riche d'intégration, qui est basé sur l'utilisation des séquences réversibles des transformations

primitives de schéma, appelé 'transformation pathways'. BAV est discutée plus en détail dans [Boyd et al 2004]].

- **Approche BGLAV**

Both Global Local As View (BGLAV) est une autre alternative de GAV et de LAV. L'approche emploie des mapping 'sources-cible' basés sur un schéma conceptuel cible prédéfini, qui est spécifié ontologiquement et indépendamment des sources. Il est plus facile de maintenir le système proposé d'intégration de données basées sur BGLAV que dans GAV et LAV, et la reformulation de requêtes est réduite au déploiement de règles. BGLAV est discutée plus en détail dans [Xu et al 2004].

5.3 Nature du processus d'intégration

Ce dernier critère permet de distinguer les approches d'intégration selon le degré d'automaticité du processus d'intégration. Notons que cette automaticité concerne la résolution ou l'élimination des conflits sémantiques entre les sources durant le processus d'intégration. On distingue les approches *manuelles*, *semi-automatiques* et *automatiques*.

5.3.1 Les approches manuelles

Les premières approches proposées étaient des approches *manuelles*. Ces approches permettent d'automatiser l'intégration des données au niveau syntaxique. Les conflits sémantiques sont gérés manuellement et nécessitent la présence d'un expert humain pour interpréter la sémantique des données. Plusieurs systèmes ont été développés selon cette approche comme les systèmes multi-bases de données, la fédération des bases de données, le système Tsimmis [Chawathe et al 1994] …. Ces approches manuelles deviennent impraticables lorsque le nombre de sources de données à intégrer est important, ou lorsque les sources évoluent fréquemment.

5.3.2 Approches semi-automatiques

Afin d'apporter plus d'automatisation dans la résolution des conflits sémantiques, plusieurs travaux se sont tournés vers les ontologies. Une ontologie permet de fournir la sémantique des concepts d'un domaine de manière formelle.
Les approches *semi-automatiques* reposent sur des ontologies **linguistiques**[1] et permettent d'automatiser partiellement la gestion des conflits sémantiques. Les ontologies linguistiques traitent des termes, et non des concepts. Ceci peut créer des conflits de noms.
Momis **[**Beneventano et al 2000] est un exemple de projet reposant sur l'ontologie linguistique Wordnet[2] pour l'intégration des sources.

5.3.3 Approches automatiques

Les approches *automatiques* consistent à associer aux données des sources une ontologie **conceptuelle** qui en définit le sens. Une ontologie conceptuelle traite les concepts d'un domaine donné. Elle est définie comme « *une spécification formelle et explicite d'une conceptualisation partagée d'un domaine de connaissance* » **[**Gruber 1995]. La sémantique du domaine est ainsi

spécifiée formellement à travers des concepts, leurs propriétés ainsi que les relations entre les concepts. La référence à une ontologie permet d'éliminer *automatiquement* les conflits sémantiques entre les sources. La connexion entre les sources et les ontologies peut se faire de plusieurs manières [Wache et al 2001]: définition des termes de la BD par les concepts ontologiques (Projet Buster **[**Stuckenschmidt et al 2000]), annotation des données (Projet SHOE [Heflin et al 1999]**)**, enrichissement de la source par des règles logiques (logique de description) (Projet Picsel [Goasdoué et al 1999])…

5.4 Niveaux d'abstraction

Si plusieurs systèmes sont en cours d'intégration, cela peut être fait à un des nombreux niveaux. A chacun des niveaux, des méthodes d'intégration différentes peuvent être identifiées:

- Au niveau des vues utilisateurs: L'idée et le but derrière les méthodes qui sont utilisées pour intégrer les vues utilisateur est que les vues des différents utilisateurs du système doivent être intégrées dans un schéma de base de données commun.

- Au niveau des schémas conceptuels: les méthodes qui sont fondées sur des schémas conceptuels décrivent l'intégration des schémas de divers systèmes. Dans ce cas, le processus d'intégration doit faire face à l'hétérogénéité structurelle et sémantique;

- Au niveau des données: Le troisième type de méthodes opère généralement au niveau des données, ici, les méthodes sont basées sur des données concrètes de bases de données aux fins de l'intégration. Les méthodes d'intégration de données doivent faire face à deux principaux problèmes : l'identification d'entités identiques et la résolution des conflits entre les valeurs d'attribut.

6. Conclusion

Dans ce chapitre nous avons présenté une étude sur les systèmes d'intégration de données hétérogènes dans un contexte distribué.
Parmi les problèmes relatifs à ce contexte tels que les schémas dissemblables, le découpage d'une requête en sous-requêtes calculables par les sources sous-jacentes, optimisation de requêtes distribuées différentes d'une optimisation de requêtes centralisées, celui de l'intégration de données hétérogènes dans les systèmes d'intégration tels que les entrepôts de données et les systèmes de médiation pose toujours des défis surtout avec l'avènement de l'Internet.

Chapitre 2 : Les systèmes de médiation

1. Introduction

L'un des moyens le plus naturel de voir un système d'intégration de données, consiste en une couche de médiation uniforme placée entre l'utilisateur et les données sources. L'utilisateur interagit avec le système par l'interrogation d'un schéma global [Lenzerini 2002], c'est à dire une représentation virtuelle des données stockées sur les sources et le système effectuera la tâche de traitement des sources pour récupérer les informations satisfaisant la requête utilisateur. D'autre part, le système d'intégration de données tiendra une représentation interne des données sur les sources, appelé schéma source, de sorte que les relations subsistent entre les entités du global et le schéma de source: ces relations sont représentées par le mapping. L'interaction entre le système d'intégration de données et les sources est rendue facile par certains modules de logiciel, appelé wrappers, qui cachent les caractéristiques des sources, permettant au système de converser avec, en utilisant son langage.

Un système de médiation est un outil puissant permettant un accès simple aux différentes informations collectées de sources de données pouvant être disparates. Il doit intégrer des données diverses afin de pouvoir offrir à l'utilisateur une vue centralisée et uniforme des données en masquant les caractéristiques spécifiques à leur localisation, méthodes d'accès et formats**.**
Le terme médiateur introduit par [Wiederhold 1992] est défini de manière consensuelle et générale comme un composant logiciel assurant la médiation entre un utilisateur et un ensemble de ressources hétérogènes et distribuées. Une ressource peut être dans notre esprit un fichier ou un ensemble de fichiers de données ou de code programme, une base de données, une métadonnée ou un ensemble de métadonnées, un programme.
Dans un contexte de médiation, les ressources pré-existent, avec leurs données et le but d'un médiateur est de réconcilier certaines de ces données de façon à donner l'illusion à un utilisateur qu'il interroge (ou navigue dans) un système d'informations homogène.

Dans ce chapitre nous présentons les systèmes de médiation, puis nous donnons un bref état de l'art des systèmes d'intégration proches d'un système construit sur notre approche d'intégration.

2. Présentation générale d'un système de médiation

Les systèmes de médiation sont de plus en plus développés et connus. Leurs composants essentiels sont : le schéma global, les mappings du schéma global avec les sources, les fonctions de réécriture de requêtes et les fonctions de composition des résultats. Tous ces composants prennent en compte l'hétérogénéité qui est un des principaux problèmes pour lesquels les systèmes de médiation sont construits.

L'objectif est de donner l'impression d'interroger un système centralisé et homogène alors que les sources interrogées sont réparties, autonomes et hétérogènes.
L'approche médiateur présente l'intérêt de pouvoir construire un système d'interrogation de sources de données sans toucher aux données qui restent stockées dans leurs sources d'origine. Ainsi, le médiateur ne peut pas évaluer directement les requêtes qui lui sont posées car il ne contient pas de données, ces dernières étant stockées de façon distribuée dans des sources indépendantes. L'interrogation effective des sources se fait via des adaptateurs, appelés des wrappers en anglais, qui traduisent les requêtes réécrites en termes de vues dans le langage de requêtes spécifique accepté par chaque source.

L'aide apportée par les systèmes de médiation peut recouvrir différentes formes : découvrir les sources pertinentes étant donnée une requête posée, puis aider à accéder à ces sources pertinentes, évitant à l'utilisateur d'interroger lui-même chacune d'elles selon leurs propres modalités et leur propre vocabulaire, enfin combiner automatiquement les réponses partielles obtenues de plusieurs sources de façon à délivrer une réponse globale. De tels systèmes offrent à l'utilisateur une vue uniforme et centralisée des données distribuées, cette vue pouvant aussi correspondre à une vision plus abstraite, condensée, qualitative des données et donc, plus signifiante pour l'utilisateur. Ces systèmes de médiation sont, par ailleurs, très utiles, en présence de données hétérogènes, car ils donnent l'impression d'utiliser un système homogène. Parmi les différentes grandes catégories d'applications de ces systèmes de médiation, nous pouvons citer les applications de recherche d'information, celles d'aide à la décision en ligne (avec entre autres l'utilisation d'entrepôts de données) et celles, de manière plus générale, de gestion de connaissances au sens large.

A titre d'illustration très simple, supposons qu'un utilisateur pose la requête suivante : quels sont les hôtels 'Sheraton' qui ont des chambres disponibles pour telle période? Et où se trouvent-ils ?
Supposons l'existence de deux sources d'information. La première, Internet Sheraton Data Base, utilise un système de gestion de bases de données relationnel et contient une liste d'hôtels, précisant pour chacun la chaîne de l'hôtel, le classement, le pays et la ville. La seconde, MondeHôtel, qui peut utiliser des fichiers XML, contient, par hôtel, les chambres disponibles, les dates de disponibilité, l'adresse et, pour chaque chambre, le type de la chambre et le prix. La réponse à la requête devra être construite en interrogeant chacune d'elles et en combinant les résultats de l'interrogation de façon à offrir à l'utilisateur une réponse globale.

3. Architecture de médiation

L'architecture de médiation fut conçue en 1992 par Gio Wiederhold dans l'article fondateur «*Mediator in the architecture of future information systems*» [Wiederhold 1992]. Il y développe une nouvelle vision de l'architecture du traitement de l'information en entreprise, il tente de régler la problématique de l'accès et de l'intégration de l'information en introduisant la notion de médiateur.

Les médiateurs doivent en général se retrouver sur des nœuds différents de ceux qui contiennent les données (BD) et des postes de travail où sont utilisées en dernière instance les données (par les utilisateurs). Il existe plusieurs sortes de médiateurs accomplissant des tâches distinctes, voici les principales :

1. Transformation et filtrage des BD.
2. Accès et intégration de plusieurs BD.
3. Traitement des données supportant un haut niveau d'abstraction et de généralisation (traitement intelligent de l'information).
4. Répertoires intelligents des bases d'information comme les catalogues, les aides à l'indexation, les structures de thesaurus (ontologies).
5. Gestion de l'incertitude reliée aux données absentes ou incomplètes et aux données mal comprises.

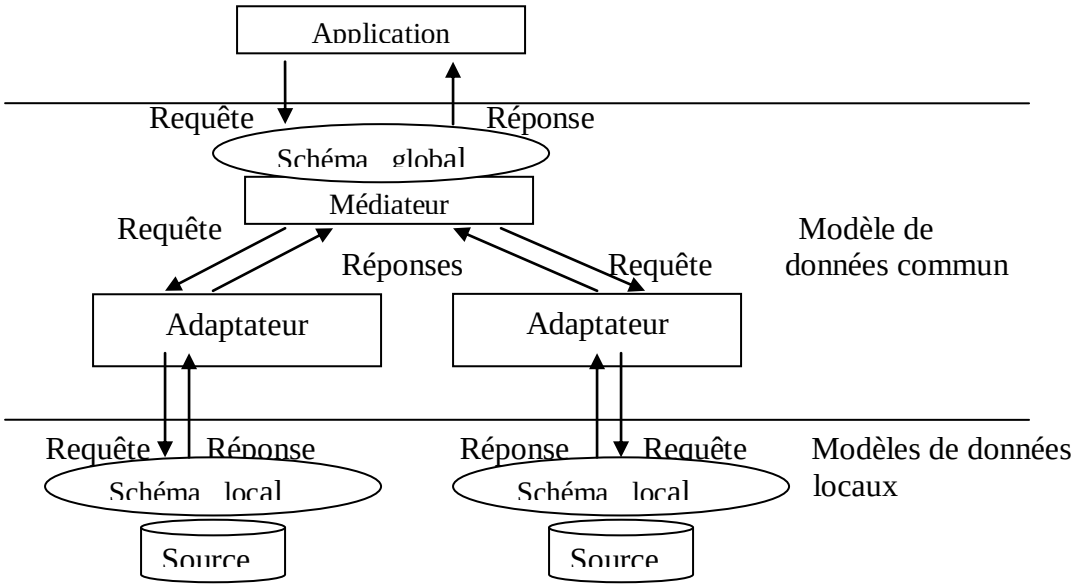


Figure 2.1 : Architecture de médiation

Comme nous pouvons le voir sur la figure 2.1, les systèmes de bases de données hétérogènes actuels adoptent une architecture distribuée qui consiste en plusieurs composants spécialisés [Wiederhold 1992] [Adali et al 1996] [Gardarin et al 1996] [Naacke 1998] [Yemeni et al 1999]. Un médiateur exporte un schéma médiateur ou (schéma global) qui est une représentation intégrée des sources de données ; son rôle est central. Les applications accèdent aux sources de données sous-jacentes à travers les médiateurs. Les médiateurs traitent les requêtes sur la représentation intégrées. Les composants essentiels d'un système de médiation sont :

3.1 Adaptateur (Wrapper):

Un adaptateur prend en compte les dimensions Distribution, Autonomie et Interopérabilité. Son objectif est d'accéder à une source, d'extraire les données appropriées et de les présenter dans un format spécifié. Un adaptateur est donc un composant qui :

• Réalise la transformation entre le modèle de données dans lequel sont représentées les données de la source et le modèle choisi au niveau médiateur.

• Effectue la transformation entre les expressions de requêtes du niveau global en expressions compréhensibles par les sources. Inversement, il traduit les réponses des sources au format du niveau global.
• Assure les traitements spécifiques non disponibles au niveau local et nécessaires pour le niveau global (augmentation du pouvoir d'expression et de traitement de la source locale).

3.2 Médiateur (Mediator)**:**

Le rôle d'un médiateur est de collecter, nettoyer et combiner les données attendues par le système de médiation et produites par les différents adaptateurs [Blakeley et al 1996]. C'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, interroger les différentes sources et intégrer les résultats partiels dans une réponse homogène et cohérente. Le médiateur permet toutes les fonctionnalités relevées par Wiederhold.

4. Problèmes étudiés

Les travaux réalisés dans le cadre des systèmes de médiation, ont porté sur les langages pour modéliser le schéma global, pour représenter les vues sur les sources à intégrer et pour exprimer les requêtes provenant des utilisateurs humains ou d'entités informatiques [Wiederhold 1992].

Des travaux ont porté sur la conception et la mise en œuvre d'algorithmes de réécriture de requêtes en termes de vues sur les sources de données pertinentes, celles-ci pouvant être connectées directement ou indirectement aux sources du serveur interrogé. Le problème à ce niveau peut consister à générer des expressions de calcul permettant de définir tous les objets du niveau global à partir des sources existantes. Le calcul de ces expressions nécessite la connaissance de l'ensemble des sources utiles à sa dérivation.

Certains travaux portent sur la conception d'interfaces intelligentes assistant l'utilisateur dans la formulation de requêtes, l'aidant à affiner une requête en cas d'absence de réponses ou de réponses beaucoup trop nombreuses [Blakeley 1996].

L'idée de médiation entre sources de données utilisant des relations sémantiques locales n'est par ailleurs pas nouvelle. Ce problème a été également étudié dans le cadre des bases de données fédérées, consistant à étudier les mises en correspondance entre relations stockées.

Dans le contexte du Web, toutefois, les techniques de bases de données fédérées ne sont pas réutilisables car le problème est étudié à plus grande échelle et les techniques proposées ne sont pas suffisamment flexibles. Il doit être bien plus facile de faire des ajouts ou des retraits de données et donc des mises en correspondance entre relations. Les systèmes accessibles via le Web sont par ailleurs particuliers dans la mesure où ils peuvent jouer des rôles multiples. Il peut s'agir de sources de données et/ou de systèmes intégrant des services.

Enfin, en raison d'un grand nombre de sources de données, contenant éventuellement des informations redondantes et de qualité variée, il est également important d'adresser le problème d'adaptabilité du système de médiation aux besoins des utilisateurs, notamment en terme de

qualité des données. Les principales questions qui peuvent être posées sont : (1) Comment automatiser la génération des requêtes de médiation ? (2) Comment détecter et résoudre les problèmes liés à l'hétérogénéité ? (3) Comment évaluer la qualité du système de médiation ? (4) Comment donner la possibilité aux utilisateurs d'exprimer leurs préférences ? (5) Comment tenir compte des préférences de l'utilisateur dans la conception du système de médiation? Ces problèmes ne sont pas spécifiques aux médiateurs, ils sont retrouvés également dans la conception des entrepôts de données et aussi dans les architectures peer-to-peer [Kostadinov et al 2005].

5. Intégration dans les systèmes de médiation

Les recherches sur les systèmes d'intégration en général et les systèmes de médiation en particulier ont fourni un ensemble riche d'approches d'intégration sur lesquelles ils peuvent être construits. En plus des approches d'intégration vues dans le chapitre précédent, les deux approches communément utilisées sont Global-As-View (GAV) dans laquelle le schéma médiateur est défini comme un ensemble de vues sur les sources de données et l'approche Local-As-View (LAV) dans laquelle les contenus des sources sont décrits comme des vues sur le schéma médiateur.

6. Evaluation de requêtes

Pour l'exécution d'une requête, une application lance une requête au médiateur et attend la réponse. Le médiateur reçoit la requête l'analyse syntaxiquement et sémantiquement puis la décompose en sous-requêtes. Chaque sous-requête correspond à une source de données et elle utilise seulement les fonctionnalités de la source de données. Le médiateur coordonne l'exécution de la requête : il envoie chaque sous-requête à l'adaptateur qui contrôle la source de données. L'adaptateur la traduit en une requête correspondante dans le langage local de la requête et envoie cette requête traduite à la source de données. La source de données calcule une réponse et la transmet à l'adaptateur qui traduit la réponse dans le format exigé par le médiateur. Le médiateur coordonne les réponses qui arrivent et les combinent en une réponse finale. Pour supporter l'exécution de la requête, le médiateur doit faire face aux différentes fonctionnalités de chaque adaptateur correspondant à chaque source de données.

Le processus de génération de requêtes doit tenir compte de l'hétérogénéité des données sources. Les opérations qui composent une requête de médiation définie sur les sources ne sont valides que si les conflits sémantiques liés aux instances sont détectés et résolus. Par exemple : Une jointure de deux relations sources sur l'attribut prix peut retourner un résultat incorrect quand les prix sont exprimés dans des monnaies différentes (ex. euro et francs). La transformation des euros en francs ou inversement est une solution au problème. •

7. Prototypes

Les recherches dans le domaine des systèmes d'intégration ont été très actives et ont conduit à l'élaboration de prototypes qui, principalement, ont mis l'accent sur les aspects inhérents aux modèles et langages de requêtes, aux algorithmes de traitement de requêtes, à la définition du schéma global et aux problèmes de translation sémantique entre les schémas[Alia et al 2004].

Les premières architectures se sont appuyées sur des sources de données relationnelles souvent centrées sur un SGBD qui joue le rôle d'un médiateur (MULTIBASE, MERMAID, INFORMATION MANIFOLD. Avec l'avènement des systèmes de bases de données objets, il a fallu tenir compte de ce nouveau modèle de données dans les architectures de médiation (DISCO, PEGASUS, IRO-DB). Enfin, sont apparues des sources à base de données semi-structurées, et les nouvelles architectures de médiation ont dû s'adapter, aussi bien dans les projets de recherche (TSIMMIS, MIX, YAT et AGORA) que dans des produits industriels (NIMBLE, XPERANTO). Les architectures les plus souvent utilisées sont des architectures GAV (TSIMMIS, GARLIC, YAT, DISCO, SilkRoute, Xperanto). Très peu d'entre elles utilisent l'approche LAV (Information Manifold, AGORA).

7.1 TSIMMIS

TSIMMIS (The Stanford-IBM Manager of Multiple Information Sources) [Chawathe et al. 1994] [Garcia et al. 1994] est un projet permettant de développer des outils qui facilitent l'intégration de sources d'informations hétérogène comportant des données structurées et semi-structurées.

De ce fait, TSIMMIS est une architecture composée des modules suivants (voir figure 2.2) :

- classifieur/extracteur de données sur le Web : depuis du texte brut, des pages HTML ou des pages XML. Le classifieur/extracteur identifie et classifie de tels objets (ex. est-ce un fichier texte? Une page web HTML ? Un courrier électronique ?), et en extrait des propriétés (ex. date, auteur). Le classifieur /extracteur est basé sur le système RUFUS [Shoens et al.1993] développé au centre de recherche d'IBM Almaden :

-traducteur (ou adaptateur) : permettant la transformation des requêtes et des données. Le traducteur convertit les requêtes spécifiques OEM-QL sur des données du modèle commun OEM en requêtes spécifiques à la source. Il convertit ensuite les données résultats de la source en données du modèle commun OEM ;

-médiateur : permettant de combiner les informations des différentes sources ;
-applications clientes : le médiateur tout comme les traducteurs, prend des requêtes OEM-QL en entrée et renvoie des données résultats OEM. Des applications clientes (MOBIE MOsaic- Based Information Explorer) permettant de naviguer dans les données à travers le web ont été développées afin d'offrir une interface conviviale à l'utilisateur final.

Un des objectifs de TSIMMIS est d'intégrer des sources qui sont très hétérogènes, qui peuvent être peu structurées et qui sont susceptibles d'évoluer rapidement. TSIMMIS a introduit le formalisme OEM [Vassalos et al 1997] comme modèle de données interne et a adopté un langage de requête dérivé d'OQL : OEM-QL. Afin de pouvoir utiliser une source de données purement semi-structurées, un SGBD semi-structuré natif LORE basé sur OEM a aussi été développé.

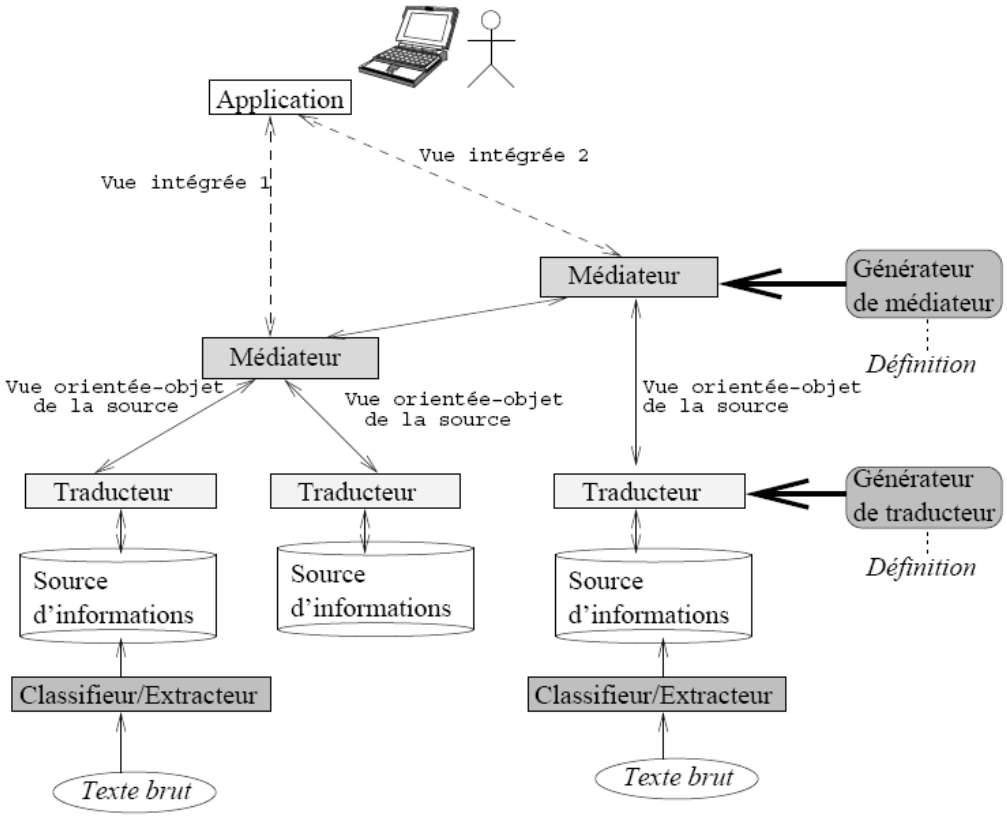


Figure 2.2 : Architecture de TSIMMIS

7.2 GARLIC et MIX

Le projet GARLIC [Haas et al 1997] est un projet semblable développé au centre de recherche d'IBM Almaden. Son objectif est l'intégration de diverses sources multimédia (images, vidéo, texte, objets médicaux, cartes géographiques).

Outre le modèle de données utilisé, la différence fondamentale entre l'approche de GARLIC et celle de TSIMMIS est dans la prise en compte des capacités des sources.
Le médiateur de TSIMMIS assume que les adaptateurs auxquels il adresse les requêtes décomposées savent tout faire [Li et al.1998], et c'est aux adaptateurs de pallier aux défaillances des sources. Le médiateur TSIMMIS ne prend pas en compte les capacités des sources, et donc, la décomposition et l'optimisation des requêtes est plus simple à réaliser au niveau médiateur. Il y a eu une bonne répartition des tâches mais l'implémentation des adaptateurs est rendue plus lourde.

Avec GARLIC, le médiateur prend en considération les capacités des sources aussi bien pour les calculs de coûts que pour les décompositions des requêtes envoyées. Son approche est plus fine et permet de traiter certaines requêtes impossibles à TSIMMIS.

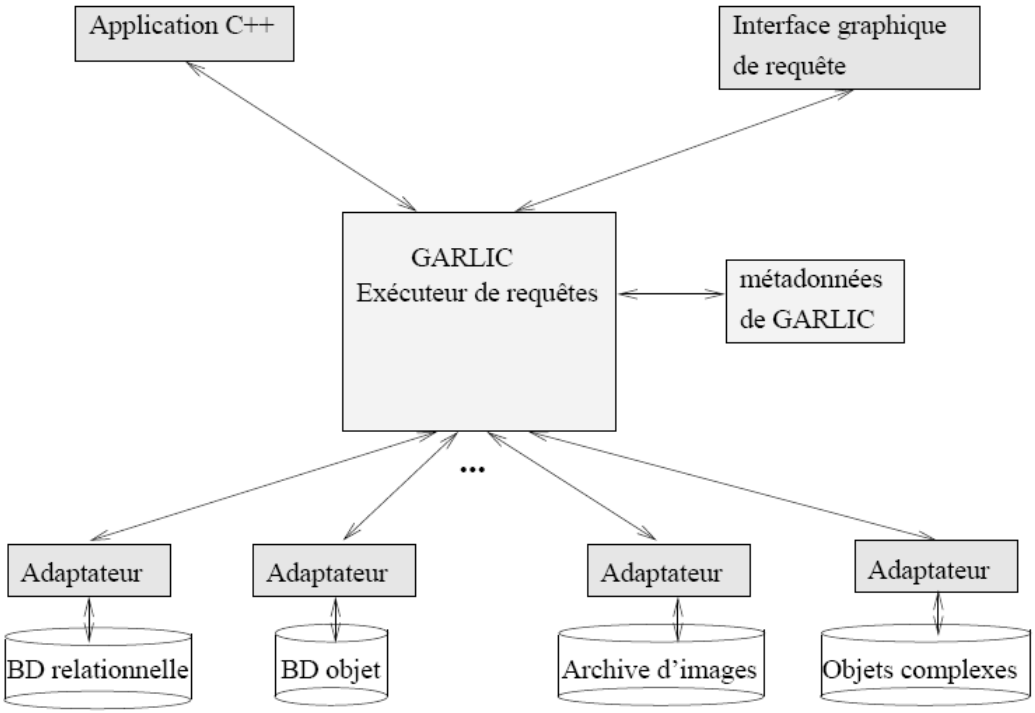


Figure 2.3 : Architecture de GARLIC

Les algorithmes de traitement du médiateur sont en revanche plus complexes et la centralisation entraine un travail supplémentaire pour le médiateur. Un langage de description des capacités des sources est aussi nécessaire. Les pré-requis fonctionnels demandés aux adaptateurs de GARLIC se base sur le plus petit ensemble commun de propriétés. Au niveau des opérations d'exécution, il est demandé à l'adaptateur de savoir au minimum retrouver les objets d'une collection, et accéder aux attributs d'un objet donné.

GARLIC n'est pas un médiateur de données semi-structurées, mais l'intégration de sources multimédias (peu ou pas structurées) avec des sources relationnelles et objet a permis à GARLIC de détailler la définition des adaptateurs et créer un modèle d'architecture pouvant intégrer des sources très hétérogènes.

Il est à noter que TSIMMIS a ensuite adopté l'approche de GARLIC [Vassalos et al.1997] et LORE a fait évoluer son format de données semi-structurées OEM vers XML. Cela a permis la conception de MIX qui utilise XML comme modèle d'échange et XMAS (Xml Matching And Structuring language) comme méta-langage de requête. De la sorte, tous les langages basés semi structuré (YATL, XML-QL, UnQL et MSL) peuvent être convertis en XMAS qui peut être ensuite utilisé par le médiateur. XMAS est le langage de requête d'échange de l'architecture de médiation MIX.

7.3 STRUDEL

STRUDEL [Fernandez et al. 1998] est un constructeur de sites web permettant de définir quelles sont les données qui seront disponibles sur le site. L'idée principale est de séparer

l'administration des données du site, la création et la gestion de la structure du site, et la représentation graphique des pages web finales. Pour cela, le constructeur de site crée un modèle uniforme de toutes les données disponibles sur le site. Puis, le constructeur de site utilise ce modèle pour définir la structure du site web en appliquant aux données, une requête de « définition de site ». Enfin, le constructeur de site spécifie la représentation graphique des pages en utilisant le langage de STRUDEL pour la définition de modèle (template) HTML. Les données résident soit sur les sources externes (SGBD, Fichiers structurés), soit dans l'entrepôt (repository) interne de STRUDEL. A chaque niveau du système STRUDEL, toutes les données (qu'elles soient internes ou externes) sont modélisées par un graphe orienté similaire à OEM. Un ensemble d'adaptateurs spécifiques se charge de convertir la représentation externe en graphe. La vue intégrée des données est appelée graphe de données.

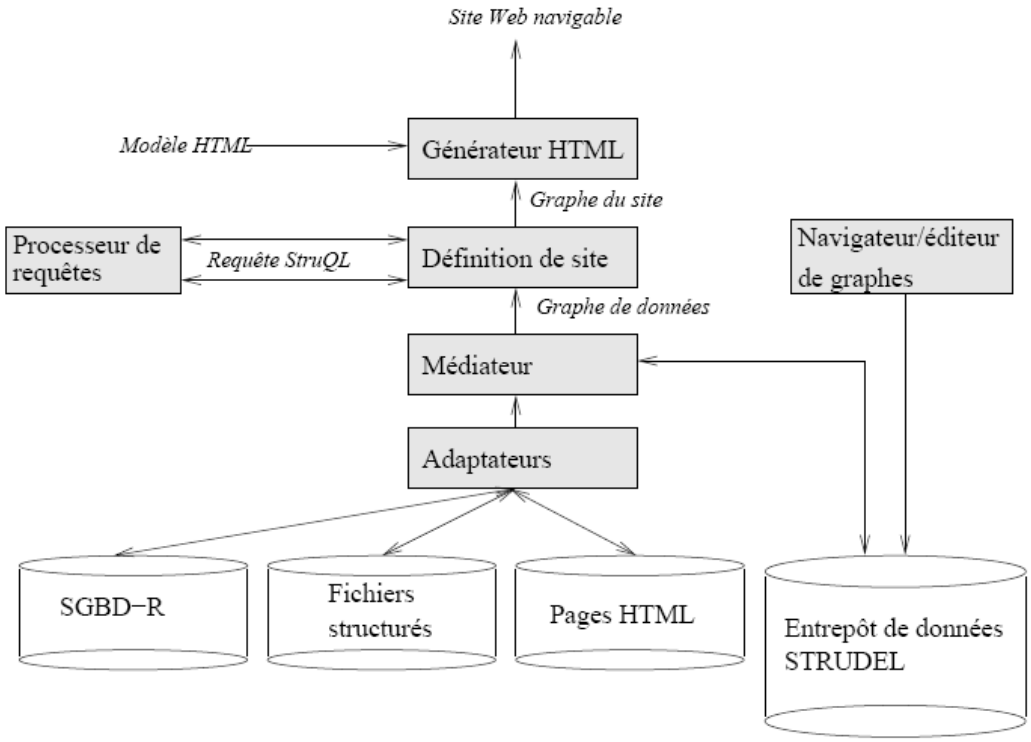


Figure 2.4 : Architecture de STRUDEL

L'architecture de STRUDEL est représentée sur la figure 2.4. Elle est composée de :

- entrepôt de données STRUDEL : le graphe de données d'un site web est stocké dans l'entrepôt de données STRUDEL. Les données sont obtenues des adaptateurs qui convertissent les données des sources externes en des données au format interne semi-structuré utilisé par STRUDEL ;

- navigateur/éditeur de graphe : il permet à l'utilisateur de créer, mettre à jour et visualiser les graphes pouvant être utilisés pour le graphe de données et le graphe du site ;

- médiateur : il fournit une vue uniforme des données sous-jacentes. Plutôt que d'interroger les sources externes à la demande au moment de l'exécution de la requête, son approche est d'intégrer les données en stockant préalablement les données des sources externes dans l'entrepôt de données STRUDEL ;

- processeur de requêtes : STRUDEL définit le langage STRUQL (STRUdel Query Language) pour réaliser des requêtes et restructurer des données semi structurées. L'interpréteur de requête de STRUDEL utilise les opérateurs physiques traditionnels (jointure, restriction, sélection) ainsi que des opérateurs nécessaires pour l'interrogation de schéma (ex. trouver tous les noms d'attributs dans un graphe) ;

- générateur HTML : pour produire la représentation graphique de chaque page du site web, un modèle HTML est associé à chaque nœud du graphe du site. Le résultat est un site web navigable.

7.4 YAT

YAT [Cluet et al 1998] intègre des sources de données dans un environnement Web. Il se base sur la représentation de graphes afin de représenter les données semi structurées provenant des différentes sources. Le langage YATL est un langage déclaratif de conversion/intégration à base de règles ; ce n'est pas un langage de requête et il ne fournit pas de possibilités de requête comme des expressions de chemins généralisées. Cependant, il permet des primitives de restructuration et des fonctions de Skolem pour manipuler des identifiants et créer des graphes complexes.

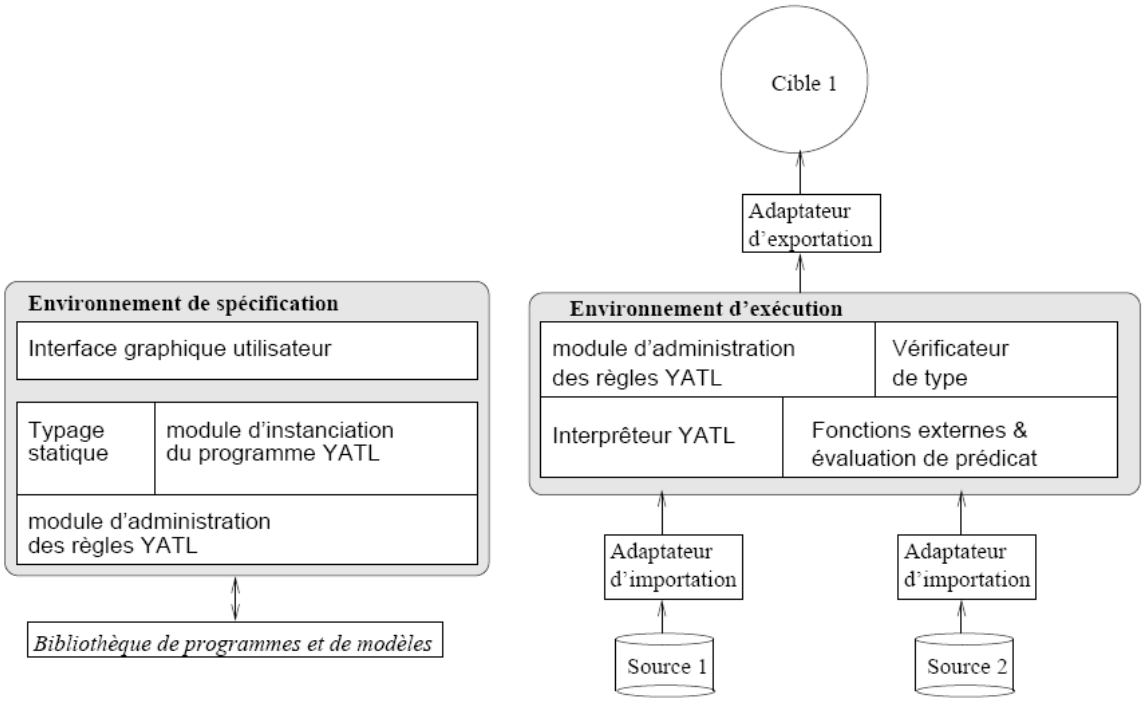


Figure 2.5 : Architecture de YAT

La figure 2.5 décrit l'architecture de YAT qui repose sur trois parties principales :

- l'environnement de spécification,
- l'environnement d'exécution,
- une bibliothèque de programmes et format.

Chacun des environnements repose sur le module de modèle YAT et d'administration de règles YATL. L'environnement de spécification offre (i) un module permettant de vérifier ou déduire le type d'un programme, (ii) une interface graphique permettant à l'utilisateur de définir les traductions en utilisant (iii) le module d'instanciation pour personnaliser les programmes existants si besoin est.

L'environnement d'exécution utilise des adaptateurs pour importer (resp. exporter) des données depuis (resp. vers) YAT et un interpréteur pour réaliser la traduction.
L'interpréteur se base sur un module séparé pour traiter des fonctions externes et des prédicats. Si cela est requis, il vérifie les types à l'aide du vérificateur de type.

7.5 AGORA / LeSelect

AGORA [Manolescu et al.2001] est le seul projet avec Information Manifold [Levy et al. 1996] à utiliser une approche LAV. A la différence de ce dernier qui utilise uniquement des données relationnelles, AGORA manipule aussi des données semi-structurées. Pour cela, il s'appuie sur le médiateur LeSelect qui est un médiateur dont le modèle de donnée est de type modèle relationnel étendu et dont le langage d'interrogation commun est de type SQL. La motivation de AGORA est de compléter les fonctionnalités de LeSelect en :

-définissant un schéma virtuel, générique et relationnel décrivant le contenu de documents XML ;

-traduisant des requêtes spécifiques à XML dans le schéma d'intégration relationnel ;

-étendant l'optimisation de requêtes afin de pouvoir gérer le cheminement ;

-utilisant l'indexation textuelle pour améliorer les performances des recherches plein texte.

L'architecture d'AGORA est représentée sur la figure 2.6.

L'objectif AGORA est de supporter les requêtes et l'intégration de sources relationnelles et semi-structurées. Le schéma global d'AGORA est une DTD XML. Les sources relationnelles et XML sont décrits comme vues sur ce schéma global. Un schéma relationnel générique est utilisé comme interface entre ce schéma et les sources. Pour chaque type de nœud XML (élément, attributs, textes, commentaires), une table relationnelle est associée dans ce schéma.

De cette façon, les requêtes XQuery sont transformées en requêtes relationnelles et les résultats sous formes de tuples son ensuite retransformés en XML.

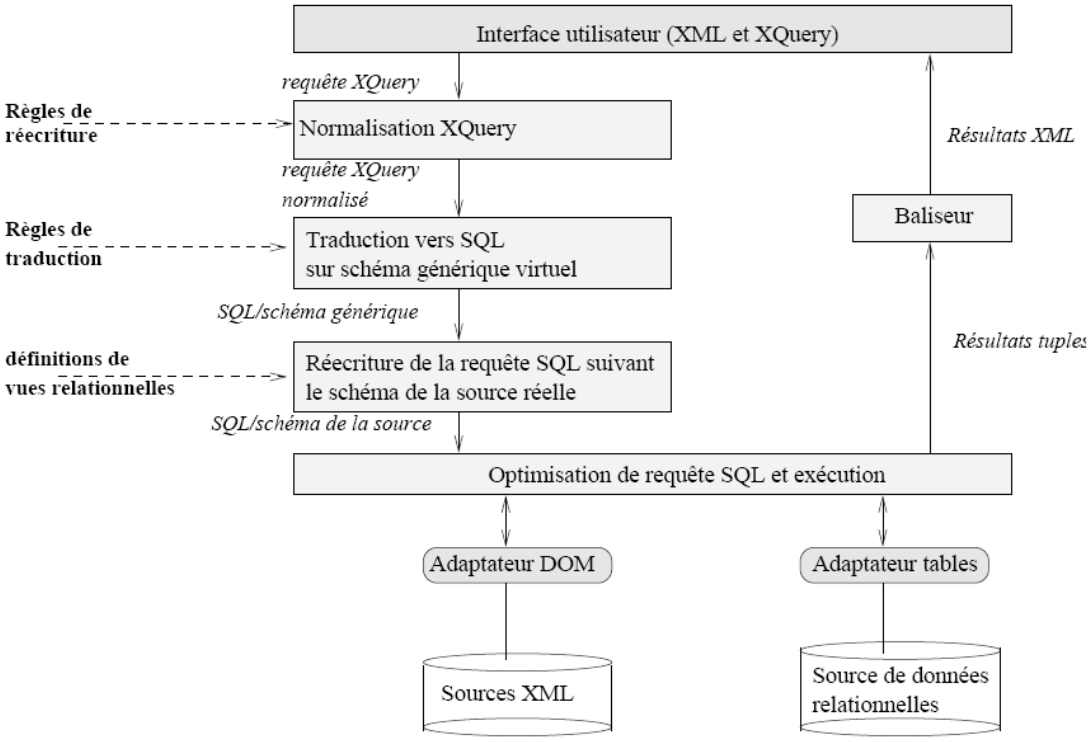


Figure 2.6 : Architecture d'AGORA

8. Conclusion

Les systèmes de médiation constituent une réponse architecturale pour un accès transparent à des sources de données distribuées. Cependant, leur mise en œuvre pose un certain nombre de problèmes, tant sur l'intégration des données de structures et de natures fondamentalement différentes que sur la décomposition d'une requête faisant intervenir plusieurs sources en des requêtes spécifiques à ces sources, que sur la recomposition du résultat. La principale difficulté étant la définition d'un schéma local ou global. Ces problèmes sont d'autant plus cruciaux lorsque les sources sont nombreuses et hétérogènes. Nous verrons dans le chapitre suivant que ces mêmes problèmes apparaissent dans les entrepôts de données.

Chapitre 3 : Les entrepôts de données (Datawarehouses)

1. Introduction

Les entreprises utilisent actuellement divers systèmes sources pour gérer leur activité et leur processus. Les systèmes décisionnels permettent d'avoir la meilleure vision possible de leur activité, cela permet aux dirigeants d'avoir les données adéquates pour prendre les bonnes décisions. Ces systèmes décisionnels puisent leurs informations au sein de l'entrepôt. Il est donc nécessaire d'alimenter ces entrepôts de données avec deux types de données : les structurées issues des progiciels et les non structurées issues d'Internet : c'est la phase d'intégration des données à destination de l'entrepôt de données.

L'approche des entrepôts de données est aujourd'hui unanimement reconnue comme étant une solution adaptée et performante, permettant d'améliorer la prise de décision dans les entreprises [Widom et al 1995] [Inmon et al 1996] [Chaudhuri et al 1997] [Gatziu et al 1999] [Jarke et al 1999].

L'objectif de ce chapitre est de dresser un état de l'art sur l'intégration de sources de données distribuées et hétérogènes dans les entrepôts de données.

2. Principe général

Un entrepôt de données répond aux problèmes de données surabondantes et localisées sur de multiples systèmes hétérogènes, c'est une architecture capable de servir de fondation aux applications décisionnelles. Pour être exploitables, toutes les données provenant des systèmes distribués doivent être organisées, coordonnées, *intégrées* et enfin stockées pour donner à l'utilisateur une vue globale des informations.

Il offre aux utilisateurs (décideurs) un accès rapide aux données et informations essentielles afin d'optimiser la prise de décisions. Un entrepôt peut être assimilé à un ensemble de vues matérialisées qui suppose une certaine anticipation des besoins des utilisateurs. Les requêtes sont traitées non pas au niveau des sources d'information mais au niveau de l'entrepôt de données.
La conception d'un entrepôt de données pose plusieurs problèmes: d'abord la localisation des sources d'informations pertinentes, ensuite l'intégration des données qui demandent la connaissance des systèmes sources pour résoudre les conflits, et enfin l'extensibilité vers de nouvelles sources.

2.1 Définition d'un entrepôt de données

De nombreuses définitions ont été proposées, soit académiques, soit par des éditeurs d'outils, de bases de données ou par des constructeurs, cherchant à orienter ces définitions dans un sens mettant en valeur leur produit.

La définition la plus appropriée est :

Définition :

[INMON et al 1994] définit un entrepôt de données (datawarehouse) comme une collection de données et d'informations intégrées, orientées sujet, non volatiles et historisées.

Cette collection est destinée à être utilisée dans le processus d'aide à la décision. Les utilisateurs interrogent les données à des fins d'analyse en se basant sur des données historisées, agrégées ou résumées [INMON et al 1994] [Doucet et al 2001]. Ces données peuvent provenir de différentes sources et sont regroupées dans une base unique conçue pour des analystes et des décideurs.

2.2 Approche entrepôt de données

La démarche consiste à voir cette intégration comme la construction de bases de données réelles, appelées entrepôt de données, regroupant les informations pertinentes pour les applications considérées [Gardarin 2001]. L'utilisateur pose alors ses requêtes ou lance un traitement directement sur les données stockées dans l'entrepôt. Les problèmes posés par sa construction à partir de plusieurs bases de données spécialisées concernent, la définition de son schéma, son peuplement et sa mise à jour, en fonction des différentes sources d'information à partir desquelles il est construit. La figure 3.1 illustre l'architecture générale d'un tel système.

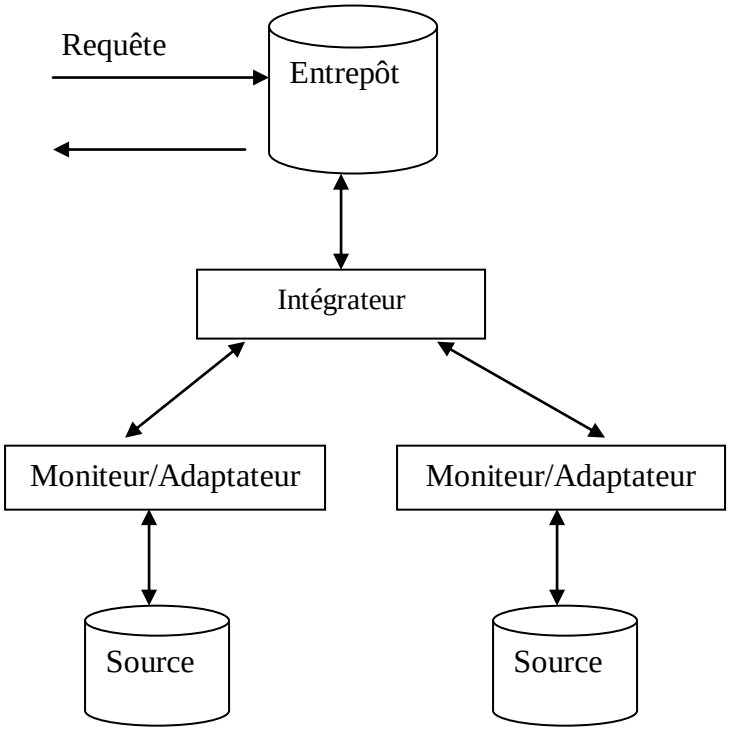


Figure 3.1: Architecture d'un entrepôt de données

Les composants de cette architecture sont:

● Les sources

Les données de l'entrepôt sont extraites de diverses sources souvent réparties et hétérogènes, et doivent être transformées avant leur stockage dans l'entrepôt. Nous avons deux types de sources des données : internes et externes à l'organisation.

Internes : La plupart des données sont saisies à partir des différents systèmes de production qui rassemblent les divers SGBD opérationnels, ainsi que des anciens systèmes de production qui contiennent des données encore exploitées par l'entreprise.

Externes : Ils représentent des données externes à l'entreprise et qui sont souvent achetées. Par exemple, les sources de données démographiques.

● Le moniteur de source

C'est un composant capable de détecter les mises à jours effectuées sur la source et de repérer les donnés à envoyer à l'entrepôt de données.

● L'adaptateur de source

Afin de préparer l'intégration des données dans l'entrepôt de données, il est nécessaire de convertir les données à envoyer dans un format plus ou moins commun. Pour ce faire, le moniteur doit être complété par un adaptateur capable de transformer les mises à jour, les questions et les réponses associées dans le modèle de l'entrepôt.

● L'intégrateur

Avant d'être déversées dans l'entrepôt, les données en provenance de sources multiples doivent être intégrées. Cette fonction est dévolue à l'intégrateur.

● L'entrepôt de données

Le support de destination des données est l'entrepôt de données. Il s'agit d'une base de données dont la structure dimensionnelle permet de faciliter le stockage et la disposition des informations afin de les analyser ultérieurement. Il existe plusieurs types de données dans un entrepôt, qui correspondent à diverses utilisations, comme :

- Données de détail courantes : Ce sont l'ensemble des données quotidiennes et plus couramment utilisées. Ces données sont généralement stockées sur le disque pour avoir un accès rapide. Par exemple, le détail des ventes de l'année en cours, dans les différents magasins.

- Données de détail anciennes : Ce sont des données quotidiennes concernant des événements passés, comme par exemple le détail des ventes des deux dernières années. Nous les utilisons pour arriver à l'analyse des tendances ou des requêtes prévisionnelles.

Néanmoins ces données sont plus rarement utilisées que les précédentes, et elles sont souvent stockées sur des mémoires d'archives.

- Données résumées ou agrégées : Ce sont des données moins détaillées que les deux premières et elles permettent de réduire le volume des données à stocker. Le type de données, en fonction de leur niveau de détail, permet de les classifier comme des données légèrement ou fortement résumées. Par exemple, les ventes mensuelles par magasin des dix dernières années sont des données faiblement résumées, tandis que les ventes semestrielles, par région, des dix dernières années sont fortement résumées.

- Les métadonnées : Ce sont des données essentielles pour parvenir à une exploitation efficace du contenu d'un entrepôt. Elles représentent des informations nécessaires à l'accès et l'exploitation des données dans l'entrepôt comme : la sémantique (leur signification), l'origine (leur provenance), les règles d'agrégation (leur périmètre), le stockage (leur format, par exemple : francs, euro,...) et finalement l'utilisation (par quels programmes sont-elles utilisées).

3. Processus d'intégration de données dans les entrepôts de données

Avant d'être stockées dans l'entrepôt, les données doivent être intégrées conformément au schéma médiateur pour donner à l'utilisateur une vue globale des informations. Nous abordons dans cette section les différents processus de l'intégration de données dans les entrepôts de données.

3.1 Etapes d'intégration

[Hacid et al 2005] distingue deux niveaux dans la construction des entrepôts de données. Le premier niveau correspond à la construction des sources de données opérationnelles, et de l'entrepôt de données global. Le second niveau englobe tous les entrepôts de données locaux. La raison de cette distinction est, qu'à chaque niveau, sont associées différentes étapes de traitement et différentes difficultés techniques.

Au premier niveau, le processus de construction est décomposé en quatre étapes principales, qui sont : (1) l'extraction des données des sources de données opérationnelles, (2) la transformation des données aux niveaux structurel et sémantique, (3) l'intégration des données, et (4) le stockage des données intégrées dans le système cible. La figure3.2 résume l'enchaînement de ces étapes de traitement.

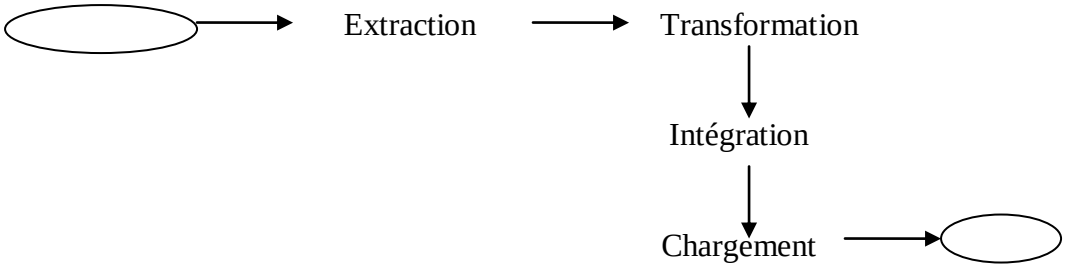


Figure 3.2 : Etapes de traitement du premier niveau de construction d'un entrepôt de données

Notons cependant que cette décomposition est seulement logique. L'étape d'extraction et une partie de l'étape de transformation peuvent être groupées dans le même composant logiciel, tel qu'un « wrapper » ou un outil de migration de données. L'étape d'intégration est souvent couplée avec des possibilités de transformation de données riches dans un même composant logiciel, qui, habituellement, réalise le chargement dans l'entrepôt de données. Toutes les étapes de traitement peuvent aussi être groupées dans un même logiciel, comme par exemple un système multibase.

Quand les étapes d'extraction et d'intégration sont séparées, les données nécessitent d'être stockées entre les deux. Une vue opérationnelle typique de ces composants est donnée par la figure 3.3. Les composants logiciels sont représentés par des rectangles. Les ellipses désignent des stockages intermédiaires des résultats de l'étape d'extraction/transformation. Toutes les données qui sont en entrée du composant intégration utilisent le même modèle de représentation de données. Finalement, un « wrapper » est associé à chaque source, fournissant ainsi une interface API à la source.

Au second niveau, le processus de construction comporte trois étapes distinctes, qui sont :

- l'extraction de données à partir d'une base de données (entrepôt de données local ou global),
- le calcul des données dérivées pour l'entrepôt de données local cible,
- le stockage des résultats dans l'entrepôt de données local.

L'étape d'extraction est un cas particulier de celle du premier niveau car les données de l'entrepôt sont stockées dans une base de données. A l'opposé, dans le premier niveau, l'extraction peut concerner des sources de données arbitraires, des fichiers par exemple.

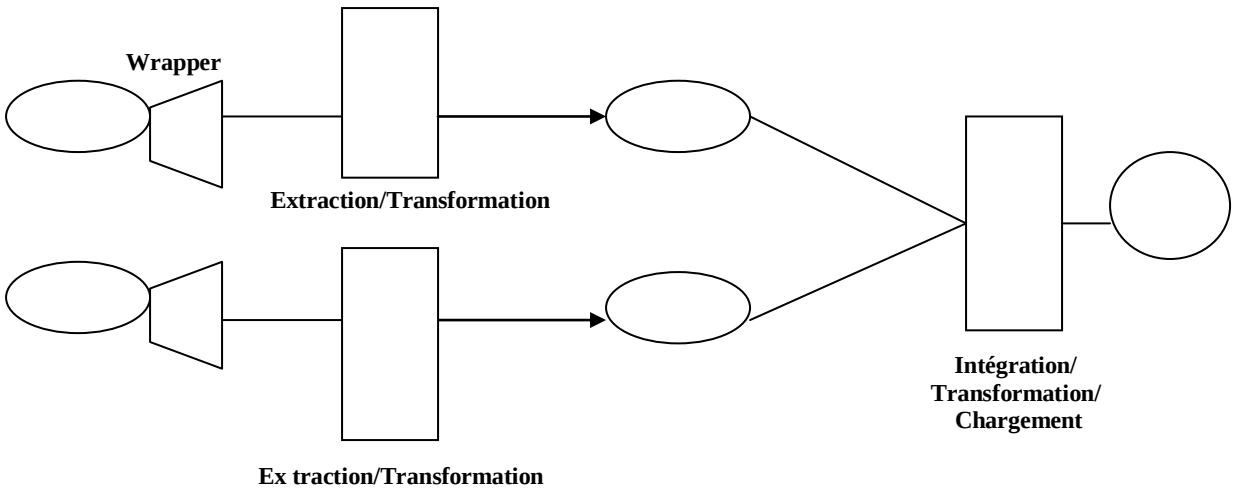


Figure 3.3 : Vue opérationnelle des composants utilisés pour la construction d'entrepôts de données

3.2. Types d'intégration

Le type d'intégration réalisé dans la conception d'un entrepôt de données est celui qui est réalisé dans le domaine de l'intégration d'information, qui a été exploré dans différents domaines comme :

- les bases de données,
- les systèmes d'information coopératifs,
- les systèmes d'information globaux,
- la représentation des connaissances.

Une première classification des différentes approches repose sur le contexte d'intégration, et par conséquent, le type des entrées/sorties du processus d'intégration, et le but du processus lui-même. Nous distinguons l'intégration de schémas, l'intégration de données virtuelle, et l'intégration de données matérialisée.

• Intégration de schémas : Dans ce cas, l'entrée de l'intégration est un ensemble de schémas sources, et la sortie est un schéma de données correspondant à la représentation intensionnelle réconciliée de tous les schémas en entrée. L'entrée comporte également la spécification de la façon d'associer les schémas des données sources à des parties du schéma résultant (cible).

• Intégration de données virtuelle (médiateurs) : L'entrée est un ensemble de données sources, et la sortie est une spécification décrivant la façon de fournir un accès global et unifié aux sources dans le but de satisfaire certains besoins en information, sans interférer avec l'autonomie des sources (figure 3.4).

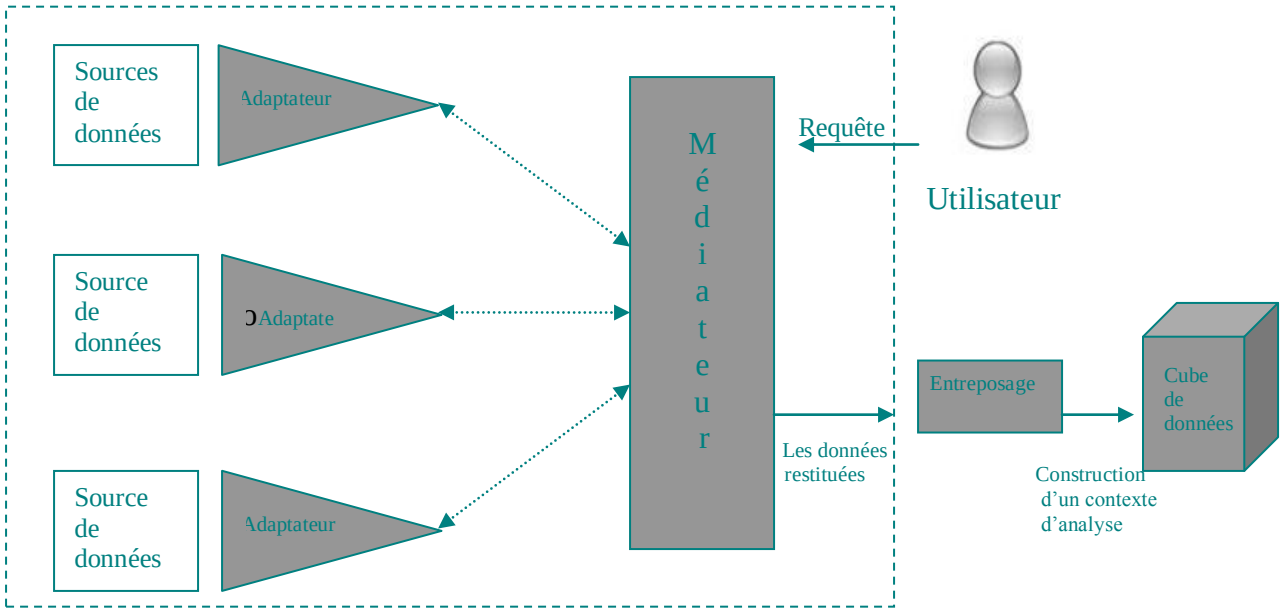


Figure 3.4 : Architecture d'entreposage virtuel

• Intégration de données matérialisée : Comme dans le cas précédent, l'entrée est un ensemble de données sources, mais ici la sortie est un ensemble de données représentant une vue réconciliée des sources, à la fois au niveau intensionnel et au niveau extensionnel.

Notre travail se situe en fait au carrefour de l'intégration de données virtuelle et l'intégration de données matérialisée. En effet, l'approche proposée dans cette thèse est une approche hybride pour l'intégration de données dans les entrepôts de données qui repose sur une intégration virtuelle.

4. Alimentation d'un entrepôt de données

Après avoir défini un entrepôt de données, intéressons nous maintenant à son alimentation depuis les données sources, en mettant en œuvre un ETL.

ETL, acronyme de *Extraction, Transformation, Loading*, est un système de chargement de données depuis les différentes sources d'information de l'entreprise (hétérogènes) jusqu'à l'entrepôt de données. Ce système ne se contente pas de charger les données, il doit les faire passer par un tas de moulinettes pour les dé-normaliser, les nettoyer, les contextualiser, puis de les charger de la façon adéquate.

Il s'agit de l'approche dite traditionnelle pour alimenter un entrepôt de données. Les outils qui s'inscrivent dans cette logique disposent en général d'un moteur (engine) et sont installés sur des serveurs distincts. Tous les traitements de transformation se font par le biais du moteur ETL. Nous pouvons citer par exemple Informatica, cognos decisionStream. C'est l'approche la plus étendue actuellement.

Il est important d'indiquer que d'une part l'entrepôt de données est le centre de chaîne décisionnelle, les utilisateurs n'auront accès qu'aux outils de requêtage, et d'analyse. Toutes les parties de l'alimentation et celles de restitution des données sont gérées par une équipe informatique interne ou externe à l'entreprise spécialisée en gestion de base de données et en décisionnel. D'autre part, il est également important de noter qu'il n'existe pas de méthodes de conceptions d'ETL. Chaque entreprise possède ses propres systèmes, sa propre logique de fonctionnement, sa propre culture. Un ETL va essayer de prendre toutes les données de l'entreprise et les mettre dans un entrepôt.

4.1 Définitions

Avant de charger les données de l'entreprise dans l'entrepôt, il faut les faire rentrer dans le schéma de ce dernier. Pour cela, les données doivent donc être:

- **Dé-normalisées :** dans un entrepôt, avoir des doublons n'est pas important, avoir un schéma en troisième forme normale est même déconseillé. Il faut que les données apparaissent là où elles doivent apparaître.

- **Nettoyées :** dans un système de production, les utilisateurs entrent les données. Les risques d'erreurs sont là : entrer la rue au lieu du pays, écrire Canoda au lieu de Canada. Ces erreurs ont des répercutions directes sur les analyses (les commandes avec Canoda comme pays ne feront pas partie des commandes faites au Canada). Il faut pouvoir détecter et corriger ces erreurs.

- **Contextualisées :** imaginez un système de production où les informations sur l'activité du personnel sont enregistrées, et un système de RH ou les informations personnelles, comptables des employés sont stockées. Un entrepôt de données possède une vision universelle, un employé est un employé, et il n'y aura qu'une seule dimension "Employé" avec toutes les informations le concernant.

- **Chargées en DW :** c'est l'étape la plus complexe, il s'agit ici d'ajouter les nouvelles lignes, voir si des lignes ont été modifiées et faire une gestion d'historique, voir si des lignes ont été supprimées et le mentionner dans l'entrepôt, tout en faisant attention de ne pas charger des données en double.

5. Prototypes

Dans cette partie, nous décrivons les projets de recherche sur les entrepôts de données. Les deux premiers (WHIPS et SIRIUS) sont centrés sur des problématiques liées à l'extraction et à la maintenance des données. Le troisième DWQ traite de la qualité des entrepôts, tandis que le dernier, le projet EVOLUTION propose le développement d'une méthodologie et d'outils pour l'aide à la conception et l'évolution des entrepôts.

5.1 WHIPS

Le but du WHIPS (Warehouse Information Prototype at Stanford) [Hammer et al 1995] [Labio et al 1997] [Xuan 2006] est de développer des algorithmes pour collecter, intégrer et maintenir des informations émanant de diverses sources. Une architecture a été développée ainsi qu'un prototype pour identifier les changements de données des sources, les transformer et les synthétiser selon les spécifications de l'entrepôt, et les intégrer de façon incrémentale dans l'entrepôt.

- Les données de l'entrepôt sont représentées selon le modèle relationnel : les vues sont définies dans ce modèle et l'entrepôt stocke les relations. Ces vues sont formulées par l'administrateur à travers le spécificateur de vue.

- Chaque vue d'entrepôt est associée à une gestion de vue. La gestion de vue est utilisée pour synchroniser des modifications survenues à cette vue en même temps.
  Pour chaque modification effectuée, une requête de mise à jour est créée et envoyée au processeur de requête.

- Le moniteur est chargé de détecter des modifications des sources. Chaque modification d'une source sera signalée à l'intégrateur pour analyser son influence sur les vues matérialisées dans l'entrepôt.

- Les adaptateurs, quant à eux, traduisent les requêtes exprimées sous forme d'une représentation relationnelle en des requêtes dans le langage natif de la source. L'utilisation d'un adaptateur par source cache les détails d'interrogation (spécifique à la source) au processeur de requêtes. L'adaptateur de l'entrepôt reçoit les définitions de vues et les modifications sur les données des vues dans un format canonique, et les traduit dans la syntaxe spécifique du système de gestion de base de données de l'entrepôt. Tous les adaptateurs supportent la même interface de requêtes bien que leur code interne dépende de leur source.

- L'intégrateur a pour rôle principal de faciliter la maintenance des vues en calculant les modifications des sources qui ont besoin d'être propagées dans les vues. Ces modifications sont ensuite transférées aux gestionnaires de vues.

- Le processeur de requête reçoit les requêtes de mise à jour des gestionnaires de vues et génère les requêtes appropriées aux adaptateurs de chaque source.

5.2 SIRIUS

Le projet SIRIUS (Supporting the Incremental Refreshment of Information warehouses) [Vavouras et al 2000] développé à l'Université de Zurich, est un système d'entrepôt de données qui a pour objectif d'étudier des techniques permettant le rafraîchissement incrémental de l'entrepôt en réduisant les temps de mise à jour. Le schéma de l'entrepôt est défini sous la forme d'un *schéma global UML*.
Chaque source de données est munie d'un adaptateur et d'un moniteur. Le moniteur détecte les évolutions de la source à laquelle il est associé. Le module de gestion du rafraîchissement est le module central (DWRM *Data Warehouse Refresh Manager*). L'adaptateur traduit les données de la source dans un modèle commun, interne au système et les transmet au module central DWRM.
L'objectif principal de cette approche est d'introduire une architecture d'intergiciel flexible, pour:

- Fournir des solutions pour le rafraîchissement de données.
- Pouvoir être utilisée de façon indépendante par rapport au stockage de données de l'entrepôt.
- Pouvoir être utilisée par des entrepôts de données composés des sources de données hétérogènes.

5.3 DWQ

DWQ (Foundations of Data Warehouse Quality) [Jarke et al 1997] est un projet de la communauté Européenne (France, Allemagne, Italie et Grèce) pour le développement des fondements sémantiques qui permettront d'aider les concepteurs d'entrepôts de données dans le choix des modèles, des structures de données avancées et des techniques d'implantation efficaces en s'appuyant sur des facteurs de qualité de service.

Les résultats comportent des méta-modèles formels de données destinés à la description de l'architecture statique d'un entrepôt de données. Les outils associés comportent des facilités de modélisation incluant des caractéristiques spécifiques aux entrepôts comme la résolution de sources multiples, la gestion de données multidimensionnelles agrégées et des techniques pour l'optimisation de requêtes et la propagation incrémentale des mises à jour.

Ce projet permet aussi l'évolution de l'entrepôt. Les outils incluent un ensemble d'opérateurs, l'analyse et l'optimisation des définitions des vues, la sélection optimisée de sources de données, des stratégies d'intégration et des vues matérialisées.
Les objectifs de recherche visent trois domaines, où les facteurs de qualité représentent le point central :

- Enrichir la sémantique de la métabase avec des modèles formels de qualité de l'information qui permettent l'optimisation de la conception de façon adaptative et quantitative.
- Enrichir la sémantique des modèles d'information qui permettent aussi bien le rafraîchissement incrémental de données et leur propagation vers l'entrepôt, que la résolution des conflits.
- Enrichir la sémantique des modèles de schéma de l'entrepôt qui permettent aux concepteurs de prendre les avantages explicites par rapport à la nature temporelle, spatiale et des agrégats des données.

Le projet DWQ fournit un modèle architectural qui considère la conception, la mise en œuvre, la maintenance et l'évolution des entrepôts. Les composants basics sont :

- Sources : les données stockées dont le contenu peut être matérialisé dans l'entrepôt.
- Adaptateur : responsable du chargement des données sources dans l'entrepôt.
- Base de données finale : l'entrepôt de données et les *data marts*.
- Méta base : conteneur d'information des autres composants, par exemple, le schéma des données sources.
- Agents d'administration : concepteurs de l'entrepôt.
- Clients : utilisateurs de l'information.

5.4 Projet EVOLUTION

Le projet EVOLUTION [Evolution 1997] propose le développement d'une méthodologie et d'un atelier d'outils de type CASE pour l'aide à la conception et l'évolution des entrepôts de données.
L'objectif de ce projet est la spécification d'un ensemble d'outils d'aide à la conception de l'entrepôt et à sa maintenance. Deux objectifs sont intégrés dans la création de ces outils : prévoir l'évolutivité de schéma et gérer la traçabilité des évolutions successives.
Pour atteindre ce but, trois objectifs, correspondant aux problèmes de recherches encore non résolus sont à atteindre :

- La définition d'un formalisme de modélisation des données et des métadonnées de l'entrepôt, ce formalisme doit permettre à la fois de le décrire et de le manipuler (pour le faire évoluer ou pour vérifier ses propriétés).
- Des algorithmes de traitement sémantique de la triple hétérogénéité de conceptualisation, de temps et de granularité.
- Des algorithmes d'optimisation des performances des ressources matérielles et logicielles de l'entrepôt pour fournir efficacement les outils de *Data Mining*.
  L'approche envisagée propose : l'intégration sémantique des différents schémas sources, la constitution d'un schéma de l'entrepôt, la définition et la mise à jour des vues, la prise en compte de l'imprécision (des données mal connues ou incertaines dans l'entrepôt), la prise en compte du temps et de la granularité, la vérification de la cohérence et de la consistance et l'optimisation des ressources (parallélisme).

6. Conclusion

Dans ce chapitre, nous avons présenté les entrepôts de données qui étendent les concepts et la technologie traditionnelle des bases de données pour créer des systèmes d'aide à la décision. En utilisant l'architecture d'un entrepôt de données, nous avons expliqué les différents composants qu'il intègre, comme les diverses sources, les types de données et les différents outils pour arriver à la visualisation de l'information. Enfin, en ce qui concerne les projets de recherche dans le domaine des entrepôts de données, nous avons décrit succinctement WHIPS, SIRIUS, DWQ et EVOLUTION. Les deux premiers sont centrés sur des problématiques liées à l'extraction et à la maintenance incrémentale des données. Le projet de la communauté Européenne DWQ traite de la qualité des entrepôts, tandis que le dernier, le projet EVOLUTION du laboratoire PRISM à Paris, propose le développement d'une méthodologie et d'outils pour l'aide à la conception et à l'évolution des entrepôts. Le modèle le plus souvent utilisé est le modèle relationnel associé au langage SQL pour définir les vues de l'entrepôt.

Dans tous les cas, l'intérêt des entrepôts de données est essentiellement de pouvoir stocker de façon centralisée des données disparates selon un schéma d'ensemble unifié et organisé pour permettre une correction des données (comme la correction des erreurs, la suppression des redondance...) et un usage approfondi des données, afin de permettre une analyse et une fouille des données efficaces dans un cadre de recherche ou de prise de décisions. Le principal inconvénient de ce type de système est la mise à jour régulière nécessaire de son contenu par rapport aux données sources.

L'intégration selon une approche entrepôt de données est fondée sur un schéma global de l'entrepôt fournissant une vue intégrée des sources. Une fois que le schéma de l'entrepôt est conçu, les données sont extraites à partir des sources, transformées au format de représentation des données de l'entrepôt, elles sont éventuellement filtrées pour ne garder que les données pertinentes, et enfin stockées dans l'entrepôt. Le rôle primordial d'un entrepôt de données apparaît ainsi évident dans une stratégie décisionnelle. L'alimentation du datawarehouse en est la phase la plus critique.

Le problème de l'intégration de sources de données hétérogènes dans les entrepôts de données se décompose en deux catégories : l'intégration des schémas et l'intégration des données elles-mêmes. Le processus d'intégration des schémas consiste à passer de plusieurs schémas distribués et hétérogènes à un schéma homogène et intégré. L'intégration des données consiste à pouvoir définir des instances d'objets répondant au schéma homogène préalablement défini correspondant aux données dont ils proviennent. Notre contribution est une approche d'intégration des schémas des sources de données hétérogènes dont le résultat est un schéma global sur lequel seront matérialisées les données de l'entrepôt.

Bilan

Nous avons présenté dans cette première partie de la thèse les systèmes d'intégration de données en général et nous avons étudié en particulier les systèmes de médiation et les entrepôts de données. Dans les systèmes de médiation l'intégration est virtuelle c-à-dire les données restent dans leurs sources d'origine. Ils offrent à l'utilisateur une vue uniforme et centralisée des données distribuées, cette vue pouvant aussi correspondre à une vision plus abstraite, condensée, qualitative des données et donc, plus signifiante pour l'utilisateur. Ces systèmes de médiation sont, par ailleurs, très utiles, en présence de données hétérogènes, car ils donnent l'impression d'utiliser un système homogène. Les médiateurs transmettent les requêtes aux adaptateurs pour extraire les résultats, puis rassemblent le résultat final localement pour construire la réponse. Ceci procure l'avantage de disposer de données fraîches mais avec l'inconvénient d'un accès qui peut s'avérer coûteux.

Dans le cas des entrepôts de données, l'intégration est matérialisée : les données sont regroupées dans une base unique. Cela signifie qu'il y a une duplication des données dans l'entrepôt et qu'il n'est plus nécessaire d'accéder aux sources initiales pour répondre à une requête. Les entrepôts de données utilisent les adaptateurs pour importer les données d'une source dans le but de les matérialiser. Les requêtes interrogeant cette source seront alors évaluées localement à partir des données entreposées. L'avantage de cette approche est qu'elle permet de contrôler les données entreposées alors qu'un de ses inconvénients majeurs est la nécessité d'un administrateur pour entretenir les données.

Il existe également un schéma global dans un entrepôt de données mais celui-ci présente des caractéristiques différentes de celles mentionnées précédemment. Selon, [Metais 2002], le schéma global n'est pas figé mais est amené à évoluer régulièrement. Un entrepôt de données est en effet dynamique et de nouvelles sources sont susceptibles d'être intégrées fréquemment, des données stockées peuvent être réorganisées (agrégation, ajout, suppression d'attributs, …), etc. Certains auteurs préconisent de suivre l'approche Local-as-View (les sources sont des vues sur le schéma global) dans les entrepôts car toute l'information présente dans les différentes sources n'est pas nécessaire [Calvanese 2001]. Il est donc préférable de définir d'abord le schéma global de l'entrepôt qui reflète l'information nécessaire pour l'analyse et la prise de décision, et puis d'établir la correspondance avec les sources (approche descendante), plutôt que de se concentrer sur les sources, avant de produire le schéma global (approche ascendante).

Par ailleurs, avec le développement du Web et des technologies de l'information ces dernières années, d'autres architectures d'intégration telle que l'architecture hybride (approche mixte) ont été proposées. Ces architectures combinent à la fois l'approche médiateur et l'approche entrepôt. Il s'agit, par exemple, d'un système médiateur qui intègre plusieurs sources de données externes et qui exploite un entrepôt de données contenant des données conformes au schéma global du médiateur. C'est dans ce type d'approche que s'inscrit le travail de cette thèse.

Nous avons également montré dans la première partie que les problèmes à résoudre dans le contexte de l'intégration de données sont bien identifiables et que des solutions existent pour bon nombre d'entre-eux. Nous nous sommes attachés à décrire les concepts et les approches d'intégration de données hétérogènes, en insistant sur leurs avantages et leurs inconvénients. En

effet, ces approches se distinguent les unes des autres par la façon dont est établie la correspondance « mapping » entre le schéma global et les schémas locaux. Deux approches principales duales existent : l'approche Global As View (GAV) et l'approche Local-As-View (LAV). D'autres approches hybrides d'intégration ont été proposées. L'accent est particulièrement mis sur la définition et le maintien des liens avec les sources de données distribuées, l'orchestration des requêtes adressées aux systèmes cibles ainsi que l'identification et la résolution des problèmes d'hétérogénéité.

Nous estimons, à ce niveau, avoir les assises nécessaires pour proposer une approche d'intégration de données dans un entrepôt de données tel que nous le verrons dans la deuxième partie de cette thèse.

PARTIE 2 :

Une approche hybride pour l'intégration de sources de données hétérogènes dans les entrepôts de données

Introduction

L'intégration des données dans un système de médiation ou dans un entrepôt de données est considérée comme une étape critique. Rappelons que l'approche par médiation est fondée sur la définition de vues [Rousset et al 02]. Les données ne sont pas stockées dans le système de médiation mais résident dans leur source d'origine. L'utilisateur a une vision unifiée des données sources : l'interrogation se fait par l'intermédiaire d'un schéma global. Il n'a pas connaissance des schémas locaux.

Une requête globale est posée via le schéma global puis elle est décomposée en sous requêtes, traduites pour être exécutées sur les différentes sources concernées.
Le médiateur est chargé de localiser les données pertinentes pour répondre à la requête (en utilisant les métadonnées). L'interrogation effective des sources se fait par des adaptateurs (ou wrappers) qui constituent une interface d'accès aux différentes sources. Ces adaptateurs traduisent les sous requêtes exprimées dans le langage de requête spécifique de chaque source. Les résultats sont ensuite renvoyés au médiateur qui se charge de les intégrer avant de les présenter à l'utilisateur.

Les médiateurs se distinguent les uns des autres entre autres par la façon dont est établie la correspondance (mapping) entre le schéma global et les schémas locaux pour traduire la requête de l'utilisateur. Deux approches principales existent : l'approche Global As View (GAV) et l'approche Local As View (LAV) [Rousset et al 02].

Il faut noter que la médiation ne s'adresse pas uniquement à des bases de données. De nombreux médiateurs permettent d'intégrer des données semi structurées dont XML en est le format. C'est le cas, notamment, de Xylème dont le but est de construire un entrepôt de données dynamique regroupant des documents XML du Web [DEL 03]. L'utilisateur interroge les sources de données à travers une description abstraite des documents. Cette description correspond au schéma global

Les entrepôts de données sont conçus dans un but bien particulier : rassembler l'ensemble des informations d'une entreprise dans une base unique, pour faciliter l'analyse et la prise de décision rapide. Les données stockées dans l'entrepôt proviennent de sources multiples souvent hétérogènes (BD de production et sources externes). Après leur extraction et leur transformation, elles sont stockées et organisées dans l'entrepôt par sujet. Ces données peuvent être ensuite manipulées par un ensemble d'outils de fouille de données (Data Mining), d'analyse en ligne (On-Line Analytical Processing) et d'interrogation (requêteurs) [Doucet et al 01].

Il existe donc une phase d'intégration lors de la conception d'entrepôts mais cette intégration est différente. Toutes les approches que nous avons présentées jusqu'à présent proposent une intégration virtuelle : les données restent dans leurs sources d'origine. Dans le cas des entrepôts de données, l'intégration est matérialisée : les données sont regroupées dans une base unique. Cela signifie qu'il y a une duplication des données dans l'entrepôt et qu'il n'est plus nécessaire d'accéder aux sources initiales pour répondre à une requête.

Avec le développement du Web et des technologies de l'information ces dernières années, d'autres approches d'intégration tels que les systèmes hybrides (approches mixtes) ont été

proposées. Ces systèmes combinent à la fois l'approche médiateur et l'approche entrepôt. Il s'agit, par exemple, d'un système médiateur qui intègre plusieurs sources de données externes et qui exploite un entrepôt de données contenant des données conformes au schéma global du médiateur. Xylème est un exemple de ce type d'architecture : Il adopte l'approche mixte dans son architecture. Le schéma global et les schémas locaux sont exprimés a l'aide d'arbres, avec un mapping GAV / LAV. Notre contribution se situe dans ce contexte et est représentée par la proposition d'une approche hybride appelée HAV : c'est une approche hybride à plus d'un titre. En effet, elle combine d'une part l'approche médiateur et l'approche entrepôt et d'autre part, elle combine les deux approches d'intégration GAV et LAV pour l'intégration des sources de données. Le processus d'intégration selon HAV consiste à prendre en entrée un ensemble de sources hétérogènes et à produire en sortie une description unifiée des schémas initiaux (le schéma intégré pour les systèmes de médiation, schémas et population de l'entrepôt de données) et les règles de traduction (mapping) qui vont permettre l'accès aux données existantes à partir du schéma intégré en utilisant des requêtes.

En résumé, les entrepôts et les systèmes de médiation prennent en compte l'hétérogénéité des sources de données qui est un des principaux problèmes pour lesquels ils sont construits. D'autres problèmes de conception émergent lors de l'utilisation de ces systèmes. Parmi ces problèmes, nous distinguons la définition du schéma global, la définition des mappings qui relient ce schéma global aux sources de données et le traitement des requêtes.

Pour contribuer à la résolution de ces problèmes, nous présentons dans cette deuxième partie de cette thèse l'approche proposée, l'architecture qui la supporte, un cadre formel pour sa définition et enfin sa validation qualitative et quantitative.

Chapitre 4 : Présentation de l'approche HAV pour l'intégration de sources de données hétérogènes dans les entrepôts de données

1. Introduction

Les entrepôts de données et les systèmes de médiation sont aujourd'hui très développés et connus. Cependant, leur mise en œuvre pose un certain nombre de problèmes, en particulier l'intégration de données, le mapping et la génération de requêtes en fonction du contenu des sources et des besoins des utilisateurs. Ce problème est d'autant plus crucial lorsque les sources sont nombreuses et hétérogènes.

L'une des caractéristiques pour la classification des approches à l'intégration de l'information, est de savoir si les données sont matérialisées dans un entrepôt, ou si elles sont conservées dans les sources, dans ce cas, l'approche est appelée virtuelle. Dans ce chapitre, nous présentons une approche virtuelle de l'intégration de données hétérogènes qui peut être utilisée aussi bien pour la construction du schéma global des systèmes de médiation que pour celui des entrepôts qui prennent avantage de l'approche virtuelle de l'intégration pour matérialiser leurs données.
« ‘’e
L'approche proposée dans la présente thèse est une alternative d'intégration qui se propose d'améliorer l'intégration de sources de données hétérogènes en combinant les deux approches de base : GAV et LAV. Son objectif principal est de soutenir et d'améliorer la conception du schéma global sur lequel seront matérialisées les données de l'entrepôt et de faciliter l'interrogation des sources.

Dans ce chapitre, nous présentons notre proposition pour l'intégration de sources de données hétérogènes dans les entrepôts de données. Cette proposition consiste en une approche hybride d'intégration nommée HAV. Nous introduirons les deux approches d'intégration de base pour ensuite montrer comment nous les combinons pour obtenir notre approche HAV. Cette présentation sera suivie par la description de l'architecture qui supportera cette approche et enfin, un cadre formel sera défini pour HAV.

2. Mapping

Il est rapidement apparu que l'un des principaux goulets d'étranglement dans la mise en place d'une application d'intégration de données est l'effort nécessaire pour créer les descriptions de sources, et plus spécifiquement, l'écriture des correspondances sémantiques entre les sources et le schéma de médiation. Les données des sources et de l'entrepôt peuvent être définies en utilisant les approches Local-As-View (LAV) ou l'approche Global As View (GAV). Nous avons présenté dans la première partie de cette thèse ces approches qui sont utilisées dans le monde des systèmes d'intégration tels que les systèmes de médiation et les entrepôts en fournissant une description générale de ces systèmes. Dans cette section, nous présentons plus en détail ces méthodes de base pour mieux saisir leurs avantages et leurs inconvénients et justifier ainsi leur combinaison dans notre approche.

2.1 Global-as-View (GAV)

L'approche GAV a été la première à être proposée pour intégrer des informations. Elle consiste à définir le schéma global en fonction des schémas des sources de données à intégrer puis à le connecter aux différentes sources comme nous pouvons le voir sur la figure 4.1
Pour cela, les prédicats du schéma global, aussi appelés relations globales, sont définis comme des vues sur les prédicats des schémas des sources à intégrer.

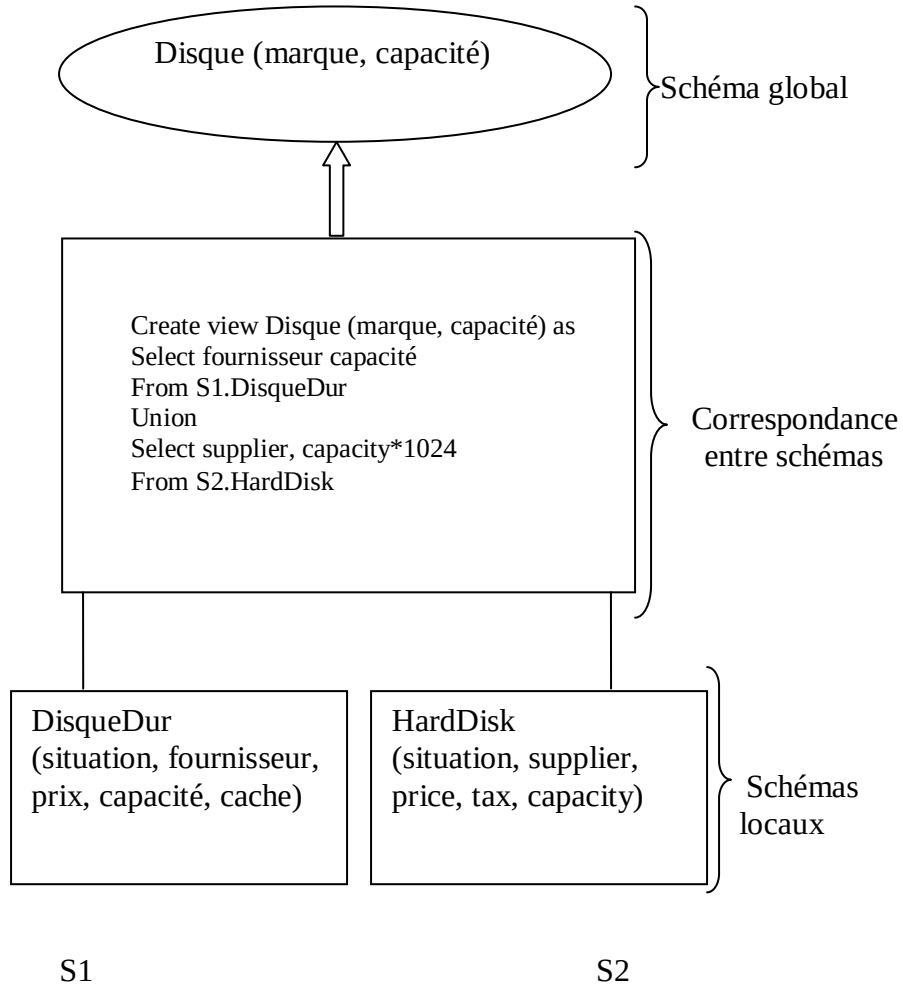
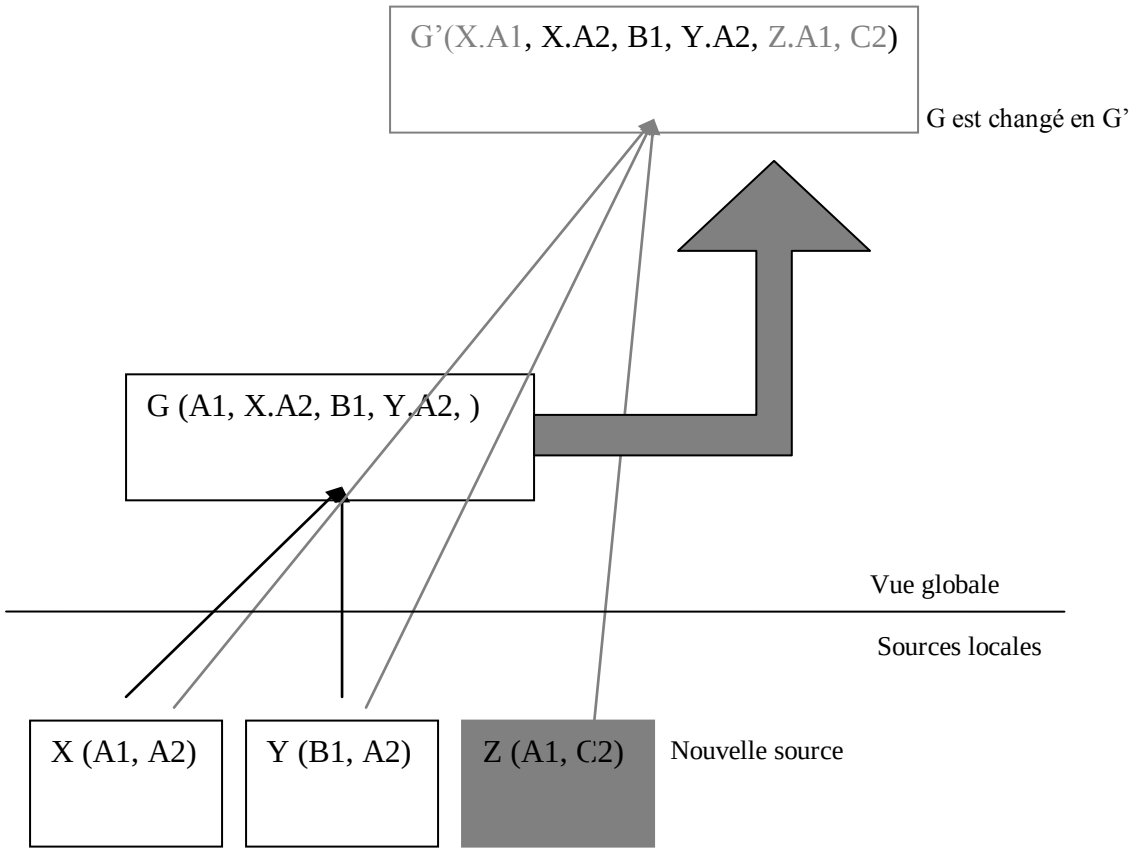


Figure 4.1 : Exemple de définition du schéma global dans GAV

Comme les requêtes d'un utilisateur s'expriment en termes des prédicats du schéma global, nous obtenons facilement une requête en termes des schémas des sources de données intégrées, en remplaçant les prédicats du schéma global par leur définition.

Comme le montre la figure 4.2, un schéma global G (A1, X.A2, B1, Y.A2) est généré en unissant des schémas de sources de X et Y. Toutes les entités à partir des schémas source ont des noms correspondants dans le schéma global, même certains d'entre eux partagent le même sens, comme X.A2 et Y.A2). Cependant, elles mènent aussi à une difficulté dans la mise à jour du schéma

global en raison de la dépendance entre le schéma global et les sources locales. Par exemple, si le schéma global a été mis à jour (ajout ou suppression de nouvelles entités), tous les nœuds locaux ont à mettre à jour leurs vues locales sur le nouveau schéma global. D'autre part, le fait d'ajouter ou supprimer des sources peut entraîner des changements considérables pour le schéma global. Comme le montre la figure 1.2, si une nouvelle source Z a été ajoutée au système, en conséquence le schéma global doit être mis à jour dans G '(A1, X.A2, B1, Y.A2, Z.A1, C2).



4.2: Exemple d'ajout d'une nouvelle source dans GAV

2.2 Local-as-View (LAV)

Contrairement à l'approche GAV, LAV suppose l'existence d'un schéma global et consiste à définir les schémas des sources de données à intégrer comme des vues du schéma global. Ces vues définissent comment les informations mappent sur le schéma global en exprimant un mapping entre une relation dans le schéma local en une (un ensemble de) relation(s) dans le schéma global [Pottinger et al 2000], un exemple de LAV est illustré dans la figure 4.3. Le principal avantage de l'approche LAV sur l'approche GAV est qu'il n'y a pas de dépendance sur le schéma global. Dans LAV, chaque schéma de source est mappé sur le schéma global.

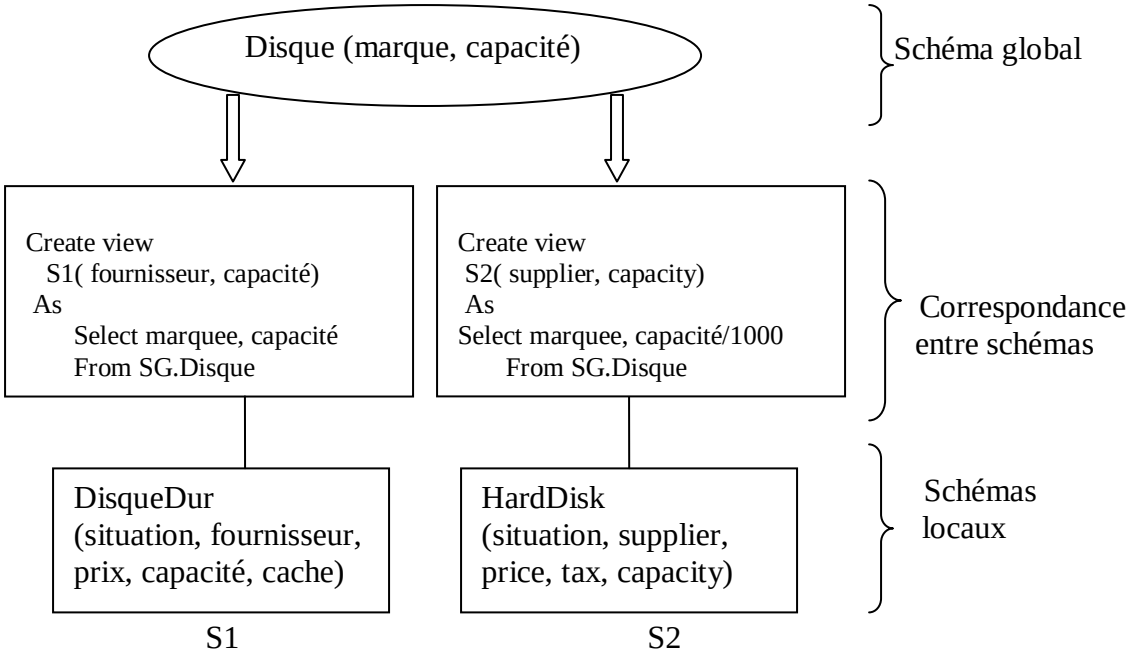


Figure 4.3 : Exemple de définition du schéma global dans LAV

L'ajout de nouvelles sources nécessite seulement les définitions des mappings nécessaires entre le schéma de la source et le schéma global comme le montre la figure 4.4. Toutefois, dans cette approche la réponse aux requêtes devient plus difficile parce que la reformulation de la requête est difficile à réaliser.

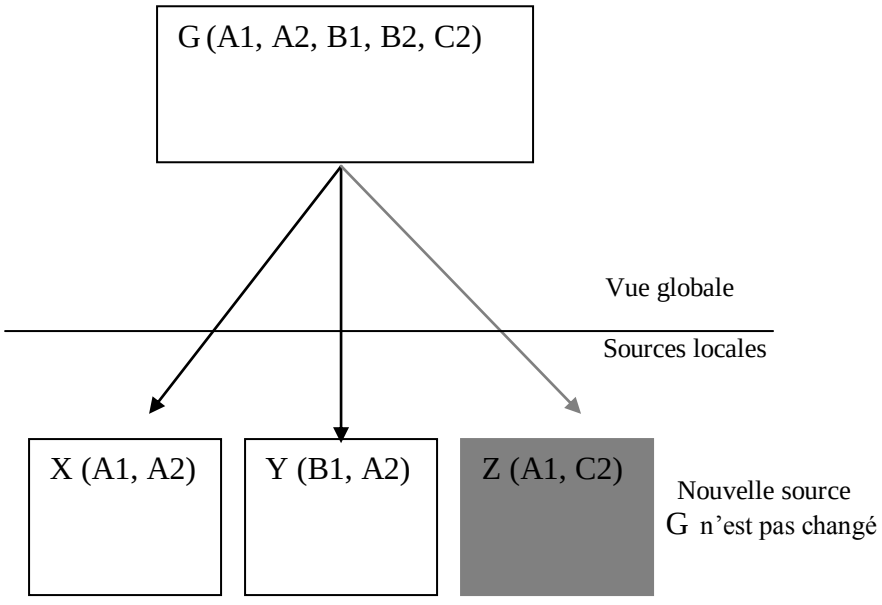


Figure 4.4 : Exemple d'ajout d'une nouvelle source dans LAV

Avant de présenter l'approche que nous proposons HAV, dressons un bilan de ces deux approches.

2.3 Bilan sur les approches GAV et LAV

Les deux problématiques soulevées dans GAV et LAV sont : la *réécriture des requêtes* (réécriture des requêtes utilisateurs sur les sources et construction de la réponse) et la *scalabilité du système d'intégration* (ajout ou suppression de sources de données) [Xuan 2006]. L'approche GAV facilite la réécriture des requêtes mais l'approche LAV est plus intéressante pour assurer la scalabilité du système d'intégration.

En effet, une requête utilisateur s'exprime en termes des relations du schéma global, sa réécriture en fonction des schémas des sources, dans une approche GAV, nécessite un simple dépliement des relations utilisées dans la requête, par leurs définitions locales. Cette réécriture, dans une approche LAV, devient un problème complexe nécessitant des inférences. D'autre part, l'ajout d'une nouvelle source de données, pouvant nécessiter une mise à jour du schéma global et du mapping entre le schéma global et les schémas des sources dans une approche GAV, est facilité dans une approche LAV. Seules les vues des nouvelles sources doivent être rajoutées (sous réserve que le schéma global ait été bien défini initialement).

3. Présentation de l'approche HAV

L'approche proposée dans cette thèse est baptisée HAV (Hybrid As View). Elle est le résultat de la combinaison du meilleur de GAV et LAV décrites plus haut, réduisant ainsi le problème de la réécriture de requêtes tout en étant facilement extensible. Il y a deux couches de médiateurs dans HAV. La couche inférieure est constituée de médiateurs multiples appelés 'médiateurs spécialisés', qui utilisent tous l'approche LAV pour intégrer les schémas d'un sous-ensemble de sources de données utilisant le même modèle de données. La couche supérieure a un seul médiateur global qui intègre les schémas spécialisés de la couche inférieure en utilisant l'approche GAV.

HAV est une approche qui peut être aussi bien utilisée dans un système de médiation pour l'intégration de sources de données hétérogènes que dans un système mixte d'intégration de données qui combine à la fois l'approche médiateur et l'approche entrepôt. Il s'agira dans ce cas, d'un système médiateur qui intègre plusieurs sources de données externes pour la matérialisation de l'entrepôt conformément au schéma global du médiateur.

En effet, GAV et LAV sont combinées pour donner l'approche hybride d'intégration de données hétérogènes: HAV. Par rapport à LAV et GAV, l'originalité de notre approche est que, en plus des caractéristiques des systèmes actuels, vise à promouvoir une architecture qui améliore l'intégration de données hétérogènes en soutenant la combinaison des deux approches GAV et LAV et de nous faire bénéficier de leurs avantages, car aucune d'elles n'est pleinement satisfaisante.

Dans l'architecture supportant HAV que nous présenterons plus loin, le schéma global n'est pas directement lié aux sources de données. En effet, un ensemble de schémas partiels joue le rôle de sources virtuelles: un mapping GAV définit la relation entre le schéma global et les schémas partiels. Chaque schéma partiel est défini sur un ensemble de source de données (sources où sont matérialisées les données) ayant le même modèle au moyen d'un mapping LAV.

L'approche HAV pallie les inconvénients de GAV et LAV, car les systèmes d'intégration de données définis selon HAV, ayant un petit nombre de schémas partiels, permettront une conception simple et stable pour le mapping GAV entre le schéma global et les schémas partiels.

D'autre part, les mappings LAV définis sur un petit nombre de sources de données (qui sont exprimées dans le même modèle de données) permettront un traitement facile de la requête.

Donc, HAV vise à combiner le meilleur des deux approches de base: la reformulation simple de requêtes de GAV et l'extensibilité de LAV. En effet, le système qui supportera notre approche est un système de sources de données hétérogènes qui adopte une architecture distribuée, qui consiste en plusieurs composants spécialisés et qui supportera à la fois le passage à l'échelle (augmentation des sources) par l'assouplissement de l'introduction de nouvelles sources de données et de faciliter l'exécution des requêtes.

3.1 Intégration de sources de données dans HAV

Les vues de chaque source définies selon LAV mappent sur le schéma spécialisé correspondant. Le schéma global G est défini selon GAV en fonction des schémas spécialisés à intégrer comme nous pouvons le voir sur la figure 4.5.

Pour cela, les prédicats du schéma global, aussi appelés relations globales, sont définis comme des vues sur les prédicats des schémas spécialisés à intégrer. En effet, un schéma global G (A1, A2, G_{p1}.B1, G_{p2}.B1, B2, C2) est généré en unissant des schémas spécialisés S_{P1}, S_{P2},…..S_{Pn} selon un mapping GAV. Les sources de même modèle mappent sur le schéma spécialisé SPi de même modèle que ces sources en exprimant un mapping LAV entre une relation dans le schéma local en relations dans le schéma spécialisé correspondant.

L'ajout de nouvelles sources nécessite seulement les définitions des mappings nécessaires entre le schéma de la source et le schéma spécialisé correspondant comme le montre la figure 4.5.

Une requête d'un utilisateur s'exprime en termes des prédicats du schéma global, nous obtenons facilement une décomposition de cette requête en sous requêtes en termes des schémas spécialisés, en remplaçant les prédicats du schéma global par leur définition. La réponse aux sous requêtes est plus facile à réaliser car la reformulation de chaque sous requête concerne un nombre réduit de sources de données.

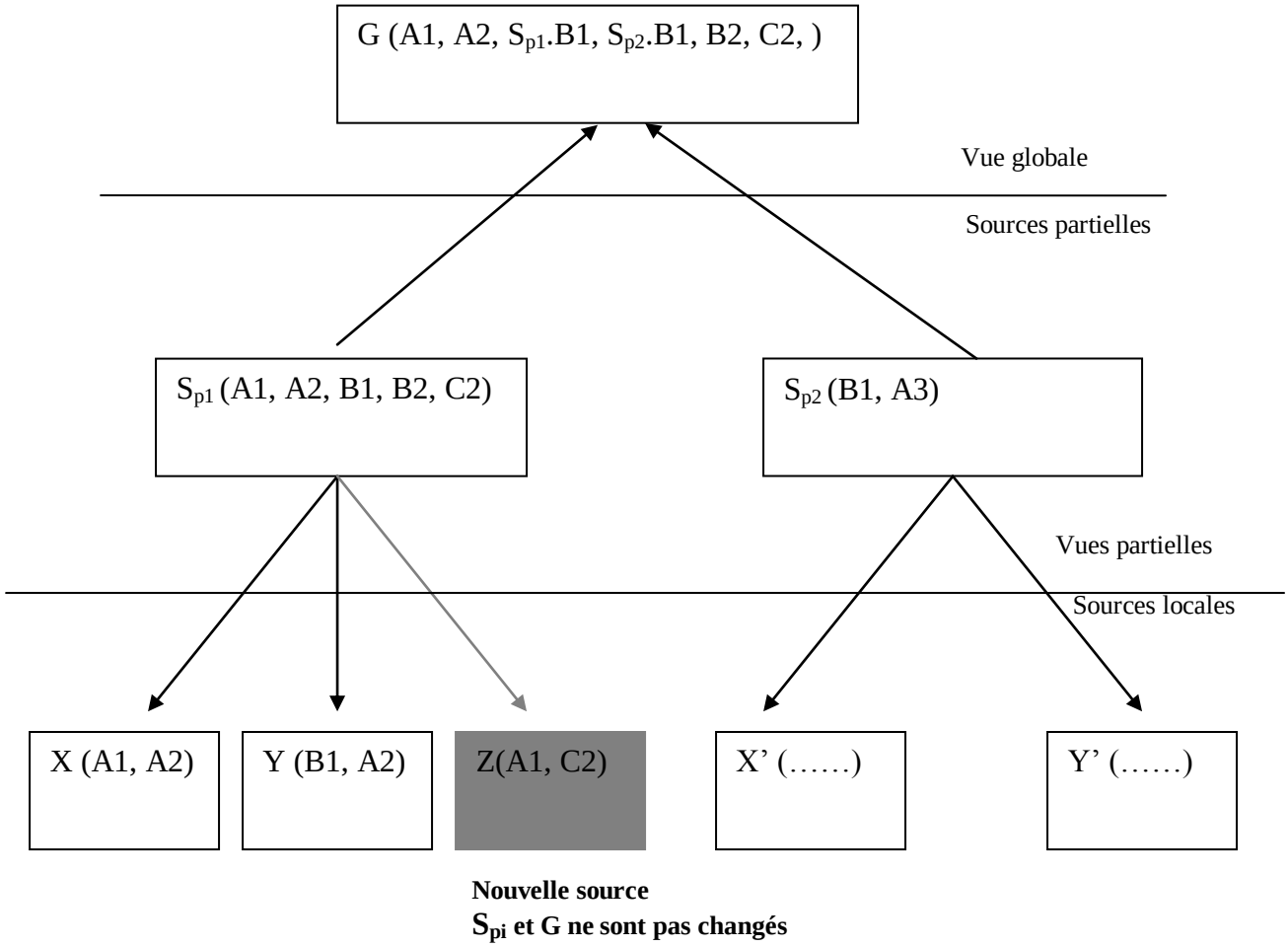


Figure 4.5: Exemple dans HAV

Un système d'intégration basé sur HAV fournit de nouvelles caractéristiques :

3.1.1 Transformation de schémas dans HAV

Le processus d'élaboration du schéma global selon l'approche HAV est divisé en trois phases : transformation de schemas, génération du schéma global et génération du mapping des schémas. Pour établir le schéma global selon HAV, plusieurs transformations de schémas doivent avoir lieu: en premier lieu, les schémas locaux de même modèle sont intégrés en un seul schéma spécialisé ensuite vers le schéma global (intégré). Ces transformations forment la transformation de schémas.

3.1.2 Transformation de langages dans HAV

Les requêtes qui sont lancées sur le système construit selon HAV et exprimées dans un langage de manipulation sont divisées en sous-requêtes qui sont envoyées à des médiateurs spécialisés, et ces sous-requêtes sont traduites dans le langage des médiateurs spécialisés pertinents. Chaque sous- requête d'un médiateur spécialisé est exprimée dans le même langage de manipulation que les sources de données sur lesquelles elle va être exécutée. Chacune d'elles sera à son tour

divisée en sous-requêtes exprimées dans le même langage pour être exécutées sur les sources de données locales, donc à ce niveau, il n'y a pas de traduction de langage car elles sont toutes exprimées dans le même modèle. Les sources de données locales répondent aux requêtes qui leur ont été envoyées, et quand elles retournent le résultat, ils sont traduits par les adaptateurs spécialisés dans la terminologie commune. Ces traductions forment la transformation de langages.

3.2 Traitement de requêtes dans HAV

Indépendamment de la méthode utilisée pour la spécification de la correspondance entre le schéma global et les sources, un service de base fourni par le système d'intégration de données est de répondre aux questions posées en termes du schéma global.
Compte tenu de l'architecture du système, le traitement des requêtes en matière d'intégration de données nécessite une étape de reformulation: la requête sur le schéma global doit être reformulée en termes d'une série de requêtes sur les sources.

Notre approche d'intégration de données peut être appliquée pour acquérir les données à partir des sources pour alimenter l'entrepôt de données conformément au schéma global, par traitement des demandes d'administrateurs ou bien pour acquérir des données à partir des sources pour répondre à une requête utilisateur dont la réponse ne nécessite pas une historisation dans l'entrepôt.

Un des problèmes difficiles rencontrés lors de la conception des systèmes de médiation est le traitement des requêtes de médiation. Dans notre cas le médiateur global intègre plusieurs sources de données pour la matérialisation de l'entrepôt conformément au schéma global du médiateur.

Dans cette section, nous proposons le principe général qui traite les requêtes de médiation sur un ensemble de sources hétérogènes dans le contexte HAV, comme illustré sur la figure 4.6.

Lorsque le médiateur global reçoit une requête utilisateur, il la décompose en sous-requêtes (si cela est nécessaire) et transmet ces dernières aux adaptateurs spécialisés concernés. Les adaptateurs spécialisés fournissent le mapping de la vue intégrée du médiateur global à son schéma partiel spécifique, c-à-dire que certains (ou tous) adaptateurs spécialisés reçoivent chacun la sous-requête qui le concerne à partir du médiateur global et la traduit dans le langage et la terminologie du schéma partiel spécifique. Chaque sous-requête est décomposée à son tour en sous-requêtes qui sont envoyées aux adaptateurs de sources correspondants. Les résultats de la requête sont retournés à partir des adaptateurs de sources aux médiateurs spécialisés correspondants qui intègrent le résultat partiel via les médiateurs spécialisés. Le médiateur global intègre l'ensemble des résultats et renvoie une réponse unique à l'utilisateur.

L'algorithme général du traitement de requêtes de médiation dans HAV comprend les étapes suivantes:

- Etape 1 : Réception de la requête par le médiateur global.

- Etape 2: recherche de médiateurs partiels contributifs: Cette étape consiste en la décomposition (par le médiateur global) de la requête globale en sous-requêtes en trouvant tous les médiateurs partiels pertinents qui peuvent contribuer au calcul de la sous-requête.

- Etape 3: Recherche de sources contributives: Cette étape consiste à trouver pour un médiateur spécialisé concerné toutes les sources sous-jacentes, pertinentes qui sont de même modèle que lui, et qui peuvent contribuer au calcul de la sous-requête correspondante.

- Etape 4: Exécution de chaque sous-requête par la source concernée.

- Etape 5: Recomposition des résultats partiels : Chaque médiateur partiel recompose les résultats obtenus à l'étape 4.

- Etape 6: Recomposition du résultat global : Pour obtenir le résultat final, le médiateur global recompose les résultats obtenus à partir des médiateurs partiels.

Pour estimer la faisabilité de notre algorithme, nous dirons qu'il est réalisable tout comme dans un système d'intégration, dans la mesure où les mêmes stratégies de réécriture peuvent être utlilisées.

En effet, les requêtes étant posées sur le schéma médiateur ou schéma virtuel (et non sur les bases sources) en utilisant un certain nombre de vues, le système essaie d'effectuer la réécriture des requêtes des utilisateurs en s'assurant que les requêtes réécrites sont soit équivalentes aux requêtes initiales, soit incluses (de façon maximale) dans chacune des requêtes initiales. La réécriture maximale de requête essaie de trouver la meilleure solution possible par rapport aux sources disponibles.

Dans le niveau GAV, la réécriture de la requête se résumera à un simple dépliement de la requête, alors que pour le niveau LAV, la réécriture utilisera les algorithmes qui existent à la différence que dans notre algorithme, la complexité de réécriture des sous-requêtes qui sont posées à un médiateur spécialisé sera réduite avec *l'homogénéité* et la *réduction* du nombre de sources concernées par LAV. Ceci parce que, dans les algorithmes de réécriture, la complexité augmente avec l'hétérogénéité et le nombre de sources.

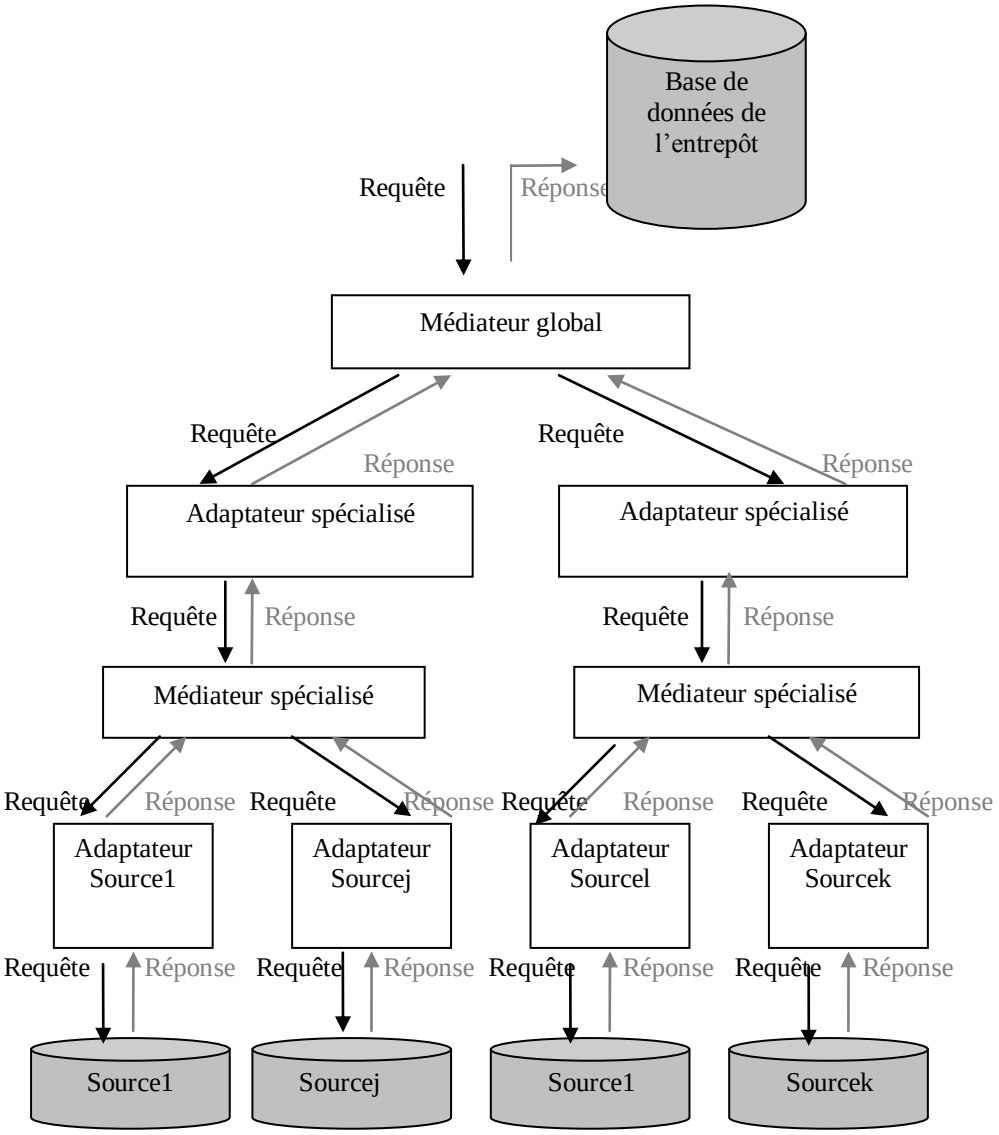


Figure 4.6: Principe de traitement de requêtes dans un système d'intégration construit selon HAV.

4. Une architecture pour HAV

L'architecture qui supporte notre approche est une architecture distribuée qui se compose de plusieurs composants spécialisés. Les utilisateurs finaux interagissent avec les applications écrites par les programmeurs d'applications. Les applications accèdent aux sources de données pertinentes par l'intermédiaire des médiateurs. Le médiateur global exporte un schéma médiateur qui est une représentation intégrée des schémas spécialisés et traite les requêtes sur la représentation intégrée. Les médiateurs spécialisés exportent chacun un schéma spécialisé qui est une représentation intégrée des schémas des sources de même modèle que lui.

Pour construire cette représentation l'administrateur fournit des informations aux médiateurs pour accomplir le traitement des requêtes. Cette information comprend des schémas des sources de données, le schéma médiateur et les vues.

Pour faire face aux différents langages d'interrogation de chaque source de données, les adaptateurs transforment les sous-requêtes envoyées aux sources de données et transforment les réponses renvoyées à partir de sources de données.

Notre objectif est de discuter de notre approche d'intégration de données. La figure 4.7 illustre l'architecture actuelle du système qui est conforme à l'approche décrite dans cette thèse, et où les médiateurs spécialisés se placent au cœur de cette architecture. Ils sont considérés comme des sources virtuelles qui seront interrogées par le médiateur global via les adaptateurs spécialisés.

Comme le montre la figure 4.7, l'architecture supportant l'approche HAV est une architecture à trois niveaux: le niveau source, le niveau spécialisé et le niveau externe.

- Le niveau source (NSR):

  Ce niveau comprend un ensemble de sources, d'adaptateurs de sources et de médiateurs spécialisés. L'interaction entre chaque médiateur spécialisé et les sources est rendue facile par des modules logiciels, appelés des adaptateurs sources. Chaque médiateur spécialisé sera concerné par un ensemble de sources de données autonomes, indépendantes et homogènes (en ce sens que les schémas sources sont exprimés dans le même modèle de données). Conformément à l'approche HAV, tous les schémas partiels concernant un médiateur spécialisé sont construits selon l'approche LAV et sont de même modèle que les sources sous-jacentes.

- Le niveau spécialisé (NSP):

  Cette couche médiane est le cœur de l'architecture de médiation de HAV. Il ya deux éléments fondamentaux: le médiateur global et un adaptateur spécialisé par schéma partiel. Les schémas partiels sont considérés comme des sources virtuelles pouvant être interrogées par le médiateur global via les adaptateurs spécialisés. Le médiateur global est construit selon l'approche GAV.

- Le niveau externe (NEX):

  Ce niveau permet l'interaction de l'administrateur de l'entrepôt de données avec le système en interrogeant le schéma de l'entrepôt. Le système effectuera la tâche de traitement des sources via les schémas partiels pour récupérer les données satisfaisant les demandes de l'administrateur. Nous notons que ces données servent à matérialiser le datawarehouse. Notons également que ce niveau permet l'interaction de n'importe quel utilisateur avec le système en tant que système de médiation.

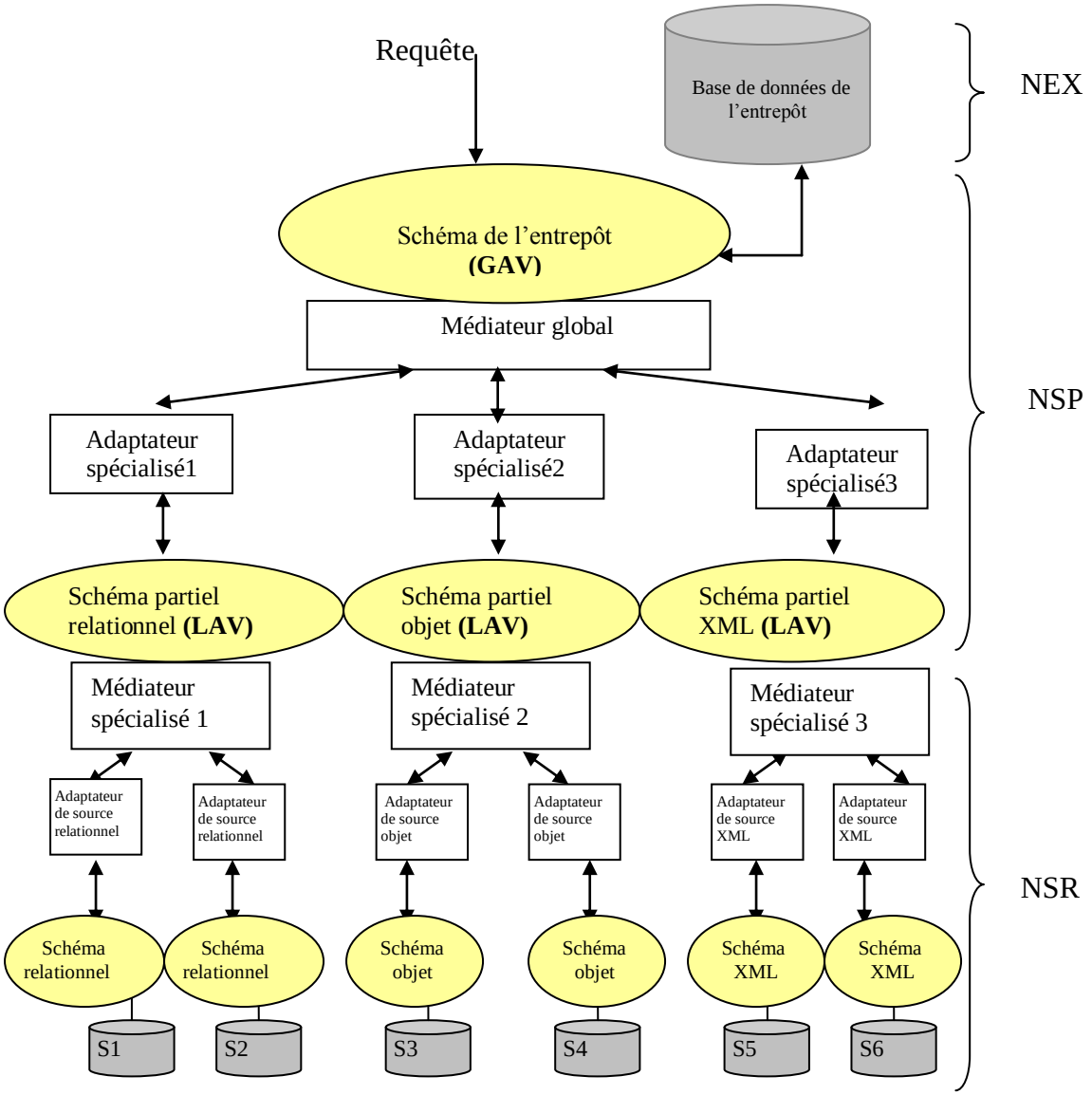


Figure 4.7: Une architecture à trois niveaux

4.1 Description des différents composants du système

Dans cette section, nous présentons les concepts essentiels et les composants de base d'un système d'intégration construit selon HAV. Ces composants sont les suivants:

● Les sources de données

Elles représentent un ensemble de sources de données, contenant les informations réelles qui intéressent les utilisateurs. Ces sources de données sont hétérogènes.

● Les adaptateurs de sources

Un adaptateur est placé au dessus de chaque source. L'objectif de chacun d'eux est d'accéder à une source, d'extraire les données pertinentes, et de présenter ces données au médiateur spécialisé correspondant. Dans un système général de médiation de données, l'adaptateur cache l'hétérogénéité au médiateur ; à ce niveau de notre architecture, il n'y a pas d'hétérogénéité, car toutes les sources concernées par l'intégration d'un médiateur spécialisé sont dans le même modèle (par exemple, elles sont toutes relationnelles).

● Les médiateurs spécialisés

Leur rôle est de collecter, nettoyer et de combiner les données produites par différentes sources. Chacun intègre toutes les sources sous-jacentes. Chaque médiateur spécialisé s'occupe d'une intégration partielle: elle procédera à l'intégration des sources dont le modèle est le même que le sien.

● Les schemas spécialisés

Chacun d'eux est une vue réconciliée, intégrée et virtuelle sur les schemas sources sous-jacents ayant le même modèle de données. Il est conçu selon l'approche LAV.

● Les adaptateurs spécialisés

Chaque adaptateur spécialisé devra établir la correspondance entre le schéma partiel sous-jacent et le schéma de l'entrepôt de données. L'objectif de chacun est d'accéder à un médiateur spécialisé, d'extraire les données pertinentes, et de présenter ces données dans un format spécifique.

● Le médiateur global

Son rôle est de collecter, nettoyer et combiner les données produites par différents adaptateurs spécialisés.

● Schéma de l'entrepôt

Le rôle de ce schéma est de fournir une vue réconciliée, intégrée et virtuelle des sous-schémas partiels sous-jacents sur lesquels sont exprimées les requêtes pour matérialiser l'entrepôt ou des requêtes de l'utilisateur sur le système de médiation. Il est conçu selon l'approche GAV. Il est proposé par l'administrateur de l'entrepôt en fonction des besoins.

● Base de données de l'entrepôt

C'est la base de données où les données résidant dans plusieurs sources de données sont matérialisées conformément au schéma de l'entrepôt de données.

Comme nous venons de le voir et en résumé à cette section, les systèmes d'intégration de données basés sur HAV, sont caractérisés par une architecture basée sur un schéma global, des schémas spécialisés et un ensemble de sources. Les sources contiennent des données réelles, tandis que chaque schéma spécialisé fournit une vue intégrée et virtuelle des sources sous-jacentes. Le schéma global quand à lui fournit une vue virtuelle et intégrée des schémas spécialisés.
L'atout principal de cette architecture est qu'elle permet de combiner les deux approches de base GAV et LAV de telle sorte à réaliser les avantages de l'une et de l'autre.

4.2 Comparaison des différentes approches avec HAV

Lors de l'étude des approches d'intégration GAV et LAV, nous avons dégagé certaines limites, qui sont : (i) la complexité de réécriture des requêtes dans LAV (ii) la difficulté d'ajout ou de suppression d'une source dans GAV. L'approche HAV que nous proposons présente certains apports qui permettent de résoudre complètement ou partiellement ces limites :

Pour soutenir les caractéristiques de HAV, nous allons comparer les trois types d'architecture supportant GAV, LAV et HAV.

Supposons que le schéma global de la figure 4.1 qui représente une architecture de médiation commune est construit sur n sources de données hétérogènes. Supposons également que nous avons parmi ces n sources par exemple : j sources de données relationnelles, k sources de données objets et m sources XML tel que : n = j+ k+ m.

- Si le schéma global est construit sur l'approche GAV, nous aurons les inconvénients suivants:
  - Plus n est large, moins GAV est stable et efficace.
  - Si une source est rajoutée, le schéma global sera reconsidéré.
- Si le schéma global est construit selon l'approche LAV, la réécriture / réponse de requêtes est difficile. Cette complexité de l'exécution de la requête augmente avec l'augmentation du nombre de sources n.
- Lorsque le schéma global est construit avec HAV (figure 4.7), les problèmes relatifs au niveau supérieur sont résolus et ceux relatifs au niveau inférieurs sont améliorés, en effet:
  - Le nombre de sources concernées par GAV est stabilisé: ce nombre est toujours égal au nombre de type de modèles source; dans notre exemple, il est égal à trois (relationnel, objet, XML).
  - Si l'on ajoute une nouvelle source, ceci n'affecte en rien le schéma global, car elle sera prise en charge par le niveau inférieur construit selon LAV. Par exemple, nous voulons ajouter une nouvelle source relationnelle, elle sera créée comme une vue sur le schéma partiel relationnel et nous aurons donc (j +1) sources relationnelles.
  - On remarque que pour chaque médiateur spécialisé les sources à intégrer sont toutes homogènes (dans le sens que les schémas sources sont exprimés dans le même modèle

que le schéma spécialisé), donc à ce niveau le problème de l'intégration de sources schématiquement hétérogènes est résolu. Par ailleurs, le nombre de ces sources est inférieur à n: il est soit inférieur ou égal à J, inférieur ou égal à k, inférieur ou égal à m. Ainsi, la complexité de reformuler les sous-requêtes qui lui sont posées sera réduite avec *l'homogénéité* et la *réduction* du nombre de sources concernées par LAV.

La comparaison de GAV, LAV HAV nous permet d'établir pour HAV les caractéristiques suivantes:

1. Le schéma global est construit avec l'approche GAV et fournit une vue unifiée des schémas de médiateurs spécialisés (appelés schémas partiels), qui sont des sources virtuelles pour le médiateur global. Les schémas partiels sont construits avec l'approche LAV sur les sources réelles. Chaque médiateur spécialisé s'occupe seulement des sources qui sont représentées dans le même modèle de données que lui. De telles représentations permettent une grande flexibilité dans la structure des données représentées.

2. Chaque adaptateur source n'aura pas à traduire les sous-requêtes qui lui sont envoyées par le médiateur spécialisé correspondant, dans le langage source, parce qu'elles sont exprimées dans le même langage. En effet, elles seront exprimées en SQL, par exemple si le médiateur spécialisé et les sources correspondantes sont relationnels.

3. Il est connu que la complexité de la vue globale augmente avec le nombre de sources et que l'approche GAV est plus efficace et réalisable si l'ensemble des sources à intégrer est petit et stable [Xu et al, 2004]. Ces deux qualités sont réalisées dans notre approche. En effet, le médiateur global aura toujours à intégrer un nombre réduit de schémas partiels qui est égal au nombre des modèles des sources de données.

4. L'inconvénient majeur de l'approche GAV comme mentionné ci-dessus est l'ajout ou la suppression d'une nouvelle source. Dans notre architecture, par exemple l'ajout d'une nouvelle source n'affecte pas le schéma global. Cette tâche est traitée par le médiateur spécialisé correspondant qui est construit selon l'approche LAV. En d'autres termes, cette source est construite comme une vue sur le schéma partiel correspondant parce que l'approche LAV est très flexible avec l'ajout (ou la suppression) de sources de données à intégrer. Alors cela n'aura aucun 'effet sur le schéma global.

5. Les requêtes seront exprimées en termes du schéma global et leur exécution se fait en atteignant chaque source concernée par le médiateur spécialisé. Selon [Rousset et al, 2002] - [Koffina et al, 2005], le prix à payer pour la flexibilité et la simplicité de la mise à jour dans un médiateur conçu selon l'approche LAV est la complexité de la construction de la réponse à une requête et que cette complexité augmente avec l'augmentation du nombre de sources. Dans notre architecture cette complexité sera réduite parce que chaque médiateur spécialisé a un nombre réduit de sources à intégrer, et ces sources sont dans le même modèle, en d'autres termes, ils sont homogènes. Ainsi, la réécriture de requêtes en termes de vue sera plus facile.

6. Les mappings sont exprimés sur les éléments du schéma source à travers des schémas partiels. Ils sont expressifs, et sont bien adaptés pour représenter les relations dans les environnements distribués d'intégration de données.

4.3 Synthèse des approches

Dans le tableau suivant, nous résumons les avantages et les inconvénients des approches vues précédemment [Koffina et al, 2005], [Xu et al, 2004], [Boyd et al, 2004], et ceux de notre approche.

Approche	Avantages	Inconvénients
GAV	- Réponse aux requêtes assez simple, - Réécriture des requêtes simple.	- N'est pas flexible pour l'ajout ou la suppression des sources de données, - N'est pas flexible pour l'ajout ou la suppression de certaines contraintes sur les sources.
LAV	- Flexible pour l'ajout ou la suppression des sources de données, - Flexible pour l'ajout ou la suppression de certaines contraintes sur les sources.	- La réponse aux requêtes est difficile, - La réécriture des requêtes est difficile. - changement du schéma global nécessite une reconsidération globale des vues
GLAV	- Donne la possibilité de mappings plus expressifs, - L'ajout/suppression est facile	La complexité de reformulation des requêtes est la même que dans LAV.
BAV	- Permet l'expression du mapping dans les deux sens, - Supporte l'évolution du schéma global et des schemas locaux.	Susceptible d'être plus coûteux de raisonner et de traiter avec BAV qu'il ne serait avec les définitions des vues correspondantes dans LAV, GAV ou GLAV.
BGLAV	- Utiliser la Source et La Cible - Mapping basé sur un schéma cible prédéfini, - Donne une algèbre relationnelle étendue.	Non documentés.
HAV	*- Efficace et praticable,* *- Flexible pour l'ajout ou la suppression des sources de données,* *- La complexité de reformulation des requêtes est réduite,* *- Pas d'hétérogénéité des sources relatives à chaque médiateur spécialisé.*	A déterminer.

Tableau 4.1: Comparaison des différentes approches

5. Cadre formel pour HAV

Les recherches dans le domaine de l'intégration de données tentent de définir une architecture de référence, de donner une bonne compréhension des concepts, et de proposer un cadre formel pour résoudre de nombreux problèmes techniques tel que l'intégration de données.

Dans cette section, nous présentons un cadre formel pour définir un système d'intégration de données basé sur l'approche HAV. Pour cela, nous utilisons récursivement la définition suggérée par LENZERINI [Lenzerini, 2002] pour un système d'intégration basé sur un schéma global.

[Lenzerini, 2002], définit de manière formelle, un système d'intégration de données I comme un triplet < G, S, M > où :

1. G est le schéma global, un ensemble de symboles de prédicats globaux (ou relations globales) , et un ensemble de contraintes exprimées sur ces symboles de prédicats;

2. S est le schéma source, un ensemble de prédicats locaux ou relations locales qui constituent la représentation des données contenues dans les sources ;

3. M représente les correspondances entre G et S. Les correspondances sont constituées par un ensemble d'assertions qui établissent le lien entre les prédicats (relations) du schéma global et les prédicats (relations) du schéma source.

L'interrogation du système constitué se fait à l'aide des prédicats globaux définis dans G. L'établissement des correspondances entre les schémas locaux et le schéma global s'effectue en tenant compte des problèmes liés à l'hétérogénéité des sources en présence.

5.1 Définition Formelle d'un système d'intégration basé sur HAV

Définition 1: Un système d'intégration de données I dans HAV est un triplet *(G, I_p, Mg,p) où*:

- G est le schéma global exprimé dans le langage L_G, sur l'alphabet A_G. Le langage L_G détermine l'ensemble de contraintes qui peuvent être définies sur lui.

- Ip, est un ensemble de systèmes d'intégration de données, chacun est un triplet (S_p, S , $M_{Sp,S}$) où:
  - S_p est le schéma partiel d'un médiateur spécialisé exprimé dans un langage L_p sur un alphabet A_p. Le langage détermine l'ensemble des contraintes qui peuvent être définies sur lui.
  - S est le schéma de la source de même modèle que S_p sur un alphabet A_s.
  - $M_{Sp,S}$ représente les règles de mapping entre S et S_p et réciproquement. Il est constitué par un ensemble d'assertions de la forme :

$$q_{Sp} \rightarrow q_s$$
$$q_s \rightarrow q_{Sp}$$

où q_{Sp} et q_s sont deux requêtes, respectivement sur le schéma partiel S_p et sur le schéma source S.

Une assertion $q_{Sp} \rightarrow q_s$ spécifie qu'un concept représenté par la requête q_{Sp} sur les schémas partiels correspond au concept dans le schéma source représenté par la requête q_s.

- *Mg,p* représente les règles de mapping entre G et S_p et réciproquement. Il est constitué par un ensemble d'assertions de la forme:

$$q_{Sp} \rightarrow q_G$$
$$q_G \rightarrow q_{Sp}$$

où q_{Sp} et q_G sont deux requêtes, respectivement sur le schéma partiel S_p et sur le schéma global G.

En d'autres termes, le schéma global G fournit une vue intégrée des schémas partiels S_p,, où chacun est le schéma résultat d'un médiateur spécialisé. Les assertions de mapping établissent la connexion entre les éléments du schéma global avec ceux des schémas partiels. D'un autre côté, les sources de même modèle sont définies comme des vues sur le schéma partiel correspondant S_p Les schémas des sources décrivent la structure des données. Dans ce cas, les assertions de mapping établissent la connexion entre les éléments du schéma partiel et ceux des sources.

Il est nécessaire d'effectuer des requêtes afin de vérifier le comportement du médiateur global. Une requête est posée sur I en termes du schéma global et est exprimée en termes du langage de requêtes L_G sur l'alphabet A_G. Elle est divisée en sous-requêtes, où chacune d'elles sera envoyée au médiateur spécialisé correspondant, en termes du schéma partiel S_p, exprimée dans le langage de requêtes L_p sur l'alphabet A_p. Chaque sous-requête reçue par un médiateur spécialisé est à son tour divisée en sous-requêtes qui seront envoyées aux adaptateurs de sources sous-jacents pour être réalisée sur les sources et les résultats sont envoyés aux médiateurs spécialisés. Les résultats partiels obtenus par chaque adaptateur spécialisé sont recomposés par le médiateur global. Le

mapping *Mg,p*, associe à chaque élément g de G une sous-requête q_p sur S_p de la forme:

$$g \longrightarrow q_p$$

D'un point de vue modélisation, à ce niveau spécialisé (GAV) de l'approche HAV, le contenu de chaque élément g du schéma global sera caractérisé en termes de la **vue q_p** sur les schémas partiels. La sous-requête q_p arrive à un médiateur spécialisé de schéma S_p conçu selon LAV. Le mapping $M_{Sp,S}$ associe dans ce cas à chaque élément s de la source S une requête q_{Sp} sur S_p. Ainsi le mapping $M_{Sp,S}$, est un ensemble d'assertions, pour chaque source s de S de la forme :

$$s \longrightarrow q_{Sp}$$

Du point de vue modélisation, à ce niveau source (LAV) de l'approche HAV, le contenu de chaque source S sera caractérisé en termes de la **vue q_{Sp}** sur le schéma partiel correspondant.

Les requêtes posées à un système d'intégration de données I conçu selon HAV, sont exprimées en termes de relations dans le schéma global de I. Etant donné une base de données partielle *P (dans I_p)* pour *I*, la réponse $q^{I,P}$ à une requête q de I par rapport à P , est l'ensemble des tuples t des relations (donnés par les réponses sur P) dans P tel que $t \in q^B$ pour chaque base de données globale *B* légale pour I par rapport à P. L'ensemble $q^{I,P}$ est appelé l'ensemble des *réponses certaines* de q pour I par rapport à P. Etant donné une base de données source D pour I_p, la réponse (réponse partielle) $q_p^{Ip,D}$, à une requête q_p pour I_p par rapport à D, est l'ensemble des tuples t d'objets dans D tel que $t \in q_p^p$ pour chaque base de données partielle P légale pour I_p par rapport à D. L'ensemble $q_p^{Ip,D}$ est appelé l'ensemble des réponses certaines de q_p pour I_p par rapport à D.
Dans notre contexte, nous appelons base de données partielle, une base de données pour le schéma partiel S_p; base de données globale, une base de données pour G satisfaisant les contraintes de G ; et base de données source, une base de données pour le schéma source S.

Afin de préciser la sémantique du système d'intégration de données conformément à HAV, nous devons spécifier quelles données satisfont le schéma global. Tout d'abord, nous commençons avec un ensemble de données sur les sources et de préciser lesquels des données satisfont le schéma partiel correspondant. Nous considérons pour I_p, la base de données D pour le schéma source S. Basé sur D, nous avons à spécifier quel est le contenu du schéma partiel correspondant. Deuxièmement, nous considérons un ensemble de données de la base de données partielle (obtenu par l'exécution de LAV sur les sources) et de préciser lesquels des données satisfont le schéma global. Soit une base de données partielle P pour le schéma partiel S_p. Basé sur P ; nous devons spécifier quel est le contenu du schéma global.

Toute base de G est appelée une base de données globale pour I. Une base de données B pour I est dite 'être légale' par rapport à D via P si elle vérifie la propriété suivante:

- B est cohérente avec G, c'est à dire toute contrainte du schéma global G est satisfaite par B.
  - P est cohérente avec le schéma partiel S_p , c'est à dire toute contrainte dans le schéma partiel S_p est satisfaite par P.
  - B satisfait le mapping par rapport à P.
  - P satisfait le mapping par rapport à D.

Définition 2: La sémantique de I par rapport à D, notée *Sem* (I,D*)*, définie sur une base de donnée source D pour I est définie comme :

Sem (I,D) = {B | B est une base de données globale légale pour I par rapport à P et P est une base de données globale légale pour I_p par rapport à D}

Lors de l'évaluation d'une requête, les réponses fournies peuvent être plus ou moins complètes. A cet effet, différentes hypothèses qui affectent la notion de satisfaction du mapping, peuvent être faites. En particulier, si nous supposons que le mapping est 'Sound' (juste mais partiel), les données fournies par les sources sont un sous-ensemble des données globales - l'extension de q_s est contenue dans l'extension de q_G. Inversement, si le mapping est considéré comme 'complet', les données fournies par les sources sont un sur-ensemble des données globales - l'extension de q_s contient l'extension de q_G. Enfin, nous supposons que le mapping est 'exact', quand il est à la fois

sound et complet. Nous rappelons que, en raison des caractéristiques générales des sources qui sont distribuées, autonomes et indépendantes, l'hypothèse du mapping 'sound' est plus raisonnable dans un environnement d'intégration de données HAV.

5.2 Définition formelle du mapping dans HAV

L'intégration des données est le processus de combiner plusieurs sources de données de telle sorte qu'elles puissent être interrogées et mises à jour via une interface commune. Cela nécessite le mapping de chaque source de données dans le schéma global de l'interface commune. Comme indiqué précédemment, l'approche HAV combine l'approche globale et l'approche locale. Dans cette approche, les schémas spécialisés sont définis indépendamment des schémas des sources locales. Chaque source est décrite comme une vue matérialisée sur le schéma spécialisé de même modèle de données. Le schéma global est défini en termes de schémas spécialisés : autrement dit, le schéma global est défini comme une vue sur les schémas spécialisés.

Définition 3 : Compte tenu de la définition 1, les mappings $M_{Sp,S}$ *et* Mg,p dans l'approche HAV sont sous la forme:

$M_{Sp,S}$: $S_{Si}(X) \longleftarrow S_1(X_1), S_2(X_2), \ldots S_k(X_J, Z_J)$
Où: $X = \mathrm{U}_i X_i$
S_i *sont les relations des schemas partiels*
S_{Si} *sont les relations locales.*

$M_{g,p}$: $Gi(X) \longleftarrow S_1(X_1), S_2(X_2), \ldots S_k(X_k)$
Où: $X = \mathrm{U}_i X_{i,}$
Gi sont les relations globales
S_i *sont les relations des schemas partiels.*

Le mapping $M_{Sp,S}$ est constitué par les correspondances LAV, associant (dans le cas du modèle relationnel) à chaque relation source une requête conjonctive sur le schéma partiel correspondant.

Le mapping $M_{g,p}$ est constitué par les correspondances GAV, en associant à chaque relation globale une requête conjonctive sur le schéma partiel.

Soit D une base de données source. Une base de données partielle P satisfait une assertion $q_s \subseteq q_{Sp}$ dans $M_{Sp,S}$ par rapport à D, si $q^d{}_s \subseteq q^p{}_{Sp}$. La base de données partielle P est dite 'légale' pour I_p par rapport à D, si elle satisfait toutes les assertions du mapping $M_{Sp,S}$, par rapport à D. Une base de données globale B satisfait une assertion $q_{Sp} \subseteq q_G$ dans Mg,p par rapport à P , si $q_{Sp}{}^P \subseteq q^B{}_G$. La base de données globale B est dite 'légale' pour I par rapport à P, si elle satisfait toutes les assertions du mapping Mg,p par rapport à P.

6. Conclusion

Ce chapitre a été consacré à la présentation de l'approche d'intégration proposée appelée hybride As View (HAV).

Après avoir introduit ce chapitre, nous avons présenté les approches qui sont à la base de notre proposition à savoir GAV et LAV. Un bilan relatif à ces approches a été dressé pour mieux justifier la combinaison de ces dernières. Nous nous sommes focalisés par la suite sur le processus d'élaboration du schéma global ainsi que sur l'acquisition des données à partir des sources. Nous avons par la suite présenté l'architecture à trois niveaux du système qui est conforme à l'approche HAV décrite dans cette thèse, et où les médiateurs spécialisés ont un rôle central. Pour soutenir les caractéristiques de HAV, nous avons dégagé un tableau comparatif des trois types d'architecture supportant GAV, LAV et HAV en mettant l'accent sur les apports de HAV. Nous avons terminé ce chapitre par la présentation du cadre formel relatif à cette approche. Cette formalisation introduit une couche intermédiaire de sources virtuelles entre le schéma global et les sources de données. Il reste à valider cette approche, ce qui sera présenté dans le chapitre suivant.

Chapitre 5: Validation de l'approche HAV

1. Introduction

Nous considérons dans ce chapitre, différentes évaluations et tests permettant de valider nos travaux. Il sera nécessaire de considérer une validation qualitative et une autre quantitative. En effet, pour valider notre approche HAV, nous procédons en deux étapes: dans la première, il faut montrer que l'architecture supportant HAV est en mesure de répondre aux différentes requêtes qui pourraient lui être posées en s'appuyant sur une étude de cas. Ainsi, des requêtes ont été effectuées afin de vérifier les comportements spécifiques des médiateurs. Cette validation correspond à la validation qualitative. Dans la deuxième étape, il s'agit de mesurer les performances de l'architecture en question dans un contexte donné. Pour cela, des expériences ont été effectuées pour évaluer les performances de l'architecture supportant HAV. Cette validation correspond à la validation quantitative.

2. Validation qualitative

Il s'agit de valider qualitativement notre approche en se rapportant à une étude de cas.

2.1 Etude de cas

Un cas d'utilisation est illustré par l'exemple suivant:

Nous considérons deux sources relationnelles S1 et S2 et deux autres semi-structurés S3 et S4. Les sources relationnelles S1 et S2 sont intégrées comme des vues locales sur le schéma partiel relationnel PS1. Les sources S3 et S4 sont intégrées pour donner le schéma partiel PS2 en XML. Cela correspond à l'intégration avec l'approche LAV. Le modèle pivot de représentation du schéma global est relationnel. Ce schéma est défini comme une vue globale sur les schémas partiels PS1 et PS2. Cela correspond à l'intégration avec l'approche GAV. Les sources à intégrer contiennent des informations sur les films. Le schéma partiel PS1 contient des films depuis 1960 et leurs critiques depuis 1990. PS2 contient des films. Le schéma global se compose de deux tables. L'une contient des informations sur les films et l'autre contient des articles relatifs aux films.

La conception du mapping est l'une des tâches essentielles dans la définition de la spécification du système l'intégration des données, nous présentons le mapping correspondant à notre étude de cas. Pour cela, il sera question d'un mapping **MSp,S** selon LAV du niveau inférieur c-à-dire du niveau source vers le niveau spécialisé et d'un deuxième mapping **Mg,p** selon GAV du niveau supérieur c-à-dire du niveau global vers le niveau spécialisé.

2.1.1 Le mapping MSp,S: .(Mapping LAV, sources-schémas partiels)

• Nous supposons que le schéma partiel (PS1) se compose de deux relations:

Film (Fid, title, productor, year)
Critiques (Fid, critique)

Nous intégrons les deux sources S1 et S2 sur le schéma partiel schéma1. La description de ces sources est la suivante:

Pour S1:
Film (Fid, title, year, productor) ← Film (Fid, title, productor, year)
Film.year>1960

Pour S2:
Critique (Fid, title, critique) ← Film (Fid, title, productor, year)
Critiques (Fid, critique)
Film.year>1990,
Film.Fid= Critiques.Fid

Règles de mapping LAV:

- Nous obtenons S1 en créant la vue suivante:

```
Create View Film(Fid, title, year, productor) as
Select F.Fid, F.title, F.productor, F.year
From Film F
Where F.year>1960
```

- Nous obtenons S2 en créant la vue suivante:

```
Create View Critique(Fid, title, critique)  as
Select F.Fid, F.title, C.critique
From Film F,Critiques C
Where F. year> 1990 And F.Fid=C.Fid
```

De la description ci-dessus, nous pouvons conclure que la relation Film de S1 contient l'identifiant du film, le titre du film, le producteur du film et l'année, seulement pour les films depuis 1960. Alors que la critique par rapport à S2 contient l'identifiant du film, le titre du film et la critique du film depuis 1990.

• Nous supposons que le schéma partiel PS2 est :

<! ELEMENT Film (Fid, title, director, kind) >

Nous intégrons les deux sources S3 et S4 sur le schéma partiel schéma2. La description de ces sources est la suivante:

Pour S3: <! ELEMENT Film(Fid, title, Productor) > ←<! ELEMENT PS2.Film (Fid, title, director) >

Pour S4: <! ELEMENT Film (Fid, kind) ← <! ELEMENT PS2.Film (Fid, kind) >

2.1.2 Le mapping: *Mg,p* **: (Mapping GAV, schéma global- schémas partiels)**

Comme nous l'avons vu ci-dessus, le schéma partiel PS1 se compose de deux relations Film et Critique. Nous supposons que, après la traduction de PS2 dans le modèle relationnel le schéma relationnel de PS2 se compose d'une relation: Film (Fid, titre, réalisateur, genre). Nous intégrons PS1 et PS2 sur le schéma global. La description du schéma global est la suivante:

Films (Fid, title, realisator, year, kind) ← PS1.Film (Fid, title, realisator, year, NULL)
PS2.Film (Fid, title, director, NULL, kind)

Articles (title, critique) ← PS1.Film (title, Null)
PS1.Critiques (title, critique)

Règles de mapping GAV :

Nous obtenons le schéma global en créant la vue suivante:

```
Create View Films (Fid, title, realisator, year, kind) as
Select Fid, title, realisator, year, NULL From PS1.Film
UNION
Select Fid, title, director, NULL, kind From PS2.Film
Create View Articles as
Select Film.title as title, Critiques.critique as critique
From PS1.Film, PS1.Critiques
Where Film.Fid= Critiques.Fid
```

2.1.3 Traitement de requêtes dans HAV

Afin de vérifier le comportement du médiateur global, Il est nécessaire d'effectuer des requêtes sur le schéma global.

Nous donnons un exemple de requêtes sur le schéma global défini ci-dessus**:**

Quelles sont les identifiants et les critiques de films ayant pour titre: «freedom»?

Nous lançons la requête globale suivante. Elle est formulée dans les termes du schéma global (du médiateur global) :

Requête globale :

```
Select Films.Fid, Articles.critique
From Films, Articles
Where Films.title = ` Freedom' And
Films.Fid = Articles.Fid
```

Les étapes du traitement de cette requête sont :

a) Traduction de la requête globale en sous-requêtes sur PS1 et PS2 (GAV):

```
Select PS1.Film.Fid, PS1.Critiques.critique, PS2.Film.Fid
From PS1.Film, PS1.Critiques, PS2.Film
 Where (PS1.Film.title='Freedom' And PS1.Film.Fid=PS1.Critiques.Fid) Or
          (PS2.Film.title='freedom'    And PS2.Film.Fid=PS1.Critiques.Fid)
```

b) Traduction des sous-requêtes sur les sources S1, S2, S3 et S4 (LAV):

```
Select S1.Film.Fid, S2.Critique.critique
From S1.Film, S2.Critique, S2.Film
Where (S1.Film.title='freedom' And  S1.Film.Fid=S2.Critique.Fid) Or
        (S2.Critique.title='freedom' And S1.Film.Fid=S2.Critiques.Fid)
```

Union (la requête XML sur S3 et S4 correspondante à la requête relationnelle suivante) :

```
Select S3.Film.Fid, S4.Film.critique
From S3.Film, S4.Film
Where S3.Film.title='freedom' And   S3.Film.Fid=S4.Film.Fid
```

c) Les données sur les sources sont récupérées :

1) Sur S1, nous récupérons les identifiants des films ayant pour titre 'freedom' (dont le Fid est égal à un Fid dans S2) et sa critique dans S2,

2) Sur S3, nous récupérons les identifiants des films ayant pour titre 'freedom' (dont le Fid est égal à un Fid dans S4) et sa critique dans S4,

3) Les résultats obtenus en 1) sont envoyés au médiateur spécialisé relationnel, et Les résultats obtenus en 2) sont envoyés au médiateur spécialisé XML,

4) L'ensemble des résultats obtenus en 3) à savoir les identifiants des films ayant pour titre 'freedom' ainsi que leurs critiques, sont renvoyés au médiateur global.

2.1.4 Ajout d'une nouvelle source

Comme il a été présenté dans le chapitre précédent, notre approche permet l'extensibilité des sources sans modification du schéma global, en effet :

Considérons une nouvelle source relationnelle S5 qui vient s'ajouter aux autres sources. Tout comme S1, S2, S3 et S4 S5 doit être définie comme une vue sur le schéma partiel relationnel PS1 dont nous rappelons le schéma :

Film (Fid, title, productor, year)
Critiques (Fid, critique)

Nous supposons que S5 contienne une relation Film dont le schéma est Film (idFilm, title, critique) ce qui signifie que la relation Film de S5 contient l'identifiant du film, le titre du film, et la critique du film.

Nous obtenons S5 en créant la vue suivante sur PS1

```
Create View Film( idFilm, title, critique) as
Select F.Fid, F.title, C.critique
From Film F, Critiques C
Where F.Fid=C.Fid
```

Il est clair que l'ajout de cette nouvelle source S5 n'a affecté en rien le schéma global ni même le schéma partiel PS1 car elle a été créée comme une vue sur le schéma global.

Supposons que la même requête précédente est lancée sur le schéma global, à savoir: Quels sont les identifiants et les critiques des films ayant pour titre: «freedom»? Rien ne change pour le schéma global, c-à-dire que nous aurons exactement la même requête:

```
Select Films.Fid, Articles.critique
From Films, Article
Where Films.title = ` Freedom' And
        Films.Fid = Articles.Fid
```

La traduction de la requête sur PS1 et PS2 ne changera pas également, c-à-dire nous aurons :

```
Select PS1.Film.Fid, PS1.Critiques.critique,
        PS2.Film.Fid
From PS1.Film, PS1.Critiques, PS2.Film
 Where (PS1.Film.title='Freedom' And PS1.Film.Fid=PS1.Critiques.Fid Or
           PS2.Film.title='freedom'    And PS2.Film.Fid=PS1.Critiques.Fid)
```

Pour la traduction de la requête sur les sources S1, S2, S3, S4 et S5, seule la sous-requête lancée sur PS1 va changer, celle de PS2 est inchangée. Nous aurons donc:

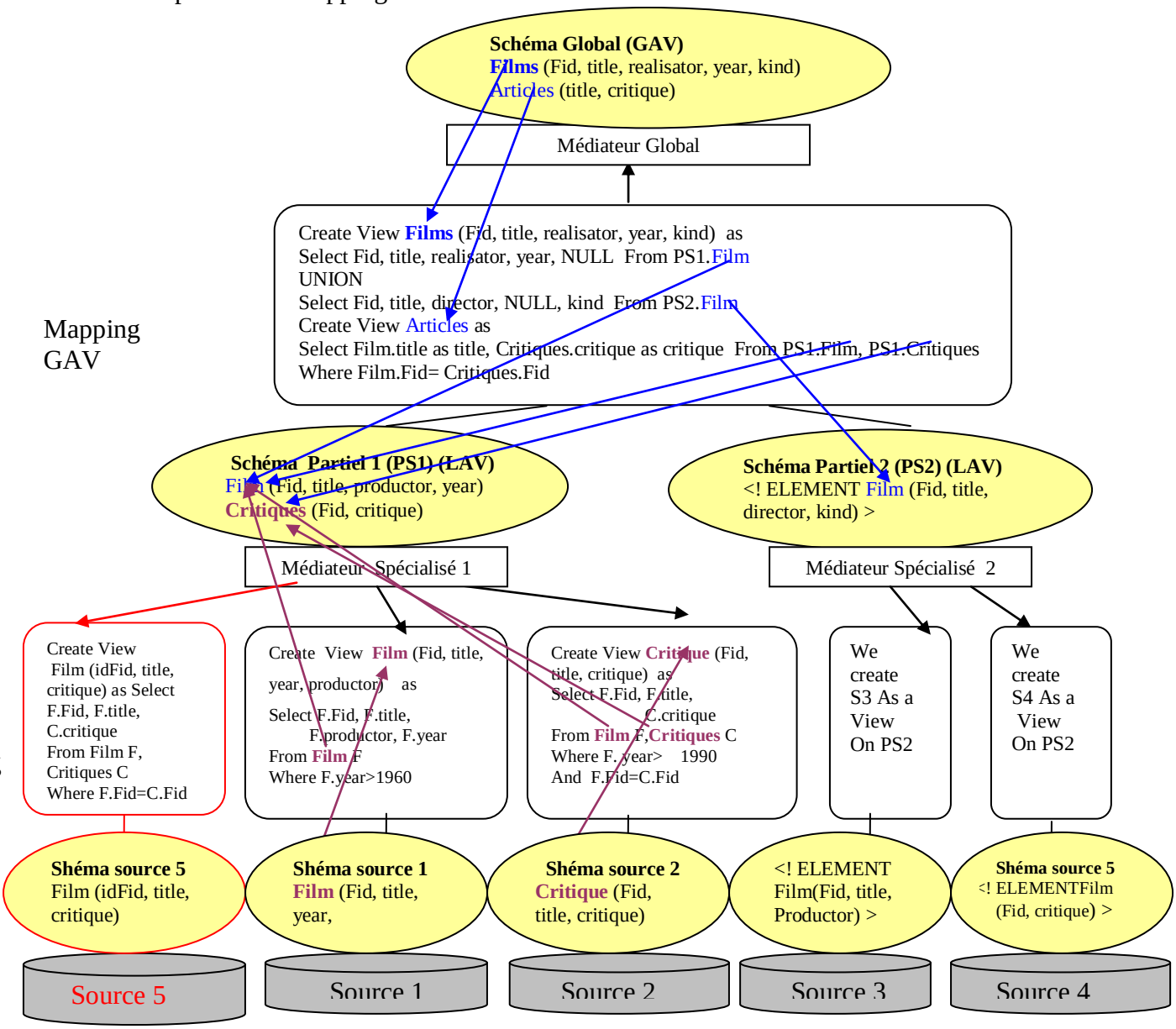
```
Select S1.Film.Fid, S2.Critique.critique
From S1.Film, S2.Critique, S2.Film
Where (S1.Film.title='freedom' And  S1.Film.Fid=S2.Critique.Fid= Or
        (S2.Critique.title='freedom' And S1.Film.Fid=S2.Critiques.Fid=  Or
        (S5.Critique.title='freedom'   And S5.Critique.idfilm= S1.Film.id) Or
        (S5.Critique.title='freedom'   And S5.Critique.idfilm= S2.Film.id)
Union
Select S3.Film.Fid, S4.Film.critique
From S3.Film, S4.Film
Where S3.Film.title='freedom' And S3.Film.Fid=S4.Film.Fid
```

2.1.5 Résumé

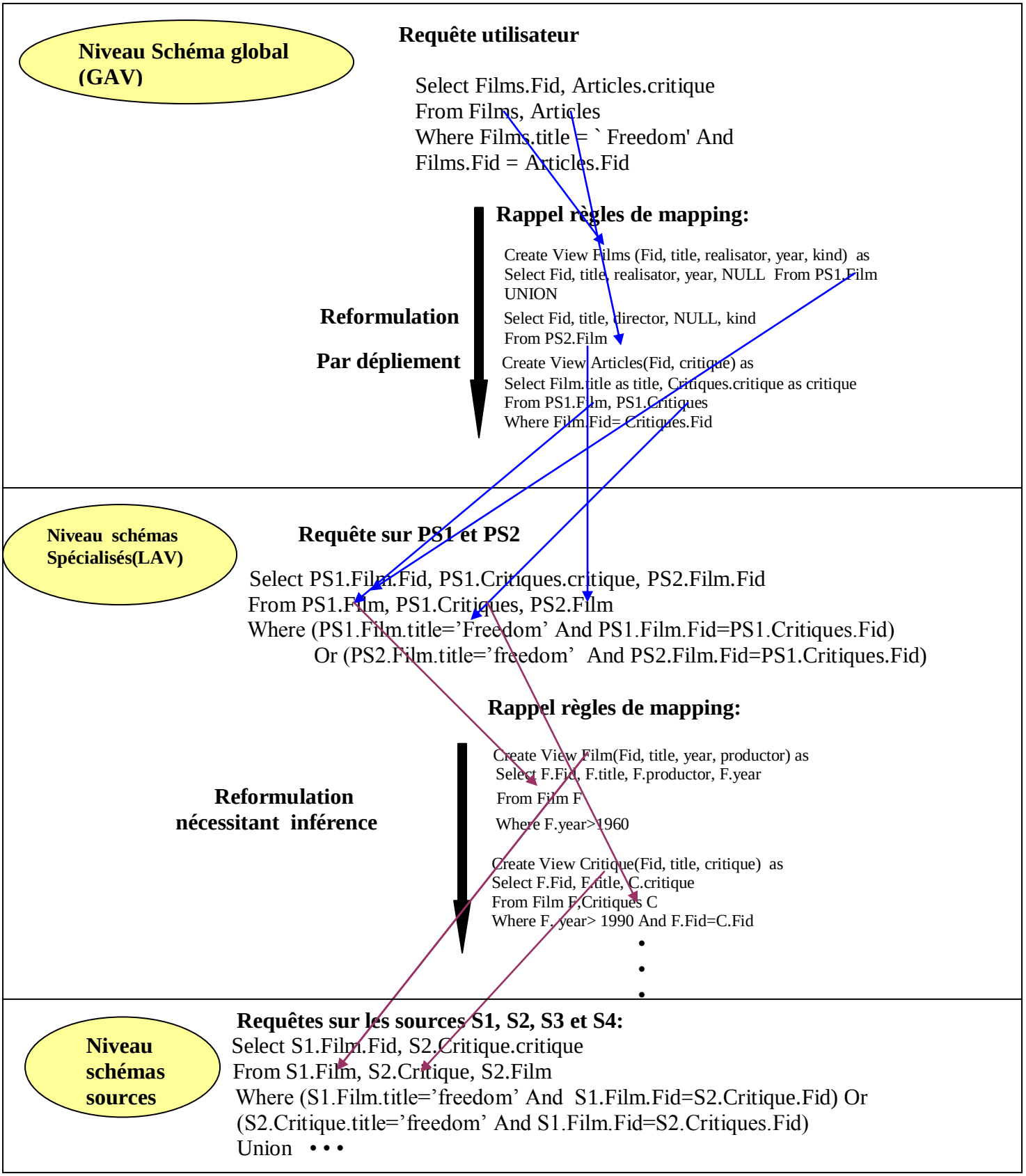
- Nous récapitulons le mapping de l'étude de cas sur le schéma suivant:

Mapping GAV

Mapping LAV



- Nous récapitulons le traitement de requêtes de l'étude de cas sur le schéma suivant:



3. La Validation quantitative

Les expérimentations quantitatives sont sans doute le meilleur moyen pour valider les performances d'un système spécifique. Dans notre cas, la validation quantitative consiste à mesurer les performances du médiateur global dans un contexte donné.

Pour valider quantitativement notre système, il serait important de s'appuyer sur un système hétérogène expérimental avec des données conçues par un générateur de banc d'essai. L'idéal dans notre travail serait de mesurer les performances du système défini sur notre approche HAV dans sa totalité et celles des systèmes définis sur les autres approches et pouvoir ainsi les comparer. Etant donné que nous ne pourrons pas réaliser ces expériences (temps d'exécution, espace mémoire…), nous nous sommes appuyés sur une validation quantitative partielle, c-à-dire que nous n'avons pas validé le système basé sur HAV dans sa totalité, mais nous avons validé certains aspects de ce système. Pour cela, nous avons montré que :

1) le surcoût induit par l'empilement des médiateurs n'est pas pénalisant,
2) la matérialisation d'un entrepôt de données sur un schéma global tel que c'est le cas pour notre architecture, ne nécessite pas une matérialisation intermédiaire comme dans la matérialisation d'un entrepôt avec un ETL,
3) l'intégration des sources de données de natures différentes (relationnel, xml, csv, xls…etc) dans un modèle global, par exemple le relationnel, nécessite un temps de traitement supplémentaire par rapport à une intégration des données où tout est relationnel aussi bien les sources de données que le schéma global. Dans ce cas le but est de prouver que, dans le cas de plusieurs sources de données hétérogènes à intégrer, un temps de conversion de modèles (réalisé par un adaptateur) est nécessaire pour tout remettre en relationnel.

3.1 Surcoût induit par l'architecture de médiateurs empilés

Une de nos premières préoccupations quant à la validation quantitative d'un système construit selon HAV comme c'est mentionné plus haut, était l'empilement des médiateurs c-à-dire que nous nous sommes demandés si le fait d'avoir plusieurs niveaux de médiateurs n'allait pas nuire à l'évaluation d'une requête globale et sur les performances d'un tel système. Pour répondre à cette question nous avons utilisé le résultat d'une expérimentation menée au sein du laboratoire PRISM [NGOG 2003] pour la classification des différents cas d'intégration de sources. Lors de cette expérimentation, l'auteur a voulu montrer le surcoût induit par les temps de communication entre les différents médiateurs, en lançant une même requête sur une architecture Mo ayant un accès direct aux données puis sur une architecture M1 ayant une arborescence de médiateurs (tel que c'est le cas dans notre architecture supportant HAV). Les résultats obtenus ont montré que le temps mesuré en millisecondes en fonction du nombre de nœuds résultat est représenté par l'auteur pour ces deux architectures Mo et M1 sur la figure 5.2:

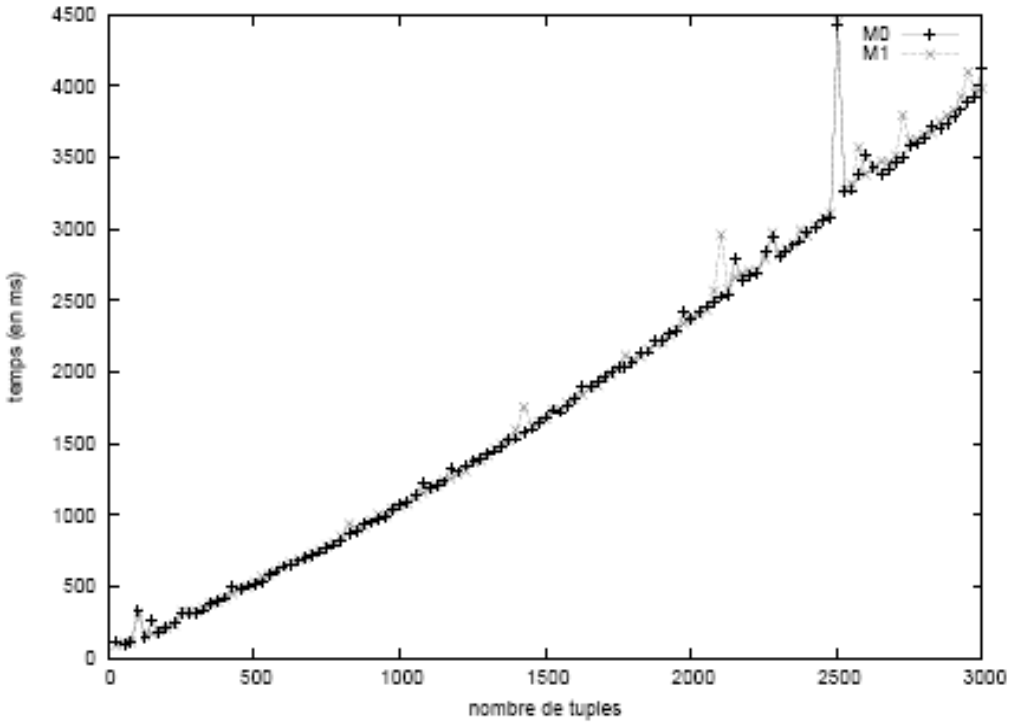


Figure 5.1 : Temps de réponse suivant l'architecture Mo et M1 [NGOG 2003]

Constatant que les courbes sont très proches, l'auteur a donc conclu qu'empiler des médiateurs ne nuit pas à l'évaluation. Dans cette expérience, Mo et M1 se trouvent sur la même machine. D'après le résultat de l'expérience, il a donc déduit que si les médiateurs se trouvaient sur des machines distinctes, seul le temps de communication réseau serait à prendre dans le surplus de temps.
Ce résultat est très important pour nous car il nous permet de répondre positivement à notre première préoccupation.

3.2 Matérialisation d'un entrepôt de données

Dans notre travail nous avons proposé une approche d'intégration des sources de données qui nous permet de construire le schéma global (intégré) sur lequel se fera la matérialisation de l'entrepôt. L'architecture la plus répandue de nos jours consiste en la mise en place d'un entrepôt de données et à l'acquisition d'un ETL, afin d'assurer les traitements d'alimentation. Cette architecture est aujourd'hui démocratisée dans le monde de l'entreprise et chez les éditeurs d'outils d'intégration.

Le but de cette validation est de montrer l'apport en termes de performance d'une architecture d'un entrepôt dont la matérialisation se fait directement des sources vers l'entrepôt conformément au schéma de ce dernier tel que c'est le cas dans notre architecture *si on fait abstraction du niveau intermédiaire de notre architecture*, en la confrontant à l'architecture d'intégration d'un ETL qui quand à elle prend en charge les différentes sources de données et les convertit dans un modèle intermédiaire indépendant en faisant abstraction aussi bien des modèles

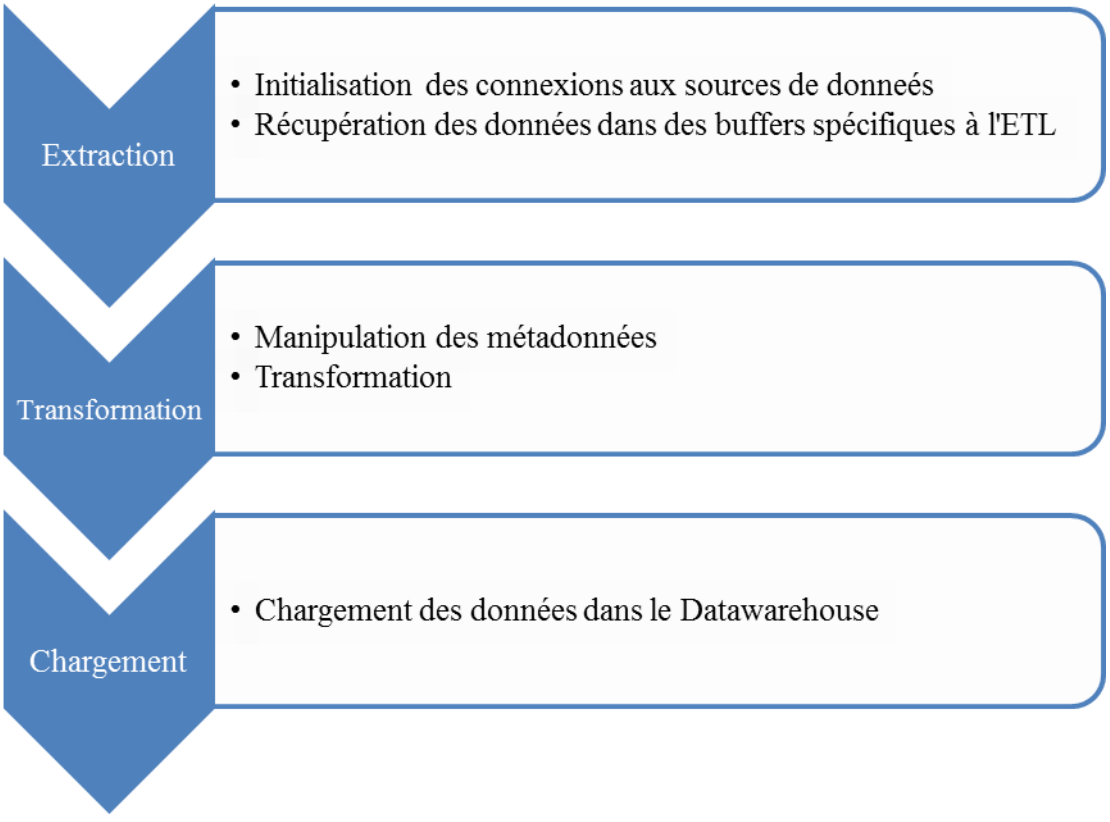
des sources de données que celui de l'entrepôt de données avant de les charger dans ce dernier conformément à son schéma .

Entre les deux approches, le schéma d'intégration garde les mêmes grandes lignes mais diffère essentiellement au niveau de l'étape de matérialisation des données. En effet, une architecture qui utilise un ETL est basée essentiellement sur la mise en place d'un flux de données provenant des différentes sources et qui sera injecté dans un buffer de modèle interne intermédiaire avant d'être chargé finalement dans l'entrepôt. Cette méthode de matérialisation des données constitue la différence majeure avec l'architecture qui propose de remplacer l'ETL par une matérialisation sur le schéma global jouant le rôle d'une couche intermédiaire entre les sources de données et le Datawarehouse.

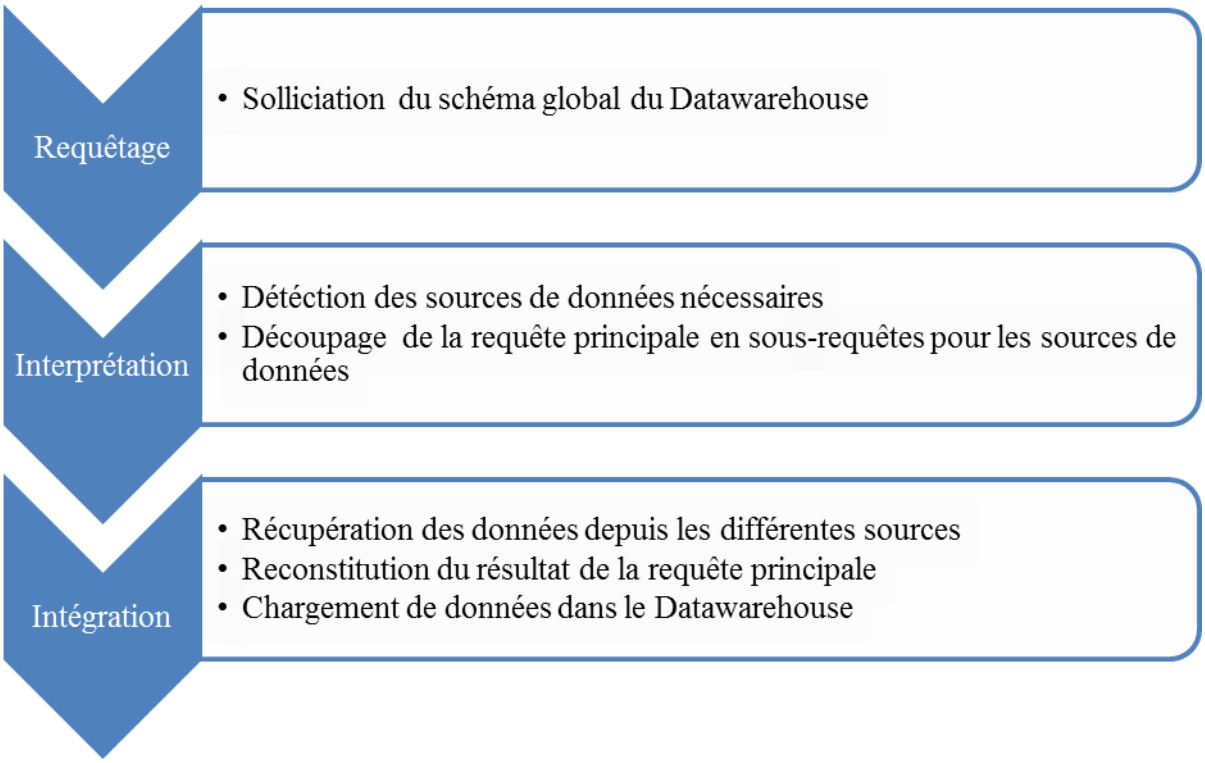
Le choix d'une telle comparaison devrait nous permettre de mettre en évidence l'apport de la matérialisation et l'alimentation d'un entrepôt conformément à son schéma d'intégration par rapport à une architecture réelle d'un ETL qui, elle, favorise l'interopérabilité entre les différentes sources de données à une optimisation du processus d'intégration qui tire profit de la puissance des modèles en amont.

Entre une intégration via un ETL et la mise en place d'un schéma global, l'approche d'intégration n'est pas la même. Ceci, dans la mesure où, notre architecture se base sur un processus inverse en termes de déroulement.

a. Description du processus d'intégration de données via un ETL :



b. Description du processus d'intégration de données basant sur le schéma global:



Lors du Processus d'intégration via un ETL, c-à-dire lors d'une phase d'extraction, de transformation et de chargement de données, les outils d'intégration existants sur le marché, notamment IBM Datastage, Informatica, Oracle Data Integrator, Microsoft SSIS, etc…, ont tous pour vocation de rendre possible la récupération des données depuis différentes sources hétérogènes, pouvoir les transformer dans leurs modèles internes et les injecter dans de différentes typologies de destination (SGBD Relationnel, fichier plat, xml, Webservices…Etc). Ils sont tous distincts, de par leurs architectures et leur philosophie de manipulation de données mais ils ont la même logique de gestion d'hétérogénéité de données. En effet, les multiples outils disposent tous de connecteurs vers les différentes typologies de sources de données. La logique reste la même et consiste à transformer l'information sous un modèle homogène intermédiaire qui permet de croiser les informations issues des différentes sources.

Ces connecteurs sont souvent, en ce qui concerne les bases de données relationnelles, des pilotes ODBC de bas niveau ou des surcouches OLE DB plus riches en fonctionnalités. Sinon, pour les fichiers plats, et les modèle non-structurés, c'est souvent des parseurs qui permettent de lire ou d'écrire les données.

Face à cette problématique d'hétérogénéité, la plupart des outils d'intégration de données du marché disposent de leur propre modèle de manipulation de données, notamment des Buffers de données tel que c'est le cas pour Microsoft SSIS. Cette étape de transformation de données

nécessite un travail préalable de la donnée avant toute manipulation. C'est pourquoi, afin d'optimiser ce processus d'intégration, une logique d'intégration des sources de données suivant le modèle de destination (celui du schéma global) (modèle relationnel dans la plupart des cas) nous fait gagner l'étape de passage par le modèle interne d'un ETL et nous laisse la possibilité de profiter de la puissance de requêtage du SGBD sur les sources de données.

3.3 Validation par comparaison

Comme nous l'avons mentionné plus haut, nous n'avons pas validé quantitativement un système basé sur HAV dans sa globalité, mais nous l'avons fait d'une manière partielle, et ce, en faisant abstraction du niveau spécialisé, c-à-dire que nous avons réalisé une implémentation dans le cas d'une architecture où le médiateur global a un accès direct aux données comme nous pouvons le voir dans ce qui suit..

Nous allons procéder à l'implémentation du processus d'intégration via un ETL, puis celle du processus d'intégration se basant sur le schéma global, pour mettre en évidence l'apport de la deuxième architecture d'intégration.

3.3.1 Méthode d'implémentation:

L'implémentation consistera en une simulation des processus avec le langage C# sur l'exemple de l'étude de cas en faisant abstraction du niveau spécialisé, c-à-dire que nous allons faire l'implémentation dans le cas d'une architecture où le médiateur global a un accès direct aux sources de données via les adaptateurs.

Le langage C# est le langage orienté objet de la plateforme .Net de Microsoft. Cette plateforme contient un nombre important de classes prédéfinies qui facilitent aux développeurs l'implémentation de leurs programmes.

a. Traitement ETL:

Les étapes dans le processus d'intégration dans une architecture ETL sont :

– Chargement de données dans le datawarehouse avec un taux de rafraichissement (quotidien, chaque heure, quasi-temps-réel) qui consiste à :

- Utiliser des connecteurs pour charger les données dans les Buffers de l'ETL,

- Démarrer l'intégration des données dans les Buffers indépendamment du modèle de données cible,

- Utiliser des connecteurs pour charger les données dans le datawarehouse conformément au schéma de l'entrepôt.

● **Implémentation du processus ETL:**

Le schéma de la figure 5.2 représente un flux de données réalisé avec l'ETL de Microsoft qui utilise l'exemple de l'étude. La suite des traitements réalisés par l'ETL sont:

1 – La lecture de deux sources relationnelles et deux sources XML,
2 – L'application de deux jointures, d'une union et d'un filtre des données,
3 – L'insertion dans la base de données de destination.

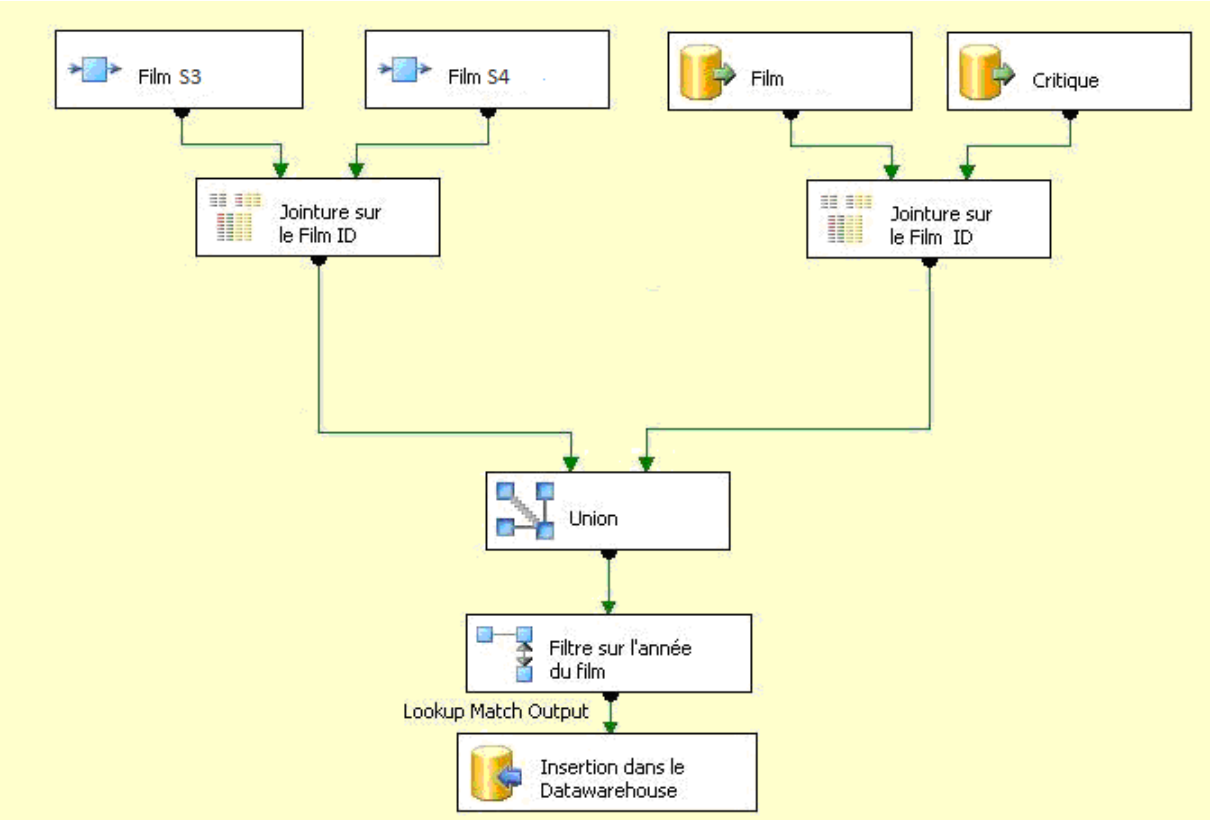


Figure 5.2 : Prise d'écran de l'implémentation de l'exemple dans l'ETL SQL Server Integration Services 2008

Afin d'illustrer le comportement en détail de cet ETL, nous présentons une implémentation de l'exemple en langage C# qui simule les différentes étapes de traitement.

Pour réaliser cette implémentation, nous utiliserons le langage C# qui est un langage de programmation orienté objet créé par la société Microsoft. Il a été créé afin que la plateforme.Net soit dotée d'un langage permettant d'utiliser toutes ses capacités. Il est très proche de Java dont il reprend la syntaxe générale.

Le code ci-dessous représente une simulation du traitement de l'intégration avec un ETL : Implémentation en langage C# des processus de, connexion, mise en Buffer et chargement de l'exemple utilisé dans l'étude de cas.

Implémentation en langage C#:

```
static void Main(string[] args)
      {
         #region 'Lecture des données à partir de la DB'

         // 1- Initialisation de la chaine de connexion à la base de données
         string _ConnectionString = "Initial Catalog=Cinema;Data Source=localhost;Integrated
Security=SSPI;";

         //2- Ouverture de la connexion
         SqlConnection myConnection = new SqlConnection(_ConnectionString);
         myConnection.Open();

         //3- La commande SQL pour récupérer les données
         SqlCommand commandFilm = myConnection.CreateCommand();
         commandFilm.CommandText = "SELECT Fid, title, year, productor FROM film";

         //4- Connecteur à la DB
         SqlDataAdapter adapterFilm = new SqlDataAdapter();
         adapterFilm.SelectCommand = commandFilm;

         //5- Lecture des MetaDonnees pour lire les données de la DB
         List<MetaDonnee> _ListMetaDataFilm = GetMetaData(adapterFilm);

         DataTable _BufferFilms = new DataTable();

         //6- Construction de la première DataTable qui contient les données récupérées de la DB –
Boucle sur les colonnes
         foreach (MetaDonnee currentMetaData in _ListMetaDataFilm)
         {
            DataColumn column = new DataColumn();
            column.ColumnName = currentMetaData.ColumnName;
            column.DataType = currentMetaData.DataType;
            column.DefaultValue = currentMetaData.DefaultValue;
            column.MaxLength = currentMetaData.MaxLength;
            _BufferFilm.Columns.Add(column);
         }

         //7- Chargement des données dans la première DataTable
         adapterFilm.Fill(_BufferFilms);
```

```
        //8- La commande SQL pour récupérer les données
        SqlCommand commandCritique = myConnection.CreateCommand();
        commandCritique.CommandText = "SELECT Fid, title, critique FROM critique";


        //9- Connecteur à la DB
        SqlDataAdapter adapterCritique = new SqlDataAdapter();
        adapterCritique.SelectCommand = commandCritique;

        //10- Lecture des Métadonnées pour lire les données de la DB
        List<MetaDonnee> _ListMetaDataCritique = GetMetaData(adapterCritique);

        DataTable _BufferCritique = new DataTable();

        //11- Construction de la deuxième DataTable qui contient les données récupérées de la
DB – Boucle sur les colonnes
        foreach (MetaDonnee currentMetaData in _ListMetaDataCritique)
        {
           DataColumn column = new DataColumn();
           column.ColumnName = currentMetaData.ColumnName;
           column.DataType = currentMetaData.DataType;
           column.DefaultValue = currentMetaData.DefaultValue;
           column.MaxLength = currentMetaData.MaxLength;
           _BufferCritique.Columns.Add(column);
        }

        //12- Chargement des données dans la deuxième DataTable
        adapterCritique.Fill(_BufferCritique);

        #endregion

        #region 'Lecture des données à partir du fichier XML'

        //13- Lecture des Métadonnées pour lire les données du fichier de la DTD du fichier XML
        List<MetaDonnee> _MetaDataFirstXML = GetMetaData("DTDXmlFirstFile");
        DataTable _BufferFromFirstXmlFile = new DataTable();

        //14- Construction de la première DataTable qui contient les données récupérées du
fichier XML – Boucle sur les colonnes
        foreach (MetaDonnee currentMetaData in _MetaDataFirstXML)
        {
           DataColumn column = new DataColumn();
           column.ColumnName = currentMetaData.ColumnName;
           column.DataType = currentMetaData.DataType;
           column.DefaultValue = currentMetaData.DefaultValue;
           column.MaxLength = currentMetaData.MaxLength;
```

```
            _BufferFromFirstXmlFile.Columns.Add(column);
        }

        //15- Chargement des données du fichier dans la DataTable
        _BufferFromFirstXmlFile.ReadXmlSchema("XlmSchemaFirstFile");

        //16- Lecture des Métadonnées pour lire les données du fichier de la DTD du fichier XML

        List<MetaDonnee> _MetaDataSecondXML = GetMetaData("DTDXmlFirstFile");
        DataTable _BufferFromSecondXmlFile = new DataTable();

        //17- Construction de la deuxième DataTable qui contient les données récupérées du
fichier XML – Boucle sur les colonnes
        foreach (MetaDonnee currentMetaData in _MetaDataSecondXML)
        {
            DataColumn column = new DataColumn();
            column.ColumnName = currentMetaData.ColumnName;
            column.DataType = currentMetaData.DataType;
            column.DefaultValue = currentMetaData.DefaultValue;
            column.MaxLength = currentMetaData.MaxLength;
            _BufferFromSecondXmlFile.Columns.Add(column);
        }

        //18- Chargement des données du fichier dans la DataTable
        _BufferFromSecondXmlFile.ReadXmlSchema("XlmSchemaSecondFile");

        #endregion

        //19- Jointure entre les 4 Buffers
        _BufferFilm.Merge(_BufferCritique);
        _BufferFromFirstXmlFile.Merge(_BufferFromSecondXmlFile);
        _BufferFilm.Union(_BufferFromFirstXmlFile);

        //20- Filtrer les films postérieurs a 1960
        DataView _dvFinalData = _BufferFilms.DefaultView;
        _dvFinalData.RowFilter = "Year > 1960";

        //21- Insertion des données dans la table de destination, en l'occurrence le DW

    DataTable filtredFinalData = new DataTable();
    //Construction de la requête
    foreach (DataRow row in filtredFinalData)
    {
        commandInsert.CommandText += "INSERT INTO DWH_Film (title, year, productor)
VALUES (" + row["title"] + ", " + row["year"] + ", " + row["productor"] + ")";
    }
```

```
        commandInsert.executeNonQuery();
          //22- Fin de l'exécution
        }

public class MetaDonnees
        {
          private string _ColumnName;
          private Type _DataType;
          private string _DefaultValue;
          private int _MaxLength;

          public string ColumnName
          {
            get { return _ColumnName; }
            set { _ColumnName = value; }
          }

          public Type DataType
          {
            get { return _DataType; }
            set { _DataType = value; }
          }

          public string DefaultValue
          {
            get { return _DefaultValue; }
            set { _DefaultValue = value; }
          }

          public int MaxLength
          {
            get { return _MaxLength; }
            set { _MaxLength = value; }
          }
        }
```

● **Compréhension de l'implémentation**:

De l'étape 1 à 12 précisées dans la partie commentaire (en vert) du corps du code, le programme initialise une connexion à la base de données, exécute une requête et récupère le résultat dans un Buffer (le type Datatable dans C#)
De l'étape 13 à 18, il se connecte à un fichier XML pour récupérer les données dans un Buffer. En premier lieu, il se connecte à la DTD afin de lire les métadonnées et préparer la structure du Buffer et par la suite il commence le chargement des données depuis le fichier vers le Buffer.

Ensuite, dans l'étape 19, il utilise ces résultats intermédiaires pour faire une jointure (merge) afin de croiser les informations des deux différentes sources puis réalise une union entre les données. Après, il applique un filtre sur le résultat intermédiaire dans l'étape 20 pour enfin insérer ces données dans la destination dans l'étape 21.

Ce qu'il faut retenir de cette approche c'est que l'ETL, pour des raisons d'interopérabilité, convertit systématiquement toutes les sources de données dans son modèle interne (Buffers) pour se consacrer par la suite aux traitements de transformation indépendamment de la nature des sources de données. Ce n'est qu'à la fin du traitement qu'il convertit les données au format de la destination pour être chargées.

b. Traitement de l'intégration se basant sur le schéma global:

Le point d'entrée étant le schéma global, le moteur se base sur la bibliothèque de métadonnées du schéma global afin de détecter à quel niveau de l'architecture se trouve la donnée pour ainsi générer des sous-requêtes destinées au niveau inférieur de l'architecture.

Exemple : Nous reprenons l'exemple utilisé dans le document (films & critiques)

Récapitulatif des étapes d'implémentation :

1- Arrivée de la requête au schéma global,

2- Interprétation de la requête par le médiateur global de la requête et sollicitation des Wrappers sous-jacents :

   a. Consultation de la bibliothèque des métadonnées afin de décomposer selon le modèle de données,
   b. Envoi des métadonnées aux Wrappers,
   c. Pour chaque Wrapper concerné par le traitement, il y a génération de la sous-requête de sollicitation de la source de données sous-jacente,

3- Renvoi des résultats des wrappers au médiateur global, ***(ces résultats sont à ce niveau représentés dans le modèle du schéma global)***

4- Recomposition par le médiateur global des résultats envoyés par les wrappers pour obtenir le résultat final.

● Implémentation en utilisant un schéma global

Le code ci-dessous représente une simulation du traitement de l'intégration en utilisant un schéma global: Implémentation en langage C# des processus d'interprétation, de renvoi et de recomposition, de l'exemple utilisé dans l'étude de cas.

Implémentation en langage C#:

```
//Schema.Film représente une vue dans le schéma global contenant toutes les informations concernant le film
// Pour exemple, la requête envoyée au schéma global est la suivante : SELECT fid, title, realisator, year, kind FROM SchemaGlobal.Film WHERE year > 1960

      string GlobalQuery = "SELECT fid, title, realisator, year, kind FROM SchemaGlobal.Film WHERE year > 1960";

 //Le corps de la fonction d'interprétation de la requête.

      public void HAV(string _GlobalQuery)
      {
         //1-Retourne une liste qui contient les structures des requêtes (la source de données, le type de la source, les champs à retourner, les tables, les prédicats ainsi que la chaine de connexion à la source)
          List<HavQuery> _ListQueries = GetQueriesFromMediatore(_GlobalQuery);

          string _DateBaseQuery = String.Empty;
          string _XMLQuery = String.Empty;
          string _CSVQuery = String.Empty;

        //2- On boucle sur les structures pour construire les requêtes
          foreach (HavQuery CurrentQuery in _ListQueries)
          {
        //3- Selon le type de la source, on appelle le Wrapper adéquat pour construire la requête
             switch (CurrentQuery.SourceType)
             {
                case "DataBase":
                   _DateBaseQuery = GetQueryFromDataBaseWarapper(CurrentQuery.SourceId, CurrentQuery.Fields, CurrentQuery.Tables, CurrentQuery.Predicates, CurrentQuery.ConnexionString);
                   break;
                case "XMLFile":
                   _XMLQuery = GetQueryFromXMLWarapper(CurrentQuery.SourceId, CurrentQuery.Fields, CurrentQuery.Tables, CurrentQuery.Predicates, CurrentQuery.ConnexionString);
                   break;
                case "CSVFile":
                   _CSVQuery = GetQueryFromCSVWarapper(CurrentQuery.SourceId, CurrentQuery.Fields, CurrentQuery.Tables, CurrentQuery.Predicates, CurrentQuery.ConnexionString);
                   break;
                default :
                   break;
             }
          }
```

```
        //4- Envoyer tous les résultats des wrappers  dans la mémoire cache du SGBD du
Médiateur global.

//5 – pour les données déjà en relationnel, aucune transformation n'est nécessaire, les données
sont directement injectées dans la mémoire cache du SGBD pour être utilisées ensuite par le
médiateur global

SqlServerDB.SqlCacheMemory DatabaseResultCache =
SqlServerDB.CacheQueryResult.execute(_DateBaseQuery);

//6– pour les données en XML, une transformation est nécessaire. On utilise les connecteurs du
SGBD pour injecter directement les données dans la mémoire cache. Il n'y aura pas de modèle
intermédiaire de données.

String _XMLQueryTransformed= SqlServerDB.XMLConnector(_XMLQuery)
//7 – A cette étape, le connecteur génère une requête SQL qui utilise les fichiers XML comme
source de données. Exemple de la requête générée : SELECT * FROM
   OPENROWSET(Bulk 'c:\exemple\films.xml', SINGLE_BLOB)

//8 – La requête sera exécutée et le résultat sera chargé dans la mémoire cache du SGBD du
schéma global
SqlServerDB.SqlCacheMemory _XMLResultCache =
SqlServerDB.CacheQueryResult.execute(_XMLQueryTransformed);

 //9- Recomposition du résultat global par lé médiateur global. A cette étape, toutes les données
sources sont dans la mémoire cache du SGBD. Il suffit au Médiateur de générer une requête de
traitement global des résultats intermédiaires en les utilisant comme des objets (tables) du SGBD.
Exemple : SELECT Fid, title, year, productor FROM DataBaseResultCache Union SELECT Fid,
title, year, productor FROM XMLResultCache

        _ DataBaseResultCache.Union(_XMLResultCache);

     //10- Filtrer les films postérieurs a 1960
        SqlServerDB.SqlCacheMemory _FinalCacheResult = _ DataBaseResultCache;
        _ FinalCacheResult.RowFilter = "Year > 1960";

      }
//Déclaration des objets utilisés
      private string GetQueryFromCSVWrapper(int p, string[] p_2, string[] p_3, string[] p_4,
string p_5)
      {
        throw new NotImplementedException();
      }


   private string GetQueryFromXMLWarapper(int p, string[] p_2, string[] p_3, string[] p_4,
string p_5)
```

```
        {
            throw new NotImplementedException();
        }

        private string GetQueryFromDataBaseWarapper(int p, string[] p_2, string[] p_3, string[]
p_4, string p_5)
        {
            throw new NotImplementedException();
        }
        private List<HavQuery> GetQueriesFromMediatore(string _Requete)
        {
            return null;
        }

        public class HavQuery
        {
            string[] _fields = null;
            string[] _tables = null;
            string[] _predicates = null;
            string _sourceType = "";
            int _sourceId = 0;
            string _connexionString = "";

            public string[] Fields
            {
                get { return _fields; }
                set { _fields = value; }
            }

            public string[] Tables
            {
                get { return _tables; }
                set { _tables = value; }
            }

            public string[] Predicates
            {
                get { return _predicates; }
                set { _predicates = value; }
            }

            public string SourceType
            {
                get { return _sourceType; }
                set { _sourceType = value; }
            }
```

```
        public int SourceId
        {
            get { return _sourceId; }
            set { _sourceId = value; }
        }

        public string ConnexionString
        {
            get { return _connexionString; }
            set { _connexionString = value; }
        }
    }
```

● **Compréhension de l'implémentation :**

De l'étape 1 à 3, le médiateur interprète la requête principale et la décompose selon son dictionnaire de métadonnées suivant la typologie des sources. Ceci, afin de relayer la tâche de recherche de données aux wrappers sous-jacents.

De l'étape 4 à 8, une fois que les données sont localisées, les Wrappers vont respectivement récupérer les données depuis les sources en les injectant directement dans la mémoire cache du SGBD du schéma global. Chaque Wrapper génère un traitement spécifique correspondant à la typologie de données traitées.

A l'étape 9, le médiateur récupère les résultats intermédiaires résultants de l'exécution des Wrappers et constitue le résultat qui encapsule la globalité des données.

Enfin, à l'étape 10, une fois que toutes les données sont fédérées, un filtre est appliqué pour restreindre les données à celles demandées initialement par la requête principale.

En résumé, la différence majeure entre la matérialisation d'un entrepôt via une architecture ETL et via une architecture se basant sur un schéma global tel que c'est le cas dans notre architecture est que la première nécessite une transformation dans un modèle intermédiaire en plus de la transformation dans le schéma global ce qui n'est pas le cas dans la deuxième architecture.

Ce qu'il faut retenir de cette approche, c'est que contrairement à la matérialisation d'un entrepôt via une architecture ETL, elle opte pour une optimisation du processus d'intégration des données. En effet, la première nécessite une transformation dans un modèle intermédiaire en plus de la transformation dans le schéma global, alors que dans ce cas d'intégration se basant sur un schéma global il n'y a qu'une étape de transformation, celle des sources vers le schéma global.

3.4 Intégration de sources de données de même modèle

Rappelons que dans le niveau source de l'architecture supportant notre approche HAV et pour une meilleure combinaison de GAV et LAV, nous avons un ensemble de médiateurs spécialisés qui intègrent chacun un ensemble de sources de même modèle que lui, c-à-dire qu'à ce niveau toutes les sources à intégrer sont homogènes. Par exemple, dans le cas où le schéma spécialisé et et les sources sous-jacentes à intégrer sont tous relationnels, ceci se concrétise par l'intégration directe des données relationnelles. c-à-dire, le wrapper spécialisé relatif à ce médiateur spécialisé n'aura pas d'étape de transformation des données du modèle source vers le modèle du schéma spécialisé. De ce fait et comme nous pouvons le voir dans l'étape 5 de la deuxième implémentation, le wrapper n'aura pas à effectuer une étape de transformation de modèle source vers le modèle global. En effet, l'utilisation des wrappers spécialisés dans notre approche permet de tirer profit de chaque typologie de modèle de données et nous garantit une optimisation du coût de traitement lié à la nature de la source. En conclusion, un regroupement du traitement des sources de données par typologie de modèles de données permet de supprimer une étape parmi celles de l'adaptateur liée à l'hétérogénéité des sources.

4. Conclusion

Ce chapitre a été consacré à la validation de notre approche. Dans un premier temps nous avons procédé à une validation qualitative en nous rapportant à une étude de cas. Dans un deuxième temps, nous nous sommes appuyés sur une validation quantitative 'partielle', à travers laquelle nous avons montré d'une part, que le fait d'avoir plusieurs niveaux de médiateurs n'est pas pénalisant pour le traitement des requêtes, et d'autre part, nous avons montré que la matérialisation d'un entrepôt de données sur un schéma global est meilleure que celle réalisée à travers un ETL. En effet la différence entre les deux types de matérialisation est bien le passage par une mémoire Buffer dans l'approche ETL. Cette architecture permet d'avoir une très grande interopérabilité et une totale abstraction de la source de données dès le début des traitements. Or, le revers de la médaille est l'impact négatif sur les performances d'une telle approche. En effet, le fait de passer par une modélisation intermédiaire nécessite beaucoup de mémoire et de temps d'écriture de l'information, contrairement à l'architecture se basant sur un schéma global qui transforme les sources de données directement dans le modèle de données du schéma global en utilisant les connecteurs spécifique à ce dernier.

Nous avons également montré qu'un regroupement du traitement des sources de données par typologie de modèles de données permet de supprimer une étape parmi celles de l'adaptateur liée à l'hétérogénéité des sources.

Conclusion et perspectives

L'intégration de données a pour objectif de combiner des sources de données autonomes, distribuées et hétérogènes afin d'obtenir une vue homogène et uniforme des données intégrées.
Une façon pour y parvenir, est de représenter les données selon un même schéma global et selon une sémantique unifiée. Les sources de données, étant au cœur de la construction du schéma global d'un entrepôt ou d'un médiateur global, leur intégration au sein du schéma global est souvent un problème délicat qui nécessite sa résolution. Partant du constat qu'un entrepôt de données intègre des informations provenant des sources de données qui sont souvent hétérogènes et distribuées, nous avons jugé utile d'étudier les principales approches d'intégration. Cette étude nous a permis de proposer une approche d'intégration de sources de données hétérogènes et distribuées. Nous avons ainsi étudié les principales approches d'intégration de données de base pour ensuite les combiner vu l'apport certain de cette combinaison. Nous avons également étudié leurs contributions en tant que schémas d'intégration des sources de données et la manière dont s'effectue le lien entre les sources et le schéma global.

La construction de ce schéma peut se faire en utilisant les deux approches de base GAV et LAV. Cette tâche inclut deux problèmes duaux selon que l'une ou l'autre des approches est utilisée: (i) la mise à jour du schéma global lors de l'ajout ou la suppression d'une source. (ii) la réécriture complexe des requêtes en termes de vues.

Pour tenter de résoudre de façon plus satisfaisante ces problèmes liés à l'intégration de données, nous avons montré dans cette thèse l'intérêt et la faisabilité de la combinaison de GAV et LAV utilisées aussi bien dans une intégration virtuelle pour la définition du schéma global d'un système de médiation que dans une intégration hybride dans les entrepôts de données qui repose sur une intégration virtuelle.

Pour résoudre les problèmes relatifs aux approches GAV et LAV quand elles sont utilisées séparément et conserver leurs avantages, nous avons donc proposé d'exploiter la complémentarité de ces deux approches et de pallier conjointement leurs limitations par combinaison de ces deux dernières.

Dans cette conclusion, nous présentons tout d'abord une synthèse de nos contributions et nous terminons par les différentes perspectives que nous avons dégagées pour nos travaux.

Après avoir positionné notre approche dans le cadre des méthodes d'intégration, nous l'avons décrite en soulignant son originalité, ses nouvelles contributions et ses apports par rapport aux limites des approches de base existantes.

Afin de résoudre de manière satisfaisante les problèmes de l'intégration de données hétérogènes dans les entrepôts, nous avons proposé et décrit une nouvelle approche baptisée HAV qui consiste à combiner les avantages des approches GAV et de LAV. Nous avons étudié les caractéristiques de HAV, l'architecture qui la supporte ainsi que les différents concepts liés à cette architecture et les relations qui existent entre eux.

Par ailleurs, nous avons montré que HAV est une approche d'intégration qui peut être utilisée aussi bien dans un système de médiation que dans un entrepôt de données. Dans notre cas, le système de médiation intègre les sources de données externes pour la matérialisation de l'entrepôt conformément au schéma global du médiateur.

Nous avons également défini notre approche en adaptant un cadre formel proposé par M.Lenzerini, à notre contexte. Nous avons donné une formalisation d'un système d'intégration de données relatif à HAV. Nous avons précisé sa sémantique et la sémantique des requêtes qui lui sont posées. Pour établir le lien entre les sources et le schéma global à travers les schémas partiels, nous avons défini le mapping. Enfin, nous avons validé qualitativement HAV en effectuant une étude de cas nous permettant de voir comment le médiateur global est capable de répondre aux requêtes qui lui sont posées et avons procédé à une validation quantitative partielle.

La contribution de l'approche HAV est d'un côté son efficacité et sa praticabilité parce que l'ensemble des schémas partiels concernés par l'approche d'intégration GAV est petit et stable. D'un autre côté, HAV est très flexible avec l'addition (ou de la suppression) de sources de données à intégrer. En effet, lorsqu'une nouvelle source est ajoutée (ou supprimée), elle sera prise en charge par le médiateur spécialisé qui est concerné par le modèle de données de cette source. En plus, la stabilité du niveau supérieur et la souplesse du niveau inférieur de l'architecture font que l'approche HAV est évolutive lorsque le nombre de sources augmente. Par ailleurs, il sera moins complexe pour les médiateurs spécialisés d'effectuer les sous-requêtes (parce que chacun a un nombre réduit de sources à intégrer et ces sources sont homogènes), que si le schéma global lui-même était construit avec l'approche LAV, donc la complexité de reformuler les requêtes est réduite. En effet, une caractéristique potentiellement importante de notre approche est que les sources sous-jacentes à un médiateur spécialisé sont de même modèle.

La validation qualitative nous a montré que le médiateur global est capable de répondre aux différentes requêtes qui lui sont posées en montrant sur une étude de cas les différentes étapes depuis la construction du schéma global jusqu'au traitement de la requête. Nous avons choisi un cas d'utilisation qui couvre un éventail assez large des possibilités qu'un médiateur devrait avoir. Pour la requête choisie, le domaine de définition, la requête SQL, sa description textuelle et le résultat qui devrait être renvoyé comme réponse à la requête sont décrits. Le domaine de définition correspond au contexte dans lequel s'appuie la requête. Il s'agit dans notre cas d'un ensemble de documents XML et d'un ensemble de tables. Nous tenons à noter que l'évaluation de la requête utilisée dans notre étude de cas a exploité les spécificités des données et fait intervenir plusieurs sources, ce qui est satisfaisant pour la validation du médiateur global et celle des adaptateurs.

La validation quantitative est une partie importante de la recherche en base de données qui se concentre sur l'amélioration de la performance d'un système particulier. Les expériences quantitatives sont le meilleur moyen de valider ces résultats. Cependant, réaliser des expériences n'est pas toujours facile. De plus la complexité du système sous test, une expérience de conception, le choix du bon environnement et des valeurs de paramètre, l'analyse des données qui sont recueillies, restent une tâche difficile. Dans cette thèse, nous avons procédé à une validation expérimentale partielle (ponctuelle) de notre approche. Pour cela, nous avons d'une part utilisé les résultats réalisés par [NGOG 2003] qui sont très prometteurs dans la mesure où ils permettent d'utiliser une arborescence de médiateurs sans nuire à l'évaluation. D'autre part les

résultats expérimentaux que nous avons obtenus montrent que nous avons atteint notre objectif de réduire le temps de traitement de requêtes du médiateur global en réduisant celui des médiateurs spécialisés dont les sources sous-jacentes sont de même modèle que lui. En effet, l'utilisation des adaptateurs spécialisés dans notre approche permet de tirer profit de chaque typologie de modèle de données et nous garantit une optimisation du coût de traitement lié à la nature homogène des sources sous-jacentes. Nous avons également mis en évidence l'apport de la matérialisation d'un entrepôt conformément à son schéma d'intégration par rapport à une architecture réelle d'un ETL.

En plus de l'approfondissement des travaux que nous venons d'exposer, certaines limites pourraient se produire parce que nous n'avons pas construit ni utilisé un tel système. En effet, Au delà de ces premières réalisations, de nombreux aspects restent à aborder avant la construction d'un prototype intégrant l'ensemble des propositions nouvelles relatives à un système d'intégration construit sur HAV. Toutefois, les résultats que nous avons obtenus montrent que nous avons atteint l'objectif de cette thèse, à savoir contribuer dans le domaine de l'intégration des sources de données dans les systèmes d'intégration en général et dans les entrepôts de données en particulier. Plus tard nous pensons nous consacrer à un certain nombre de travaux de recherche, qui permettront de compléter utilement notre travail. En particulier, pour une meilleure évaluation en termes de propriétés de performance, il serait très intéressant d'adapter un bon d'essai parmi ceux qui existent à notre contexte. Il serait également intéressant de comparer l'approche HAV avec certaines approches hybrides telle que GLAV.

Références

[Adali et al 1996] S. Adali, K.S.Candan, Y. Papakonstantinou, and V.S. Subrah-Maniam, " Query caching and optimization in distributed mediator systems," in Proceedings of the ACM SIGMOD Conference, on Management of Data, 1996, pp. 137-148.

[Alexe et al 2008] B. Alexe L. Chiticariu R. J. Miller W. Tan. MUSE: Mapping Understanding and deSign by Example. *Proceedings of International Conference on Data Engineering (ICDE) 2008.*

[Ashish et al 1997] N. Ashish, C. Knoblock. Semi-automatic Wrapper Generation for Internet Information Sources. In Second IFCIS International Conference on Cooperative Information Systems (CoopIS), Charleston, SC, 1997.

[Beneventano et al 2000] D. Beneventano, S. Bergamaschi, S. Castano, A. Corni, R. Guidetti, G. Malvezzi, M. Melchiori, and M. Vincini. Information integration : The MOMIS project demonstration. In *The VLDB Journal*, pages 611–614, 2000.

[Blakeley 1996], Jose Blakeley, "Data access for masses through OLE DB", in proceeding of the ACM SIGMOD International Conference on Management of data, New York, 1996, vol. 2 of ACM SIGMOD Record, pp. 161-172, ACM Press.

[Bogdan et al, 2008] Bogdan, A., Chiticariu, Miller , L.R., Tan, W.c , *Muse: Mapping Understanding and design by Example*, International Conference on Data Engineering (ICDE).10-19, 2008.

[Boulçane.F 2006]. *An approach of mediation*. In proceedings of the IEEE International Conference on Information & Communication Technologies, ICTTA'06. Volume 2, Page(s): 3546-3551, 2006.

[Boulçane.F 2007]. *The HAV Data Integration Approach: The Mapping in HAV,* proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS'07), Page(s): 490-493, 2007.

[Boulçane.F, 2008]. A Hybrid Data Integration Approach based on specialized mediators, proceedings of the International Review on Computers and Software, Vol.3 N.5, pages(s) 554-563, Septembre 2008.

[Boyd et al 2004] Boyd, M., Lazanitis, C., Kittivoravitkula, S., Brien, P.M., Rizopoulos, N., *AutoMed: A BAV Data Integration System for Heterogeneous Data Sources*,In Advanced Information Systems Engineering 16th International Conference, CAiSE 2004, Riga, Latvia, Proceedings Springer-Verlag, 2004.

[Bright et al 99] BRIGHT L., GRUSER J.-R., RASCHID L., VIDALM, « A Wrapper Generation Toolkit to Specify and Construct wrappers for Web accessible Data Sources (WebSources) », *Journal of Computer Systems Science & Engineering. Special Issue: Semantics on the World Wide Web*, vol. 14, no 2, 1999.

[Cali 2003] A. Calì. Reasoning in data integration system: why LAV and GAV are siblings. in Proc. of the 14th International Symposium on Methodologies for Intelligent Systems (ISMIS), 2003.

[Calvanese et al 2001] D. Calvanese, G. D. Giacomo, M .Lenzerini, and M. Y.Vardi. View-Based Query Answering and Query Containment over Semistructured Data. In DBPL Workshop, pages 40–61, 2001.

[Calvanese et al, 2004] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R., *What to ask to a peer: Ontology-based query reformulation*, Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning, 2004.

[Carey et al, 1995] Carey, M. J, Haas, L. M., Schwarz, P. M. , Arya, M., Cody, W.F., Fagin, R., Flickner, M., Luniewski, A., Niblack, W., Petkovic, D., Thomas, J., Williams, J.H., Wimmers, E.L., Towards *heterogeneous multimedia information systems: The Garlic approach*. In Proc. of the 5th Int. Workshop on Research Issues in Data Engineering – Distributed Object Management (RIDE-DOM'95), pages 124–131, IEEE Computer Society Press, 1995.

[Chaudhuri et al 1997] Chaudhuri S., Dayal U., "An Overview of Data Warehousing and OLAP Technology", ACM SIGMOD Record, 26(1), 1997.

[Chawathe et al 1994] S. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. D. Ullman, and J. Widom. The tsimmis project : Integration of heterogeneous information sources. *Proceedings of the 10th Meeting of the Information Processing Society of Japan*, pages 7–18, Mars 1994.

[Cluet et al 98] CLUET S., DELOBEL C., SIMEON J., SMAGA K., « Your Mediators Need Data Conversion », *Proceedings of the ACM SIGMOD Conference*, 1998, p. 177-188.

[Cluet et al 2001], S. Cluet, P. Veltri, and D. Vodislav. Views in a Large Scale XML repository. In *Proc. VLDB*, Rome, Italy, September 2001.

[Cody et al 95] CODY W., HAAS L., NIBLACK W., ARYA M., CAREY M., FAGIN R., LEE D., PETKOVIC D., SCHWARZ P., THOMAS J., ROTH M. T., WILLIAMS J., WIMMERS E., « Querying Multimedia Data from Multiple Repositories by Content : The Garlic Project », *IFIP 2.6 Third Working Conference on Visual Database Systems (VDB-3)*, Lausanne, Switzerland, March 1995, http ://www.almaden.ibm.com/cs/garlic.

[Doan et al 2000] Anhai Doan , Pedro Domingos , Alon Y. Levy , Learning Source Descriptions for data integration *Proceedings of the International Workshop on The Web and Databases (WebDB) 2000*.

[Doucet et al 2001] Doucet A., Gançarski S., « *Bases de données et Internet, Modèles, Langages et Système* », chapitre Entrepôts de données et bases de données multidimensionnelles, Hermès-Lavoisier - Sous la direction de A. Doucet et G. Jomier, 2001.

[Etzioni et al 1994] Etzioni O. & Weld D. (1994). A Softbot-Based Interface to the Internet. *Communications of the ACM*. Vol. 37(7). p. 72-76.

[Evolution 1997] Projet Evolution - Http: //www.prism.uvsq.fr/recherche/themes/ anciens/ teams/dataware/coop/evolution.html, 1997.

[Fernandez et al 1998] M.F.Fernandez, D.Florescu, J.Kang, A.Y.Levy, et D.Suciu. Catching the Boat with Strdel: Experiences with a Web-Site Management System. In SIGMOD Conference, pages 414-425, 1998.

[Friedman et al 1997] Friedman M. & Weld D. S. (1997). Efficiently executing informationgathering plans. *In 15th International Joint Conference on Artificial Intelligence*. p. 785-791, Nagoya. Japan.

[Friedman et al 1999] Friedman, M., Levy, A., Millstein, T., *Navigational plans for data integration*, Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications, Pages: 67 – 73, Orlando, Florida, United States, 1999.

[Fuxman et al 2005], Fuxman A, Kolaiti P, et al., Peer Data Exchange. In PODS ,2005

[Garcia et al 1994] H.Garcia-Molina, J.Hamer, K.Ireland, Y.Papakonstantinou, J.Ullman, et J.Widom. Integrating and Accessing Heterogeneous Information in TSIMMIS. In 10th Anniv. Meeting of Information Processing Society of Japan, pages 7-18, Tokyo, Japan, 1994.

[Garcia et al 2002] H. Garcia-Molina, J. Ullman, J. Widom. Database Systems: The Complete Book, Prentice hall, 2002.

[Gardarin et al 1996] G. Gardarin et al, "IRO-DB: A distributed system federating object and relational databases", in Object-Oriented Multi-database Systems _ A solution for Advanced Applications, O.A.Bukhres and A.K. Elmagarmid, Eds. Prentice Hall, 1996.

[Gardarin 2001] George Gardarin. Les data webhouses arrivent, laboratoire PRISM et E-xml media, 2001.

[Gatziu et al 1999] Gatziu S., Jeusfeld M.A., Staudt M., Vassiliou Y., "Design and Management of Data Warehouses", Report on the DMDW'99, ACM SIGMOD Record, 28(4), Dec. 1999.

[Goasdoué et al 1999] Goasdoué F., Lattes V., Rousset M (1999). The use of carin language and algorithms for information integration: The picsel project. International Journal of Cooperative Information Systems (IJCIS), 9(4):383 – 401.

[Genesereth et al 1997] M. R. Genesereth, A. M. Keller, and O. M. Duschka. Infomaster : an information integration system. In *ACM SIGMOD International Conference on Management of Data*, pages 539–542, 1997.

[Goh et al 1999] C. Goh, S. Bressan, E. Madnick, and M. D. Siegel. Context interchange : New features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems*, 17(3) :270–293, 1999.

[Gomez et al 2005] G.-L. G. Gomez. *Construction automatisée de l'ontologie de systèmes médiateurs. Application à des systèmes intégrant des services standards accessibles via le Web.* PhD thesis, Universit´e Paris XI Orsay, 2005.

[Gruber 1995] T. Gruber. A translation approach to portable ontology specification. Knowledge Acquisition, 5(2) :199–220, 1995.

[Haas et al 1997] L.M.Haas, D.Kossman, E.L. Wimmers, et J.Yang. Optimizing Queries Across Diverse DataSources. In 23th International Conference on Very Large Data Base, pages 276-285, Athens, Greece, 1997.

[Hacid et al 2005] M.-S. Hacid and C. Reynaud. L'intégration de sources de données. *La revue I3 : Information - Interaction - Intelligence*, Vol5 n˚1, 2005.

[Halevy et al, 2005] A. Halevy, G. Zakary, D. Suciu, I.Tatarinov, Schema Mediation for Large Scale Semantic Data Sharing.,*VLDB Journal*, Volume 14, Issue 1, 2005

[Hammer et al 1995] J. Hammer, H. Garcia-Molina, J. Widom, W. Labio, and Y. Zhuge. The Stanford data warehousing project. *IEEE Quarterly Bulletin on Data Engineering ; Special Issue on Materialized Views and Data Warehousing*, 18(2) :41–48, 1995.

[Heflin et al 1999] Heflin J., Hendler J., Luke S (1999). Shoe: A knowledge representation language for internet applications. Technical CS-TR-4078, Institute for Advanced Computer Studies, University of Maryland.

[Idrissi et al, 2008] Idrissi, Y.B.; Vachon, J 2008. Contextualized Linguistic Matching for Heterogeneous Data Source Integration e-Technologies, 2008 International MCETECH Conference on Volume , Issue , 23-25 Jan. Page(s):136 – 147.

[Inmon et al 1994] Inmon W.H, John Wiley & Sons. Using the Data Warehouse. 304 p.ISBN: 0471059668.

[Inmon et al 1996] Inmon W.H, John Wiley&Sons. "Building the Data Warehouse", ISBN 0471-14161-5.

[Ives et al 2002] Z. Ives. Efficient Query Processing for Data Integration. A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy University of Washington, 2002.

[Jarke et al 1997] M. Jarke and Y. Vassiliou. Data warehouse quality design : A review of the dwq project. In *Invited Paper, 2nd Conference on Information Quality. Massachusetts Institute of Technology, Cambridge*, 1997.

[Jarke et al 1999] Jarke M., Lenzerini M., Vassiliou Y., Vassiliadis P., "Fundamentals of Data Warehouses", Ed. Springer Verlag, ISBN 3-540-65365-1, 1999.

[Kapitskaia et al 1997] Olga Kapitskaia, Anthony Tomasic, and Patrick Valduriez, "Dealing with discrepancies in wrapper functionality", Tech. Rep.RR-3138, INRIA, 1997.

[Koffina et al 2005] Koffina,I., Serfiotis, G., Christophides, V., *Foundations for Information Integration: A State of the Art.* Sixth Framework Programme, 2005.

[Labio et al 1997] W. J. Labio, Y. Zhuge, J. L. Wiener, H. Gupta, H. Garc´ıa-Molina, and J. Widom. The WHIPS prototype for data warehouse creation and maintenance. In *Proceedings of SIGMOD*, pages 557–559, 1997.

[Lacroix 00] LACROIX Z., « Scientific Data Integration: Wrapping Textual Documents with a Database View Mechanism and an XML Engine », *IEEE International Symposium on Bio-Informatics and Biomedical Engineering (BIBE)*, Washington, DC, November 2000.

[Lawrence et al, 2002] Lawrence, R., Barker, K., Using *Unity to Semi-Automatically Integrate Relational Schema*, Demonstration at ICDE 2002, 18th International Conference on Data Engineering, 2002.

[Lenzerini, 2002] Lenzerini, M., Data Integration: *A Theoretical perspective. Università di Roma "La Sapienza".* Proceeding of the twenty-first ACM SIGMOD-SIGACT-SIGART, Symposium on Principles of Database Systems, pages 233–246, 2002.

[Leonidas et al, 2003] Leonidas R. J., Galanis, Y.W., DeWitt, D.J., *Locating data sources in large distributed systems*, in the 29th VLDB Conference, Berlin, Germany, 2003.

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

[Levy et al, 1996] Levy A. Y., Rajaraman, A., Ordille, J.J., *Querying heterogeneous information sources using source descriptions*, in VLDB, pp. 251–262, 1996.

[Levy et al 2000] A. Levy. **Logic-based techniques in data integration**. In J. Minker, editor, Logic Based Artificial Intelligence. Kluwer Academic Publishers, 2000.

[Li et al 1998] C.Li, R.Yerneni, V.VassalosV., H.Garcia-Molina, Y.Papaconstantinou, J.D. Ullman, et M.V Valivetti. Capability Based Mediation in TSIMMIS. In proceedings ACM-SIGMOD International Conference on Management of Data, pages 564-566, Seattle, Washington, 1998.

[Manolescu et al, 2001] Ioana Manolescu, D. Florescu, D.K., *Answering XML Queries over Heterogeneous Data Sources,* Proceeding of 27th International Conference on Very Large Data Bases, Rome, Italy, pages 241-250, 2001.

[McBrien et al 2003] P. McBrien, A. Poulovassilis. Data integration by bi-directional schema transformation rules. In: 19th International Conference on Data Engineering, ICDE'03, March 5 - March 8, 2003

[Mena et al 1996] Mena E. & Kashyap V. & Sheth A. & Illarramendi A. (1996). OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *In 4th Int. Conf. on Cooperative Information Systems*. p. 14-25. Bruessels. Belgium.

[Mesrobian et al 1996] Edmond Mesrobian, Richard Muntz, Eddie Shek, Silvia Nittel, Mark LaRouche and Marc Kriguer, "OASIS: an open architecture scientific information system, "in Proceedings of RIDE", New Orleans 1996, IEEE Press.

[Metais et al 2002] Metais E. et Sèdes F. Appariement d'informations dans les entrepôts de données: quelques approches pour le filtrage flexible, Revue I3 (Information - Interaction - Intelligence), 2(2), pp. 63-89. 2002.

[Naacke et al 1998] Hubert Naacke, Georges Gardarin, and Anthony Tomasic, " Leveraging mediator cost models with heterogeneous data sources", in Proceedings of the 14th International Conference on Data Engineering (ICDE 98), Orlando, Florida, 1998.

[NGOG 2003] T.T.Dang NGOG : Fédération de données semi-structurées avec XML. Thèse soutenue à l'université de Versailles, Juin 2003.

[Pottinger et al 2000] Pottinger, R., A.Y. Levy., A *scalable algorithm for answering queries using views*. Proc. of the 26th Int. Conf. on Very Large Data Bases (VLDB), pages 484–495. 2000.

[Roth et al 97] ROTH M., SCHWARZ P., « Don,t Scrap It, Wrap It ! A wrapper Architecture for Legacy Data Sources », *Proceedings of the International Conference on Very Large Data-Bases*, 1997.

[Rousset et al 2002] M.C.Rousset, A.Bidault,.C.Froidevaux, H.Gagliardi, F.Goasdoué, C.Reynaud, B.Safar, Construction de médiateurs pour intégrer des sources d'information multiples et hétérogènes , *le projet Picsel. Information-Interaction-Intelligence*, Revue 13 : 2002.

[Subrahmanian et al 1995] Subrahmanian V.S. & Adali S. & Brink A. & Emery R. & Lu J. J. & Rajput A. & Rogers T. J. & Ross R. & Ward C. (1995). HERMES: A heterogeneous reasoning and mediator system. *Technical Report. Univ. of Maryland*.

[Shoens et al 1993] K.Shoens, A.Luniewski, P.Schwarz, J.Stamos, et J.Thomas. The Rufus System: InformationOrganization for Semi-Structured Data. In R.Agrawal. S.Baker, et D.Bell, editors, 19th International Conference on Very Large Data Bases, pages 97-107, Dublin, Ireland, Auguct 1993.

[Stuckenschmidt et al 2000] Stuckenschmidt H., Wache H., Vogele T., Visser U (2000). Enabling technologies for interoperability. In Ubbo Visser and Hardy Pundt, editors, Workshop on the 14th International Symposium of Computer Science for Environmental Protection, pages 35–46, Bonn, Germany.

[Vassalos et al 1997] V.VassalosV, H.Garcia-Molina, Y.Papaconstantinou. Describing and Using Capabilities of Heterogeneous Sources. In Proceeding of Intrenational Conference on Very Large Data Base, Athens, Greece, August, 1997.

[Vassiliadis et al 2005] P. Vassiliadis, A. Simitsis, P. Georgantas, M. Terrovitis, and S. Skiadopoulos. A generic and customizable framework for the design of etl scenarios. *Information Systems*, 30(7):492–525, 2005.

[Vavouras et al 2000] A. Vavouras, S. Gatziu, and K. Dittrich. Modeling and Executing the Data Warehouse Refreshment Process. Technical Report ifi-2000.01,
11, 2000.

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

[Wache et al 2001] H. Wache, T. V¨ogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. H¨ubner. Ontology-based integration of information - a survey of existing approaches. Proceedings of the International Workshop on Ontologies and Information Sharing, pages 108–117, August, 2001.

[Widom 1995] J. Widom. Research problems in data warehousing. In *Int'l Conf. on Information and Knowledge Management*, 1995.

[Wiederhold 1992] G. Wiederhold, "Mediators in the architecture of future information systems », IEEE Computer, vol.25, n°3, pp 38-49, 1992.

[Wiederhold 1995] G.Wiederhold. Mediaton in information systems. *ACM Computing Surveys*, 27(2):265–267, June 1995.

[Xuan 2006] Nguyen Xuan D. Intégration de bases de données hétérogènes par articulation à priori d'ontologies : application aux catalogues de composants industriels.' Thèse de doctorat, ENSMA/Université de Poitiers, 2006.

[Xu et al 2004] Xu, L., Embley, W., *Combining the Best of Global-as-View and Local-as-View for Data Inte*gration, In Information Systems Technologies and Its Applications - ISTA. 123–136, 2004.

[Yemeni et al 1999], Yerneni R , Chen L , et al. , Computing Capabilities of Mediators. In SIGMOD, 1999.