An Advanced XML Mediator for Heterogeneous Information Systems Based on Application Domain Specification

Mostafa. Ezziyani, Mustapha Bennouna and Loubna Cherrat

Information and Telecommunication Systems Laboratory,
Abdelmalek Essaadi University, Morocco
e-mail: ezziyyani@ieee.org, president@uae.ma, cherrat@ieee.org

Abstract

The explosion of the number of web-based information sources has drastically increased the need for intelligent mediation tools to be implemented between the users and these information resources. These tools must overcome the limitations of the current search engines while making it possible for the users to submit more sophisticated queries than simple key words. On the other hand, we can incorporate brief replies coming from various sources to build total responses to the users' queries. This technique is based on the integration of heterogeneous information sources.

The central problem of information sources integration resides on their heterogeneity on the level of their format, the design and/or the semantic aspect. It is not possible to change these existing sources in order to make them homogeneous (for example XML or others). But, building a mediator on the top of these sources can solve the problem and allows a to query an information system which is centralized and homogeneous.

In this paper, we will present and illustrate the various approaches which we followed for the design and the development of an Advanced Xml Mediator (AXMed). The goal of this mediator is to ensure an integration of several resources of non-materialized heterogeneous data. The AXMed design which we propose is based on application domains specification. This system of mediation represents the fruit of a thorough research of internal architecture and the existing operating mode of several mediators, namely Médience, LeSelect, Tsimmis, Agora, etc.

The AXMed architecture system provides uniform access to the mentioned kind of data sources. The advantage of this system is very easy to configure and maintain by writing simple XML files, describing the structure and mapping of a new source. Indeed, adding additional data sources does not need restarting and redefinition of the system core.

Keywords: Heterogeneous, Integration, GAV, LAV, Rewriting Queries, Wrapper, Web Service, semantic Cache, XML, XQuery.

1. INTRODUCTION

To meet the needs for communication and data exchange between increasing number of hardware and software available today on the Web, the design and the development of a flexible and effective mediation system is necessary. The goal of such a system is to intercept the users' queries and to find the more adequate data and services from several heterogeneous resources, to answer the queries of the user, to pass the specific parameters, to call upon the service and to turn over the result in a transparent way to the users. The latter do not need to know the nature, the type or the localization of

the data, where the services are called upon, in which language they were programmed and on which operating system they are lodged, or no other system aspects which do not form part of the interface of the required services.

To face these integration problems and to ensure the inter-working between the various services on the Web and the availability of the heterogeneous resources on the Web, the development of the specific tools which facilitates the transparent use of these resources is necessary. This software component plays the part of a footbridge between the user and the sets of heterogeneous resources, from where comes the name of mediator [1]-[4].

This paper is structured as follows. The next section contains an overview of the existing theoretical approaches of Information Systems mediation and describes the overall architecture of these systems, while section 4 explains the proposed Advanced Xml Mediator (AXMed) in more detail. Then, an illustrative case study example is presented. In the last section, we propose the UML design of AXMed.

I.1 PROBLEMS OF INTEGRATION INFORMATION SYSTEMS

Information Systems are expected to be a completely new generation of software systems. Their main task is to operate at a global level over existing data sources. It is important to consider that these sources have certain characteristics making the integration process very difficult:

Heterogeneity: the data sources are mostly developed for a special purpose. This often results in different solutions for storing information of the same real-world objects. Information can be stored in databases with different models (relational, object oriented), or be available as Web Services. It is obviously that these kinds of sources are accessed through different interfaces, protocols and languages (Syntactical Heterogeneity). Even the same data model can cause mapping conflicts due to different understandings of the real world.

Autonomy: Data sources do not give up their autonomy. First of all they keep their Design autonomy. It's up to them how the contained information is stored. Furthermore they are able to decide which other systems are allowed to communicate with them. Additionally each component is independent in deciding how the incoming queries are scheduled and executed.

Distribution: Sources do not always reside on the same host. It is likely that they are on different hardware platforms and operating systems and can only be accessed through certain network protocols.

2. THE INFORMATION SYSTEMS MEDIATORS

A mediator gathers within a unifying framework the description of a domain as well as the contents of the various available sources relative to this domain. Thus, the user does not have to know the format nor the contents of the available sources which are relative to his domain of interest. He expresses his query in terms of the vocabulary of domain description. It is the task of the mediator to take in entry the query of the user and to transform it into achievable queries on the sources, according to the description of the content and the format of these sources available to it [5][6].

The mediators ensure at the same time the integration of the data and the resources attached to these data, namely, the tools of research, simulation, analysis and adapted visualization. To allow the integration of all available resources in the design of complex gates meeting the needs of an industry for example, several challenges are to be taken up**.**

Such a system is composed primarily of two parts: the mediator and wrappers. The mediator is an interface between the user and the collection of data sources which gives him/her the possibility to question a homogeneous and centralized information system by providing him/her with the integrated

Schema . The key role of the mediator is to divide, according to this diagram, the user's queries into several sub-queries supported by the sources.

The wrapper is an interface allowing the translation of information between the mediator and the data sources. It translates, initially, the queries submitted by the mediator into understandable queries by the data sources. Then, it translates the answers sent by the data sources into comprehensible answers by the mediator (Figure 1).

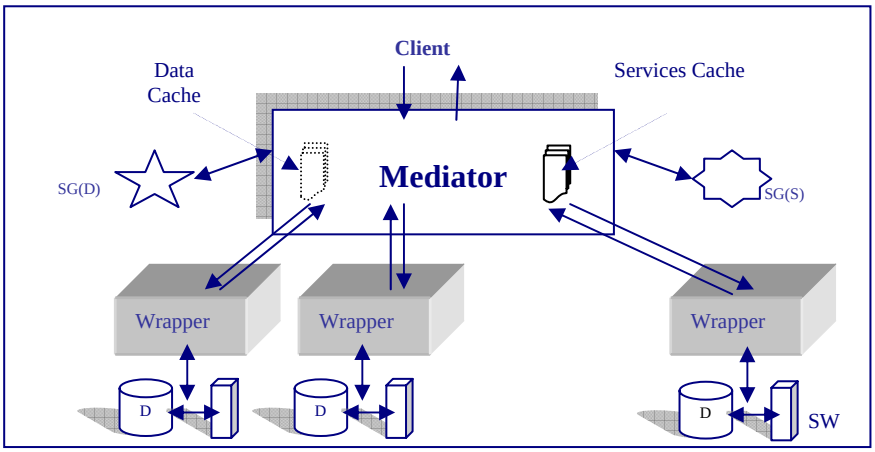

Figure 1: Mediation level

The integration systems (by mediation) are characterized by the way in which the Schema s of the local sources are related to the unified global Schema : there are mainly two approaches [7][8]:

a) The first approach known as Global-As-View: Consists in defining the Global Schema as a sight on the local Schema s. In other words, for each relation R in the Global Schema , one defines a sight made up of sources terms. The main advantage of this approach lies in the fact that the rewriting of the queries is simple. Indeed the relations of the integrated Schema are made up of terms of the Schema sources relations. Another advantage of this approach is the possibility of re-using the views like sources to build the hierarchy of the mediator. On the other hand, it is difficult to add new sources to the system. Indeed, it is necessary to take account of which of the new relations in the expression of the integrated Schema will be amended. This makes it necessary to rewrite practically all the relations of the integrated Schema .

b) The second approach, known as Local-As-View: it is the local sources that are defined like views on the global Schema . With this approach the problem of the rewriting of the queries is in general Np-complete: however in many cases it is polynomial (according to the number of data sources). On the other hand, the addition of a new source is facilitated because it consists to write the whole of the relations of the source according to the relations of the Global integrated Schema .

With the definition of the global Schema , the mediator presents virtual integrated views of the data sources. Thus, a query formulated with the mediator is submitted independently of the localization of the various data intervening to calculate the result. That introduced three difficulties:

- Decomposition of a query: it acts starting from a query submitted on an integrated view, to locate the data intervening in its resolution, to produce specific sub-queries to each source, to order these sub-queries and if required to introduce operators on the level of the component of mediation in order to supplement this whole of sub-queries. The localization of the under-queries requires specific structures of the metadata management.
- Recombination of the results: once the sub-queries submitted to the different sources, it is necessary to combine the various results. The results of each sub-query can possibly be the

subject of an additional treatment to ensure the referential constraint between the data and the definition of an inter-queries execution scheduling.

Several mediation systems were developed (e.g. Médience, Tsimmis, e-XMLMedia...) [1]-[2] to address certain specifications in particular domain. These systems are the subject of a thorough study which allows us to develop and propose a new architecture.

Different architectural aspects of these systems are taken into consideration, namely, type of data sources, internal representation, external representation, integration approach, etc. The following table (*Table 1*) presents the main features of some mediators:

Mediator	Data Sources	Internal Representation	External Representation	Integration Approach
Medience	*Relational*	*Relational*	*Relational*	*GAV*
Tsimmis	*Relational*	*Relational*	*Semi-structured*	*GAV*
Agora	*XML*	*Relational*	*XML*	*LAV*
Xyleme	*XML*	*XML*	*XML*	*LAV/GAV*
e-*XMedia*	*XML*	*XML*	*XML*	*GAV*

Table 1: Mediators description

In the following section, we will discuss in detail the different approaches used to build the advanced XML mediator together with its architecture and main features.

3. THE ADVANCED XML MEDIATOR

The AXMed mediator aims to propose a user friendly mediation platform for the query and the integration of several heterogeneous data sources located in different servers on the Web. This system also ensures the transparent and parallel treatments by the integration of the software resources available on the questioned sites.

3.1. AXMed CONCEPTUAL SPECIFICATIONS

To address this deep technological progress and the evolution of the uses which accompanies it, the proposed AXMed mediation system deals with the whole of the structural specifications and the following functional constraints:

Data sources integration: The extraction of the data from several heterogeneous distributed resources satisfies the response to the users' queries. The main objective of a data integration system is to facilitate users to focus on specifying what data they want, rather than on describing how to obtain them. To achieve this, the system provides an integrated view of the data stored in the underlying data sources. In a data integration system, users are interested mainly in querying the integrated data rather than updating the data through the integrated view.

The opening system: The possibility of handling data and software sources of a resource in other similar and available resources, by ensuring the transparency and the rights of access.

Data security: Since information systems are open, the data confidentiality vis-à-vis spying attacks is a major stake. Therefore AXMed mediator must solve the problems related to data coding with a multiple access, define and manage access rights on multiple autonomous and distributed resources, and also solve the information access methods problem.

For both technical and pragmatic reasons, it is unreasonable to expect that a single security technology can be defined that will both address all mediation system security challenges and be adopted in every hosting environment. Existing security infrastructures cannot be replaced overnight. For example, each source integrate by a mediation system is likely to have one or more registries in which user accounts are maintained; such authentication mechanisms deployed in an existing environment that is

reputed secure and reliable will continue to be used. Each source typically has its own authorization infrastructure that is deployed, managed and supported. It will not typically be acceptable to replace any of these technologies in favor of a single model or mechanism. Thus, to be successful, mediation system security architecture needs to step up to the challenge of integrating with existing security architectures and models across platforms and hosting environments. This means that the architecture must be implementation agnostic, so that it can be instantiated in terms of any existing security mechanisms (e.g., Kerberos); extensible, so that it can incorporate new security services as they become available; and integratable with existing security services.

Reliability: An advanced technique of duplication and restoration of data (Transaction) to avoid data loss in the event of a breakdown.

Distribution and autonomy: Each interoperable resource dedicated to a given service works independently from the others. Consequently, the distribution is taken into account in two levels, namely, the software and the data.

Evolution and extensibility: The system is neither limited in time nor in space. Consequently, it is necessary to use homogeneous and complete data model to be able to support different applications data.

Effective and selective access to information: To satisfy the criteria of relevance and performance desired by the user, new techniques of query and filtering must be proposed (i.e. criteria preferences, approximate filtering, partial results, etc.).

Follow-through and update: detailed specifications will enable us to follow the evolution of the system in time and space.

The development of such a system of mediation is original because it aims not only the integration of data but also that of the applications such as forms and tools for simulation and scientific calculations, and other tools which facilitate the access to the data and their handling. The scientific objectives are articulated around two axes:

a) Data and applications representation and the integration.

b) Query optimization and services execution.

3.2 FUNCTIONAL CONSTRAINTS AND AXMed ARCHITECTURE INNOVATIONS.

This mediator is the result of a detailed study of the advantages and disadvantages of several existing mediators. The implementation of the core is based on the management technology of the objects distributed around the two data models: the relational/object model and the XML model. Concerning the adopted approach it is a mixed approach between GAV and LAV. This last choice is justified by the simplicity of the queries' rewriting operation, through the use of GAV approach, and also by the evolution and the flexibility of the systems with the introduction of LAV approach [12]. The main contribution in the core of architecture that we propose and the originality of this system is summarized in the method of the global Schema definition which is based on the processor of refinement by specialization of the domains [13]. This new technology will be presented in the following section. And the new methodology for the management of the semantic cache, and in the use of the Web Service Technology to ensure the tools integration. The generic architecture of this mediator is illustrated in Figure2.

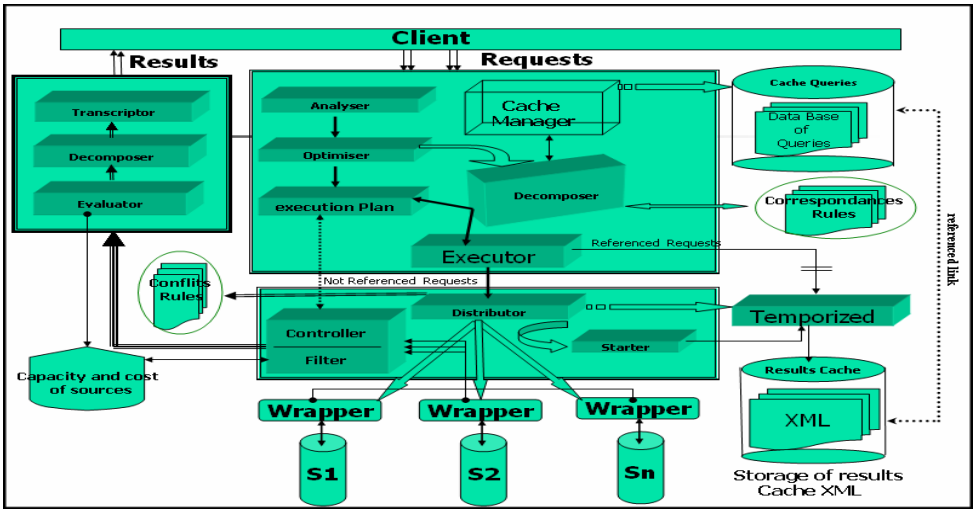

Figure 2: AXM[ed] Architecture.

The flowing table (Table 2) presents the role of the basic composites of the **AXM[ed]** mediator:

Analyzer	This component allows to analyze the queries of the users for the syntactic and lexical checking.
Optimizer	This component optimizes the query according to preset rules' of optimization.
Decomposer	It carries out the operation of the query rewriting of the users. It generates the sub-queries and sends them to the specific wrappers local sources.
Execution plan Generator	It defines an execution order of deferent sub-query generated by Decomposer
Queries Executor	It carries out the operation of transmission of the sub-queries to the different wrappers and to the manager of the semantic cache.
Temporization	It allows synchronizing between the execution of the sub-queries on the local sources and the semantic cache query.
Starter	It make possible to start the operations of the overlapped sub-queries execution and the data filtering.
Controller/filter	It carries out the operations of the overlapped sub-queries execution and the data filtering.
Evaluator	Control the cost of various resources.
Decomposer	It allows to combine the results received from various queried local sources and those of the semantic cache.
Transcriptor	Supplies the final result at the users
Cache Queries Database	This source contains the users queries history for the queries submitted to the mediator
Cache results Database	This source contains the users' queries execution results.
Correspondences Rules	These rules used to bind the elements of the sources Schema to those of the Global Schema (inter-Schema s correspondence)
Conflicts Rules	These rules used to manage the Mapping phase, to solve the inter-Schema conflicts and to establish the inter-Schema correspondences
Wrapper	It is responsible for wrapping a data source in such a way that the source can interact with the rest of the integration system

Table 2: The role of the basic composites of the AXM[ed] mediator.

3.3. GLOBAL SCHEMA AND QUERIES REWRITING.

In this part, we are interested in the definition of the mediator's global Schema and the necessary stages to rewrite the users' queries and to generate sub-queries to adapt them to the different sources integrated by the mediator [9]-[11].

3.3.1. DEFINITION OF THE GLOBAL SCHEMA .

The Global Schema definition is based essentially on the identification of all the domains which model the whole of the data and services case study. These domains are modeled by a hierarchical structure, where each node represents a domain grouping sub-domains defined by the children of this node. Consequently, each node of the Global Schema integration tree is characterized by: a name and a description of the domain, a list of its attributes, a list of the integrated sources, a list of the integrated tools and a list of sub-domains generated by the father domain. The main reports of the Global Schema integration definition is based on the process of successive refinement by specialization starting from a basic federator domain (i.e. Root of the integrating tree). Moreover we suppose that each source represents a view on a sub-domain in the integration tree hierarchical structure (L.A.V Approach). This Global Schema can be described by the following integration tree:

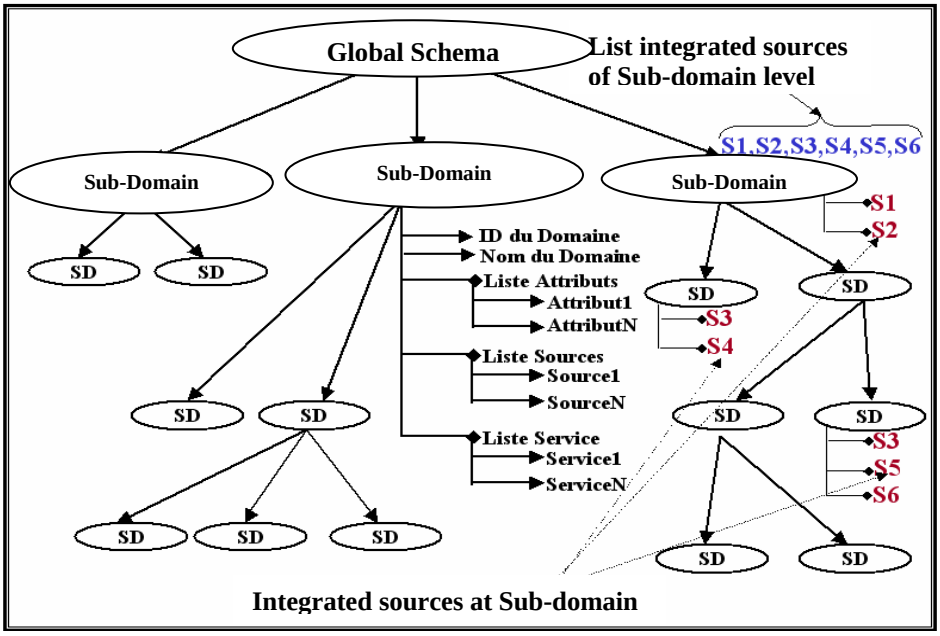

Figure 3: Integration Tree structure.

The improvement of these domains structuring allows to facilitate and optimize, at the same time, the phase of the mediator's query by the users and to generate the queries execution plan. Indeed, the user can easily explore the integration Global Schema tree to determine an optimum list of sources which can be queried by the mediator. Indeed, after having to carry out the rewriting of requests and affecting each sub-request a specific Domain in the tree of Global Schema, the mediator generates a plan of execution preestablishes, following an in-depth course of the tree. The results of execution of each sub-request are stored in the temporary memories associated the domain of the tree. At the time of the customer request evaluation and to generate the finale result required by the customer, these data will be amalgamated starting from the answers partial recorded to the level of the sheets to the root of the tree of Global Schema.

Consequently, in the mapping phase only the necessary sources are treated. This structure will also enable us to define an execution plan of sub-queries generated by the mediator. After the rewriting

phase of a query, the order of execution starts with sub-queries generated for the sources integrated into the low level domains (possibly sheets) until the federator domain [14]-[17].

For the Global Schema integration definition we propose the following basic constraints:

1. A source can be integrated by several domains.
2. The list of the sources integrated by a domain is the list of all the sources integrated by all sub-domains of this domain.
3. Sub-domains are disjoined: sources can be affected only to one and one sub-domain of the same domain.
4. If two domains (or several) of the same levels (even depth in the integration tree) integrate same sources, the level of integration of this sources moves on the level of the father domain of these domains.

The Global Schema integration can be modeled by the following DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT ShemaGlobal (Domaine*)>
<!ATTLIST ShemaGlobal
    IDShemaGlobal         ID #REQUIRED
    DescriptionShemaGlobal CDATA #REQUIRED>
<!ELEMENT Domaine (ListeSources, ListeAttributs ,
                   ListeServices, Domaine*)>
<!ATTLIST Domaine
    IDDomaine             ID #REQUIRED
    NomDomaine            CDATA #REQUIRED
    CheminDomaine         CDATA #REQUIRED>
<!ELEMENT ListeAttributs (Attribut*)>
<!ELEMENT Attribut EMPTY>
<!ATTLIST Attribut
    IDAttribut            ID #REQUIRED
    NomAttribut           CDATA #REQUIRED
    TypeAttribut          CDATA #REQUIRED>
<!ELEMENT ListeSources (Source*)>
<!ELEMENT Source EMPTY>
<!ATTLIST Source
    IDSource              ID #REQUIRED
    NomSource             CDATA #REQUIRED
    TypeSource            CDATA #REQUIRED>
<!ELEMENT ListeServices (Service*)>
<!ELEMENT Service EMPTY>
<!ATTLIST Service
    IDServices            ID #REQUIRED
    NomServices           CDATA #REQUIRED
    TypeServices          CDATA #REQUIRED>
```

3.3.2. DEFINITION OF THE MAPPING SCHEMA

One of the main problems arising from the data integration consists in carrying out the correspondence between a data source schema and the Global Schema . Generally, it is a question of laying down the rules which make it possible to bind the elements of the Schema of a source to those of the Global Schema (inter-Schema s correspondence). This makes it possible to the mediator to answer the queries of the user which are submitted on the Global Schema .

Correspondences Identification.

When the Global Schema reaches the desired level of conformity, the following stage consists in identifying the common correspondence rules. With each time that is possible, the correspondences are defined in intention. The integration process consists in finding these correspondences between the elements of the sources and those of the Global Schema . These rules form part of the integration process result. In our model, the elements in correspondences can be entities, attributes or many access paths to the attributes and methods signature, etc. These elements are varied according to the sources model (i.e. Relational, object, XML.). The correspondence elements can be summarized in the following table (Table 3):

	source Element	**global Schema Element**
Relational	Relation	Entity
	Attribute	Attribute
	Trigger	Service
Object	Class	Entity
	Attribute	Attribute
	Method	Service
XML	Entity	Entity
	Attribute	Attribute
	Path	Path

Table 3: The elements schema correspondence.

In order to well manage the mapping phase and to solve the inter-Schema conflicts problems; we define a list of basic rules to establish the following inter-Schema correspondences:

1. Each source can be identified by a view on a part of the integration tree of the Global Schema (GAV approach).
2. If a source element is in correspondence between two elements of two different domains, the constraints on the choice of a correspondence must be fixed for the management of the inter-Schema conflicts.
3. If two elements of the same source are in correspondence with two elements of two different domains, the priorities between these two domains must be defined for the generation of the execution plan.
4. Each element of a source can be in correspondence only with one element of the source integration domain sub-domains.

The Mapping Schema can be represented by the following DTD.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT ShemaMapping (Domaine*)>
<!ATTLIST ShemaMapping
    IDShemaMapping            ID #REQUIRED
    DescriptionShemaMapping CDATA #REQUIRED>

<!ELEMENT Domaine (Sources*, Services*)>
<!ATTLIST Domaine
    IDDomaine              ID #REQUIRED
    NomDomaine           CDATA #REQUIRED
    CheminDomaine        CDATA #REQUIRED>

<!ELEMENT Sources(Entité*, Attribut*)>
<!ELEMENT Entité  EMPTY>
<!ATTLIST Entité
    IDEntité               ID #REQUIRED
   Correspondence        CDATA #REQUIRED
    NomEntité             CDATA #REQUIRED
    TypeEntité            CDATA #REQUIRED>

<!ELEMENT Attribut EMPTY>
<!ATTLIST Attribut
    IDAttribut            ID #REQUIRED
    Correspondence        CDATA #REQUIRED
    RangAttribut          CDATA #REQUIRED
    NomAttribut           CDATA #REQUIRED
    TypeAttribut          CDATA #REQUIRED
    TailleAttribut        CDATA #REQUIRED >

<!ATTLIST Service
    IDServices            ID #REQUIRED
   Correspondence         CDATA #REQUIRED
    NomServices          CDATA #REQUIRED
    TypeServices         CDATA #REQUIRED
    LangageService       CDATA #REQUIRED >
```

AXMed query process.

Firstly, AXMed receives a query formulated in terms of the unified Schema and queries the cache manager, which generates tow sub-queries: the local query and the distance query. The first query is used to extract the local data stored in semantic cache. The second query is decomposed by the rewriter component into sub-queries and addressed to specific data sources. This decomposition is based on source descriptions by global Schema and mapping Schema , which play an important role in sub-queries' execution plan optimization. Finally, the sub-queries are sent to the wrappers of the individual sources, which transform them into queries over the sources. The results of these sub-queries are sent back to the mediator. At this point the answers are merged with the result of cache query by the local query and returned to the user. Besides the possibility of making queries, the mediator has no control over the individual sources [18]-[22].

The latter component (wrapper) is responsible for wrapping a data source in such a way that the source can interact with the rest of the integration system. It provides the mediator with data from the source that it is in charge of, as asked by the query execution engine. In consequence, it presents a data source as a convenient database, with the right Schema and data, appropriate for being understood and used by the mediator. This presentation Schema may be different from the real one, i.e., the internal to the data source.

AXMed query rewriting algorithm

With the collection of views, the problem to consider lies in determining how much from the real answer we get by using the pre-computed views only; and also in determining the maximum we can get in terms of the kind of views we have available. Two basic algorithms can be used [16-20], namely,

- The Bucket Algorithm: The main idea underlying the Bucket Algorithm is that we can reduce the number of query rewritings that needs to be considered if we consider each sub-goal in the query separately to determine which views may be relevant to each sub-goal.
- MiniCon Algorithm: It is an improved version of the Bucket Algorithm. As in the Bucket Algorithm, there are two steps: computing the buckets (one for each sub-goal of the query) and then computing the rewritings by using the buckets. In addition, MiniCon Algorithm pays special attention to the interaction of the variables in the query and in the view definitions, in order to prune some of the views that will be added into the buckets. This way, the number of views to be considered for the rewriting step is reduced. MiniCon Algorithm considers conjunctive queries, as Bucket Algorithm does.

For AXMed mediator the algorithm that we propose is based on a hierarchical determination of the locales sources to query. Indeed, the course of the structure of the AXMed global schema tree will enable us to define sets of sub-queries attached to the sub-domains. So, given a query Q this Algorithm finds a rewriting of Q in six steps:

1. The first stage consists in separating between the three components from the query: the list of the attributes (Select Clause), list of the fields (From clause) and lists conditions (Where clause).
2. The algorithm creates a list of attached local sources for each domain.
3. The algorithm creates a list of attached query condition sources for each source.
4. The algorithm creates a list for each sub-domain in Q that contains the views (i.e., data sources) that are relevant to answering that particular sub-domain.
5. For each sub-domain the algorithm defines an order of execution between sub-queries.
6. The algorithm tries to find query rewritings that are conjunctive queries, each consisting of one conjunct from every list of sources. For each possible choice of element from each list, the algorithm checks whether the resulting conjunction is contained in the query Q, or whether it can be made to be contained if additional predicates are added to the rewriting. If so, the rewriting is added to the answer.

4. EXAMPLE: SCIENTIFIC PUBLICATIONS MANAGEMENT

The following example presents a case study for the integration of three heterogeneous sources which manage the published articles in scientific research: Thereafter, we give the definitions of all the necessary data structures to query various local sources for the basic example presented in the preceding section. For the example of query, we suppose that the user is interested at all the articles which treat the topic of databases and which are published between 01/01/2002 and 01/01/2005 (R1 query). The three sources are:

R1 { ***Select*** *Code_Article, Nom_Article, Date_Pub*
From *Article.* }

Source 1: (**Articles**) This source contains a list of the scientific publications. We suppose that this source has only one entity: "***Papiers".***

Papiers
Code_Art
Titre_Art
Theme_Art
Date_Art

Source 2: (**Communications**) This source contains the scientific papers and also all information about the paper editors. We suppose that this source has two entities: "***Communications"***, "***Professeurs"***

Communications
Code_Com
Code_Ens
Theme_Com
Date_Com

Enseignant
Code_Ens
Nom_ Ens
Pre_ Ens
Ville_ Ens

Source 3 : (**Publications**) This source records the information of the publications and all data of laboratories concerned. We suppose that this source has two entities: "***Publication"***, "***Laboratoire"***

Publication
Code_Pub
Code_Lobo
Theme_Pub
Date_Pub

Laboratoire
Code_Labo
Nom_ Labo
Adr_ Labo
Ville_ Labo

According to the case study the basic domain application is scientific research ***(recherche)***. This domain can be refined by specializing to three sub-domains: Laboratories ***(Laboratoires)***, Articles (***Articles)*** and Researchers (***Chercheurs)***. The tree of the global Schema of this example can be modeled in the following way:

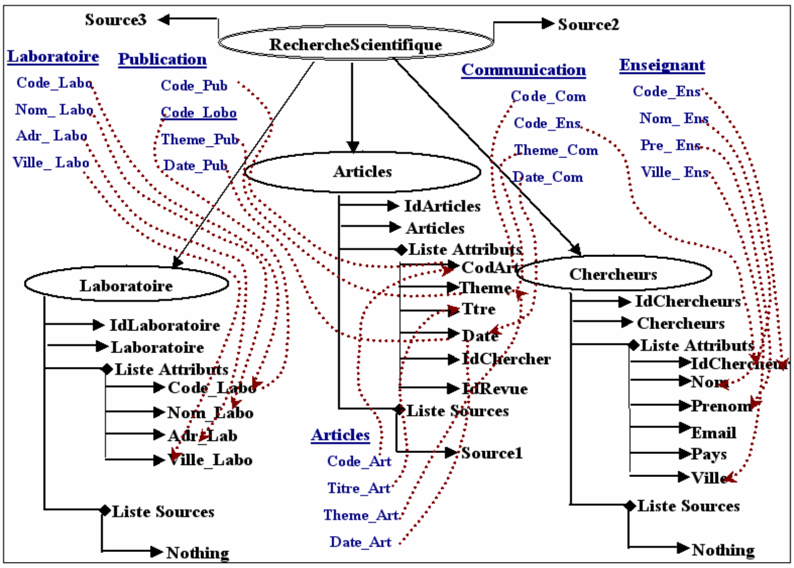

Figure 4: Integration Tree structure with local sources mapping.

The following XML source allows managing all these correspondences between the sources and the global Schema . This document “MappingSources.XML” serves the mapping phase to generate adapted queries to local sources.

<MappingSource>
<Recherche>
// Listes des attributs
// du domaine recherche
<Domaine>
<ListeSourceMappées>
<Source>
ReferenceSource #Ref(ListeSource)
NonSource=Articles
<Entite>
RefEntite= ‘RfeArti’
Nom=’Articles’
<Propriete>
Code Article=’Code_Art’
Nom Article= ‘Nom_Art’
Titre Article=’Tit_Art’
Theme Article=’The_Art’
Date Article=’Dat_Art’
</Propriete>
</Entite>
</Source>

Continu……..

<Source>
ReferenceSource #Ref(ListeSource)
<Entite>
RefEntite= ‘RfePubli’
Nom=’Publication’
<Propriete>
Code Article=’Code_Pub’
Nom Article= ‘Nom_Pub’
Titre Article=’Tit_Pub’
Theme Article=’The_Pub’
Date Article=’Dat_Pub’
</Propriete>
<Reference>
Entite=’Professeur’
Ref_Entite= ‘refProf’
Attribut=’Code_Proffeur’
</Reference>
</Entite>
</Source>
</ListeSourceMappées>
</Domaine>
<Domaine> ……….
//Etc..., add all the sources
// integrated by mediator with the same way.

With such a structure the mediator interprets the queries according to the four following phases:

Query analysis: this phase allows identifying the domains list and the attributes which take part in the definition of the users' queries, one being based on the XML document which presents the applicability and the global Schema "Schema Global.xml". With the *R1* query of the example one has:

Domains List	Attributes List
Articles, Professeurs, Laboratoires	*Nom_Article, Themes_Article, Date_Pub, Nom_Prof, Code_Prof*

Table 4 : Query analysis.

Domains	Sources List
Articles	*Source1, Source2, Source3*
Professeurs	*Source2*

Table 5 : Identification of the distant sources.

Identification of the distant sources: the role of this phase is to identify the different sources where the data required by the user can be extracted. It is a question of defining the integrated sources list at the level of each field. With our example:

Mapping of the sources: this phase presents the most important phase thus making it possible to generate the different sub-queries to send them towards the wrappers. With this intention the mediator bases itself on the XML document "MappingSources.xml" to transform the query into sub-queries respecting the local Schema of various resources integrated by the mediator. In our example and according to the identification of the resources, the mediator generates the three sub-queries addressed to the wrappers of the three sources:

Sources	Queries
Source1	*Select Code_Art, Thème_Art, Date_Art From Articles Where Date_Art beteewen D1 and D2*
Source 2	*Select Code_Pub, Thème_Pub, Date_Pub From Communications Where Date_Pub beteewen D1 and D2*
Source3	*Select Code_Art, Thème_Art, Date_Art From Publications Where Date_Art beteewen D1 and D2*

Table 6 : Mapping of the sources.

Query of the sources: The preceding tree phases make it possible to define all necessary information to query the target sources concerned. In this phase, the mediator uses this information to query suitably the different resources.

Merging the results: After the query phase, the results are sent by the adapters to the mediator, the latter merge them and presents the final result to the user.

5. AXMED MEDIATOR UML DESIGN

This part focuses on the conceptual modeling of AXMed mediator by UML method.

5.1 DEFINITION OF THE ACTORS

Usage cases are described in a textual way, decorated with a few interaction Schema s. The interactions represent the main events which occur within a mediator. The analysis begins with the research of the actors (categories of users) from the access management system. An actor represents a person, group of people or a thing which interacts with the system. The actors recruit themselves among the mediator users and also among the persons in charge of its configuration and maintenance. They are shared out in the following

Actors	Uses Case
User	An interlocutor interconnected to AXMed mediator via Internet
AXMed	A system with which the client communicates to ask for information, via a query.
WRAPPER	It hides the sources heterogeneity and offers to the mediator a homogeneous sources view. They transform the local sub-queries according to the comprehensible formulation by the target source.
Administrator	It is the system supervisor
Supplier	Provides the registered information in the local data sources.

Table 7: List of the Actors

categories:

5.2. ROLES OF EACH ACTOR.

User:

The user can thus generate queries of different formats: SQL, XQuery or XPath. The main operations of the user can be defined in the following way:

a) Ask AXM[ed] User Connection: Establishment of a connection to AXMed via a URL,
b) Requires AXM[ed] User Inscription,
c) Writing of a query: Generation of sub-queries in a predefined model.

AXMed:

The mediator is an interface between the user and the collection of given resource and services giving him the possibility to query a homogeneous and centralized information system by providing him with an integrated global Schema . The main operations of AXMed can be defined in the following way:

a) **Query Analysis**: It carries out the syntactic analysis (in accordance with grammar) and semantic in accordance with the referred view or with the query Schema ,
b) **Query Translation**: This case of use makes it possible to translate the user's query under the XML query language,
c) **Optimization of the Query**: It is the main role of the mediator to divide, according to global Schema , the users' query in several sub-queries supported by the sources,
d) **Translation of the Result to the User's Format**: To reformulate the answer to be validated in accordance with the user's query language,
e) **AXMed cache manager**. Manage the semantic cache of the mediator

WRAPPER:

It is an interface allowing the translation of information between the mediator and the data sources. Then, the principal operations of the WRAPPER can be defined in the following way:

a) Queries Translation: It translates the queries sent by the mediator into comprehensible sub-queries by the data sources,
b) Send translated queries to the sources,
c) Retrieval and new translation of the answers: Translate the answers sent by the data sources into XML answers,
d) Send answers to the mediator.

Administrator:
It is the supervisor of the system. The main operations of the administrator can be defined in the following way:

a) Management of the AXMed User,
b) Management of the Global Schema,
c) Management of the Local Schemas: to create the local schema for each source to be integrated.

Supplier:
It is the distributor of data from the local sources,

a) Extraction of the data,
b) Control of the data integrity and safety.

5.3. DIAGRAM OF USE CASES.

Thereafter, we are interested in the representation of the diagrams for the two basic actors: user and AXMed.

User:

c) List of uses case:

Name
Client request of the AXMed connection.
Client request of the AXMed Inscription.
Writing of the queries.

Table 8: Uses case of the client.

d) Use Cases Diagram:

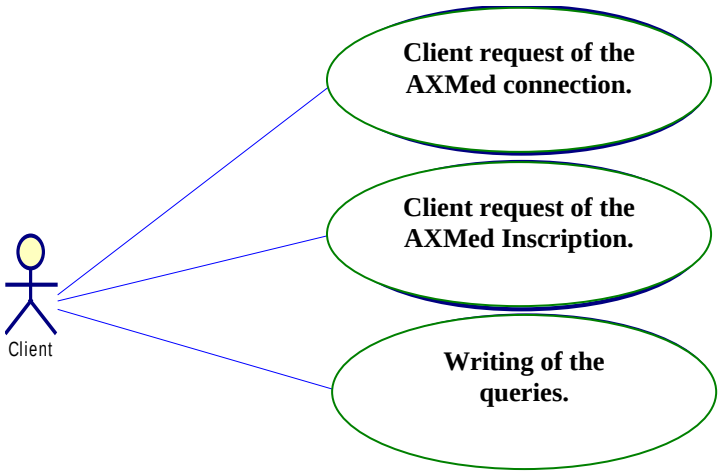

Figure 5: Use cases diagram of client

AXMed :

d) List of use cases

Name
Queries translation.
Analysis of the queries.
Queries Optimization.
Results translation to client format.
Management of the AXMed Cache

Table 9: Uses case of the AXMed Mediator.

e) Use Cases Diagram:

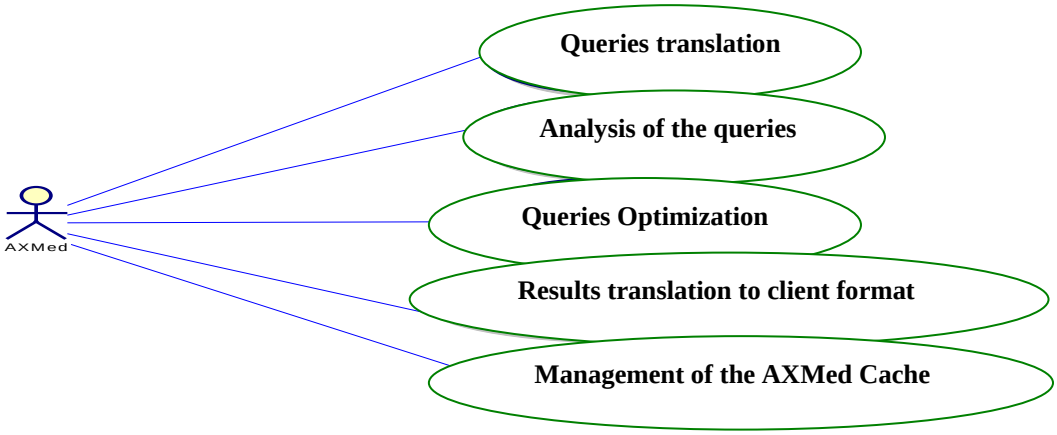

Figure 6: Use cases diagram of AXMed Mediator

5.4. SEQUENCE DIAGRAMS:

The sequence diagrams show interactions between objects. Its representation concentrates on the sequence of the interactions according to a chronological point of view. They are, in general, adapted to model in real time the dynamic aspects of the systems and the complex scenarios involving different objects.

User: Establish of the connection:

a) List of the messages:

Name	Receiver	Sender
Connection query	AXMed	Client
Open connection verification	AXMed	AXMed
Enter login and password	AXMed	Client
Access Right verification	AXMed	AXMed
Establish the connection	AXMed	Client

Table 10 : List of the messages between the Client and AXMed Mediator.

b) Sequence Diagram:

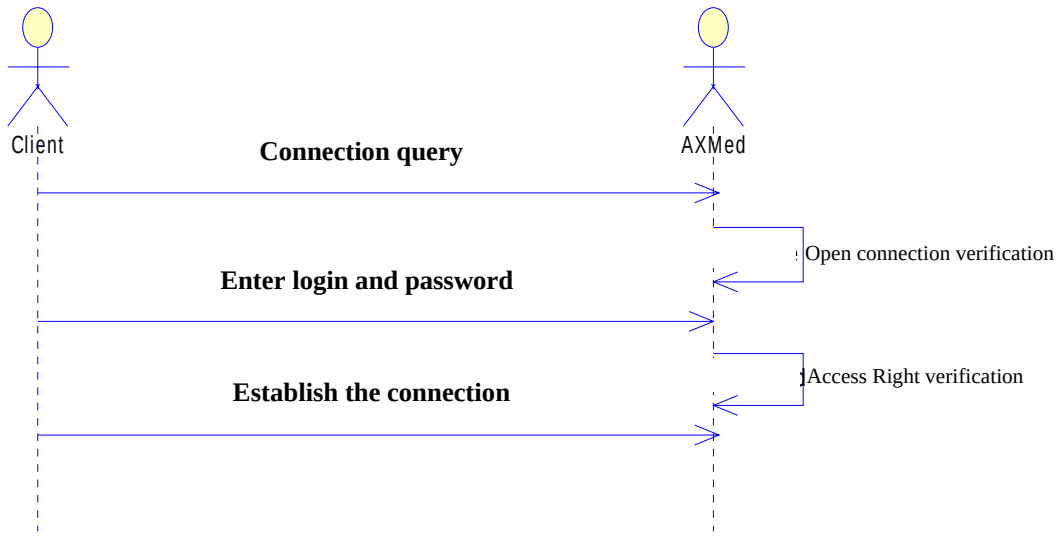

Figure 7: Sequence diagram : Establish of the connection

AXMed :

a) List of messages:

Name	Receiver	Sender
Send a query	AXMed	Client
Translate the query into a comprehensible language by the source	Wrapper	Wrapper
Send the query translated to the sources	Supplier	Wrapper
Extract information required	Supplier	Supplier
Send the corresponding results	Wrapper	Supplier
Retranslate the answers in into a comprehensible language by the mediator	Wrapper	Wrapper
Send the Translation result	AXMed	Wrapper
User Result Format	Client	AXMed
Optimizing the query	AXMed	AXMed
Send the Sub query	WR	AXMed
Analyzes the Translation query	AXMed	AXMed
Assemble the query results	AXMed	AXMed

Table 11 : List of the messages between the Client, AXMed Mediator, Wrapper and Supplier.

b) Diagram of Sequence

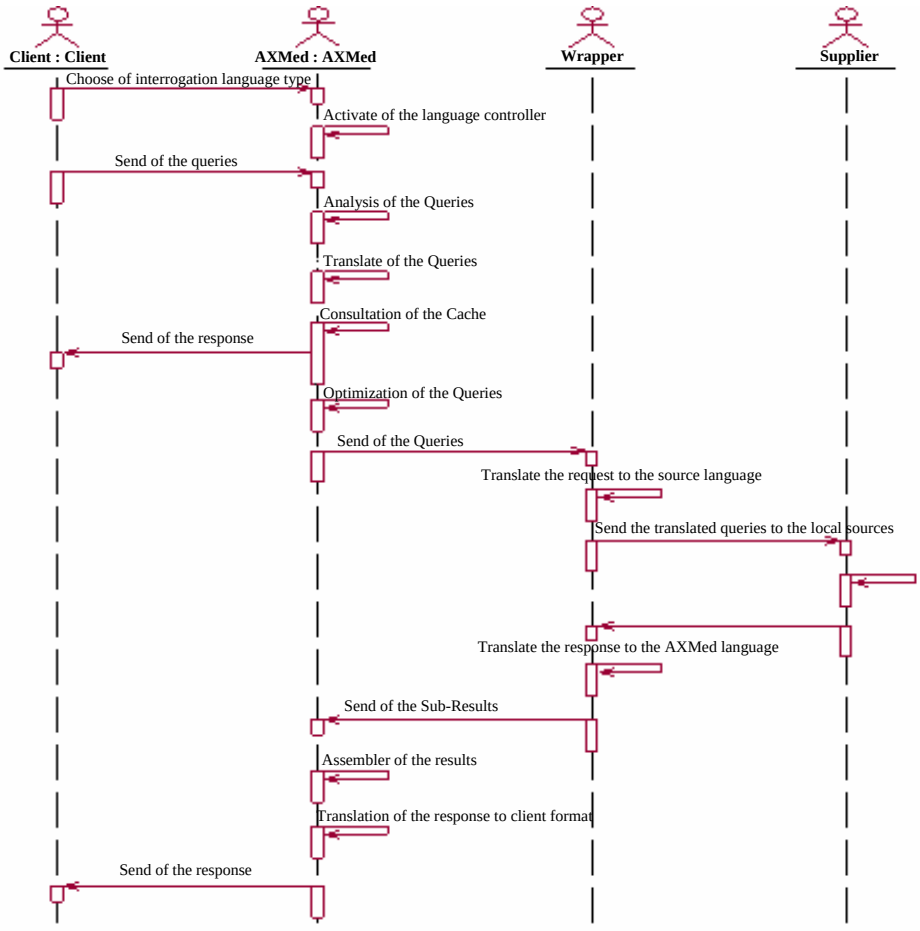

Figure 8: Sequence diagram: AXMed Interrogation

6. CONCLUSIONS

The proposed architecture satisfies almost all requirements for a mediator allowing an efficient integration of heterogeneous information systems. Besides the integration of different kinds of data sources it offers now a more flexible way of extending the system. Our Mediation system currently provides following features:

- With such mediation architecture for information systems it is possible to make several information systems with different designs and architectures cooperate.
- Several heterogeneous data sources can be easily integrated, updated or just removed from the system by simply changing the global Schema .
- A large amount of available databases, structured text files and Web Services are supported due to already available wrappers. Of course it is possible to write own wrappers that import other currently not supported data sources.
- The mapping process is carried out by certain simple mapping actions

As the sources may contain overlapping information several services may refer to specific tasks. Furthermore, AXMed mediator must provide a common interface to access to the different services used by all the integrating locale sources. These tasks still have to be done in future.

7. Bibliography

[1]. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, J. Widom, "*The TSIMMIS Project: Integration of Heterogeneous Information Sources*", IPSJ Conference, pp 7-187, Tokyo, Japan, October 1994.

[2]. G. Gardarin, A. Mensch, T. Dang-Ngoc, L. Smit, "*Integrating Heterogeneous Data Sources with XML and XQuery. e-XMLMedia*".

[3]. G. Wiederhold, "*Intelligent Integration of Information*", ACM SIGMOD Conf. On Management of Data, pp. 434-437, Washington D.C, USA, May 1993.

[4]. Z. Lacroix, O. Boucelma, "*InterMed : Interfaçe de médiation pour systèmes d'information*", *Revue ISI. Volume X-n° x/2002, pp. 1- X, 2002.*

[5]. L. Bright, L-R. Gruser, L. Raschid, M.E Vidal, "*A Wrapper Generation Toolkit to Specify and Construct Wrappers for Web accessible Data Sources (WebSources)*", Journal of Computer systems Science and Engineering. Special Issue: Semantics on the World Wide Web, 2002.

[6]. L. Manolescu, D. Florescu, D. Kossmann, "*Answering XML Queries over Heterogeneous Data Sources"*, 27th Very Large Data Bases, pp. 241-2560, Roma, Italy, Sept 2001.

[7]. J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderburk, "*Querying XML Views of relational Data*", 27th Very Large Data Bases, pp. 241-2560, Roma, Italy, Sept 2001.

[8]. B. Amann, C. Beeri, I. Fundulaki, M. Scholl, "*Ontology-Based Integration of XML Web Resources"*, In International Semantic Web Conference (ISWC), Sardinia, Italy, 2002.

[9]. C. Baru, A. Gupta, B. Lud¨ascher, R. Marciano, Y. Papakonstantinou, P. Velikhov, V. Chu. "*XML-based information mediation with MIX*", In Demonstrations, ACM/SIGMOD, pp. 597–599, 1999.

[10]. V. Christophides, S. Cluet, J. Simeon, "*On Wrapping Query Languages and Efficient XML Integration*", In Proc. of ACM SIGMOD, Dallas, USA, May 2000.

[11]. A. Levy, D. Srivastava, T. Kirk, "*Data Model and Query Evaluation in Global Information Systems*", Journal of Intelligent Information Systems, 1995.

[12]. M. Ezziyyani, M. Bennouna, M. Esaaidi, , "*Advanced XML Mediator For Heterogeneous Information Systems – AXMed* ", ICTIS'05, Maroc, Tetoan, June 2005In Proc.

[13]. M. Ezziyyani, M. Bennouna, M. Esaaidi, , "*Information System Integration through Web Services Technology*", ICTIS'05, Maroc, Tetoan, June 2005.

[14]. A. Levy, A. Rajaraman, J. Ordille, "*Querying Heterogeneous Information Sources Using Source Descriptions*", In Proc. VLDB, pp. 251–262, Mumbai (Bombay), India, September 1996.

[15]. R. Pottinger, A. Levy, "*A Scalable Algorithm for Answering Queries Using Views*", In Proc. VLDB, pp. 484–495, Cairo, Egypt, September 2000.

[16]. Y. Papakonstantinou, H. Garcia-Molina, J. Widom, "*Object Exchange Across Heterogeneous Information Sources*", In Proc. ICDE, pp. 251–260, Taipei, Taiwan, March 1995.

[17]. S. Cluet, C. Delobel, J. Siméon, K. Smaga, "*Your Mediators Need Data Conversion*", ACM SIGMOD Intl. Conf. on Management of Data, pp. 177-188, Seattle, Washington, USA, 1998.

[18]. I. Fundulaki, B. Amann, C. Beeri, M. Scholl, "*STYX : Connecting the XML World to the World of Semantics*", In Proceedings of EDBT, Prague, Czech Republic, March 2002.

[19]. F. Goasdoue, V. Lattes, M-C. Rousset, "*The use of CARIN language and algorithms for information integration: The PICSEL System*", International Journal on Cooperative Information Systems, 2000.
[20]. Y. Halevy, "*Theory of answering queries using views*", SIGMOD Record, pp. 40–47, 2000.
[21]. I. Manolescu, D. Florescu, D. Kossmann, "*Answering XML Queries over Heterogeneous Data Sources*", In Proc. VLDB, Rome, Italy, September 2001.
[22]. L. Haas, D. Kossman, E. Wimmers, J. Yang, "*Optimizing Queries across Drivers Data Sources*", 23ed Very Large Data Bases, August 1008, Athens, Greece 1997.
[23]. D. Chamberlin, D. Florescu, J. Robie, J. Siméon, M. Stefanescu, 2000, "*XQuery: A Query Language for XML*", W3C, http://www.w3.org/TR/xmlquery.
[24]. C. Li, A Eike. A. Rundensteiner, Song Wang, "*Xcache - A semantic Caching System for XML Queries*", ACM SIGMOD, June 2002.
[25]. P. Buneman, S. B. Davidson, W. Fan, C. S. Hara, W. C. Tan, "*Keys for XML*", In Proc.WWW10, pp. 201–210, 2001.
[26]. D. Chamberlin, D. Florescu, J. Robie, J. Siméon, M. Stefanescu, "*XQuery : A Query Language for XML*", *W3C, 2004.*
[27]. J. Clark, S. DeRose, "*XML Path Language (XPath) Version 1.0*", W3C Recommendation, http://www.w3c.org/TR/xpath, November 1999.