Warehousing complex data from the web

O. Boussaïd*, J. Darmont*, F. Bentayeb
and S. Loudcher

ERIC
University of Lyon 2
5 Avenue Pierre Mendès-France
69676 Bron Cedex, France
Fax: +33 478 772 375
E-mail: Omar.Boussaid@univ-lyon2.fr
E-mail: Jerome.Darmont@univ-lyon2.fr
E-mail: Fadila.Bentayeb@univ-lyon2.fr
E-mail: Sabine.Loudcher@univ-lyon2.fr
*Corresponding authors

Abstract: Data warehousing and Online Analytical Processing (OLAP) technologies are now moving onto handling complex data that mostly originate from the web. However, integrating such data into a decision-support process requires their representation in a form processable by OLAP and/or data mining techniques.

We present in this paper a complex data warehousing methodology that exploits eXtensible Markup Language (XML) as a pivot language. Our approach includes the integration of complex data in an ODS, in the form of XML documents; their dimensional modelling and storage in an XML data warehouse; and their analysis with combined OLAP and data mining techniques. We also address the crucial issue of performance in XML warehouses.

Keywords: data warehousing; Online Analytical Processing; OLAP; Decision Support System; DSS; complex data; data web; complex data ETL process; complex data warehouse; XML warehousing; XML cube; X-warehousing.

Reference to this paper should be made as follows: Boussaïd, O., Darmont, J., Bentayeb, F. and Loudcher, S. (2008) 'Warehousing complex data from the web', *Int. J. Web Engineering and Technology*, Vol. 4, No. 4, pp.408–433.

Biographical notes: Omar Boussaïd is an Associate Professor in Computer Science at the School of Economics and Management of the University of Lyon 2, France. He received his PhD degree in Computer Science from the University of Lyon 1, France in 1988. Since 1995, he has been in charge of the Master's degree 'Computer Science Engineering for Decision and Economic Evaluation' at the University of Lyon 2. He is a member of the Decision Support Databases research group within the ERIC laboratory. His main research subjects are data warehousing, multidimensional databases and OLAP. His current research concerns complex data warehousing, XML warehousing, data mining-based multidimensional modelling, combining OLAP and data mining, and mining metadata in RDF form.

Jerome Darmont received his PhD degree in Computer Science from the University of Clermont-Ferrand II, France in 1999. He has been an Associate Professor at the University of Lyon 2, France since then, and became the head of the Decision Support Databases research group within the ERIC

laboratory in 2000. His current research interests mainly relate to the evaluation and optimisation of database management systems and data warehouses (benchmarking, auto-administration, optimisation techniques, *etc.*), but also include XML and complex data warehousing and mining, and medical or health-related applications.

Fadila Bentayeb has been an Associate Professor at the University of Lyon 2, France since 2001. She is a member of the Decision Support Databases research group within the ERIC laboratory. She received her PhD degree in Computer Science from the University of Orléans, France in 1998. Her current research interests involve database management systems, including the integration of data mining techniques into DBMSs and data warehouse design, with a special interest for schema evolution, XML and complex data warehousing, benchmarking and optimisation techniques.

Sabine Loudcher Rabaseda is an Associate Professor in Computer Science at the Department of Statistics and Computer Science of the University of Lyon 2, France. She received her PhD degree in Computer Science from the University of Lyon 1, France in 1996. Since 2000, she has been a member of the Decision Support Databases research group within the ERIC laboratory. Her main research subjects are data mining, multidimensional databases, OLAP and complex data. Since 2003, she has been the Assistant Director of the ERIC laboratory.

1 Introduction

Decision-support technologies, including data warehouses and Online Analytical Processing (OLAP), are nowadays technologically mature. However, their complexity makes them unattractive to many companies; hence, some vendors develop simple web-based interfaces (Lawton, 2006). Furthermore, many decision-support applications necessitate external data sources. For instance, performing competitive monitoring for a given company requires the analysis of data available only from its competitors. In this context, the web is a tremendous source of data, and may be considered as a farming system (Hackathorn, 2000).

There is indeed a clear trend towards online data warehousing, which will give way to new approaches such as virtual warehousing (Belanger *et al.*, 1999) or eXtensible Markup Language (XML) warehousing (Baril and Bellahsène, 2003; Hummer *et al.*, 2003; Nassis *et al.*, 2005; Park *et al.*, 2005; Pokorný, 2002; Vrdoljak *et al.*, 2005; Zhang *et al.*, 2005). Data from the web are not only numerical or symbolic, but may be:

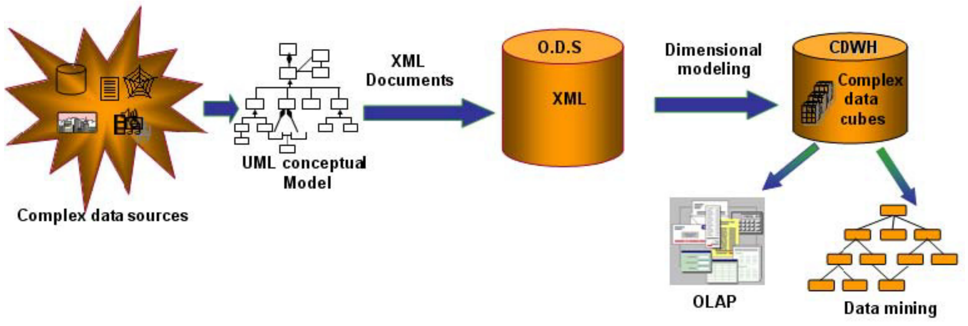
- represented in various formats (databases, texts, images, sounds, videos, *etc.*)
- diversely structured (relational databases, XML documents, *etc.*)
- originating from several different sources
- described through several channels or points of view (a video and a text that describe the same meteorological phenomenon, data expressed in different scales or languages, *etc.*)
- changing in terms of definition or value (temporal databases, periodical surveys, *etc.*).

We term data that fall in several of the above categories ‘complex data’ (Darmont *et al.*, 2005). Managing such data involves a lot of different issues regarding their structure, storage and processing (Darmont and Boussaïd, 2006); and classical data warehouse architectures must be reconsidered to handle them.

In this context, our motivation is to integrate complex data from the web into a decision-support process, which requires the integration and the representation of complex data under a form that can be processed by online analysis and/or data mining techniques (Darmont *et al.*, 2003). We propose a full, generic data warehousing and online analysis process (Figure 1) that includes two broad axes:

- data warehousing, including complex data integration and modelling
- complex data analysis.

Figure 1 Complex data warehousing and analysis process (see online version for colours)



To preserve the generic aspect of this process, we must exploit a universal formalism for modelling and storing any form of complex data. The XML formalism has emerged as a dominant W3C[1] standard for describing and exchanging semistructured data among heterogeneous data sources. Its self-describing hierarchical structure enables a manipulative power to accommodate complex, disconnected and heterogeneous data. Furthermore, XML documents may be validated against an XML Schema. It allows describing the structure of a document and constraining its contents. With its vocation for semistructured data exchange, the XML language offers great flexibility for representing heterogeneous data, and great possibilities for structuring, modelling and storing them.

The approach we propose consists in representing complex data as XML documents and in physically integrating them into an Operational Data Storage (ODS), which is a buffer ahead of the actual data warehouse. Then, we recommend an additional layer to model complex data and prepare them for analysis. Complex data in the form of XML documents are thus multidimensionally modelled to obtain an XML data warehouse. Finally, complex data analysis can take place from this warehouse, with online analysis, data mining or a combination of the two approaches.

One originality of our approach is that we do not envisage data mining only as a front-end, stand-alone analysis tool. We exploit data mining techniques throughout the whole complex data warehousing process:

- at the data integration stage, *e.g.*, to extract semantic information from complex data
- in the multidimensional modelling phase, *e.g.*, to discover pertinent measures or dimensions
- when analysing data, *e.g.*, by coupling OLAP and data mining
- in support of database administration, *e.g.*, to automatically select pertinent indexes or materialised views.

The objective of this paper is to present the issues we identified on the path to designing and implementing this approach, as well as the solutions we devised to solve them. The remainder of this paper is organised as follows: Section 2 presents our approach for complex data integration. Section 3 details the *X-Warehousing* XML warehousing platform. Section 4 describes sample complex data analyses. Section 5 addresses performance problems in XML data warehouses. Finally, Section 6 concludes this paper and discusses open issues.

2 Complex data integration

2.1 Context and issues

Companies collect huge amounts of heterogeneous and complex data. They aim at integrating these data in their Decision Support Systems (DSSs), and some efforts are needed to structure them and to make them as homogeneous as possible. In data warehousing, the prime objective of storing data is to facilitate the decision process. To achieve the value of a data warehouse, incoming data must be transformed into an analysis-ready format. In the case of numerical data, data warehousing systems often provide tools to assist in this process. Unfortunately, standard tools are inadequate for producing a relevant analysis axis when data are complex. In such cases, the data warehousing process should be adapted in response to evolving data and information requirements. We need to develop tools to provide the needed analysis.

In a data warehousing process, the data integration phase is crucial. Data integration is a hard task that involves reconciliation at various levels (data models, data schema, data instances, semantics). Indeed, the special nature of complex data poses different and new requirements to data warehousing technologies, over those posed by conventional data warehousing applications. Hence, to integrate complex data sources, we need more than a tool for organising data into a common syntax.

Two main and opposing approaches are used to perform data integration over heterogeneous data sources. In the mediator-based approach (Rousset, 2002), the different data remain located at their original sources. User queries are executed through a mediator-wrapper system (Goasdoué *et al.*, 2000). A mediator reformulates queries according to the content of the various accessible data sources, while the wrapper extracts the selected data from the target source. The major advantage of this approach is its flexibility, since mediators are able to reformulate and/or approximate queries to better satisfy the user. However, when the data sources are updated, the modified data are lost. This is not pertinent in a decision support system where the historicity of data is necessary.

On the opposite side, in the data warehouse approach (Inmon, 1996; Kimball, 1996), all the data from the various data sources are centralised in a new multidimensional database, the data warehouse. In a data warehouse context, data integration corresponds to the Extract, Transform, Load (ETL) process that accesses, cleans and transforms the heterogeneous data before they are loaded into the data warehouse. This approach supports the dating of data and is tailored for analysis.

In this section, we present our approach for complex data integration based on both data warehouse technology and Multiagent Systems (MASs). Our aim is to take advantage of the MASs, intelligent programs composed of a set of agents, each one offering a set of services, to achieve complex data integration. Indeed, we can assimilate the three steps of the ETL process to services carried out by specific agents.

2.2 *Proposed solution*

2.2.1 *Complex data ETL process*

The classical ETL approach proceeds in three steps. The first phase, *extraction*, includes understanding and reading the data source, and copying the necessary data in a buffer called the preparation zone. Then, the *transformation* phase proceeds in successive steps: clean the data from the preparation zone; discard some useless data fields; combine the data sources; and build aggregates to optimise the most frequent queries. In this phase, metadata are essential to store the transformation rules and various correspondences. The third phase, *loading* stores the prepared data into multidimensional structures (data warehouse or data marts).

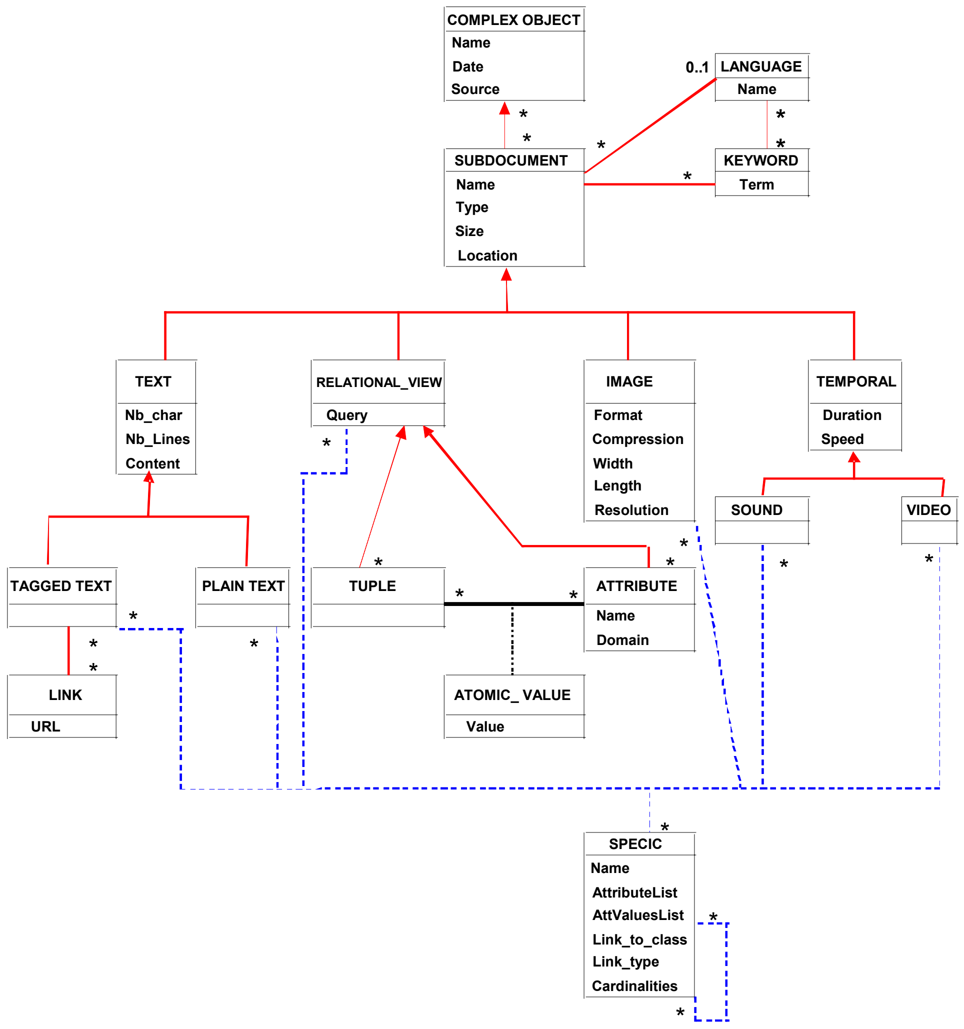
To achieve complex data integration following the warehouse approach, the traditional ETL process is ill-adapted. We present here our approach to accomplish the ETL process in an original way. We propose a modelling process to achieve the integration of complex data into a data warehouse (Boussaïd *et al*., 2003). We first design a conceptual UML model for a complex object (Boussaïd *et al*., 2006). The UML conceptual model is then directly translated into an XML Schema, which we view as a logical model. The obtained logical model may be mapped into a relational, object-relational or XML-native database.

2.2.2 *Complex data UML model*

First, we present a generic UML model that allows us to model not only low-level but also semantic information concerning the complex data to be analysed. After we transform this UML model into an XML grammar (as a Document Type Definition – DTD – or an XML Schema), we generate the XML documents describing the complex data. Therefore, we integrate complex data as XML documents into an ODS as a first step in complex data warehousing.

We choose to integrate the characteristics of data rather than the original data themselves. We use XML to describe the characteristics of our complex data as it encloses not only the content of the complex data, but also the way they are structured. The basic characteristics (*e.g.*, *file size*, *file name*, *duration* for films or sounds, and *resolution* for images) can be extracted automatically. These characteristics capture low-level information concerning the original data. The generic model that we present (Figure 2) allows us to add semantic characteristics of data in order to enrich their description by manual annotations or by knowledge automatically extracted from data mining techniques.

Figure 2 Generic complex data UML model (see online version for colours)



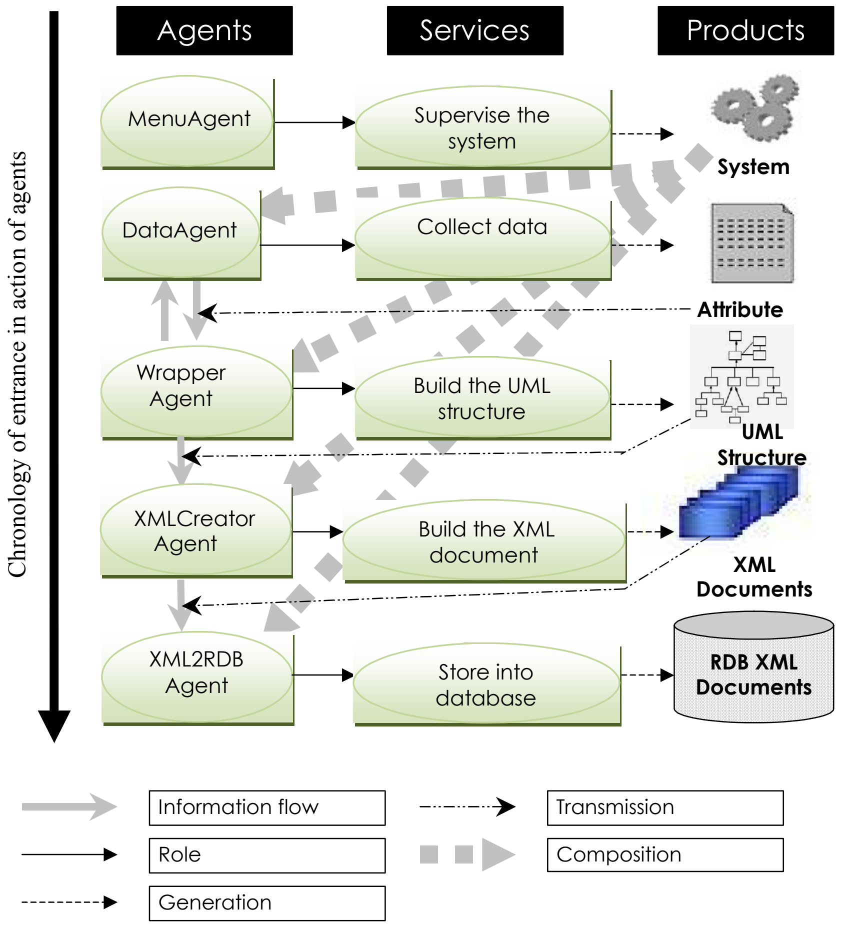
The UML class diagram represents a complex object generalising all complex data types. Note that our goal here is to propose a general data structure: the list of attributes for each class in this diagram is intentionally not exhaustive. Our generic model defines a complex object which is composed of complex data represented as subdocuments. The subdocuments have special predefined low-level characteristics that depend on the type of complex data they contain. The (meta)class *Specific* in our model is a generic class that allows us to define new classes and relationships in the UML model, and thus enables modelling semantic characteristics of complex data. It allows not only describing the semantic properties of data, but also any other useful characteristic. Every class linked to the class *Specific* in the UML model can take advantage of it when instantiating the model (at implementation time), by defining its own new characteristics.

2.2.3 MAS-based ETL approach

A MAS is a collection of actors that communicate with each other (Sycara and Zeng, 1996). Moreover, each agent (actor) is able to offer specific services and has a well-defined goal. Each agent is able to perform several tasks, in an autonomous way, and communicate the results to a receiving actor (human or software). A MAS must respect the programming standards defined by the Foundation for Intelligent Physical Agents (FIPA, 2002).

Our MAS-based integration approach is a flexible and evolutive architecture to which we can add, remove or modify services, and even create new agents (Figure 3) (Boussaïd *et al.*, 2003). To validate our approach, we have developed a MAS-based ETL prototype: SMAIDoC, which is freely available online (BDD-ERIC, 2003).

Figure 3 MAS-based ETL architecture for complex data integration (see online version for colours)



We have instantiated five agents that allow the integration of complex data. The purpose of this collection of agents is to perform several tasks. The first main agent in our prototype, *MenuAgent*, pilots the system, supervises agent migrations and indexes the accessible sites from the platform. Some other default pilot agents help in the management of the agents and provide an interface for the agent development platform. Two agents, named *DataAgent* and *WrapperAgent*, model the input complex data into UML classes. Finally, the *XMLCreator* agent translates UML classes into XML documents that are mapped into a relational database by the *XML2RDBAgent* agent or are stored as a collection of XML documents.

To develop our prototype, we have built a platform using JADE (2002) version 2.61 and the Java language (Sun Microsystems, 2002), which is portable across agent programming platforms.

2.3 Perspectives

From a technical point of view, we can extend the services offered by SMAIDoC, especially for extracting data from their sources and analysing them. For example, the *DataAgent* agent could converse with online search engines and exploit their answers. We could also create new agents in charge of modelling data in a multidimensional way, and applying analysis methods such as OLAP or data mining.

Finally, we aim at studying a metadata representation of the results of data mining techniques that generate rules in mixed structures, combining XML Schema and the Resource Description Framework (RDF). These description languages are indeed well suited for expressing semantic properties and relationships between metadata.

Therefore, SMAIDoC is designed as an incremental and progressive platform and as a technical support for our complex data integration method, whose main objective is to describe and store complex data into the XML documents.

3 The X-warehousing platform

3.1 Context and issues

In our approach to complex data integration, we chose XML to describe and to store data in an ODS. At this stage, it is possible to mine the stored XML documents directly with the help of adapted techniques such as XML structure mining (Section 4.1). Otherwise, in order to analyse these XML documents efficiently, it is interesting to warehouse them. Therefore, new efforts are needed to integrate XML in classical business applications. Feeding data warehouses with XML documents is also becoming a challenging issue, since the multidimensional organisation of data is quite different from the semistructured organisation. The difficulty consists in carrying out a multidimensional design within a semistructured formalism like XML.

Nevertheless, in the literature, we distinguish two separate approaches in this field. The first approach focuses on the physical storage of XML documents in data warehouses. XML is considered an efficient technology to support data within structures well suited for interoperability and information exchange, and can definitely help in feeding data warehouses. Baril and Bellahsène (2003) introduce the View Model, onto which they build an XML warehouse called DAWAX (DAta WArehouse for XML). Hummer *et al.* (2003) propose an approach that focuses on the exchange and the transportation of data cubes over networks, rather than multidimensional modelling with XML.

The second approach aims at using XML to design data warehouses according to classical multidimensional models such as star and snowflake schemas. Pokorný (2002) uses a sequence of DTDs to make explicit dimension hierarchies that are logically associated, about the same way referential integrity is achieved in relational databases. Golfarelli and Rizzi (1999) introduce a Dimensional Fact Model represented by Attribute Trees. They also use XML Schemas to express multidimensional models, by including relationships in subelements. Trujillo *et al.* (2004) also provide a DTD model from which valid XML documents are generated to represent multidimensional models at a conceptual level. Nassis *et al.* (2004) propose a similar approach, where an Object Oriented (OO) standard model is used to develop a conceptual model for XML Document Warehouses (XDW). An XML repository, called xFACT, is built by integrating OO concepts with XML Schemas. They also define virtual dimensions by using XML and UML package diagrams in order to help in the construction of hierarchical conceptual views.

Since we are able to describe complex data in XML documents and we need to prepare them for future OLAP analyses, storing them in a data repository is not a sufficient solution. Rather we need to express through these documents a more interesting abstraction level that is completely oriented towards analysis objectives. To achieve this goal, we propose an approach called X-Warehousing (Boussaïd *et al.*, 2006), which is entirely based on XML, to warehouse complex data. It allows building a collection of homogeneous XML documents. Each document corresponds to an OLAP fact where the XML formalism structures data according to a multidimensional model.

3.2 Proposed solution

We include in our approach a methodology that enables the use of XML as a logical modelling formalism for data warehouses. This methodology starts from analysis objectives defined by users according to a Multidimensional Conceptual Model (MCM). Therefore, we focus on analysis needs rather than on the data themselves. The *X-Warehousing* approach (Figure 4) accepts a reference MCM and XML documents as input. In fact, through the reference MCM, a user may design a data warehouse by defining facts, dimensions and hierarchies. Despite the use of a star schema or snowflake schema, the MCM depicts an analysis context independently of its logical and physical representation. The MCM is then transformed into a logical model via an *XML Schema* (XSD file).

In a second step, an *attribute tree* is automatically generated from this XSD file. An *attribute tree* is a directed, acyclic and weakly connected graph that represents a warehouse schema (Golfarelli *et al.*, 2001). Once the reference model is defined, we can submit XML documents to feed the designed warehouse. *XML Schemas* and *attribute trees* are also extracted from the input XML documents. We transform the reference model and the XML documents into *attribute trees* in order to make them comparable.

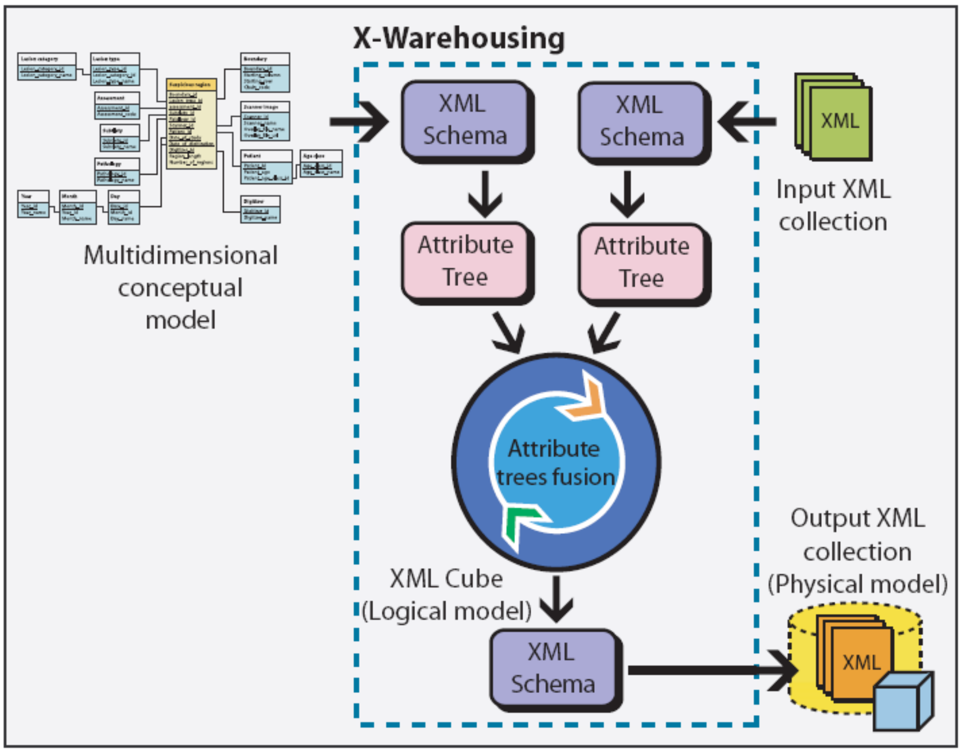
In fact, two *attribute trees* can easily be merged through a fusion process based on *pruning* and *grafting* functions (Golfarelli *et al.*, 2001). At this stage, two cases are possible:

1 If an input document contains the minimum information required in the reference MCM, the document is accepted and merged with the MCM. An instance of the XML documents is created and validated against the resulting *XML Schema*. This new *XML Schema* represents the logical model of the final *XML Cube*.

2 If a submitted document does not contain enough information to represent an OLAP fact according to the reference MCM, the document is rejected and no output is provided.

The goal of this condition is to obtain a homogeneous collection of data with minimum information to feed the final *XML Cube*.

Figure 4 Overview of the *X-Warehousing* approach (see online version for colours)



The benefit of our approach is quite important since organisations are treating domains of complex applications. In these applications, a special consideration is given to the integration of heterogeneous and complex information in DSSs. For example, in *breast cancer researches* (Heath *et al*., 2000), experts require efficient representations of mammographic exams. Note that information about a mammogram comes from different sources such as texts, annotations by experts and radio scanners. We think that structuring such a set of heterogeneous data within an XML format is an interesting solution for warehousing them. Nevertheless, this solution is not sufficient for driving future analyses. We propose to structure these data in XML with respect to the multidimensional reference model of a data warehouse. Output XML documents of the

X-Warehousing process represent the physical model of the data warehouse. Each output document corresponds to the multidimensional structured information of an OLAP fact.

3.2.1 Modelling a warehouse with XML

According to the properties of the XML documents, we propose to represent the above conceptual data warehouse models (*star schema* and *snowflake schema*) with XML. More precisely, we use *XML Schemas* to define the structure of a data warehouse. To formulate a *star schema* of a data warehouse in XML, we define the notion of an *XML star schema* as follows:

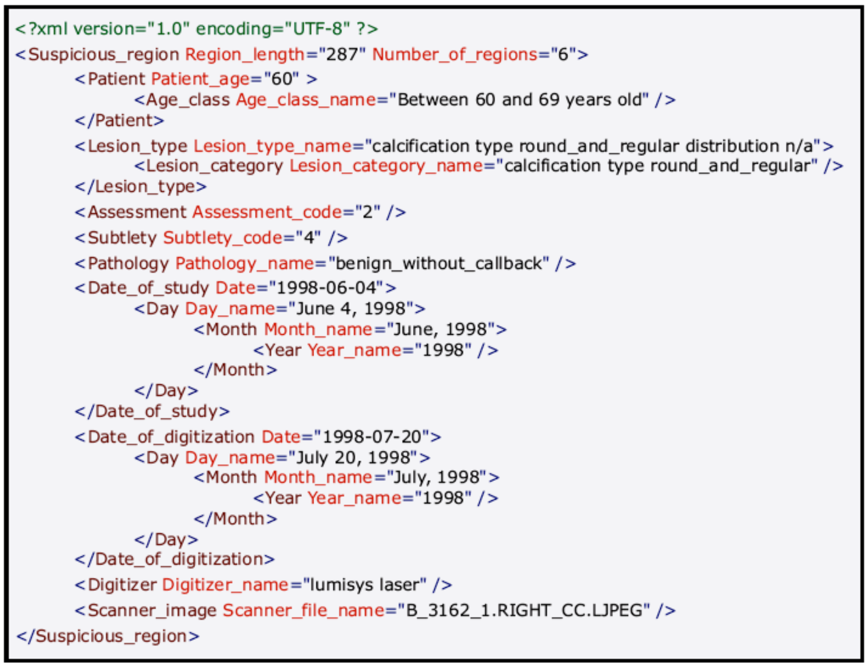
Definition: XML star schema. Let $(F, \mathcal{D})$ be a star schema, where F is a set of facts having m measure attributes $\{F.M_q, 1 \leq q \leq m\}$ and $\mathcal{D} = \{D_s, 1 \leq s \leq r\}$ is a set of r independent dimensions where each D_s contains a set of n_s attributes $\{D_s.Ai, 1 \leq i \leq n_s\}$. The XML star schema of $(\mathrm{F}, \mathcal{D})$ is an XML Schema where (1) F defines the XML root element in the XML Schema; (2) $\forall q \in \{1,\ldots, m\}$, $F.M_q$ defines an XML attribute included in the XML root element; (3) $\forall s \in \{1,\ldots, r\}$, D_s defines as many XML subelements of the XML root element as the number of times it is linked to the set of facts F; (4) $\forall s \in \{1,\ldots, r\}$ and $\forall i \in \{1,\ldots, n_s\}$, $D_s.A_i$ defines an XML attribute included in the XML element D_s.

Knowing that the XML formalism allows us to embed multilevel subelements in one XML tag, we use this property to represent XML hierarchies of dimensions. Let $H = \{D_1,\ldots, D_t,\ldots, D_l\}$ be a dimension hierarchy. We can represent this hierarchy by writing D_1 as an XML element and $\forall t \in \{2,\ldots, l\}$, D_t being an XML subelement of the XML element $D_{t\text{–i}}$. The attributes of each D_t are defined as XML attributes included in the XML element D_t. Since a dimension may have some hierarchies, it is possible to describe each one by an XML element with its embedded subelements. Therefore, we can also define the notion of the *XML snowflake schema*, which is the XML equivalent of a conceptual *snowflake schema*:

Definition: XML snowflake schema. Let $(F, \mathcal{H})$ be a star schema, where F is a set of facts having m measure attributes $\{F.M_q, 1 \leq q \leq \mathrm{m}\}$ and $\mathcal{H} = \{H_s, 1 \leq s < r\}$ is a set of r independent hierarchies. The XML snowflake schema of $(F, \mathcal{H})$ is an XML Schema where (1) F defines the XML root element in the XML Schema; (2) $\forall q \in \{1,\ldots, m\}$, $F.M_q$ defines an XML attribute included in the XML root element; (3) $\forall q \in \{1,\ldots, r\}$, H_s defines as many XML dimension hierarchies, like subelements of the XML root element, as the number of times it is linked to the set of facts F.

Based on the properties of the XML formalism, *XML Schemas* enable us to write a logical model of a data warehouse from its conceptual model. Our approach does not only use the XML formalism to design data warehouses (or data cubes), but also feeds them with data. We use XML documents to support information related to the designed facts. As an XML document supports values of elements and attributes, we assume that it contains information about a single OLAP fact. We say that an XML document supports an *XML fact* when it is valid against an *XML star schema* or an *XML snowflake schema* representing the logical model of a warehouse. Figure 5 shows an example of an *XML fact.*

Figure 5 Example of XML fact (see online version for colours)



```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Suspicious_region Region_length="287" Number_of_regions="6">
        <Patient Patient_age="60" >
                <Age_class Age_class_name="Between 60 and 69 years old" />
        </Patient>
        <Lesion_type Lesion_type_name="calcification type round_and_regular distribution n/a">
                <Lesion_category Lesion_category_name="calcification type round_and_regular" />
        </Lesion_type>
        <Assessment Assessment_code="2" />
        <Subtlety Subtlety_code="4" />
        <Pathology Pathology_name="benign_without_callback" />
        <Date_of_study Date="1998-06-04">
                <Day Day_name="June 4, 1998">
                        <Month Month_name="June, 1998">
                                <Year Year_name="1998" />
                        </Month>
                </Day>
        </Date_of_study>
        <Date_of_digitization Date="1998-07-20">
                <Day Day_name="July 20, 1998">
                        <Month Month_name="July, 1998">
                                <Year Year_name="1998" />
                        </Month>
                </Day>
        </Date_of_digitization>
        <Digitizer Digitizer_name="lumisys laser" />
        <Scanner_image Scanner_file_name="B_3162_1.RIGHT_CC.LJPEG" />
</Suspicious_region>
```

3.2.2 Building XML Cubes

The comparison of *attribute trees* is realised by fusion operations according to *pruning* and *grafting* adapted functions (Boussaïd *et al*., 2006). In some cases, when an input XML document does not contain enough information required by the analysis objectives, the fusion provides a poor output XML document, which represents an OLAP fact with missing data. It is naturally useless to feed the warehouse with such a document. In order to check whether an input XML document contains enough information to feed the warehouse or not, we introduce the *minimal XML document content.* The *minimal XML document content* is an information threshold entirely defined by users when submitting the MCM to express analysis objectives. At this stage, a user can declare for each measure, dimension and dimension hierarchy whether it is mandatory or optional according to his/her objectives and to the information he/she needs to see in the final *XML Cube*. The *minimal XML document content* corresponds to the *attribute tree* associated with mandatory elements declared by the user when submitting the data cube model. It is naturally not possible to decide with an automatic process which element in a future analysis context may be optional or not. It is entirely up to the user to define the *minimal XML document content.* Nevertheless, by default, we suppose that all measures and dimensions attributes of a submitted data cube model are mandatory in the final *XML Cube.* Moreover, we require that not all measures can be optional elements in the data cube. Indeed, in an analysis context, OLAP facts without a measure cannot be exploited by OLAP operators such as aggregation. Hence, users are not allowed to set all the measures to optional elements. At least one measure in the submitted data cube model must be mandatory.

At the fusion step, the *attribute tree* of an input XML document is checked. If it contains all the mandatory elements required by the user, it is merged with the *attribute tree* of the data cube model. Otherwise, it is rejected, the fusion process is cancelled, and therefore no output document is created. To validate our approach, we have implemented it as a Java application (Boussaïd *et al*., 2006).

3.3 Perspectives

A lot of issues need to be addressed in our *X-Warehousing* approach. The first perspective is a performance study of OLAP queries in order to achieve analysis on XML documents as provided in *XML Cubes*. We should also deal with experimental tests on the reliability of the developed application. This includes studies on complexity and time processing of loading input XML documents, building attribute trees, merging attribute trees and creating output XML documents. Second, we should solve the problem of updating the *XML Cube* when the reference MCM is modified in order to change analysis objectives.

We also carried out different scenarios concerning different XML representations of the physical model in order to study the behaviour of aggregation queries on *XML cubes*. The obtained results seem to highlight the problem of performances in *XML cubes*. In the case of certain configurations, the results show that the response time of the roll-up aggregation is prohibitory, but that the physical model is scalable. For other configurations, the response time of aggregation seems more acceptable but linear. Therefore, the concerned configurations are not truly scalable.

These solutions have the merit to show that the conception of *XML cubes* is not trivial. Many problems must be solved. The generation of a new aggregate fact must preserve the same structure as that of the original one. The detailed facts derived from an aggregate fact must also respect the same XML grammar. This latter is definitely defined by the cube's *XML Schema.* When the roll-up or drill-down operations are achieved, it is necessary to keep trace of the OLAP facts' changes when they are transformed from a granularity level to another into the *XML cube*. This can be achieved through the indices mechanism already used in the classical databases.

Some research already exists that uses the concept of referential integrity (ID/IDREF) to tackle the problem of response times (Pokorný, 2002). The idea consists in working with views of the XML documents that significantly reduce the facts by deleting non-useful information for given queries. The definition of the XML documents views, the fragments of XML views or XML documents have been proposed in different articles (Baril and Bellahsène, 2003). Combining these mechanisms of indices and views would be the solution for better performances in *XML cubes*. The choice of a configuration for the physical model of an *XML cube* is an open problem.

4 Complex data analysis

After having presented the problems related to warehousing complex data and the solutions we propose to handle them, we address in this section the issue of complex data analysis.

The possible approaches to analyse complex data include data mining and OLAP analysis. Indeed, the integration of complex data into an ODS under the form of XML documents enables us to consider several ways to analyse them. The first way consists in exploring XML documents directly with data mining techniques. The second way consists in using OLAP analyses to aggregate complex data and to explore them in a multidimensional way. However, classical OLAP tools are ill-adapted to deal with complex data. It seems that OLAP facts representing complex data need appropriate tools and new ways of aggregation to be analysed.

4.1 XML structure mining

4.1.1 Context and issues

The success met by XML is primarily due to its flexibility and its capacity to describe all kinds of data. Consequently, it becomes essential to set up suitable techniques to extract and to exploit information contained in XML documents. Mining the XML documents involves the traditional mining techniques, in particular classification and clustering. There are two main approaches in XML mining. On one hand, XML content mining applies mining methods to document contents. On the other hand, XML structure mining takes interest in information extraction from the structure of XML documents (Garofalakis *et al.*, 1999). Content mining is usually based on text mining techniques. Some authors have also taken interest in association rules extraction from the content of XML documents tags (Braga *et al.*, 2002). Nevertheless, this work took very little into account the hierarchical aspect that exists between tags. Mining the structure of XML documents (intra- and interdocument structure) relates to information contained in the hierarchical organisation of tags (Nayak *et al.*, 2002). The advantage of this approach is that it takes the hierarchical structure into account. In this context, we will quote some work related to the extraction of a DTD from a set of XML documents whose structure is similar (Moh *et al.*, 2000).

We are also interested in the extraction of knowledge from the structure of XML documents. In particular, we wish to release the existing bonds between tags of a set of homogeneously structured documents. Among the existing mining methods, association rules extraction appears to be the best adapted in this case. Indeed, this technique has proven great effectiveness in the discovery of interesting relationships among a large amount of data.

Association rules extraction from the structure of XML documents poses specific problems, mainly due to the hierarchical organisation of tags in XML documents.

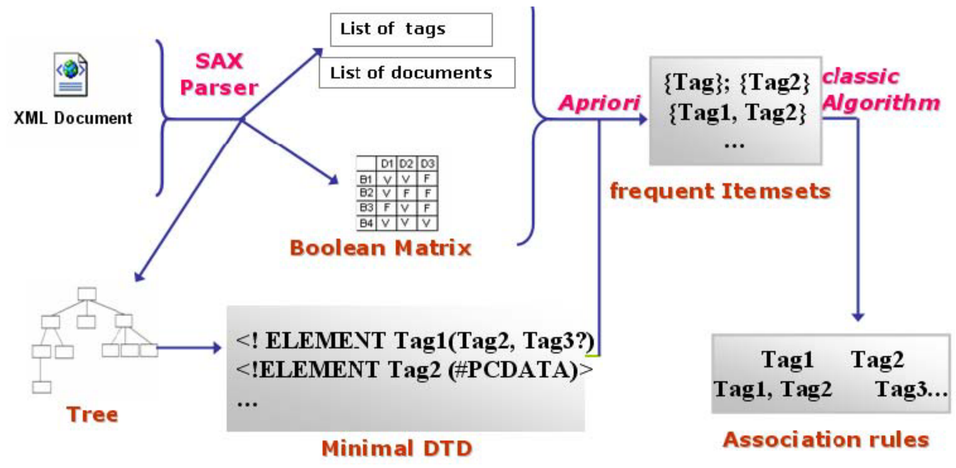
First, XML documents are not directly usable for association rules extraction. Indeed, most frequent itemset search-and-association rule-extraction algorithms are normally intended to be used within relational databases. They reach the items they need for the constitution of frequent itemsets thanks to query languages such as Structured Query Language (SQL). This type of use is not possible in the case of XML documents tags. To be able to use frequent itemset search-and-association rule-extraction algorithms, it is necessary to extract tags from documents and to structure these data. Thus, it is essential to carry out a preprocessing step in order to preformat and retrieve data.

Second, XML document tags are hierarchically organised, unlike tuples in a relational database. Thus, it is necessary to develop a strategy to manage and to take into account the tags' hierarchical structure during association rule extraction. Third, in a well-formed XML document (a document that respects the XML syntax), the same tag may appear in various places in the hierarchy, but it always represents the same information. These tags must be managed specifically.

4.1.2 Proposed solution

To stage these problems, we carry out an effective preformatting of XML documents and we create a minimal DTD representing the hierarchy of all the tags met (Duffoux *et al.*, 2004). Moreover, we show that this preformatting and the minimal DTD make them exploitable by traditional extraction algorithms, and set up an adequate structure for hierarchy management. We obtain a restricted set of relevant rules while improving processing time. We apply an adapted version of the *Apriori* algorithm (Agrawal *et al.*, 1993) for frequent itemset search. Lastly, association rules are extracted and results are presented in XML documents (Figure 6).

Figure 6 Structure mining method in XML documents (see online version for colours)



In order to test our approach, we used two sets of XML documents: Medline (a US dictionary of medicine) and a set of French scientific articles. Then, we applied the adapted traditional algorithm for association rule extraction. We obtain a set of association rules, which we score thanks to the *Discriminating Probabilistic Indicator* (IPD) quality measure. We achieve a clear improvement in terms of quality and processing time for the rules we obtain, compared to a system that does not use a minimal DTD (Duffoux *et al.*, 2004).

Our work can be useful for the creation of a management platform for XML documents (repairing and creating a minimal DTD for documents that do not bear the same schema). This approach also constitutes the first stage of a broader step in XML mining. The association of structure and content mining should enable us to widen and improve the XML content mining techniques, in particular for tags that are bounded. Lastly, in a step of complex data representation, this mining method can be relevant.

4.1.3 Perspectives

This approach highlights the interest in mining the XML document. An XML document encloses more information than a common text. Its structure supports some relevant information. We intend to resort to structure mining as a preliminary task to enhance content mining. Both mining tasks constitute an efficient XML mining that is different from the traditional text mining.

Structure mining may be perceived as an interesting way to discover the relevance of some tags, which may be selected as measures or dimensions in the multidimensional modelling task assisted by data mining techniques (Boussaïd *et al*., 2006). In order to achieve this assisted modelling, we can exploit the associations among some tags discovered by our structure mining approach.

4.2 A data-mining-based OLAP operator for complex data

4.2.1 Context and issues

OLAP is a powerful means of exploring and extracting pertinent information from data through multidimensional analysis. In this context, data are organised in multidimensional views, commonly called data cubes. However, classical OLAP tools are not always able to deal with complex data. For example, when processing images, sounds, videos, texts or even XML documents, aggregating information with the classical OLAP does not make any sense. Indeed, we are not able to compute a traditional aggregation operation, such as a sum or average, over such data. However, when users analyse complex data, they need more expressive aggregates than those created from elementary computation of additive measures. We think that OLAP facts representing complex objects need appropriate tools and new aggregation means, since we wish to analyse them.

Furthermore, a data cube structure can provide a suitable context for applying data mining methods. More generally, the association of OLAP and data mining allows elaborated analysis tasks exceeding the simple exploration of a data cube. Our idea is to take advantage of OLAP, as well as data mining techniques, and to integrate them into the same analysis framework in order to analyse complex objects. Despite the fact that both OLAP and data mining have long been considered two separate fields, several recent studies have proven the capability of their association to provide an interesting analysis process (Imielinski and Mannila, 1996). The major difficulty when combining OLAP and data mining is that traditional data mining algorithms are mostly designed for tabular data sets organised in individual-variable form (Fayyad *et al*., 1996). Therefore, multidimensional data are not suited to these algorithms. Nevertheless, a lot of previous research motivated and proved an interest in coupling OLAP with data mining methods in order to extend OLAP analysis capabilities (Goil and Choudhary, 1998; Han, 1998; Sarawagi *et al*., 1998). In addition, we propose another contribution to this field by developing a new type of online aggregation for complex data. It is a new OLAP Operator for Aggregation by Clustering (OpAC) (Ben Messaoud *et al*., 2004).