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:

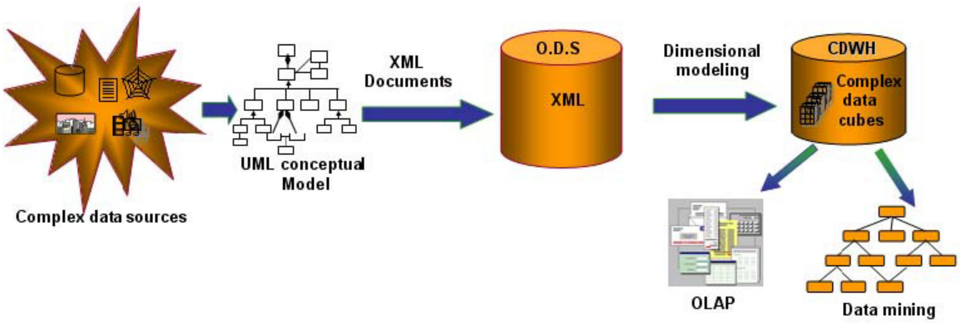
- represented in various formats (databases, texts, images, sounds, videos, *etc.*)
- diversely structured (relational databases, XML documents, *etc.*)
- originating from several different sources
- described through several channels or points of view (a video and a text that describe the same meteorological 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).

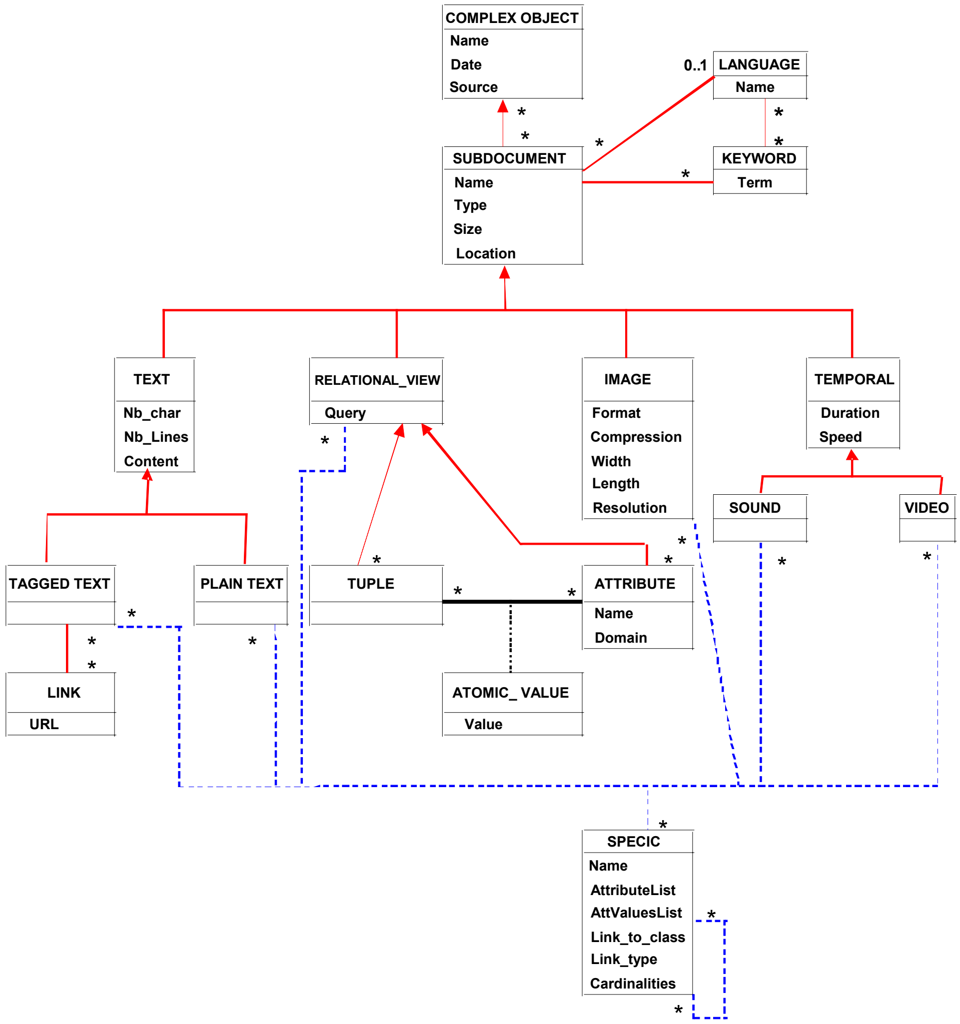
To achieve complex data integration following the warehouse approach, the traditional ETL process is ill-adapted. We present here our approach to accomplish the ETL process in an original way. We propose a modelling process to achieve the integration of complex data into a data warehouse (Boussaïd *et 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)



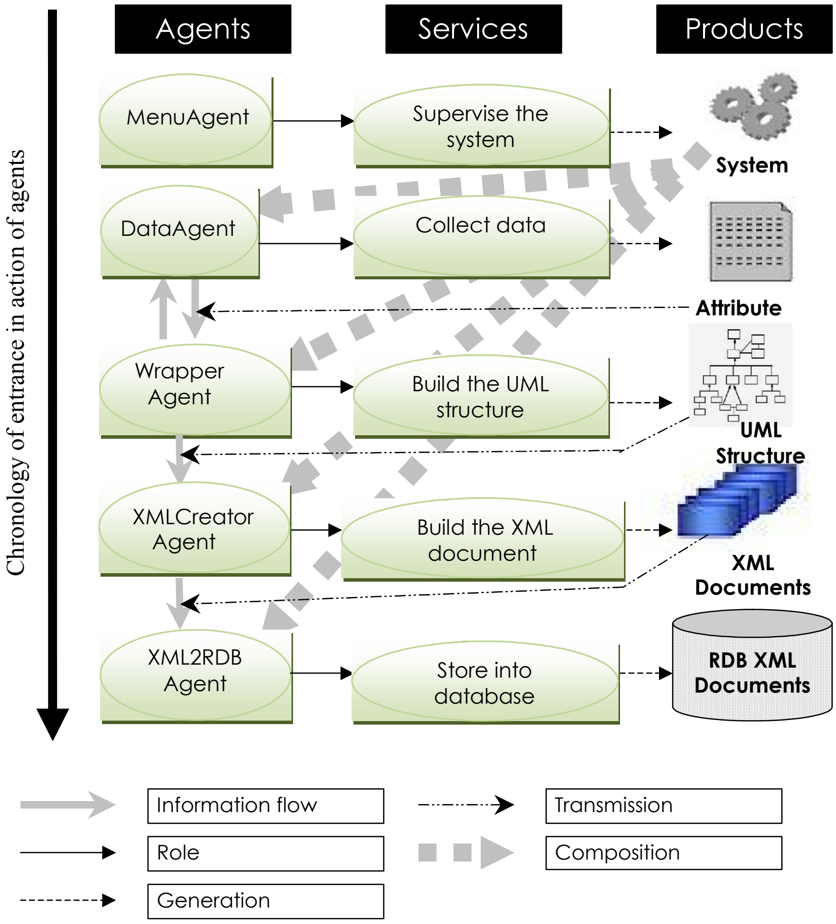
The UML class diagram represents a complex object generalising all complex data types. Note that our goal here is to propose a general data structure: the list of attributes for each class in this diagram is intentionally not exhaustive. Our generic model defines a complex object which is composed of 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.

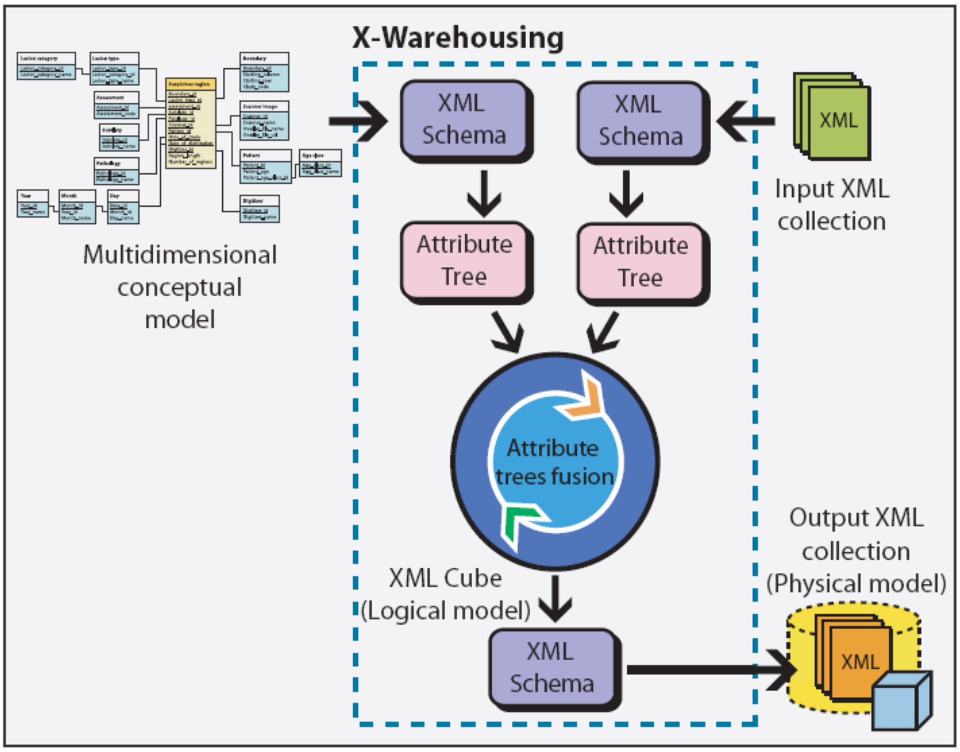
In fact, two *attribute trees* can easily be merged through a fusion process based on *pruning* and *grafting* functions (Golfarelli *et al*., 2001). At this stage, two cases are possible:

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 (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.

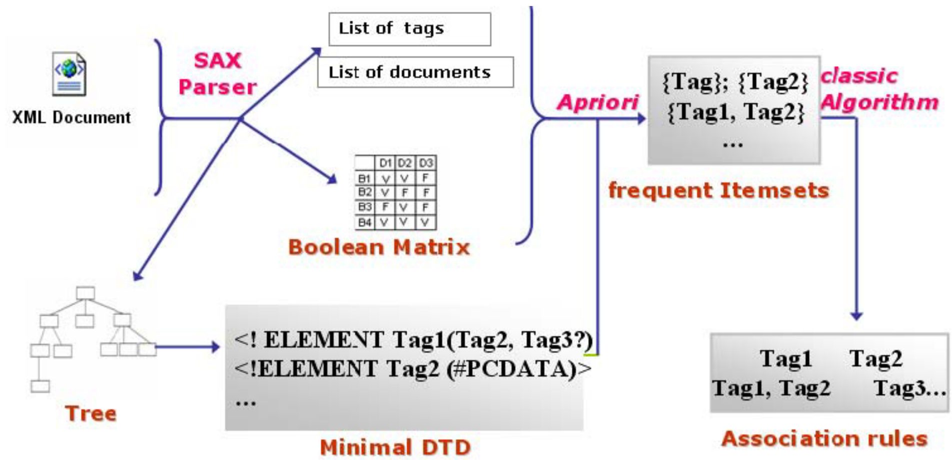
First, XML documents are not directly usable for association rules extraction. Indeed, most frequent itemset search-and-association rule-extraction algorithms are normally intended to be used within relational databases. They reach the items they need for the constitution of frequent itemsets thanks 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).

4.2.2 Proposed solution

To aggregate information about complex data, we must often gather similar facts into a single group and separate dissimilar facts into different groups. In this case, it is necessary to consider an aggregation by computing both descriptors and measures. Instead of grouping facts only by computing their measures, we also take their descriptors into account to obtain aggregates expressing semantic similarities. Our new OpAC data combine OLAP with an automatic clustering technique. We use the Agglomerative Hierarchical Clustering (AHC) as an aggregation strategy for complex data. Our operator enables significant aggregates of facts expressing semantic similarities. More generally, the aggregates provided by OpAC give interesting knowledge about the analysed domain. OpAC is adapted for all types of data, since it deals with data cubes modelled in XML (Ben Messaoud *et al*., 2006).

Furthermore, we also propose some evaluation criteria that support the results of our operator. These criteria aim at assisting the user and helping him/her in choosing the best partition of aggregates that will fit well with his/her analysis requirements. We also developed a web application for our operator. We provided performance experiments and led a case study on XML documents dealing with the breast cancer research domain (Ben Messaoud *et al*., 2006).

The main idea of OpAC is to exploit the cube's facts describing complex objects to provide a more significant aggregation over them. In order to do so, we use a clustering method and automatically highlight aggregates that are semantically richer than those provided by the current OLAP operators. Hence, the clustering method provides a new OLAP aggregation concept. This aggregation provides hierarchical groups of objects, summarising information and enables navigating through levels of these groups. Existing OLAP tools, such as the Slicing operator, can create new restricted aggregates in a cube dimension, too, but these tools always need manual assistance, whereas our operator is based on a clustering algorithm that automatically provides relevant aggregates. Furthermore, with classical OLAP tools, aggregates are created in an intuitive way in order to compare some measure values, whereas OpAC creates significant aggregates that express deep relations with the cube's measures. Thus, the construction of such aggregates is interesting for establishing a more elaborate online analysis context. According to the above objectives, we choose the AHC as an aggregation method. Our choice is motivated by the fact that the hierarchical aspect constitutes a relevant analogy between AHC results and the hierarchical structures of dimensions. The objectives and the results expected for OpAC match perfectly with the AHC strategy. Furthermore, the AHC adopts an agglomerative strategy that starts with the finest partition where each individual is considered a cluster. Therefore, OpAC results include the finest attributes of a dimension. But, like almost all unsupervised mining methods, the main defect of the AHC is that it does not give an evaluation of its results, *i.e.*, the partitions of clusters. It is quite tedious to choose the best partition, and it is more difficult when we deal with a great number of individuals. In the case of the OpAC operator, we propose to use many criteria to help users to select the best partition of aggregates. According to the data mining literature, we can cause intra- and intercluster inertias or the Ward's method. In addition, we also propose a new criterion based on the cluster separability (Ben Messaoud *et al*., 2006).

The AHC is compatible with the exploratory aspect of OLAP. Its results can also be reused by classical OLAP operators. In fact, the AHC provides several hierarchical partitions. By moving from a partition level to a higher one, two aggregates are joined together. Conversely, by moving from a partition level to a lower one, an aggregate is divided into two new ones. These operations are strongly similar to the classical Roll-up and Drill-down operators. AHC is a well-suited clustering method to summarise information into OLAP aggregates from complex facts.

4.2.3 Perspectives

This work has proved the benefit of associating OLAP and data mining in order to enhance the power of online analysis. We believe that, in the future, this association will provide a new generation of efficient OLAP operators well suited to complex data analyses. Extending the online analysis towards the new capabilities such as explanation and prediction is an exciting challenge that we plan to achieve. In order to accomplish this objective, we intend to develop a new algebra based on the OLAP and data mining coupling to define some new online mining operators to deeply explore the complex data.

5 Performance issues in XML warehousing

5.1 Context and issues

While we advocated in the previous sections that XML data warehouses form an interesting basis for decision-support applications exploiting complex data, XML-native Database Management Systems (DBMSs) usually show limited performances when the volume of data is very large and queries are complex. This is typically the case in data warehouses, where data are historicised and analytical queries involve several join and aggregation operations. Hence, it is crucial to devise means of optimising the performance of XML data warehouses. In such a context, indexing and view materialisation are presumably some of the most effective optimisation techniques.

Indexes are physical structures that allow direct data access. They avoid sequential scans and thereby reduce query response time. Materialised views improve data access time by precomputing intermediary results. Therefore, end-user queries can be efficiently processed through data stored in views, and do not need to access the original data. However, exploiting either indexes or materialised views requires additional storage space and entails maintenance overhead when refreshing the data warehouse. The issue is then to select an appropriate configuration of indexes and materialised views that minimises both query response time, and index and view maintenance cost given a limited storage space (an NP-hard problem).

Most of the existing XML indexing techniques (Bertino *et al*., 2004; Boulos and Karakashian, 2006; Bruno *et al*., 2002; Chien *et al*., 2002; Chung *et al*., 2002; Goldman and Widom, 1997; He and Yang, 2004; Jiang *et al*., 2003; Kaushik *et al*., 2002; Milo and Suciu, 1999; Qun *et al*., 2003; Zhang *et al*., 2001) are applicable only to XML data that are targeted by simple path expressions. However, in the context of XML data warehouses, queries are complex and include several path expressions. Moreover, these indices operate on one XML document only, whereas in XML warehouses, data are managed in several XML documents and analytical queries are performed over these documents. The Fabric index (Cooper *et al*., 2001) does handle multiple documents, but it is not adapted to XML data warehouses either, because it does not take into account the relationships that exist between XML documents in a warehouse (facts and dimensions). Fabric is thus not beneficial to decision-support queries. In consequence, we propose a new index that is specifically adapted to XML multidimensional data warehouses (Mahboubi *et al*., 2006b). This data structure allows us to optimise the access time to several XML documents by eliminating join costs, while preserving the information contained in the initial warehouse.

In sum, the literature about materialised view selection is abundant in the context of relational data warehouses but, to the best of our knowledge, no such approach exists in XML databases and XML data warehouses in particular. Hence, we proposed an adaptation of a query-clustering-based relational view selection approach (Aouiche *et al*., 2006) to the XML context (Mahboubi *et al*., 2006a). This approach clusters XQuery queries and builds candidate XML views that can resolve multiple similar queries belonging to the same cluster. XML-specific cost models are used to define the XML views that are important to materialise.

5.2 Proposed solutions

5.2.1 Reference XML data warehouse

Several authors address the issue of designing and building XML data warehouses. They use XML documents to manage or represent the facts and/or dimensions of the warehouse (Hummer *et al*., 2003; Park *et al*., 2005; Pokorný, 2002). We select XCube (Hummer *et al*., 2003) as a reference data warehouse model. Since other XML warehouse models from the literature are relatively similar, this is not a binding choice. The advantage of XCube is its simple structure for representing facts and dimensions in a star schema: one document for dimensions (*Dimensions.xml*) and another one for facts (*Facts.xml*).

5.2.2 XML data warehouse indexing

Building the indexes cited in Section 5.1 on an XML warehouse causes a loss of information in decision-support query resolution. Indeed, clustering or merging identical labels in the XML graph causes the disappearance of the relationship between fact measures and dimensions. We illustrate this problem in the following example. The *Facts.xml* document is composed of *Cell* elements. Each cell is identified by its attributes and one or more measures. Figure 7 shows the structure of the *Facts.xml* document and its corresponding 1-index (we selected the 1-index (Milo and Suciu, 1999) as an example). The 1-index represents cells linearly, *i.e.*, all labels for the same source are represented by only one label. Hence, recovering a cell characterised by its measures and its dimension identifiers is impossible.

An index should be able to preserve the relationships between dimensions and fact measures. Thus, the structure of our index is similar to that of the *Facts.xml* document, except for the *attribute* element. As usual with XML indexes, our index structure is stored in an XML document named *Index.xml* (Figure 8). Each *Cell* element is composed of dimensions and one or more facts. A *Fact* element has two attributes, *@id* and *@value*, which respectively represent measure names and values. Each *dimension* element is composed of two attributes: *@id*, which stores the dimension name, and

@node, which stores the value of the dimension identifier. Moreover, the dimension element has children *attribute* elements. These elements are used to store the names and values of the attributes from each dimension. They are obtained from the *Dimensions.xml* document. An *attribute* element is composed of two attributes, *@name* and *@value*, which respectively store the name and value of each attribute.

Figure 7 *Facts.xml* structure (a) and its corresponding 1-index (b)

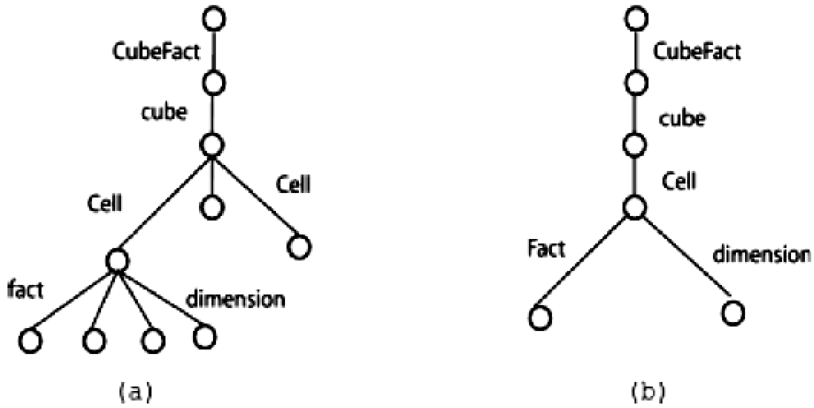
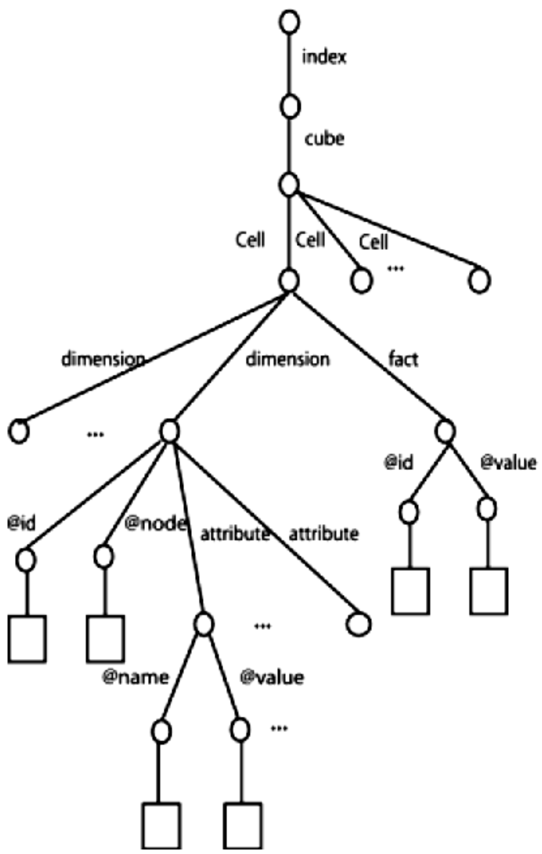


Figure 8 XML join index structure



Data migration from *Dimensions.xml* and *Facts.xml* to the index structure helps in storing facts, dimensions and their attributes in the same cell. This feature wholly eliminates join operations, since all the information that is necessary for a join operation is stored in the same cell. Finally, queries need to be rewritten to exploit our index. The rewriting process consists in preserving the selection expressions and the aggregation operations.

5.2.3 XML materialised view selection

The first step in our view selection strategy is to build, from a workload of representative queries, a clustering context. We extract from the queries representative attributes, *i.e.*, attributes that are present in the selection predicates and grouping clauses. Then, we store the relationships between the query workload and the extracted attributes in a 'query-attribute' matrix. The matrix lines are the queries and the columns are the extracted attributes. The general term q_{ji} of this matrix is set to one if extracted attribute a_j is present in query q_i, and to zero otherwise. This matrix represents our clustering context.

Then, since it is hard to search for all the syntactically relevant views (candidate views) because the search space is very large, we cluster together similar queries. Similar queries are those having a close binary representation in the query-attribute matrix. Two similar queries can be resolved by using only one materialised view. We define similarity and dissimilarity measures which ensure that queries within the same cluster are strongly related to each other, whereas queries from different clusters are significantly different. The number of candidate views we obtain is generally as high as the input workload is large. Thus, it is not feasible to materialise all the proposed views because of storage space constraints. To circumvent this limitation, we devised cost models and objective functions that exploit them and that help in selecting only the most pertinent materialised views.

Finally, our view selection algorithm is based on a greedy search within the candidate view set V, with respect to an objective function F. In the first iteration, the values of F are computed for each view within V. The view υ_{max} that minimises F, if it exists ($F_{/S}(\upsilon_{max}) > 0$), is then added to S. The values of F are then computed for each remaining view in $V - S$, since they depend on the selected views present in S. This helps in taking into account the interactions that probably exist between the views. We repeat these iterations until there is no improvement ($F_{/S}(\upsilon) \leq 0$), all the views have been selected ($V - S = \emptyset$) or the storage space is full.

5.3 Perspectives

To validate our proposals, we performed many experiments with XML-native and relational, XML-compatible DBMSs. Our tests show that using either the index structure we propose, or the materialised views our strategy helps in building, significantly improves response time for typical analytical queries expressed in XQuery. Gains in performance range from a factor of 8000 to 25 000, depending on the host system. Furthermore, our tests also demonstrate that, properly indexed, XML-native DBMSs can compete with, and even best relational DBMSs in terms of performance when XML documents are bulky. This is because relational DBMS engines combine XQuery to SQL and must convert the result from relations to XML. On the other hand, XML-native DBMSs preserve the hierarchical structure of XML data, which allows path scans to be efficiently processed by XQuery engines.

This research opens three broad axes of perspectives. First, we have to complement our experiments with other tests, presumably by using other systems and data warehouse configurations. Our aim is to assert the gain in performance versus the overhead for generating and refreshing indexes and materialised views in each configuration. Second, it is widely acknowledged that indexes and materialised views are mutually beneficial to each other. We have designed a method for simultaneously selecting indexes and materialised views in the relational context, which we aim to adapt to the XML context. Finally, our performance optimisation strategies could be better integrated in a host XML-native DBMS. This would certainly help in developing an incremental strategy for the maintenance of indexes and materialised views. Our selection strategy is currently static. Studies dealing with incremental data mining may be exploited to make it dynamic, so that a configuration of indexes and materialised views can be updated instead of being rebuilt from scratch. Moreover, the mechanism for rewriting queries would also be more efficient if it was part of the system.

6 Conclusion

Nowadays, data from the web are complex. To handle this kind of data in a decision-support context, we propose a full, generic data warehousing and an online analysis process. In this paper, we identified some problems related to both these processes and we suggested different solutions.

First of all, we exposed an approach to integrate complex data. We presented a generic UML model that allows modelling not only low-level but also semantic information concerning the complex data to be analysed. We integrated complex data as XML documents into an ODS as a first step in complex data warehousing. This complex data integration approach is based on both the data warehouse technology and MASs. Our integration approach indeed necessitates several tasks that may be assimilated to services offered by well-defined agents in a system intended to achieve such an integration. We developed the SMAIDoC system, which allows this integration. It is based upon a flexible and progressive architecture in which we can add, remove or modify services, and even create new agents.

Secondly, we proposed a methodology entirely based on the XML formalism to warehouse complex data. Our X-Warehousing approach does not simply populate a repository with XML documents, but also expresses an interesting abstraction level by preparing XML documents for future analyses. In fact, it consists in validating documents against an XML Schema, which models a data warehouse. We defined a general XML formalisation for star and snowflake schemas. We also exploited the concept of attribute trees to help in the creation and the warehousing of homogeneous XML documents, by merging initial XML sources with a reference multidimensional model. Constraints on the created XML documents may be required and expressed by users. To validate our X-Warehousing approach, we implemented a Java application, which takes as input a reference multidimensional model and XML documents, and provides logical and physical models for an XML cube composed of homogeneous XML documents.

These proposed solutions constitute the first axis related to the complex warehousing process. We also proposed two approaches regarding the online analysis process of complex data. The first one applies a data mining technique to discover association rules among the tags of XML documents. Structure mining is indeed a relevant preliminary task for content mining over XML documents. Our second proposal carries on a new online analysis context for complex data. Our approach is based on coupling OLAP with data mining. The combination of the two fields can be a solution to capitalise their respective benefits. We have created OpAC, a new OLAP aggregation operator based on an automatic clustering method. Unlike classical OLAP operators, OpAC enables precise analyses and provides semantic aggregates of complex objects.

Finally, the implementation of our X-Warehousing method highlighted performance problems when storing warehouses in XML-native DBMSs. Hence, we proposed a new join index that is specifically adapted to XML multidimensional data warehouses. This data structure allows optimising the access time to several XML documents by eliminating join costs, while preserving the information contained in the initial warehouse. We also presented a view selection algorithm, which we combine with the index structure to improve the overall efficiency of our strategy. The experiments we performed show a significant improvement in response time for analytical queries.

All the solutions we have presented in this paper are a modest contribution to the problem of complex data warehousing and online analysis. Although we mentioned various perspectives for our work at the end of each of this paper's sections, a lot of scientific issues are still unsolved.

Acknowledgements

The authors would like to thank Prof. Robert Wrembel and Prof. Jaroslav Pokorný, the editors, for inviting them to publish an article in this special issue.

References

Agrawal, R., Imielinski, T. and Swami, A. (1993) 'Mining association rules between sets of items in large databases', *1993 ACM SIGMOD International Conference on Management of Data (SIGMOD 93),* Washington, DC, pp.207–216.

Aouiche, K., Jouve, P.E. and Darmont, J. (2006) 'Clustering-based materialized view selection in data warehouses', *10th East-European Conference on Advances in Databases and Information Systems (ADBIS 06), LNCS*, Thessaloniki, Greece, Vol. 4152, pp.81–95.

Baril, X. and Bellahsène, Z. (2003) 'Designing and managing an XML warehouse', *XML Data Management: Native XML and XML-enabled Database Systems*, Boston: Addison Wesley, pp.455–473.

BDD-ERIC (2003) 'SMAIDoC download page', http://bdd.univ-lyon2.fr/?page=logiciel&id=4.

Belanger, D., Churchet, K. and Hume, A. (1999) 'Virtual data warehousing, data publishing, and call detail', *International Workshop on Databases in Telecommunications, LNCS*, Edinburgh, Scotland, Vol. 1819, pp.106–117.

Ben Messaoud, R., Boussaïd, O. and Loudcher Rabaséda, S. (2004) 'A new OLAP aggregation based on the AHC technique', *7th ACM International Workshop on Data Warehousing and OLAP (DOLAP 04)*, Washington, DC, pp.65–72.

Ben Messaoud, R., Boussaïd, O. and Loudcher Rabaséda, S. (2006) 'A data mining-based OLAP aggregation of complex data: application on XML documents', *International Journal of Data Warehousing and Mining*, Vol. 2, No. 4, pp.1–26.

Bertino, E., Catania, B. and Wang, W.Q. (2004) 'XJoin index: indexing XML data for efficient handling of branching path expressions', *20th International Conference on Data Engineering (ICDE 04)*, Boston, USA, p.828.

Boulos, J. and Karakashian, S. (2006) 'A new design for a native XML storage and indexing manager', *10th International Conference on Extending Database Technology (EDBT 06), LNCS*, Munich, Germany, Vol. 3896, pp.755–772.

Boussaïd, O., Ben Messaoud, R., Choquet, R. and Anthoard, S., (2006) 'X-Warehousing: an XML-based approach for warehousing complex data', *10th East-European Conference on Advances in Databases and Information Systems (ADBIS 06), LNCS*, Thessaloniki, Greece, Vol. 4152, pp.39–54.

Boussaïd, O., Bentayeb, F. and Darmont, J. (2003) 'A multi-agent system-based ETL approach for complex data', *10th ISPE International Conference on Concurrent Engineering: Research and Applications (CE 03)*, Madeira, Portugal, pp.49–52.

Boussaïd, O., Bentayeb, F., Duffoux, A. and Clerc, F. (2003) 'Complex data integration based on a multi-agent system', *1st International Conference on Industrial Applications of Holonic and Multi-Agent Systems, (HoloMAS 03), LNAI*, Prague, Czech Republic, Vol. 2744, pp.201–212.

Boussaïd, O., Tanasescu, A., Bentayeb, F. and Darmont, J. (2006) 'Integration and dimensional modelling approaches for complex data warehousing', *Journal of Global Optimization*, to appear.

Braga, D., Campi, A., Klemettinen, M. and Lanzi, P.L. (2002) 'Mining association rules from XML data', *4th International Conference on Data Warehousing and Knowledge Discovery (DaWak 02), LNCS*, Aix-en-Provence, France, Vol. 2454, pp.21–30.

Bruno, N., Koudas, N. and Srivastava, D. (2002) 'Holistic twig joins: optimal XML pattern matching', *2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 02)*, Madison, USA, pp.310–321.

Chien, S.Y., Vagena, Z., Zhang, D., Tsotras, V.J. and Zaniolo, C. (2002) 'Efficient structural joins on indexed XML documents', *28th International Conference on Very Large Data Bases (VLDB 02)*, Hong Kong, China, pp.263–274.

Chung, C., Min, J. and Shim, K. (2002) 'APEX: an adaptive path index for XML data', *2002 ACM SIGMOD International Conference on Management of Data (SIGMOD 02),* Madison, USA, pp.121–132.

Cooper, B.F., Sample, N., Franklin, M.J., Hjaltason, G.R. and Shadmon, M. (2001) 'A fast index for semistructured data', *27th International Conference on Very Large Data Bases (VLDB 01)*, Roma, Italy, pp.341–350.

Darmont, J. and Boussaïd, O. (Eds.) (2006) *Processing and Managing Complex Data for Decision Support*, Hershey: Idea Group Publishing.

Darmont, J., Boussaïd, O., Bentayeb, F., Rabaseda, S. and Zellouf, Y. (2003) 'Web multiform data structuring for warehousing', *Multimedia Systems and Applications*, Dordrecht: Kluwer Academic Publishers, Vol. 22, pp.179–194.

Darmont, J., Boussaïd, O., Ralaivao, J.C. and Aouiche, K. (2005) 'An architecture framework for complex data warehouses', *7th International Conference on Enterprise Information Systems (ICEIS 05)*, Miami, USA, pp.370–373.

Duffoux, A., Boussaïd, O., Lallich, S. and Bentayeb, F. (2004) 'Fouille dans la structure de documents XML', 4èmes *Journées Francophones d'Extraction et de Gestion des Connaissances (EGC 04), RNTI*, Clermont-Ferrand, Vol. 2, pp.519–524.

Fayyad, U.M., Shapiro, G.P., Smyth, P. and Uthurusamy, R. (1996) *Advances in Knowledge Discovery and Data Mining*, Cambridge: AAAI/MIT Press.

FIPA (2002) 'The foundation for intelligent physical agents', http://www.fipa.org.

Garofalakis, M., Rastogi, S.S. and Shim, K. (1999) 'Data mining and the web: past, present and future', *2nd International ACM Workshop on Web Information and Data Management (WIDM 99)*, Kansas City, USA, pp.43–47.

Goasdoué, F., Lattès, V. and Rousset, M.C. (2000) 'The use of CARIN Language and algorithms for information integration: the PICSEL system', *International Journal of Cooperative Information Systems*, Vol. 9, No. 4, pp.383–401.

Goil, S. and Choudhary, A. (1998) 'High performance multidimensional analysis and data mining', *High Performance Networking and Computing Conference (SC 98)*, Orlando, USA.

Goldman, R. and Widom, J. (1997) 'DataGuides: enabling query formulation and optimization in semistructured databases', *23rd International Conference on Very Large Data Bases (VLDB 97)*, Athens, Greece, pp.436–445.

Golfarelli, M. and Rizzi, S. (1999) 'Designing the data warehouse: key steps and crucial issues', *Journal of Computer Science and Information Management*, Vol. 2, No. 3, pp.88–100.

Golfarelli, M., Rizzi, S. and Vrdoljak, B. (2001) 'Data warehouse design from XML sources', *4th ACM International Workshop on Data Warehousing and OLAP (DOLAP 01)*, Atlanta, USA, pp.40–47.

Hackathorn, R. (2000) *Web Farming for the Data Warehouse*, San Francisco: Morgan Kaufmann.

Han, J. (1998) 'Toward on-line analytical mining in large databases', *SIGMOD Record*, Vol. 27, No. 1, pp.97–107.

He, H. and Yang, J. (2004) 'Multiresolution indexing of XML for frequent queries', *20th International Conference on Data Engineering (ICDE 04)*, Boston, USA, pp.683–694.

Heath, M., Bowyer, K., Kopans, D., Moore, R. and Kegelmeyer, P. (2000) 'The digital database for screening mammography', *5th International Workshop on Digital Mammography (IWDM 00)*, Toronto, Canada, pp.212–218.

Hümmer, W., Bauer, A. and Harde, G. (2003) 'XCube: XML for data warehouses', *6th International Workshop on Data Warehousing and OLAP (DOLAP 03)*, New Orleans, USA, pp.33–40.

Imielinski, T. and Mannila, H. (1996) 'A database perspective on knowledge discovery', *Communications of the ACM*, Vol. 39, No. 11, pp.58–64.

Inmon, W.H. (1996) *Building the Data Warehouse*, Indianapolis: John Wiley & Sons.

JADE (2002) 'The Java Agent development framework', http://sharon.cselt.it/projects/jade/.

Jiang, H., Lu, H., Wang, W. and Ooi, B.C. (2003) 'XR-Tree: indexing XML data for efficient structural joins', *19th International Conference on Data Engineering (ICDE 03)*, Bangalore, India, pp.253–263.

Kaushik, R., Shenoy, D., Bohannon, P. and Gudes, E. (2002) 'Exploiting local similarity to efficiently index paths in graph-structured data', *18th International Conference on Data Engineering (ICDE 02)*, San Jose, USA, pp.129–140.

Kimball, R. (1996) *The Data Warehouse Toolkit*, Indianapolis: John Wiley & Sons.

Lawton, G. (2006) 'Making business intelligence more useful', *Computer*, Vol. 39, No. 9, pp.14–16.

Mahboubi, H., Aouiche, K. and Darmont, J. (2006a) 'Materialized view selection by query clustering in XML data warehouses', *4th International Multiconference on Computer Science and Information Technology (CSIT 06)*, Amman, Jordan, pp.68–77.

Mahboubi, H., Aouiche, K. and Darmont, J. (2006b) 'Un index de jointure pour les entrepôts de données XML', *6èmes Journées Francophones Extraction et Gestion des Connaissances (EGC 06)*, *RNTI*, Lille, France, Vol. E-6, pp.89–94.

Milo, T. and Suciu, D. (1999) 'Index structures for path expressions', *7th International Conference on Database Theory (ICDT 99), LNCS*, Jerusalem, Israel, Vol. 1540, pp.277–295.

Moh, C.H., Lim, E.P. and Ng, W.K. (2000) 'DTD-Miner: a tool for mining DTD from XML documents', *2nd International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems (WECWIS 00)*, Milpitas, USA, pp.144–151.

Nassis, V., Rajugan, R., Dillon, T.S. and Rahayu, J.W. (2004) 'Conceptual design of XML document warehouses', *6th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 04), LNCS*, Zaragoza, Spain, Vol. 3181, pp.1–14.

Nassis, V., Rajugan, R., Dillon, T.S. and Rahayu, J.W. (2005) 'Conceptual and systematic design approach for XML document warehouses', *International Journal of Data Warehousing and Mining*, Vol. 1, No. 3, pp.63–86.

Nayak, R., Witt, R. and Tonev, A. (2002) 'Data mining and XML documents', *2002 International Conference on Internet Computing (IC 02)*, Las Vegas, USA, pp.660–666.

Park, B.K., Han, H. and Song, I.Y. (2005) 'XML-OLAP: a multidimensional analysis framework for XML warehouses', *7th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 05), LNCS*, Copenhagen, Denmark, Vol. 3589, pp.32–42.

Pokorný, J. (2002) 'XML data warehouse: modelling and querying', *5th International Baltic Conference (BalticDB&IS 02)*, Tallinn, Estonia, pp.267–280.

Qun, C., Lim, A. and Ong, K.W. (2003) 'D(k)-Index: an adaptive structural summary for graph-structured data', *2003 ACM SIGMOD International Conference on Management of Data (SIGMOD 03)*, San Diego, USA, pp.134–144.

Rousset, M.C. (2002) 'Knowledge representation for information integration', *13th International Symposium on Methodologies for Intelligent Systems (ISMIS 02), LNCS*, Lyon, France, Vol. 2366, pp.1–3.

Sarawagi, S., Agrawal, R. and Megiddo, N. (1998) 'Discovery-driven exploration of OLAP data cubes', *6th International Conference on Extending Database Technology (EDBT 98), LNCS*, Valencia, Spain, Vol. 1377, pp.168–182.

Sun Microsystems (2002) 'The source for Java technology', http://java.sun.com.

Sycara, K. and Zeng, D. (1996) 'Coordination of multiple intelligent software agents', *International Journal of Cooperative Information Systems*, Vol. 5, Nos. 2–3, pp.1–31.

Trujillo, J., Lujan-Mora, S. and Song, I.Y. (2004) 'Applying UML and XML for designing and interchanging information for data warehouses and OLAP applications', *Journal of Database Management*, Vol. 15, No. 1, pp.41–72.

Vrdoljak, B., Banek, M. and Rizzi, S. (2005). 'Designing web warehouses from XML schemas', *5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 03), LNCS*, Prague, Czech Republic, Vol. 2737, pp.89–98.

Zhang, C., Naughton, J.F., DeWitt, D., Luo, Q. and Lohman, G. (2001) 'On supporting containment queries in relational database management systems', *2001 ACM SIGMOD Conference on Management of Data (SIGMOD 01)*, Santa Barbara, USA, pp.425–436.

Zhang, J., Wang, W., Liu, H. and Zhang, S. (2005) 'X-warehouse: building query pattern-driven data', *14th International Conference on World Wide Web (WWW 05)*, Chiba, Japan, pp.896–897.

Note

1 http://www/w3.org