J Glob Optim (2007) 37:571–591
DOI 10.1007/s10898-006-9064-6

ORIGINAL PAPER

Integration and dimensional modeling approaches for complex data warehousing

O. Boussaid · Adrian Tanasescu · Fadila Bentayeb · Jérôme Darmont

Received: 22 June 2006 / Accepted: 23 June 2006 / Published online: 9 August 2006

Abstract With the broad development of the World Wide Web, various kinds of heterogeneous data (including multimedia data) are now available to decision support tasks. A data warehousing approach is often adopted to prepare data for relevant analysis. Data integration and dimensional modeling indeed allow the creation of appropriate analysis contexts. However, the existing data warehousing tools are well-suited to classical, numerical data. They cannot handle complex data. In our approach, we adapt the three main phases of the data warehousing process to complex data. In this paper, we particularly focus on two main steps in complex data warehousing. The first step is data integration. We define a generic UML model that helps representing a wide range of complex data, including their possible semantic properties. Complex data are then stored in XML documents generated by a piece of software we designed. The second important phase we address is the preparation of data for dimensional modeling. We propose an approach that exploits data mining techniques to assist users in building relevant dimensional models.

Keywords Complex data · Data integration · Data mining · Dimensional modeling · Data preparation · Data warehousing

O. Boussaid (✉) · F. Bentayeb · J. Darmont
ERIC - Université Lumière Lyon2, 5 avenue Pierre-Mendès-France,
69676 Bron Cedex – France
e-mail: omar.boussaid@univ-lyon2.fr

F. Bentayeb
e-mail: bentayeb@eric.univ-lyon2.fr

J. Darmont
e-mail: jerome.darmont@univ-lyon2.fr

A. Tanasescu
LIRIS - Université Claude Bernard Lyon 1, 43 boulevard. du 11 novembre 1918,
69422 Villeurbanne Cedex, France
e-mail: Adrian.Tanasescu@liris.cnrs.fr

1 Introduction

Traditional databases aim at data management, i.e., they help organizing, structuring and querying data. They are oriented toward transaction processing and are often qualified as production databases. Data warehouses have a very different vocation: analyzing data (Kimball and Ross 2002; Inmon 2005) (i.e., extract information from data). To achieve this goal, data warehouses exploit specific models (star schemas and their derivatives): data are organized around indicators called measures, and observation axes called dimensions, which form an analysis context. These models are not normalized, because they do not aim at avoiding data redundancy (like in the classical relational context), but at enhancing the performance of decision support queries (Kimball and Ross 2002). Furthermore, data warehouses store huge volumes of data. They are also the core of decision support systems (DSS) (Kimball and Merz 2000).

With the expansion of multimedia data and the worldwide development of the Web, many data are now available to various decision support fields (customer relationship management, marketing, competition monitoring, medicine...). However, these data are not only numerical or symbolic. For example, medical DSS might require the analysis of various and heterogeneous data, such as patient records, medical images, biological analysis results, etc. (Saad 2004). We term such data *complex data*. In summary, data may be qualified as complex if they are (Darmont et al. 2005):

- *multiformat*, i.e., represented in various formats (databases, texts, images, sounds, videos...);
  and/or
- *multistructure*, i.e., diversely structured (relational databases, XML document repositories...);
  and/or
- *multisource*, i.e., originating from several different sources (distributed databases, the Web...);
  and/or
- *multimodal*, i.e., described through several channels or points of view (radiographies and audio diagnosis of a physician, data expressed in different scales or languages...);
  and/or
- *multiversion*, i.e., changing in terms of definition or value (temporal databases, periodical surveys...).

Though many people have actually been working on subsets of complex data for years, the idea of a new, broader research field has recently emerged (Darmont and Boussaid 2006). However, complex data are often still handled in separate ways. For instance, distinct solutions are proposed for multimedia data management and information retrieval (Grabczewski et al. 2001) or knowledge discovery in texts, images or videos (text mining, image mining or more generally multimedia mining). In opposition, we consider complex data as a whole field and more precisely take interest in their integration, organization and analysis. Decision support technologies, such as data warehousing, data mining and on-line analytical processing (OLAP), must definitely address this issue. These techniques and tools have indeed showed their efficiency when data are "classical" (i.e., numerical or symbolic), but substantial adaptations are necessary to handle complex data.

In this paper, we propose an original approach for complex data warehousing. Our approach covers the three main phases of the classical data warehousing process and includes new methods to adapt them to complex data. These phases are:

1. *data integration*: complex data are represented in a unified format and stored in a database;
2. *dimensional modeling*: complex data are modeled to build analysis contexts (facts and dimensions);
3. *data analysis*: complex data are analyzed by appropriate tools to extract relevant information.

The first phase of the warehousing process is very important. We particularly focus on improving existing techniques for integrating complex data (Boussaid et al. 2003). To achieve this goal, we propose a framework that is based on conceptual, logical and physical modeling.

We build a generic UML model that enables the representation of complex data through their low-level and semantic descriptors. For instance, low-level characteristics of an image are color, texture, shape, etc. They are helpful for complementing the image's basic description, which includes file size, file location, creation date, etc. On the other hand, semantic characteristics deal with the image's content. This kind of information is difficult to obtain through an automatic process. It is generally represented as manual or semi-automatic annotations.

With the help of a piece of software we developed, our generic UML model enables the generation of XML documents describing the complex data that are to be integrated. These XML documents are physically stored either into a relational or an XML-native database. In both cases, we consider this repository as an Operational Data Storage (ODS) allowing data to later be loaded into a dimensional structure, namely a data warehouse, data mart or data cube. A data warehouse covers the whole activity of a company, while a data mart focuses on a given department within that company. A data cube is a narrower view of a data warehouse or a data mart, where only some measures and dimensions are considered.

Before the second phase of dimensional modeling takes place, complex data must be prepared. To achieve this goal, we propose a data mining step to enrich designer information about the stored data and to help building better-adapted and relevant data cubes.

The last phase of our approach is out of the scope of this paper. Nevertheless, we have achieved some advances in building on-line analysis tools adapted to complex data. The reader can find in BenMessaoud et al. (2004) a detailed presentation on a new operator that helps aggregating complex data. In a few words, this tool is based on a clustering technique, namely agglomerative hierarchical clustering. Every output class corresponds to an aggregation. Thus, we can summarize complex data according to different hierarchies. Furthermore, as we use XML to represent complex data, we have proposed an XML structure mining technique that exploits association rules to highlight relationships between XML tags in documents. The extracted information can then be used for content mining processing.

The aim of this paper is to present the first two phases of our approach and to illustrate how we prepare complex data for analysis. As a concrete example, we also present a whole case study. We also show how a data mining technique can be used to highlight the relevance of some particular characteristics in the dimensional modeling process.

The remainder of this paper is organized as follows. Section 2 discusses the context of our work. Section 3 details the complex data integration process we propose and presents the generic UML model we built to produce XML documents of complex data. Section 4 presents the complex data dimensional modeling phase. We also introduce the data mining step we advocate for, which helps acquiring better knowledge about complex data. Section 5 is dedicated to our case study. In Sect. 6, we present a discussion about how we situate our approach in the DSS context and we show our contribution in the complex data warehousing. We finally conclude the paper and discuss future research directions in Sect. 7.

2 Related work

Many approaches relate to data integration, such as mediator-based (Rousset and Reynud 2004) or warehousing approaches (Inmon 2005; Kimball and Ross 2002). We situate our work in the warehousing approach, which we adapt to handle complex data. Before detailing our contribution, we briefly present some related research.

2.1 Integration

Data integration was first addressed in the context of federative databases (Gardarin et al. 1984). Since then, mediation-based integration has been the scope of a lot of research. Some of these studies concern on the fly information integration for answering a query using existing information sources that are distributed and possibly heterogeneous. Other researches combine mediation-based integration and data warehousing (Rousset 2002; Rousset and Reynaud 2004). Finally, logic description frameworks or languages (e.g., CARIN) have been proposed for information integration (Calvanese et al. 1998; Goasdoué et al. 2000).

We address in this paper the issue of integrating complex data into a database, a subject that has been scarcely studied before. Jensen et al. though proposed a general system architecture for integrating XML and relational data sources at the conceptual level into a web-based OLAP database (Jensen et al. 2001). A "UML snowflake diagram" is built by choosing the desired UML classes from any of the two sources. The process is deployed through a graphical interface that deals only with UML classes and makes data sources transparent to the designer. Other approaches for an easy integration of complex data semantics have also been published. For instance, Jaimes et al. introduced a new framework for video content understanding. It is an expert system that exploits a rule-based engine, domain knowledge, visual detectors and metadata to enhance video detection results and to allow the semi-automatic construction of multimedia ontologies (Jaimes et al. 2003). Finally, semantic indexing techniques for complex data also exist (Stoffel et al. 1997). These techniques are based on domain knowledge under the form of ontologies.

To process complex data, it is also very important to consider their metadata. To take this information into account, appropriate management tools are necessary. For instance, an integrative and uniform model for metadata management in data warehousing environments, using a uniform representation approach based on UML to integrate technical and semantic metadata and their interdependencies, has been proposed (Stöhr et al. 2002). Standards for describing resources that we consider as complex data also exist. For example, the Resource Description Framework

(Lassila and Swick 1999) is a W3C-approved standard that uses metadata to describe Web contents. More generally, RDF is a language that allows to define and represent resources. Another important standard is MPEG-7 (Manjunath et al. 2002). Its most important goal is to provide a set of methods and tools for the different aspects of multimedia content description. MPEG-7 focuses on the standardization of a common interface for describing multimedia materials (representing information about contents and metadata).

Another important aspect of our work concerns dimensional modeling, which has little evolved since the birth of data warehousing.

2.2 Dimensional modeling

The database community has been devoting lots of attention to the data warehousing approach since the mid-nineties (Widom 1995; Wu and Buchmann 1997; Chaudhuri and Dayal 1997). Multidimensional databases and OLAP technologies indeed provide efficient solutions to manipulate and aggregate data in databases (Chaudhuri and Dayal 1997). Nowadays, database system features for business data analysis have become popular. This trend is obvious, given the popularity of many OLAP (Codd 1993) such as Essbase (Arbor Software) or Express (Oracle Corporation). These systems are based on a multidimensional, conceptual view of data. They are specifically tailored for data analysis and their characteristics are significantly different from those of relational databases. The analysis process concerns basic or aggregated data containing relevant information. OLAP allows data to be modeled in a dimensional way and to be observed from different perspectives. This approach consists in building data cubes (or hypercubes) on which OLAP operations are performed. A data cube is a set of facts described by measures to be observed along analysis axes (dimensions) (Kimball and Ross 2002; Inmon 2005; Chaudhuri and Dayal 1997). A dimension may be expressed through several hierarchies. Many aggregation levels for measures can be computed to obtain either summarized or detailed information using OLAP operators. Thus, hierarchies allow sophisticated analysis and data visualization in a multidimensional database (Jagadish et al. 1999).

Dimensional modeling has been designed for numerical, aggregative data. Hence, it must be adapted to handle the specifics of complex data. Our contribution addresses this issue. It is developed in the following sections.

3 A framework for complex data integration

The aim of the methodological framework we propose is to prepare complex data for warehousing and analysis. In the following, we detail the approach used to achieve complex data integration based on UML modeling.

3.1 Integration approach

We first proposed a formalization of complex data integration into a relational or XML-native database (Darmont et al. 2003). In the case of a relational database, XML documents describing complex data are mapped into relational tables allowing the use of analysis tools based on Relational DataBase Management Systems (RDBMS). Otherwise, the user may store XML documents into a repository and directly apply

XML-based analysis tools. Thus, the choice of the database type depends on the analysis tools the user intends to use.

Our integration approach is based on the classical modeling levels: conceptual, logical and physical. It is a generalization of the UML-XML-based approach introduced in (Darmont et al. 2003) that represents a complex object capable of figuring a wide range of complex data types. The complex object (root class, Fig. 1) can contain several subdocuments, and each subdocument may be of *text*, *relational view*, *image*, *video*, or *sound* type. This general conceptual model is converted into a logical model consisting of an XML grammar (Document Type Definition, DTD). Moreover, complex data are represented at the physical level by XML documents. The choice of XML as an implementation tool for complex data is interesting as it allows both the description and the contents of any document to be represented. The obtained database is considered as an ODS from which data are to be loaded into a future dimensional structure. Furthermore, an algorithm allows the XML documents to be built out of the complex data characteristics.

Structuring data for storage as well as their preparation for future analysis is necessary. In our approach we decide to process complex data represented by their characteristics gathered in vectors. Some basic characteristics (*file size*, *file name*, *duration*—for videos or sounds, *resolution*—for images, and so on) can be extracted automatically by using standard methods or ad hoc automatic extraction algorithms (Darmont et al. 2002). The extraction of other characteristics necessitate external processing techniques (i.e., image processing, etc.). All these characteristics capture low-level information concerning the original data.

The generic model that we present allows us to include also semantic characteristics of data to enhance their description by manually capturing this information (manual annotations). Our generic model defines a complex object that is composed of complex data represented as subdocuments (Fig. 1). The subdocuments have predefined low-level characteristics that depend on the type of complex data they contain. They also have an associated language and are annotated by keywords.

Subdocuments represent the basic data types and/or documents we want to integrate.

- Text documents are subdivided into plain texts and tagged texts (namely HTML, XML, or SGML documents). Tagged texts are further associated to a certain number of links. Since an hypertext document may point to external data (other pages, images, multimedia data, files...), those links help to relate these data to their referring document.
- Relational views are actually extractions from any type of database (relational, object, object-relational—we suppose a view can be extracted whatever the data model) that will be materialized in the data warehouse. A relational view is a set of attributes (columns, classically characterized by their name and their domain) and a set of tuples (rows). At the intersection of tuples and attributes is a data value. In our model, these values appear as ordinal, but in practice they can be texts or BLOBs containing multimedia data. At the physical level, the view is a temporary table that is not stored. The RDBMS only stores the query that generated the view. For instance, it might be inadequate to duplicate huge amounts of data, especially if the data source is not regularly updated. On the other hand, if successive snapshots of an evolving view are needed, data will have to be stored.

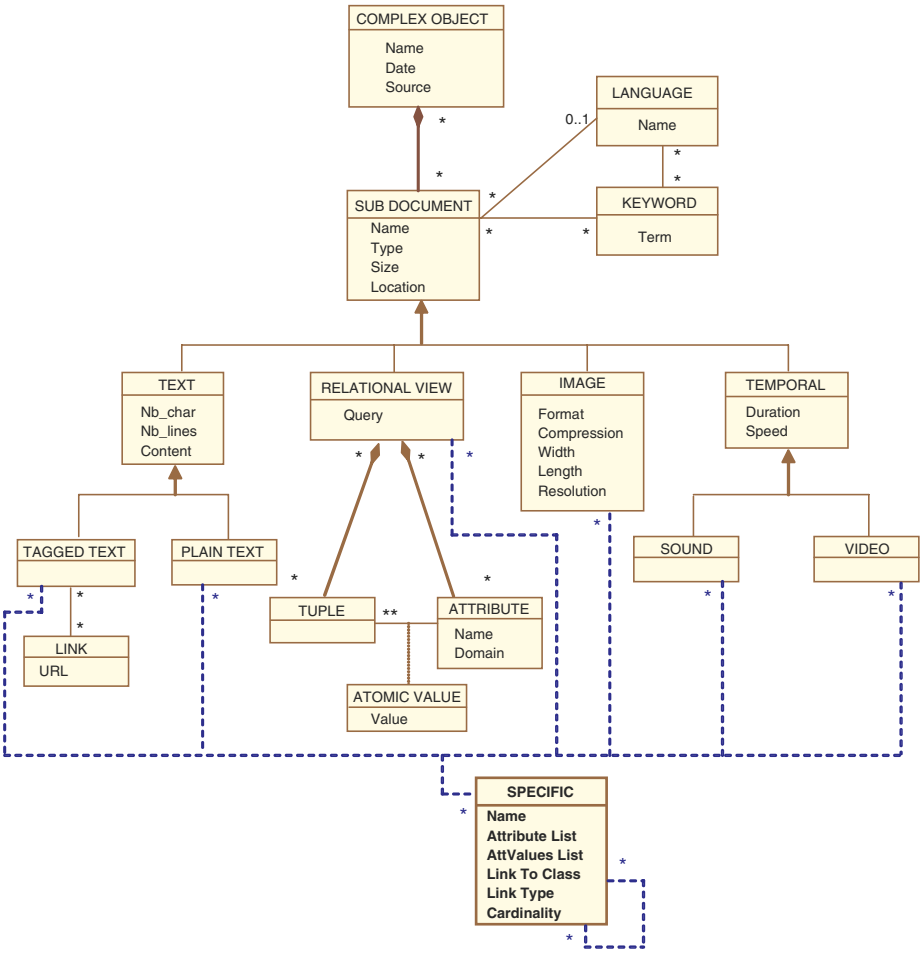

Fig. 1 Generic UML model for complex data

– Images may bear two types of attributes: some that are usually found in the image file header (format, compression rate, size, resolution), and some that need to be extracted by program, such as color or texture distributions.
– Sounds and video clips are part of a same class *Temporal* because they share continuous attributes that are absent from the other types of data we considered.

In Fig. 1, if we don't consider the class *Specific* and the set of dashed lines, the presented model is general, i.e., all the classes concern low-level information. To take into account information about the contents of complex data, we complete the model with a specific part composed of new classes and links representing the semantics. Therefore, we enrich the model with a metaclass called *Specific*. This metaclass is a generic class that allows us to define new classes and relationships in the UML model and thus enables the modeling of the semantic characteristics of complex data (in Sects. 5.1 and 5.2, the use of this metaclass is explained through an example). It allows us not only to describe semantic properties of data but also any other useful low-level

characteristics. The dashed lines in Fig. 1 are not predefined relations in the UML model. They point out possible relations that may be established between the classes of the model and classes instantiated from the metaclass. The relation type (association, aggregation, etc.) will be determined by an attribute of the instantiated class. For example, the instantiated classes *Circles* and *Scanner* and their associated aggregation relationships with the class *Image* are obtained from the metaclass *Specific* (Fig. 2). Note that when a class is instantiated from the metaclass it is linked to another class.

Let us formalize the concepts we use in our approach. We define the metaclass *Specific* based on the class definition used in Jensen et al. (2001).

Specific = {(*name*, ***AttributeList***, ***AttValuesList***, *link_to_class*, *link_type*, *cardinality*) | *link_type* ∈ ***Links*** and *cardinality* ∈ ***Cardinalities***}, where

- *name* - name of the new class to be created as an instance of ***Specific***
- ***AttributeList*** = {(*attribute_name*)}—list of attributes of the new class
- ***AttValuesList*** = {(*attribute_value*)}—list of attribute values (used at implementation time)
- *link_to_class*—name of the class to which the new class is linked
- ***Links*** = {association, aggregation, inheritance}—type of link
- ***Cardinalities*** = {0..1, 1, 1..*, *}

In other words, the metaclass *Specific* allows users to obtain a UML instance of the generic model that fulfills their specific needs.

We now consider an example: a set of complex data composed of mammographies and medical reports we want to describe using our generic UML model. We obtain a diagram composed of classes: *Complex Object, Subdocument, Text, PlainText*, and *Image* (Fig. 2). We suppose that physicians annotate mammographies by circling the areas suspected of being cancerous. To allow this specific description of our complex data, we instantiate the metaclass to obtain a class *Circles* containing an attribute expressing the type of pathology, and one or several attributes for the circle location. We also generate (instantiate) the class *Scanner* to describe the digitalizer used for the mammographies. The *Circles* class contains semantic information, while the *Scanner* class represents low-level information.

3.2 Generic model implementation

In order to validate our approach, we implemented a prototype named *CDO2XML* (Tanasescu and Boussaid 2003) that can create XML documents describing our complex data. The XML documents are meant to be integrated in an XML-native database. The program works in two steps.

1. The first step allows the visual definition of a UML model for the complex data, in a given context, using our generic model as a background. The program converts the obtained UML model into an XML DTD.
2. The second step allows the user to physically generate valid XML documents according to the UML model defined in the first step (Fig. 3).

Once the desired UML model is created, by instantiating the initial generic model, CDO2XML generates a corresponding DTD. In the second step, some characteristics are automatically extracted by ad hoc algorithms. Furthermore, to complete these characteristics, the program asks the user to browse the sub-documents that compose the complex object and eventually to declare the special characteristics s/he defined

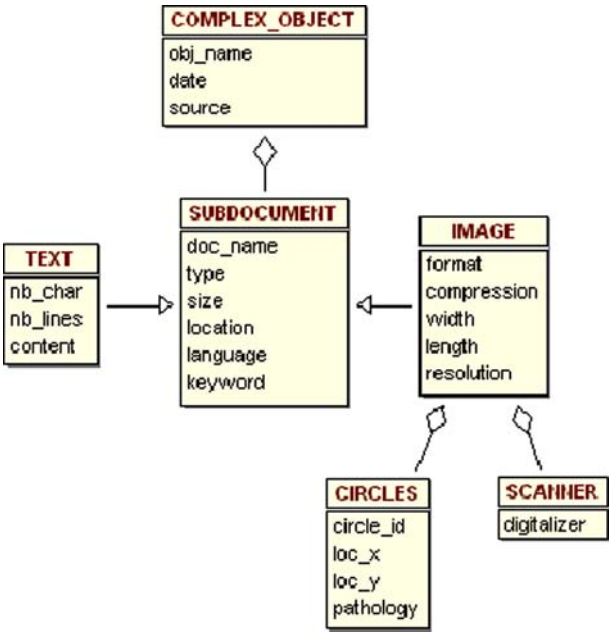

Fig. 2 Instantiated UML model used with mammographies

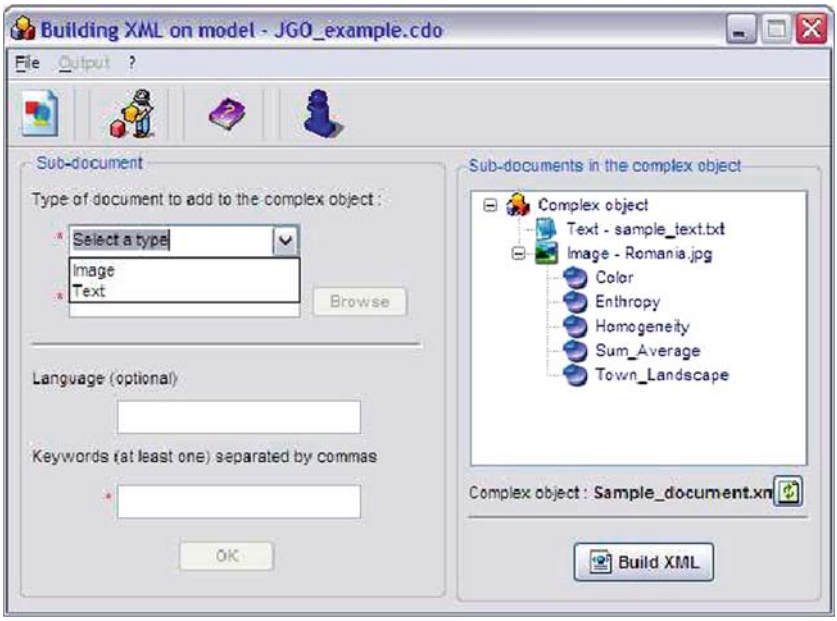

Fig. 3 CDO2XML screenshot—Building XML documents

for the complex data. When all the required information is available, the application can generate valid XML documents describing the complex data (a generated XML document is provided in Appendix 2). Note that no XML document is created until all the types of complex data composing the complex object are included. For instance, if the complex data contains a piece of text and an image, the user has to provide one text document and one image before proceeding to the generation of the XML document.

Globally, this application brings three contributions to the users who need to integrate complex data.

- It allows users to create an instance of the generic model that is adapted to their needs (Fig 4).
- It creates a DTD describing the UML model (see Appendix 1), which is a straightforward correspondence with the UML model in Fig. 4. This DTD can be used to define the structure of the relational database where the XML documents will be mapped if a relational structure is chosen for storage.
- It allows the user to build valid XML documents that describe the complex data, according to the previously generated XML DTD.

The integration of complex data, presented as XML documents, is concluded by their physical integration into relational or XML-native database. The obtained database represents an ODS for the complex data warehousing process that is followed by the dimensional modeling phase.

4 Complex data dimensional modeling

4.1 A dimensional modeling framework

The storage of complex data in relational databases On-Line Transactional Processing (OLTP) cannot provide data analysis capabilities.

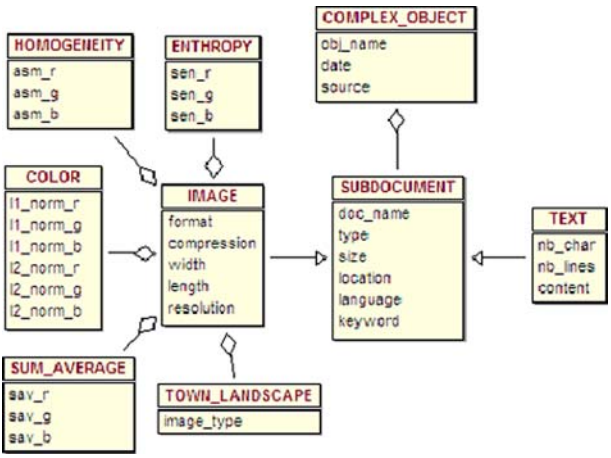

Fig. 4 Sample implementation of our generic model using CDO2XML

Though, correct relational integration of complex data is necessary to store them into an ODS. We need to represent complex data in a dimensional way. In fact, in a warehousing process, dimensional modeling allows the creation of appropriate analysis contexts and the preparation of data for analysis. Dimensional modeling consists of structuring data into a star schema. Such a schema includes a fact table containing measures that are indicators and several dimension tables enclosing the descriptors of these measures. These dimensions represent different observation axes and may be developed by hierarchies representing different granularity levels of data. In this case, we obtain a snowflake schema. When it contains more than one fact table the dimensional model is called a constellation (Chaudhuri and Dayal 1997; Inmon 2005; Kimball and Ross 2002). Figure 6 represents a dimensional model in the star schema form.

Complex data are very rich in content and thus, need adapted dimensional structures allowing their analysis. The difference with classic data is that complex data often bear characteristics with a higher level of abstraction. This can be very difficult from the designer's point of view as s/he might not take part in the definition of all the characteristics describing the complex data to analyze. Therefore, we define a framework for the process of complex data dimensional modeling, that supports the designer to aquire the right data to be modeled.

1. *Define analysis goals*
2. *Identify data sources*
   Often, complex data are stored in heterogeneous and distributed data sources. This step is crucial and is influenced by the selected analysis goals.
3. *Build a data and metadata repository for modeling support*
   This repository is a crucial component to the complex data dimensional modeling process. It contains both data and metadata, which are information about data. Metadata are essentially definitions and descriptions, i.e., the source of data, their nature, their type, their domain and so on. Other metadata can describe structures and rules used to extract data from their original sources. This repository must bring a consistent support in the definition of measures and dimensions. It must:
   (a) list exhaustively the necessary data for the dimensional modeling process;
   (b) describe the characteristics of data and define their role in the dimensional model to be created;
   (c) help to choose the facts to be modeled.
4. *Select relevant data according to the analysis goals using data mining techniques*
   The main objective of this step is to extract information about the data to be modeled in a dimensional way. The obtained information is defined in the form of rules, that constitute a useful knowledge, necessary in the complex data modeling. From these extracted rules, we can discover relevant variables that will be considered as facts and dimensions in the dimensional model.
5. *Select facts to be analyzed according to the analysis goals*.
   This consists of describing the measures according to the analysis goals, describing the dimensions (analysis axes) and the dimension hierarchies, and making sure that the latter aggregate correctly.

In the dimensional modeling process, we decide not to handle complex data themselves but only their characteristics (see Sect. 3.1, 3rd paragraph). This is why, basic

and semantic characteristics are extracted or created in order to represent original data in the most accurate way. Building a dimensional model based on characteristics is more difficult than modeling classical data for warehousing. The relevant characteristics should be extracted according to the analysis goals. If the analysis objective is defined once the data are already stored in an ODS, we need to find which characteristics of our complex data are more relevant with respect to the analysis goals. This is not an easy task for the designers even if they have sufficient information about the data and their characteristics. Therefore, data mining techniques can help designers in the choice of the facts to be analyzed. These facts must be expressed in term of measures and dimensions representing the relevant characteristics highlighted by the mining process.

4.2 Mining complex data for dimensional modeling

The data integrated into the data warehouse is meant to be analyzed in a dimensional way by building data cubes. A data cube is a partial or a complete view of the data warehouse. It may be composed by one or more measures and some or all dimensions, that offer different views of measures upon different analysis axes. Dimensional modeling has a vocation to prepare data for On-Line Analysis (OLAP). This preparation consists of aggregating data to reduce their huge volume and summarize the information contained in the data. Then, the user can observe these aggregates from different points of view.

To model the data coming from the ODS in a dimensional way, it is necessary to beforehand have information about these data in the form of domain knowledge or metadata. Another way consists of directly extracting this information from the data by using data mining techniques.

The choice of these measures and dimensions is not an obvious task. Data mining techniques can help to find relevant characteristics to analyze. Data mining techniques are very useful in analyzing classical data to find correlations between variables and highlight causal relations among them. We use these techniques to discover the relevance of the data to the analysis goals.

Hence, we consider it to be useful to evaluate the relevance of the variables stored in the ODS using data mining techniques (mainly decision trees). We consider that this step can improve the construction of the data cube. In the case of complex data, this technique can be applied not only on the basic characteristics of data but also on the semantic ones. Additional information obtained from the data mining step can be integrated in the metadata repository and then used in the definition of the complex data cubes.

In the following case study, we present how we use a data mining technique, namely a decision tree method, to build an interesting data cube.

5 A case study of complex data dimensional modeling

In this section, we show how our approach works through an example. To process complex data, we first extract their characteristics and then we apply our approach to represent them into valid XML documents. These documents are integrated into a database that constitutes an ODS. Data mining techniques are then applied on the data to help the user in the dimensional modeling process.

5.1 Example

In this example, we consider complex data represented by images (towns and landscapes) and texts. The basic description of the images (resolution, file size, etc.) are extracted by ad hoc algorithms of our application. Other low-level characteristics (color, homogeneity, entropy, etc.—Fig. 4) are already extracted using image processing techniques. This phase is external to our approach and is achieved beforehand. Furthermore, we decide to add a semantic characteristic specifying whether images represent towns or landscapes. We discovered that some of these characteristics were very relevant in the dissociation between images representing natural landscapes and towns. It is quite obvious that different colors are supposed to dominate in the two classes of images, but the same characteristics are insufficiently discriminating in other situations. Therefore, data mining techniques, namely decision trees, can highlight the importance of each characteristic in such a situation.

In the first phase of our approach, we obtained the UML diagram from Fig. 4, instantiated from our generic UML model (Fig. 1). The town and landscape images, as complex data, are represented by the *Complex Object* class composed by two subdocuments (*Text* and *Image* classes). Furthermore, five classes are generated from the metaclass *Specific* to describe the images by color and texture characteristics (*Entropy, Homogeneity, Color, Sum_Average, Town_Landscape*).

5.2 Integrating images and text into an ODS

Figure 4 shows how the generic model can be instantiated by the user in order to obtain the characteristics he desires about all the images. The complexity of the example is voluntarily reduced for an easier understanding. All the generated classes (e.g., *Homogeneity*, *Color*, etc.) are instances of the metaclass *Specific* (Fig. 1). Nevertheless, the use of this metaclass through the prototype CDO2XML is invisible to the user. The translation between the metaclass *Specific* and its instances is made as follows:

- *Specific*.name—its value gives the name of the class (i.e., *Homogeneity*);
- *Specific*.AttributeList—these values contain the attribute names of the new class (i.e., *ASM_R*, *ASM_G*, *ASM_B*);
- *Specific*.link_to_class—its value (Image) shows that our new class links to *Image* class;
- *Specific*.link_type—its value (aggregation) shows the link type with *Image*;
- *Specific*.cardinality—shows cardinalities of *Specific*.link_to_class attribute.

Similar results are obtained for the other instantiated classes.

In our example, we describe some *Red*, *Green*, *Blue* (*RGB*) characteristics of images (Scuturici 2002). Thus we consider *Angular Second Moment* (*ASM*) that measures the homogeneity of an image, *L1Normalised* (medium color characteristic of an image), and *Sum of Entropy* (*SEN*) as relevant characteristics (Haralick et al. 1973) for our future analysis. We decline these characteristics for each of the three color channels (*RGB*) and we obtain the following variables *ASM_R*, *ASM_G*, *ASM_B*, *L1Norm_R*, *L1Norm_G*, *L1Norm_B*, and so on.

For each image, the user specifies the values corresponding to these attributes. Once the characteristics are provided, CDO2XML produces, from the instantiated UML model, the corresponding DTD (Appendix 1) and generates the valid XML

documents (Appendix 2) representing the complex data. At this point of our approach, the data are integrated into an ODS which is either an XML repository or a relational database. We can consider the integration process completed. We now address the dimensional modeling phase.

5.3 Decision trees for dimensional modeling

In order to model the complex data as a data cube, the designer must define the main analysis goals. In our example, we consider analyzing images representing towns and landscapes and we decide to analyze the way image characteristics influence this dissociation. We know beforehand that information about color and texture can help discriminating between a landscape and a town picture (field knowledge).

From the stored XML documents and the corresponding XML DTD, we extract data and metadata to build a repository as modeling support. This repository is a long list of terms (attributes) concerning a field the user is generally not familiar with. Thus, it is necessary to collect information about these terms. In our example, we use the decision tree technique, which is an automatic classification method.

Decision tree-based methods are supervised learning methods to automatically classify individuals into well-defined classes. Generally, the individuals population is divided in two sets, a learning set and a test set. The aim of this method is to extract classification rules from the learning set and validate them on the test set. When the number of individuals is small, the learning phase is applied on the whole population and the test phase is achieved through a cross-validation method (for more details see Quinlan 1993), like in our example.

Building a decision tree consists of successively partitioning a population (images in our case) in order to obtain well-defined classes of individuals (Fayyad et al. 2001; Witten and Frank 2005). To achieve this goal, data are represented in an *individual-attribute* array containing images as individuals and characteristics as attributes. These characteristics (such as *ASM_R*, *ASM_G*, *ASM_B*, and so on) represent predictive attributes, while the attribute to be predicted, the class attribute, (*Image_Type*) has two modalities: *town* and *landscape*. Both modalities are the labels of the well-defined classes in the decision tree we want to build. The paths from the decision tree root to its leaves are expressed as rules that classify each individual into the corresponding class. With the built decision tree, we obtain a set of rules that may not be all relevant. There are methods to help selecting the relevant ones (Fayyad et al. 1996).

The machine learns how to correctly classify new images into the corresponding class. Thus, we obtain the confirmation that characteristics revealed by the rules are relevant for discriminating the two types of images. We use the C4.5 algorithm (Quinlan 1993) to build the decision tree and extract the rules . Some of these rules are shown in Table 1.

The decision tree (Fig. 5) shows that the most relevant characteristic to dissociate between towns and landscapes, is *L2Norm_G* [1] that measures the "weight" of the green color in the images. Note that the characteristic *L2Normalised*, combined with other characteristics, allows to determine whether images represent towns or landscapes. We find it interesting to observe it in relation to the other relevant characteristics. Thus, we decide to consider *L2Normalised* (for each of the three colors) as the fact to analyze.

[1] In Fig. 5, L2Norm_G is represented as L2Norm_V (Green is said *Vert* in French).

Table 1 Sample rules produced by the decision tree

– if *L2Norm_G* >= 0.38 and *ASM_R* < 0.001 then Image is of type *Landscape*;
– if *L2Norm_G* < 0.38 and *SEN_R* >= 4.61 then Image is of type *Town*;
– if *L2Norm_G* >= 0.38 and *ASM_R* > 0.001 and *L2Norm_R* < 0.37 then Image is of type *Landscape*.
– …

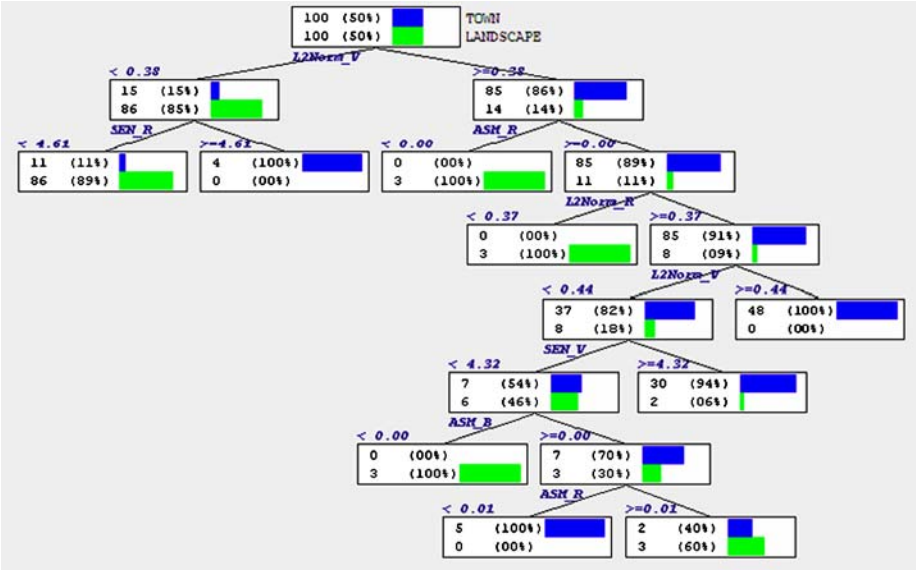

Fig. 5 Sample decision tree built with Sipina Research Edition

Using the information obtained through the data mining step, we can build the star schema (Kimball and Ross 2002) of our dimensional model. In this model the fact table contains the indicator *L2Normalised* declined into the following measures; *L2Norm_R*, *L2Norm_G*, *L2Norm_B*. These measures will be observed through four dimensions representing the homogeneity (*Homogeneity* Dimension), the entropy (*Entropy* Dimension), the sum average (*Sum average* Dimension) and finally the *Town_Landscape* Dimension. These characteristics are the predictive attributes in the individual-attribute array, which are determined as relevant by the decision tree algorithm.

Figure 6 shows the dimensional model obtained following our approach.

The obtained dimensional model can be sufficient according to the initial analysis goals or can be complemented by the users if their needs evolve, for instance, if they want to add dimension hierarchies to create other levels of observation.

The three measures of the dimensional model (Fig. 6) can now be analyzed through the four defined dimensions that were revealed as relevant by the data mining process. No hierarchies were defined for these dimensions. The example was built as simple as possible and highlights the relevance of some characteristics to the analysis. Through the knowledge discovery in the complex data the users are helped to choose the most relevant characteristics according to the analysis objectives.

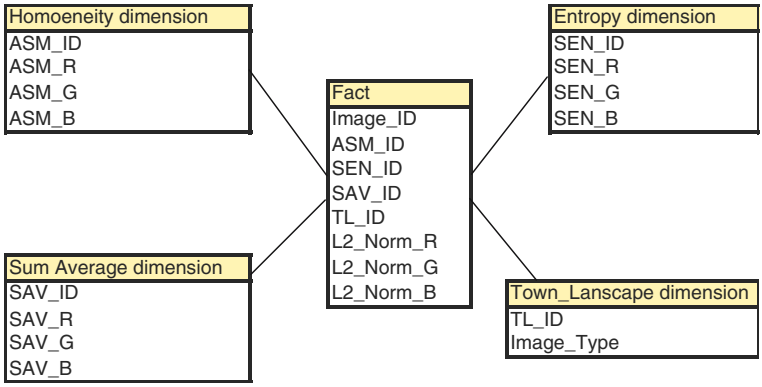

Fig. 6 Sample dimensional model built using the obtained metadata

6 Discussion

Research in data warehousing and OLAP has produced important technologies for the design, management and use of information systems for decision support. Nevertheless, even though the high maturity of these technologies, there are new data needs in companies. These needs not only demand more capacity or storing necessities, but also new methods, models, techniques or architectures. Some of the hot topics in data warehouses include web data, multimedia data or biomedical data that we call more generally complex data. Warehousing such data involves a lot of different issues regarding their structure, storage, processing and analyzing. In data warehousing, the prime objective of storing data is to facilitate decision process in a company. 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 relevant analysis in 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.

The special nature of complex data poses different and new requirements to data warehousing technologies, over those posed by conventional data warehouse applications. This paper presents a number of interesting new research challenges posed by complex data warehousing, to be met by the database research community. These include the need for complex data modeling features, the integration of complex data and the complex data analysis.

To integrate complex data sources, we need more than a tool for organizing data into a common syntax. Data integration is a hard task that involves reconciliation at various levels (data models, data schema, data instances, semantics). Moreover, OLAP systems typically employ multidimensional data models to structure their data. However, current multidimensional data models fail in their abilities to model the complex data found in some real-world application domains.

In this paper, we proposed a general framework to warehouse complex data. We used XML as the canonical standard to transform and store complex data from original data sources and used a DTD to define a global ODS schema. We shown that XML can greatly help the task of complex data integration.

Data integration corresponds to the ETL phase in the data warehousing process. To achieve complex data integration, the traditional ETL approach is not adapted. Indeed, the variety of data types (images, texts, sounds, videos, databases) increases the complexity of data. It is necessary to structure them in a nonclassical way. Moreover, because data are complex, they need more information to be described. Hence, it is important to consider this information and to represent it as metadata. The choice of the XML formalism is thus fully justified since its self-describing hierarchical structure allows to represent both data and metadata. Complex data are then represented as XML documents that we generated through an prototype software.

Starting from the obtained XML documents that we consider as the sources of the complex data warehouse, we proposed an approach for building the multidimensional conceptual schema for a complex data warehouse. This paper defines a new strategy for modeling complex data in a multidimensional way using data mining methods. Indeed, the models obtained from data mining methods, that represent information about the data, are used to define relevant analysis axes in the multidimensional model.

The choice of a data model for a complex data warehouse requires more than only data sources and analysis goals. Complex data are very rich in content and thus need a more sophisticated process to take into account both data and their semantics represented under the form of characteristics. Building a multidimensional model based on the characteristics extracted from complex data rather than complex data themselves is not an easy task. It needs the use of appropriated methods such as data mining. Moreover, the obtained characteristics should respond to the user's analysis goals.

Our approach for complex data warehousing presents several advantages. We can cite the unified, XML complex data format and the use of data mining techniques for extracting relevant information that are necessary for building dimensional models. The main contribution of our research is then to combine data mining techniques with complex data warehousing. However, there are some limitations of our approach. In this paper, we used decision trees and the rules we obtained are not all relevant. The selection of the more relevant rules is a problem that can affect our multidimensional model. On the other hand, other data mining techniques exist, but finding the best technique and assessing its impact on the multidimensional model is intricate. In addition, only some basic characteristics are extracted by our software. Other external techniques are necessary to complement the description of complex data significantly.

7 Conclusions and future work

In this paper, we exposed an approach to warehouse complex data. First, we presented a generic UML model that allows us to model not only low-level but also semantic information concerning the complex data to be analyzed. We integrated complex data as XML documents into an ODS as a first step in complex data warehousing. To validate our approach, we have implemented a prototype application that assists the designer in the description of complex data as XML documents. Then, we proposed an approach for supporting complex data dimensional modeling. It consists of using data mining techniques to enrich the knowledge of designers about the stored complex data and help them build better adapted data warehouses. We have illustrated both our approaches, the integration and the dimensional modeling of complex data, with an example.

There are several future research issues related to our work. The first one concerns the complex data integration phase. First, to represent complex data in XML format we used a DTD as an XML grammar. While DTDs offer only one data type (PCDATA), XML Schema offers more data types and also allows to define complex types. In addition, an XML Schema consists of type definitions, which can be derived from each other. XML Schema gives a more accurate representation of the XML structure constraints than DTDs. Second, the extraction of complex data characteristics is a crucial problem. Often, specific methods such as image processing are needed to obtain the color, texture and shape information about images; or data mining methods are used to extract semantic information. We are exploring ways for automatically importing the extracted characteristics in our integration process.

Another perspective in the dimensional modeling phase is to carry on the representation of data mining generated rules as metadata in a mixed structure combining XML Schema and RDF Schema. This structure is better suited for expressing semantic properties and relationships between metadata.

Finally, in the last phase of our approach, we pursue the development of complex data analysis tools combining data mining and OLAP techniques and mining XML documents.

Appendix 1: XML DTD generated by CDO2XML

```
<!ELEMENT COMPLEX_OBJECT (OBJ_NAME, DATE, SOURCE, SUBDOCUMENT+)>
  <!ELEMENT OBJ_NAME (#PCDATA)>
  <!ELEMENT DATE (#PCDATA)>
  <!ELEMENT SOURCE (#PCDATA)>
<!ELEMENT SUBDOCUMENT (DOC_NAME, TYPE, SIZE, LOCATION, LANGUAGE?, KEYWORD*,
  (IMAGE | TEXT))>
  <!ELEMENT DOC_NAME (#PCDATA)>
  <!ELEMENTTYPE (#PCDATA)>
  <!ELEMENT SIZE (#PCDATA)>
  <!ELEMENT LOCATION (#PCDATA)>
  <!ELEMENT LANGUAGE (#PCDATA)>
  <!ELEMENT KEYWORD (#PCDATA)>
<!ELEMENT IMAGE (COMPRESSION, FORMAT, RESOLUTION, LENGTH, WIDTH,
ENTROPY*, HOMOGENEITY*, COLOR*, SUM_AVERAGE*, TOWN_LANDSCAPE*)>
  <!ELEMENT COMPRESSION (#PCDATA)>
  <!ELEMENT FORMAT (#PCDATA)>
  <!ELEMENT RESOLUTION (#PCDATA)>
  <!ELEMENT LENGTH (#PCDATA)>
  <!ELEMENT WIDTH (#PCDATA)>
<!ELEMENT ENTROPY (SEN_R, SEN_G, SEN_B)>
  <!ELEMENT SEN_R (#PCDATA)>
  <!ELEMENT SEN_G (#PCDATA)>
  <!ELEMENT SEN_B (#PCDATA)>
<!ELEMENT HOMOGENEITY (ASM_R, ASM_G, ASM_B)>
  <!ELEMENT ASM_R (#PCDATA)>
  <!ELEMENT ASM_G (#PCDATA)>
  <!ELEMENT ASM_B (#PCDATA)>
<!ELEMENT COLOR (L1_NORM_R, L1_NORM_G, L1_NORM_B, L2_NORM_R, L2_NORM_G, L2_NORM_B)>
  <!ELEMENT L1_NORM_R (#PCDATA)>
  <!ELEMENT L1_NORM_G (#PCDATA)>
  <!ELEMENT L1_NORM_B (#PCDATA)>
  <!ELEMENT L2_NORM_R (#PCDATA)>
  <!ELEMENT L2_NORM_G (#PCDATA)>
  <!ELEMENT L2_NORM_B (#PCDATA)>
<!ELEMENT SUM_AVERAGE (SAV_R, SAV_G, SAV_B)>
  <!ELEMENT SAV_R (#PCDATA)>
  <!ELEMENT SAV_G (#PCDATA)>
  <!ELEMENT SAV_B (#PCDATA)>
<!ELEMENT TOWN_LANDSCAPE (IMAGE_TYPE)>
  <!ELEMENT IMAGE_TYPE (#PCDATA)>
```

```
<!ELEMENT TEXT (NB_CHAR, NB_LINES, CONTENT)>
  <!ELEMENT NB_CHAR (#PCDATA)>
  <!ELEMENT NB_LIGNES (#PCDATA)>
  <!ELEMENT CONTENT (#PCDATA)>
```

This XML DTD is a translation of the instantiated UML model presented in Fig. 4, where each class and each attribute is described by an ELEMENT tag, which encloses the attributes of the class. We consider this DTD as a logical model to describe complex data.

Appendix 2: XML document generated by CDO2XML

```
<!DOCTYPE ComplexData SYSTEM "JGO_example.dtd">
<COMPLEX_OBJECT>
<OBJ_NAME>Sample_document.xml</OBJ_NAME>
 <DATE>jeudi, 20 july 2005  a  10:47</DATE>
 <SOURCE>Local</SOURCE>
 <SUBDOCUMENT>
  <DOC_NAME>France.jpg</DOC_NAME>
  <TYPE>Image</TYPE>
  <SIZE>210.73Ko</SIZE>
  <LOCATION>E:\France.jpg</LOCATION>
  <KEYWORD>country</KEYWORD>
  <KEYWORD>mountains</KEYWORD>
  <IMAGE>
   <COMPRESSION>N/A</COMPRESSION>
   <FORMAT>jpg</FORMAT>
   <RESOLUTION>200x150</RESOLUTION>
   <LENGTH>150</LENGTH>
   <WIDTH>200</WIDTH>
   <COLOR>
    <L1_NORM_R>0.54</L1_NORM_R>
    <L1_NORM_G>0.63</L1_NORM_G>
    <L1_NORM_B>1.24</L1_NORM_B>
    <L2_NORM_R>0.32</L2_NORM_R>
    <L2_NORM_G>0.64</L2_NORM_G>
    <L2_NORM_B>0.23</L2_NORM_B>
   </COLOR>
   <ENTROPY>
    <SEN_R>3.54</SEN_R>
    <SEN_G>4.64</SEN_G>
    <SEN_B>1.25</SEN_B>
   </ENTROPY>
   <HOMOGENEITY>
    <ASM_R>0.001</ASM_R>
    <ASM_G>0.0021</ASM_G>
    <ASM_B>0.0054</ASM_B>
   </HOMOGENEITY>
   <SUM_AVERAGE>
    <SAV_R>2.54</SAV_R>
    <SAV_G>6.52</SAV_G>
    <SAV_B>2.25</SAV_B>
   </SUM_AVERAGE>
   <TOWN_LANDSCAPE>
    <IMAGE_TYPE>landscape</IMAGE_TYPE>
   </TOWN_LANDSCAPE>
  </IMAGE>
 </SUBDOCUMENT>
</COMPLEX_OBJECT>
```

This XML document is a valid document with respect to the DTD from Appendix 1, and represents a complex object containing one image type subdocument. It is described by its characteristics (type, size, compression, color, entropy, etc.) represented by classes and attributes. The XML document represents a physical model in our integration process.

References

BenMessaoud, R., Boussaid, O., Rabaseda, S.: A new OLAP aggregation based on the AHC technique. In: ACM 7th International Workshop on Data Warehousing and OLAP (DOLAP 04), pp. 65–72. Washington DC, USA (2004)

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

Calvanese, D., Giacomo, G.D., Lenzerini, M., Nardi, D., Rosati, R.: Description logics framework for information integration. Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), pp. 2–13. Trento, Italy (1998)

Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. SIGMOD Record **26**(1), 65–74 (1997)

Codd, E.: Providing OLAP (on-line analytical processing) to user-analysts: an IT mandate. Tech. rep., E.F. Codd and Associates (1993)

Darmont, J., Boussaid, O. (eds.): Processing and Managing Complex Data for Decision Support. Idea Group Publishing, Hershey, PA, USA (2006)

Darmont, J., Boussaid, O., Bentayeb, F., Rabaseda, S., Zellouf, Y.: Web Multiform Data Structuring for Warehousing, Vol. 22 of Multimedia Systems and Applications, pp. 179–194. Kluwer Academic Publishers. In: Djeraba, C. (ed.) Multimedia Mining: A Highway to Intelligent Multimedia Documents (2003)

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

Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. MIT Press, Cambridge (1996)

Fayyad, U., Grinstein, G., Wierse, A.: Information Visualization in Data Mining and Knowledge Discovery. Morgan Kaufmann (2001)

Gardarin, G., Simon, E., Verlaine, L.: Temporal view: a tool for real time distributed data bases. In: Distributed Data Sharing Systems (DDSS), pp. 195–202. Parma, Italy (1984)

Goasdoué, F., Lattès, V., Rousset, M.-C.: The use of CARIN language and algorithms for Information Integration: The PICSEL Project. Int. J. Coop. Inform. Syst. **9**(4), 383–401 (2000)

Grabczewski, E., Cosmas, J., Santen, P.V., Green, D., Itagaki, T., Weimer, F.: 3D MURALE: multimedia database system architecture. In: 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage, pp. 315–322. Glyfada, Greece (2001)

Haralick, R., Shanmugan, K., Dinstein, I.: Texture Features for Image Classification, Vol. 3 of Man and Cybernetics, pp. 610–622. IEEE Transactions Systems (1973)

Inmon, W.: Building the Data Warehouse, 4th edn. John Wiley and Sons (2005)

Jagadish, H.V., Lakshmanan, L.V.S., Srivastava, D.: What can hierarchies do for data warehouses? In: 25th International Conference on Very Large Data Bases (VLDB'99), pp. 530–541. Edinburgh, Scotland, UK (1999)

Jaimes, A., Tseng, B.L., Smith, J.R.: Modal keywords, ontologies, and reasoning for video understanding. In: Image and Video Retrieval, Second International Conference, CIVR 2003, pp. 248–259. Urbana-Champaign, IL, USA (2003)

Jensen, M.R., Møller, T.H., Pedersen, T.B.: Specifying olap cubes on xml data. In: 13th International Conference on Scientific and Statistical Database Management, pp. 101–112. Fairfax, Virginia, USA (2001)

Kimball, R., Merz, R.: The Data Webhouse. Eyrolles (2000)

Kimball, R., Ross, M.: The Data Warehouse Toolkit. John Wiley and Sons (2002)

Lassila, O., Swick, R.: RDF Model and Syntax Specification, http://www.w3.org/TR/REC-rdf-syntax/ (1999)

Manjunath, B., Salembier, P., Sikora, T.: Introducton to MPEG-7: Multimedia Content Description Interface. Wiley (2002)

Quinlan, R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)

Rousset, M.: Knowledge representation for information integration. In: XIIIth International Symposium on Methodologies for Intelligent Systems (ISMIS 2002), Lyon, France, Vol. 2366 of LNAI, pp. 1–3 (2002)

Rousset, M., Reynaud, C.: Knowledge representation for information integration. Inf. Syst. **1**(29), 3–22 (2004)

Saad, K.: Information-based medicine: a new era in patient care. ACM 7th International Workshop on Data Warehousing and OLAP (DOLAP 04), p. 58. Washington, USA (2004)
Scuturici, M.: Contribution to object oriented techniques in the management of video sequences for Web servers, PhD thesis, INSA Lyon, France (2002)
Stoffel, K., Saltz, J., Hendler, J., Dick, J., Merz, W., Miller, R.: Semantic indexing for complex patient grouping. In: Annual Conference of the American Medical Informatics Association (1997)
Stöhr, T., Müller, R., Rahm, E.: An integrative and uniform model for metadata management in data warehousing environment. In: 5th ACM international workshop on Data Warehousing and OLAP (DOLAP02), pp. 35–42. McLean, USA (2002)
Tanasescu, A., Boussaid, O.: CDO2XML - Complex Data Object to XML - XML documents generation prototype, http://bat710.univ-lyon1.fr/~atanases/CDO/install_en.zip (2003)
Widom, J.: Research problems in data warehousing. In: 1995 International Conference on Information and Knowledge Management (CIKM'95), pp. 25–30. Baltimore, Maryland, USA (1995)
Witten, I., Frank, E. (eds.): Data Mining: Practical Machine Learning Tools and Techniques. 2nd edn. Morgan Kaufmann (2005)
Wu, M.-C., Buchmann, A.P.: Research issues in data warehousing. In: Datenbanksysteme für Business, Technologie und Web (BTW'97), pp. 61–82. Ulm, Germany (1997)