Preservation of Data Warehouses: Extending the SIARD System with DWXML Language and Tools

Carlos Aldeias, Gabriel David, and Cristina Ribeiro
INESC TEC / DEI, Faculdade de Engenharia, Universidade do Porto, Portugal

ABSTRACT

Data warehouses are used in many application domains, and there is no established method for their preservation. A data warehouse can be implemented in multidimensional structures or in relational databases that represent the dimensional model concepts in the relational model. The focus of this work is on describing the dimensional model of a data warehouse and migrating it to an XML model, in order to achieve a long-term preservation format. This chapter presents the definition of the XML structure which extends the SIARD format used for the description and archive of relational databases, enriching it with a layer of metadata for the data warehouse components. Data Warehouse Extensible Markup Language (DWXML) is the XML language proposed to describe the data warehouse. An application that combines the SIARD format and the DWXML metadata layer supports the XML language and helps to acquire the relevant metadata for the warehouse and to build the archival format.

Keywords: Database Preservation, Data Warehouse Preservation, Metadata, DWXML, SIARD Format.

INTRODUCTION

The technological generation in which we live has gradually modified the method to create, process and store information, using digital means for this purpose. The institutions, enterprises and governments rely more and more on information systems that increase the availability and accessibility of information. These information systems typically require relational databases, which become valuable assets for those entities.

However, rapid technological changes degenerate into rapid obsolescence of applications, file formats, media storage and even databases management systems (DBMS) (Date, 2004). If nothing is done, access to large chunks of stored information may become impossible and it will eventually be lost. So, it is important that entities which have major responsibilities in preserving information in digital form become aware of this problem and join initiatives all over the world, seeking for the best methodology for long-term digital preservation, and in particular for database preservation.

The work presented here has been developed in the context of DBPreserve, a research project funded by the Portuguese Foundation for Science and Technology (FCT), in collaboration with INESC Porto, University of Minho and the Portuguese National Archives (DGARQ). The project

goal is to study the feasibility of data warehousing technologies to preserve complex electronic records, such as those constituting databases. The DBPreserve project approaches the long-term preservation of relational databases issue with a new concept, a two step migration:

– A model migration from the relational model to the dimensional model, using data warehouse concepts to simplify the model simplification and increase efficiency (Rahman, David & Ribeiro, 2010);

– An XML migration from the dimensional model to an XML (Consortium, 2008) format that represents the data warehouse, to ensure a long-term preservation format.

A data warehouse is structured by star or snowflake representations. A star is made up of a fact table that stores the facts, and dimensional tables that contextualize the facts. There are also bridge tables used to resolve a many to many relationship between a fact table and a dimension table, or to flatten out a hierarchy in a dimension table. A snowflake is similar to a star but the dimension tables have been subject to a partial normalization, resulting in subdimensions. Data marts are subsets of a data warehouse.

We propose the Data Warehouse Extensible Markup Language (DWXML), an XML dialect for describing a Data Warehouse (DW) (Inmon, 2002; Kimball & Ross, 2002; Date, 2004). It has been defined and refined according to data warehouse's properties and tested using a case study of SiFEUP[i]. It is used in the project as a complement to the SIARD format (Archives, 2008) used for the description and archive of relational databases. This enrichment leverages past efforts to define an archive format suitable for data tables from databases and adds a layer of metadata for the data warehouse components.

BACKGROUND

Digital preservation concerns sustainable and efficient strategies for the long-term preservation of digital objects (Ferreira, 2006). However, databases and data warehouses are different from conventional digital objects as they have an internal structure, and include schemas and integrity constrains which are vital for interpreting data.

Digital Preservation Projects

There are already many efforts and projects developed under the digital preservation scope. Projects such as CAMiLEON (Hedstrom & Lampe, 2001), InterPARES (Force, 2002), FEDORA (Lagoze, Payette, Shin & Wilper, 2006) or PLANETS (Hoeven, 2007; Zierau & Wijk, 2008; Sinclair, 2010) contributed to the study of requirements, strategies and proposals for preserving digital objects and ensure their authenticity.

Regarding complex digital objects, such as databases, projects like SIARD (Archives, 2008), Chronos (Brandl & Keller-Marxer, 2007) or RODA (Ramalho, Ferreira, Faria & Castro, 2007), analyzed in detail the preservation of relational databases. The PLANETS project built a framework that deals with Access, MS SQL Server and Oracle databases, as well as the SIARD format (PLANETS , 2009), a preservation format for relational databases.

The PresDB'07 workshop report states that *"existing preservation techniques for fixed digital objects are not suited for databases, thus some of our most critical digital assets are endangered - both economically and technically - in the long term"* (Christophides & Buneman, 2007).

This section introduces the concepts, requirements and strategies for digital preservation in the long term. *"Long term is long enough to be concerned with the impacts of changing*

technologies, including support for new media and data formats, or with a changing user community. Long term may extend indefinitely" (CCSDS, 2002).

Thibodeau's organization of digital preservation strategies relates them to their applicability and objective (Thibodeau, 2002). The figure below shows a simplified version of this bidimensional mapping, according to Ferreira' perspective (Ferreira, 2006), that is sufficiently clear and synthesized for our purposes. As Thibodeau's organization, this viewpoint arranges on the left the strategies focusing on preservation of the physical/logical object, and on the right side the strategies focused on preserving the conceptual object.

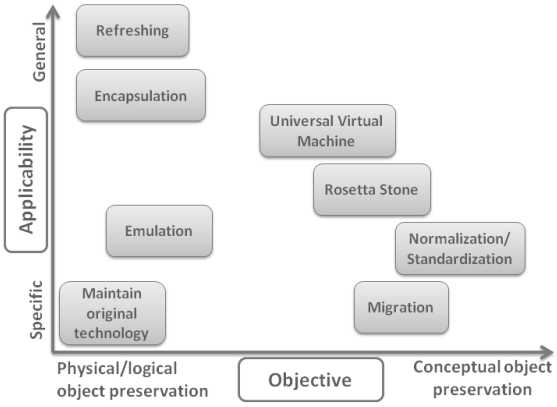

Figure 1 – Digital Preservation Methods (Ferreira, 2006)

Among all these strategies, the one that has been considered most feasible for database preservation is data migration to a standard XML format. XML stands for eXtensible Markup Language and is an open standard defined by the World Wide Web Consortium (W3C). It is a very flexible text format derived from SGML (ISO8879) (Sperberg-McQueen & Burnard, 1994), and it is widely used to structure, exchange and store data (Consortium, 2008). XML is platform and application independent, has a simple text format and is human readable, and is therefore an effective technology for the long-term preservation of relational databases.

The Open Archival Information System (OAIS) Reference Model (CCSDS, 2002), approved as an ISO standard in 2003, introduces, in the context of long-term preservation, the terminology for communication between the concerned parties in the preservation of digital objects, and defines the functional components necessary to implement a digital archive. An Open Archival Information System is *"an organization of people and systems that has accepted the responsibility to preserve information and make it available for a designated community"* (CCSDS, 2002). The term 'Open' emphasizes the fact that it has been developed in an open public forum, in which any interested party was encouraged to participate.

Data Warehouse Metadata

The research produced around digital preservation of databases does not account for the concepts of the dimensional model. Concepts like facts, dimensions, bridges, hierarchies, levels, data marts, star schemas or snowflake schemas are essential for the full description of a data warehouse.

Data warehouses are often implemented using relational database technology, and thus they are made up of tables that store data. A deeper inspection leads to the finding of facts, dimensions, bridges tables, indexes, level keys and views. However, there are some important differences between a database used in an operational system and in a data warehouse.

W. H. Inmon defined a data warehouse as *"a subject-oriented, integrated, nonvolatile, time variant collection of data in support of managements decisions"* (Inmon, 1992). Data warehouses fulfill two major purposes: provide a single, clean and consistent source of data for decision support and unlink the decision platform from the operational system (Date, 2004).

In a data warehouse the tables and joins are simple and de-normalized, in order to reduce the response time for analytical queries. For the characterization of a data warehouse, additional metadata is required that defines the dimensional model and allows data interpretation across different perspectives. The structure of a data warehouse is referred to as a dimensional schema, where dimensional tables, forming star schemas, surround the fact tables. A fact table is often located at the center of a star schema and consists of facts of a business process (e.g., measurements, transaction values).

To understand the facts it is necessary to introduce the context and meaning of the dimensional model, captured in the dimensions, representing the relevant vectors of analysis of the business process facts. The dimensions allow us to identify the how, what, who, when, where and why of the data. Dimensions are usually represented by one or more dimensional tables. A dimensional table contains attributes to define and group the data for data warehouse querying.

The dimensions are characterized by a set of levels with defined hierarchies. Hierarchies are logical structures that use levels to organize and aggregate data, define navigation paths or establish a family structure (Inmon, 1992; Kimball & Ross, 2002). A common example is a time dimension, where the hierarchy might aggregate data from the day level to the week, month, quarter or year levels.

Figure 2 shows an example of a star schema in a real-world case study used in the project, a *"Course Evaluation"* information system where statistics about user satisfaction (anonymous students) are collected in an academic environment, specifically on professor and class evaluation.

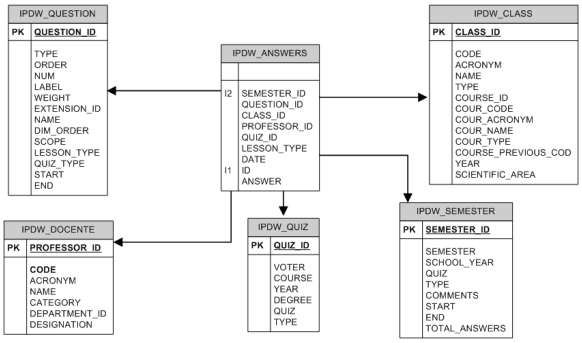

Figure 2 - Star schema example

In the center, a fact table contains the submitted answers (`IPDW_ANSWERS`). As dimensional tables, there are the question table (`IPDW_QUESTION`), the quiz table (`IPDW_QUIZ`), also the semester table (`IPDW_SEMESTER`), the class table (`IPDW_CLASS`) and the professor table (`IPDW_PROFESSOR`). Because the answers are anonymous, there is no relationship with the students who actually answered the questionnaires. An important step in the data warehouse

building process is to declare the dimensions. The following sample code shows the declaration of a dimension with the `CREATE DIMENSION` SQL statement (Oracle , 2011) using Oracle.

Example of a dimension declaration

```
CREATE DIMENSION class_dim
 LEVEL class IS (IPDW_CLASS.CLASS_ID)
 LEVEL course IS (IPDW_CLASS.COURSE_ID)
 HIERARCHY class_rollup(
  class CHILD OF
  course)
 ATTRIBUTE class DETERMINES
  (IPDW_CLASS.CODE, IPDW_CLASS.ACRONYM,
   IPDW_CLASS.NAME, IPDW_CLASS.TYPE)
 ATTRIBUTE course DETERMINES
  (IPDW_CLASS.COUR_CODE, IPDW_CLASS.COUR_ACRONYM,
   IPDW_CLASS.COUR_NAME, IPDW_CLASS.COUR_TYPE,
   IPDW_CLASS.COURSE_PREVIOUS_COD);
```

This declaration defines a dimension (`class_dim`) with a hierarchy (`class_rollup`) of two levels: the level course with `COURSE_ID` as the level key, and a child level class with `CLASS_ID` as the level key. This dimension uses the data from the table `IPDW_CLASS`. The `ATTRIBUTE` clause specifies the attributes that are uniquely determined by a hierarchy level. Thus it is possible to analyze the data in a more global perspective, through the course level, or get a more detailed view using the class level.

Another data warehouse concept is a bridge table. A bridge table is used to resolve a many to many relationship between a fact table and a dimension table and is also used to flatten out a hierarchy in a dimension table (Kimball & Ross, 2002).

Storing snowflake schemas and data marts is also needed. The snowflake schema is similar to the star schema, but dimensions are normalized into multiple related tables. A data mart is a subset of a data warehouse (Kimball & Ross, 2002; Hackney, 1997).

A DATA WAREHOUSE PRESERVATION FORMAT

The main goal of this proposal is to provide a preservation format that suits the characteristics of a generic data warehouse. This format should allow the definition of the relevant metadata from the data warehouse perspective and archive the metadata as well as the data from the tables in a format that would guarantee long-term preservation. The use of XML to satisfy these requirements appeared as an obvious option.

The study of the work already produced around the preservation of databases (Brandl & Keller-Marxer, 2007; SFA, 2008; Sinclair, 2010), including the model migration approach developed in the DBPreserve project (Rahman, David & Ribeiro, 2010), and on XML representation of a data warehouse (Pokorny, 2002; Hummer, Bauer & Harde, 2003), resulted in the decision to adopt and complement the SIARD format, an XML based format for the archival of relational databases, in order to adapt it to the characteristics of the dimensional model used in data warehouses.

The SIARD format proved to be the most appropriate starting point for this representation given the inherent modularity of data warehouses, with independent stars sharing some

dimensions. SIARD has a segmented structure of directories and files, unlike the DBML (Database Markup Language) used in RODA (Ramalho, Ferreira, Faria & Castro, 2007), which represents everything in a single file, making data handling harder.

Thus, reusing the archival format that stores the definition of the tables and their data, we propose to add a metadata layer for data interpretation according to the data warehouse perspective. Given the simplicity of the dimensional model in terms of relationships between tables, it becomes possible to analyze the archived data with greater efficiency through simplified queries applied directly to the XML files using XQuery[ii] and XPath[iii].

Relational Database Preservation with SIARD

The Swiss Federal Archives (SFA) have developed an open storage format for relational databases called SIARD (Software Independent Archiving of Relational Databases), as well as a set of conversion tools named the SIARD Suite (Thomas, 2009), in order to convert relational databases (e.g., Access, Oracle and SQL Server) into the archival SIARD format, edit the SIARD format and reactivate an archived database, restoring from the SIARD Format to a database.

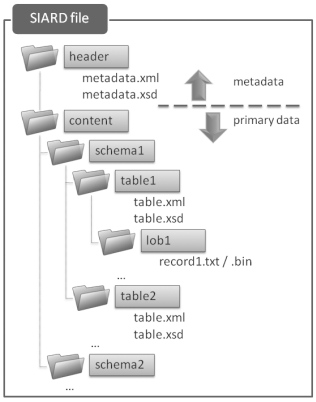

Figure 3 - Structure of the SIARD Archive File

The SIARD format is a nonproprietary and published open standard, based on open standard (e.g., ISO norms Unicode, XML, SQL1999) and the industry standard ZIP. In May 2008, the European PLANETS project accepted SIARD format as the official format for archiving relational databases (Archives, 2008).

The SIARD format is a ZIP64 (PKWARE, 2007) uncompressed package based on an organizational system of folders, storing the metadata in the header folder and table data in the content folder. This organization is shown in Figure 3.

For database's metadata characterization a single XML file is used that contains the entire structure of the database (schemas, tables, attributes, keys, views, functions...) and the corresponding XSD[iv] schema for XML validation.

As to the primary data, each schema is stored in different folders and sequentially numbered, as well as the tables of each schema. The data from each table is stored in an XML file with simplified structure (only rows and columns) and its XSD. If there are Large Objects - LOB (BLOB - Binary Large Objects and CLOB - Character Large Objects), these data are stored in binary files or text, within a folder for each attribute of these types, being referred to its path in the respective XML of the table.

One of the major benefits of this segmented archiving of the primary data is that it will reduce the size of each XML file, because the data will be distributed into the corresponding XML table files. This will increase the efficiency of parsing and querying of the XML data and can be extremely useful for parsing and querying of simultaneous XML table files, in order to solve a query involving more than one XML file (table).

Another reason for using the SIARD format is the existence of a tool that already allows us to create these packages from relational databases in Oracle, MSAccess and MSSQLServer.

The SIARD project produced a set of tools - SIARD Suite (Thomas, 2009) - comprised of three components: the *SiardEdit*, a graphical user interface for migration and metadata processing; the *SiardFromDb*, a command line application for extracting and storing a database generating the SIARD file; and the *SiardToDb*, a command line application to reactivate a database from a SIARD file.

Thus, the effort in this work will focus on the description of the dimensional model, complementing the existing one for the relational metadata format. The migration of the primary data into an XML format according to SIARD has also to be ensured.

However, the reuse and expansion of an existing open format like SIARD should not prevent the use of the applications that supports it, the SIARD Suite. Existing applications should be executed as if no changes to the format were made. Thus, using SIARD Suite it must be possible to manage the relational level metadata and the primary data.

DWXML definition

As XML languages exist for many domains and applications, existing XML representations for data warehouses were considered. There are some works in this area (Jensen, Muller & Pedersen, 2001; Hummer, Bauer & Harde, 2003). These works concern a multidimensional schema representation, i.e. data cubes. The XCube (Hummer, Bauer & Harde, 2003) is a data cube XML representation and it was developed to exchange data warehouse data over networks. The XCube was designed for interoperability purposes in MOLAP systems. The representation of the cube is divided into several XML documents to characterize each entity involved in the multidimensional system. This approach is interesting in the context in which it was developed, as it allows slicing the cube and sending small packets of information over the network, just as requested by the client. Even trying to adapt it for dimensional models, the diversity of documents produced would obscure the representation of the data warehouse. Moreover, it doesn't have any reference to tables (which store the facts and the dimensions in ROLAP systems), views, star or snowflake schemas.

To extend the SIARD format for archiving data warehouses in ROLAP systems, providing dimensional metadata to the SIARD format, a new XML file is added for characterizing the data warehouse and providing the concepts associated with the dimensional model, which are not covered by the base SIARD format. The XML schema (XSD) will also be added for validation of the XML file produced. This new XML representation was named Data Warehouse Extensible Markup Language (DWXML).

As a SIARD format extension for archiving data warehouses, the proposed DWXML bridges the gap between the relational model description and the dimensional model description, adding a metadata file (dw.xml) and its schema definition (dw.xsd[v]). Figure 4 shows an excerpt of the extended SIARD format, which includes the description of a data warehouse.

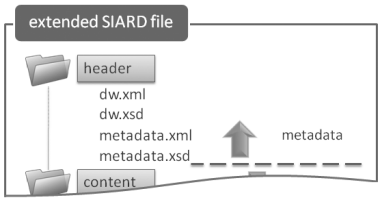

Figure 4 - DWXML added to the SIARD Archive File

The data warehouse is composed by a set of stars and a set of dimensions, implemented by tables and views organized in schemas. Data marts are also defined as a set of stars. Figure 5 characterizes the DWXML basic structure and the star element.

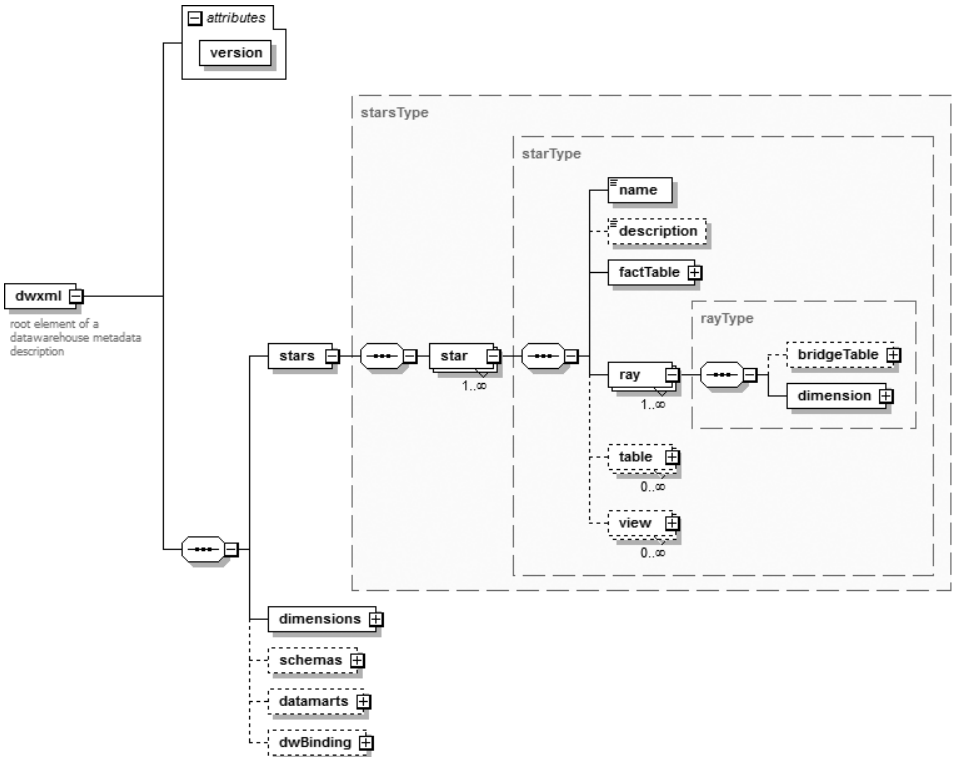

Figure 5 - DWXML schema showing the star element

The *version* attribute represents the version of the DWXML definition. The element *dwBinding* supports the description of the DWXML file, the information related to the owner of the data, the credentials of the connection to the data warehouse and the names and versions of the applications involved in the DWXML creation, including the DBMS where the data warehouse was working and the migration date.

Table 1 describes the elements of the data warehouse metadata. The column *Opt*. indicates whether the identifier is optional. Table 2 describes the elements of the data warehouse binding metadata.

Table 1 - Data warehouse metadata description

IDENTIFIER	OPT.	DESCRIPTION
version	no	DWXML format version
stars	no	List of stars in the data warehouse
dimensions	no	List of dimensions, dimensional tables and views in the data warehouse
schemas	yes	List of schemas in the data warehouse
datamart	yes	List of datamarts in the data warehouse
dwBinding	yes	Additional metadata for data warehouse connection description and DWXML file generation

Table 2 - Data warehouse binding metadata description

IDENTIFIER	OPT.	DESCRIPTION
description	yes	Description of the data warehouse's meaning and content
dataOwner	no	The owner of the data, who has the right to grant the access to the data
xmlApplication	yes	Name and version of program that produced the DWXML from the data warehouse
migrationDate	yes	Date when the DWXML was produced from the data warehouse
dwProduct	no	Product name and version of the DBMS containing the data warehouse
dwUser	no	The user of the data warehouse who carried out the XML migration
dwConnection	no	The connection string to the data warehouse that contains the dimensional model

Stars and Facts

A star is composed by a fact table and a set of "rays" which establish relationships to dimensions and, possibly, bridge tables. The *factTable* element references the respective table description in the schemas element, indicates the columns responsible for the joins between fact tables and dimensions or bridge tables and contains information about its granularity and about the facts. Regarding the facts, these elements indicate the table column that represents each of them, as well as their measure type: non-additive, semi-additive or additive. Table 3 describes the elements of the star metadata.

Table 3 - Star metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the star
description	yes	Description of the star meaning and content
factTable	no	Fact table implementing the star
ray	no	Ray of the star connecting the fact table with a dimension or a bridge table (referenced by schema and name).

table	yes	List of extra tables to accommodate special cases (referenced by schema and table name)
view	yes	List of views which may represent a virtual fact table (referenced by schema and view name)

Figure 6 shows the schema of a fact table element, including its facts and possible join column definitions. These are used when a bridge table sits between a fact table and a dimension, with a many-to-many relationship between the fact and the bridge table.

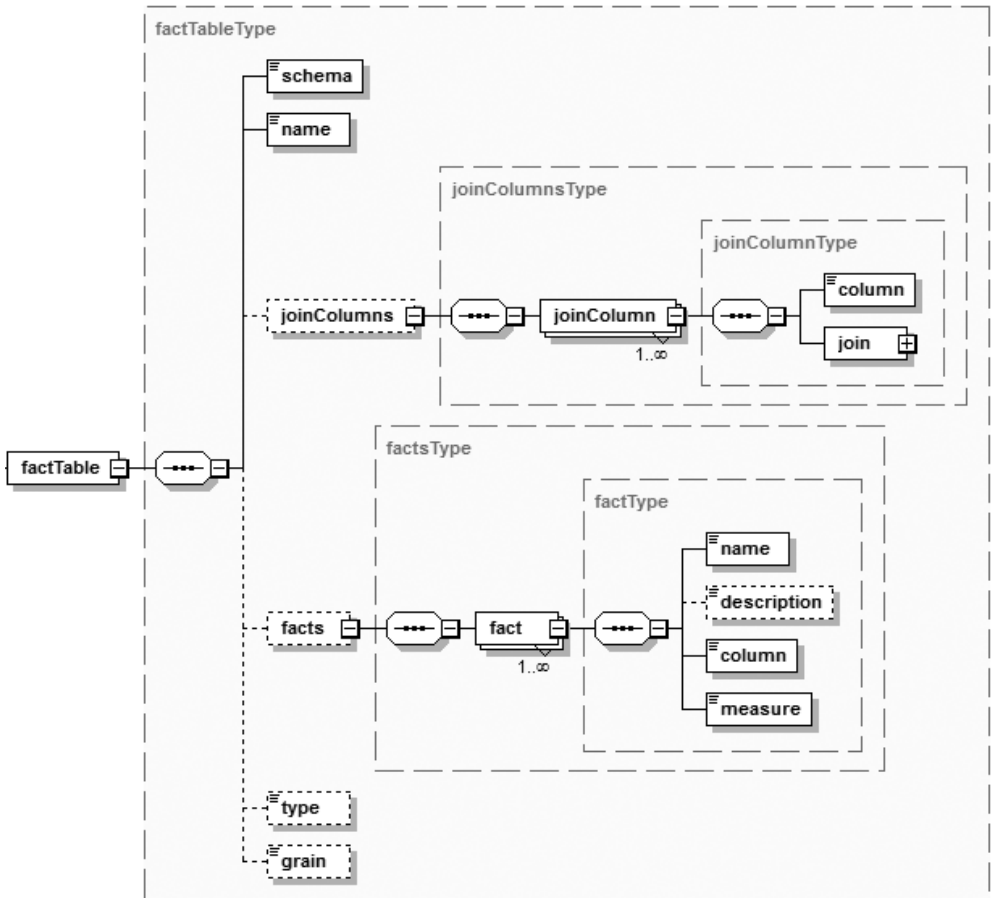

Figure 6 - DWXML schema showing a fact table element

Table 4 describes the elements of the fact table metadata.

Table 4 - Fact table metadata description

IDENTIFIER	OPT.	DESCRIPTION
schema	no	Schema of the fact table
name	no	Name of the fact table
joinColumns	yes	List of columns used in the joins between the fact table and possible bridge tables

facts	yes	List of facts represented in the fact table
type	yes	The type of the fact table (TRANSACTIONS, CUMULATIVE or SNAPSHOT)
grain	yes	The grain of the fact, the meaning and content of a row in the fact table

Joins with dimensions are dealt by the ray element. The joinColumns element is useful to indicate which column of the fact table is responsible for the relationship with a column of a bridge table. Table 5 describes the elements of the join column metadata.

Table 6 describes the elements of the fact metadata.

Table 5 - Join column metadata description

IDENTIFIER	OPT.	DESCRIPTION
column	no	Fact table column used in the join
join	no	Column of the bridge table involved in the join (schema, table and column)

Table 6 - Fact metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the fact
description	yes	Fact meaning and content
column	no	Column of the fact table where the fact is stored
measure	no	Measure type (constrained to ADDITIVE, NON ADDITIVE or SEMI ADDITIVE)

In a star, each *ray* element represents a relationship between the fact table and a dimension. In special situations, for instance when there is a many to many relationship between the fact table and the dimension, a bridge table may be added. In this case, the ray element is composed by a *bridgeTable* element that references the related table, followed by a reference to the dimension element. Table 7 describes the elements of the ray metadata.

Table 7 - Ray metadata description

IDENTIFIER	OPT.	DESCRIPTION
bridgeTable	yes	Reference to the bridge table (schema and name)
dimension	no	Reference to the dimension (schema and name)

The following example shows a star definition using DWXML. The `IPDW_ANSWERS_STAR` is composed by the `IPDW_ANSWERS` fact table, which holds the data of the additive fact `ANSWER_F` represented on the fact table's column `ANSWER`, and by two ray elements. One of them establishes a connection to the dimension `QUESTION_DIM`.

Example of a DWXML star definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<dwxml version="1.0" xsi:noNamespaceSchemaLocation="dw.xsd"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <stars>
      <star>
        <name>IPDW_ANSWERS_STAR</name>
        <description>Star related to the answers</description>
        <factTable>
          <schema>CALDEIAS</schema>
          <name>IPDW_ANSWERS</name>
          <facts>
            <fact>
               <name>ANSWER_F</name>
               <column>ANSWER</column>
               <measure>ADDITIVE</measure>
             </fact>
          </facts>
        </factTable>
        <ray>
          <dimension>
            <schema>CALDEIAS</schema>
            <name>QUESTION_DIM</name>
          </dimension>
        </ray>
        <ray>
          ...
        </ray>
      </star>
    </stars>
    ...
  </dwxml>
```

As snowflake schemas can be seen as extensions of star schemas, their representation starts as a star schema, but the dimensions of a snowflake schema are implemented by tables (dimension tables) that are partially normalized, resulting in relationships with other tables (sub-dimension). So, inspecting the dimension table's foreign keys, it is possible to differentiate between a snowflake schema and a star schema. If a foreign key of a dimension table refers a sub-dimension, the schema is a snowflake schema.

Datamarts are subsets of data warehouses, i.e. sets of star or snowflake schemas. Table 8 describes the elements of the datamart metadata.

Table 8 - Datamart metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the datamart
description	yes	Description of the datamart meaning and content
stars	no	List of the names of the stars that defines the datamart

Dimensions

Dimensions may be shared by different stars. So, the metadata related to the dimensions is stored in a list of dimension elements that are referenced by the stars. Dimensions explain the meaning of the measures stored in the fact table and support the data analysis. Figure 7 displays the dimensions element schema. The dimension element has been defined following the syntax of the `CREATE DIMENSION` Oracle SQL statement (Oracle, 2010).

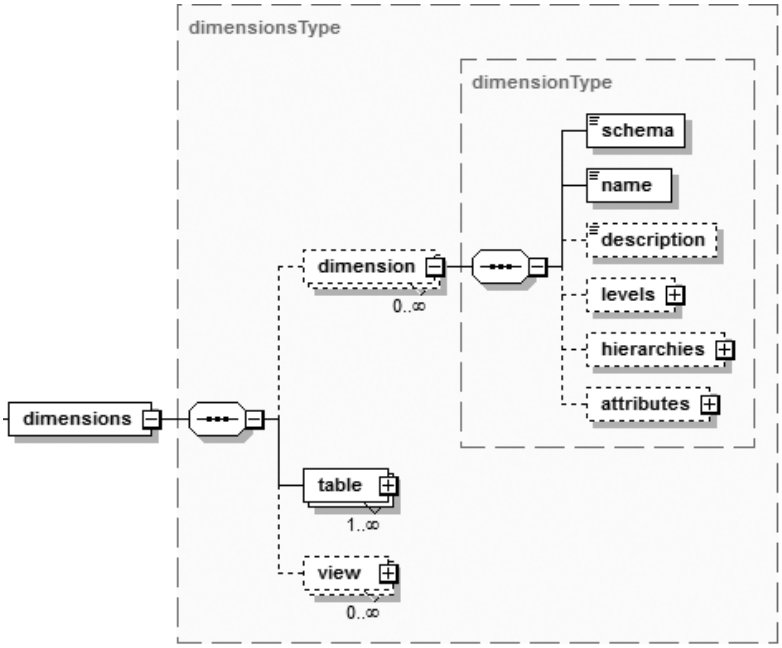

Figure 7 - Dimensions element schema

Each dimension element represents a relevant entity in the problem domain and is characterized by a set of attributes. Attributes may be grouped in levels, which are organized in hierarchies. Dimensions and levels have keys. The tables and views elements contain the references to the tables and views (schema and name) that support the declared dimensions; their structure is described in the schemas element.

Table 9 describes the elements of the dimension metadata.

Table 9 - Dimension metadata description

IDENTIFIER	OPT.	DESCRIPTION
schema	no	Schema of the dimension
name	no	Name of the dimension
description	yes	Description of the dimension meaning and content
levels	yes	List of levels in the dimension
hierarchies	yes	List of hierarchies in the dimension
attributes	yes	List of attributes in the dimension

Figure 8 displays the level element schema in a dimension. Levels have a level key that identifies each level. The level key of the lowest level corresponds to the dimension key. This key represents the dimension in the data warehouse.

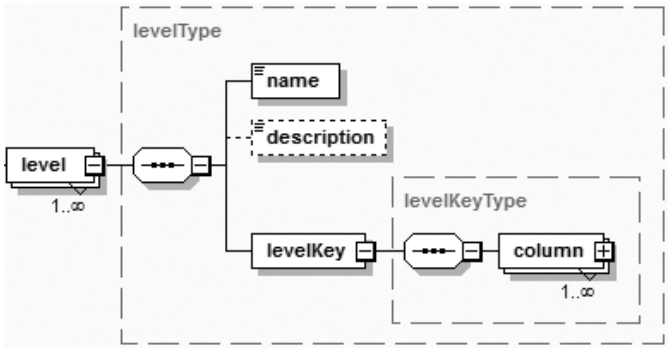

Figure 8 - Level element schema

Table 10 describes the elements of the level metadata.

Table 10 - Level metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the level
description	yes	Description of the level meaning and content
levelKey	no	Key of the level (one or more columns in the dimension table)

Figure 9 displays the hierarchy element schema in a dimension.

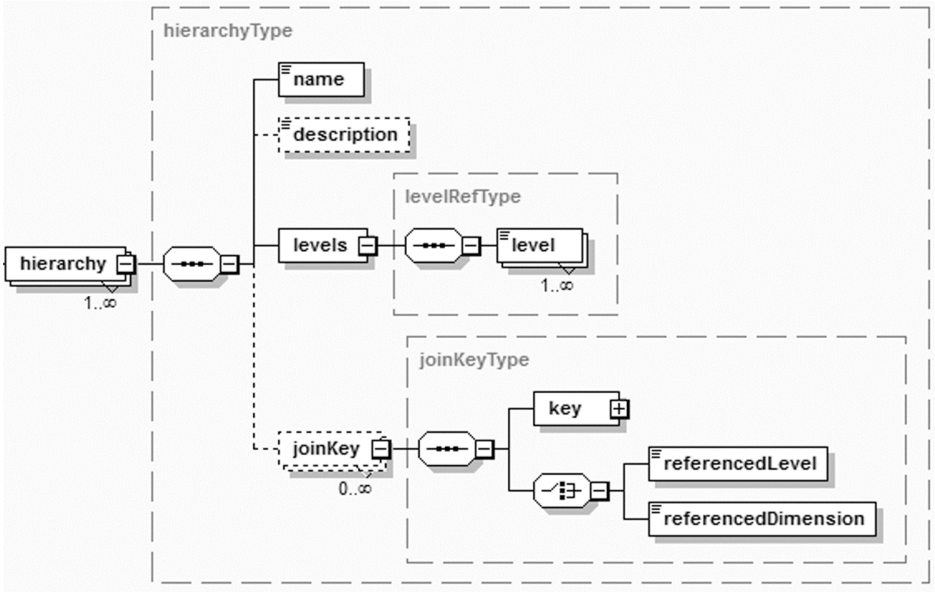

Figure 9 - Hierarchy element schema

Table 11 describes the elements of the hierarchy metadata. Table 12 describes the elements of the attribute metadata. Attributes represent the characteristics of a level, which is identified by its level key.

Table 11 - Hierarchy metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the hierarchy
description	yes	Description of the hierarchy meaning and content
levels	no	List of the levels of the hierarchy, starting with the more general level
joinKey	yes	The key that joins the levels of the hierarchy when they are implemented using different dimension tables.

When a dimension is implemented by just one table, each record contains attributes for all the levels. If some levels are detached in a subdimension (snowflake or partially normalized schema) an extra join attribute must be included in the main dimension to reference the subdimension.

Table 12 - Attribute metadata description

IDENTIFIER	OPT.	DESCRIPTION
attributeName	yes	Name of the attribute
level	no	Identifies the level and the attributes determined by its level key. There must be at least one level element

Figure 10 displays the attribute element schema in a dimension.

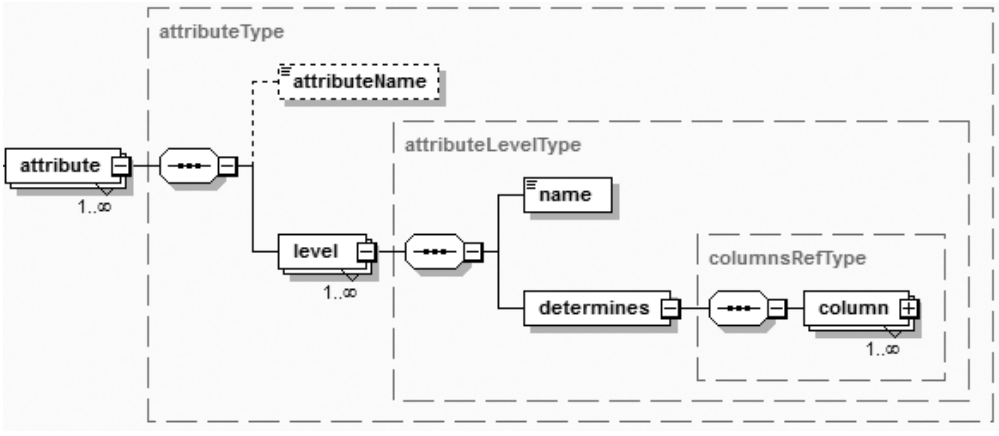

Figure 10 - Attribute element schema

Table 13 describes the identifiers of the level referenced by an attribute.

Table 13 - Attribute levels metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the level
determines	no	Attribute determined by the level

The next example shows a dimension definition using DWXML. This dimension (CLASS_DIM) is composed by the levels COURSE and CLASS, identified by their level keys COURSE_ID and CLASS_ID, respectively. Both these levels are implemented by the IPDW_CLASS relational table. The dimension has a defined hierarchy of levels named (CLASS_ROLLUP) stating that CLASS level is child of COURSE level, according to the order of appearance (the first is parent of the second and so on). The attribute element states which attributes are defined by each level. So, the attributes COUR_PREVIOUS_COD, COUR_TYPE, COUR_NAME, COUR_ACRONYM belong to the COURSE level and the attributes TYPE, NAME, ACRONYM belong to the CLASS level.

Example of a DWXML dimension definition:

```
<?xml version="1.0" encoding="UTF-8"?>
<dwxml version="1.0" xsi:noNamespaceSchemaLocation="dw.xsd"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  ...
  <dimensions>
    <dimension>
       <schema>CALDEIAS</schema>
      <name>CLASS_DIM</name>
      <levels>
        <level>
          <name>CLASS</name>
          <description />
          <levelKey>
                <column>
                    <schema>CALDEIAS</schema>
                    <table>IPDW_CLASS</table>
                    <column>CLASS_ID</column>
                </column>
             </levelKey>
             </level>
              <level>
          <name>COURSE</name>
          <levelKey>
                 <column>
                   <schema>CALDEIAS</schema>
                   <table>IPDW_CLASS</table>
                  <column>COURSE_ID</column>
                 </column>
               </levelKey>
           </level>
      </levels>
      <hierarchies>
            <hierarchy>
               <name>CLASS_ROLLUP</name>
```

```
            <levels>
              <level>COURSE</level>
              <level>CLASS</level>
            </levels>
             </hierarchy>
        </hierarchies>
        <attributes>
               <attribute>
            <attributeName>COURSE</attributeName>
            <level>
              <name>COURSE</name>
              <determines>
                 <column>
                    <schema>CALDEIAS</schema>
                    <table>IPDW_CLASS</table>
                    <column>COUR_PREVIOUS_COD</column>
                 </column>
                 <column>
                    <schema>CALDEIAS</schema>
                    <table>IPDW_CLASS</table>
                    <column>COUR_TYPE</column>
                 </column>
            ...
              </determines>
            </level>
             </attribute>
          <attribute>
          ...
               </attribute>
        </attributes>
      </dimension>
      ...
  </dimensions>
  ...
</dwxml>
```

Tables and Views

The schemas, tables and views follow a simplified representation with respect to the SIARD format and some elements are replicated in this description to make the data warehouse metadata self-contained. However, this DWXML version does not contemplate the representation of the primary data in XML, since it is used in conjunction with the SIARD format, which already performs the primary data migration to XML format.

A schema contains a group of tables and a group of views. Figure 11 displays the schema element. Table 14 describes the elements of the schema metadata.

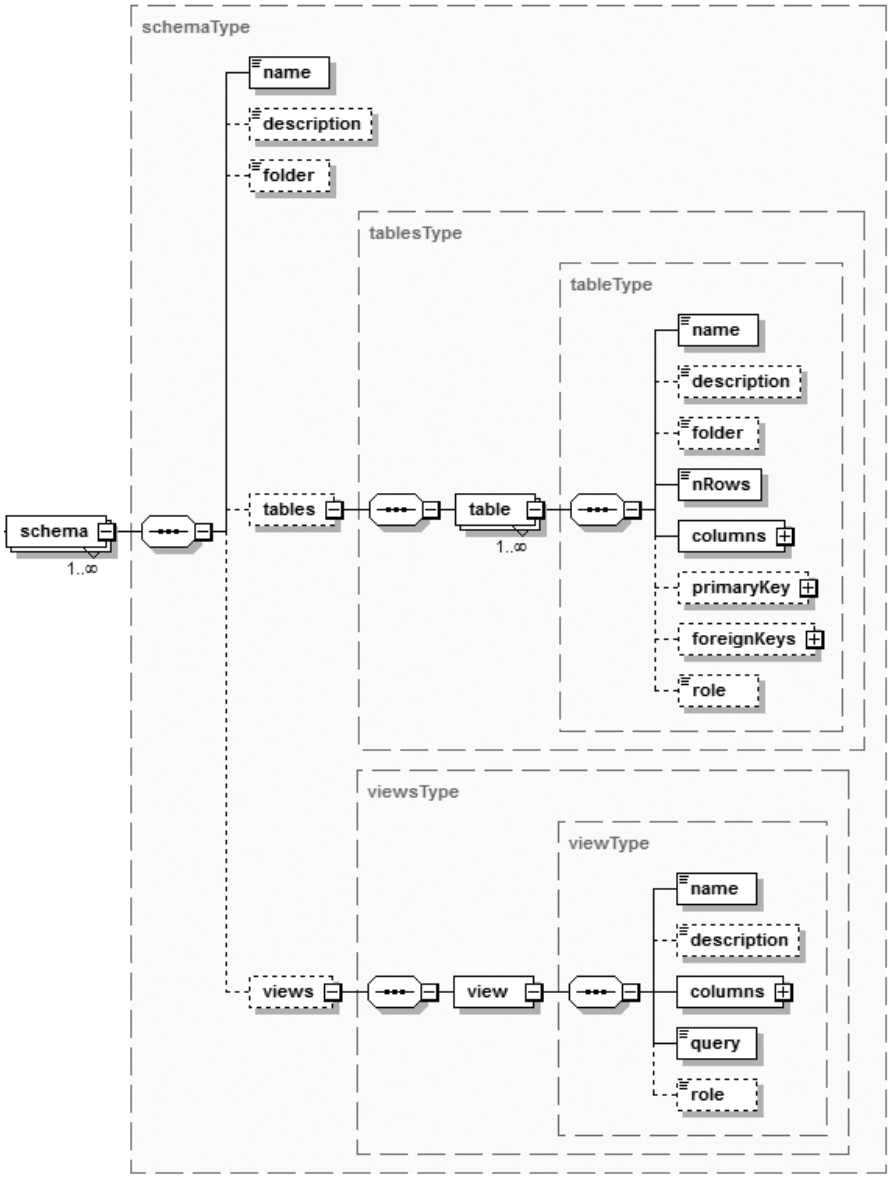

Figure 11 - The schema element

Table 14 - Schema metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the schema
description	yes	Description of the schema meaning and content
folder	yes	The name of the folder in the SIARD format
tables	yes	List of tables of the schema and their definition
views	yes	List of views of the schema and their definition

Table 15 to 19 describe the elements of the table, column, primary key, foreign key and view metadata.

Table 15 - Table metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the table
description	yes	Description of the table meaning and content
folder	yes	The name of the data folder in the SIARD format
nRows	no	Number of rows of the table
columns	no	List of columns of the table and their definition
primaryKey	yes	The primary key of the table
foreignKeys	yes	List of foreign keys of the table and their definition
role	yes	The role of the table in the dimensional model (FACT TABLE, DIMENSION TABLE, BRIDGE TABLE, DEGENERATED DIMENSION TABLE)

Table 16 - Column metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the column
description	yes	Description of the column meaning and content
folder	yes	The name of the folder in the SIARD format for LOBs storage
type	yes	Data type of the column in the data warehouse
defaultValue	yes	Default value of the column
nullable	no	Indicates if the column value can be null

Table 17 - Primary key metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the primary key
description	yes	Description of the primary key's meaning and content
column	no	Column that belongs to the primary key. There must be at least one column element

Table 18 - Foreign key metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the foreign key
description	yes	Description of the foreign key's meaning and content
referencedSchema	no	Name of the schema of the referenced table
referencedTable	no	Name of the referenced table
reference	no	Name of the referencing column and referenced column (must be non-empty)

Table 19 - View metadata description

IDENTIFIER	OPT.	DESCRIPTION
name	no	Name of the view
description	yes	Description of the view's meaning and content
columns	no	List of the columns of the view (schema, table and column names)
query	no	Query that represent the view
role	no	The role of the view in the dimensional model

Application Architecture

The DBPreserve Suite, the application that supports the data warehouse migration process to the proposed preservation format, had the following general requirements:

- Migrate the data warehouse model implemented using relational database technologies to the SIARD format;
- Acquire the metadata to describe the dimensional model of the data warehouse;
- Help in the process of building a DWXML representation, upon the metadata collected;
- Enable metadata editing supported by graphical interfaces;
- Generate the DWXML from the metadata collected/edited and embed it into the generated SIARD format;
- Enable primary data browsing using the proposed preservation format.

The DBPreserve Suite application is a Windows desktop application that has a modular and extensible architecture, composed by 5 major new modules as shown in the overall architecture of the application in Figure 12.

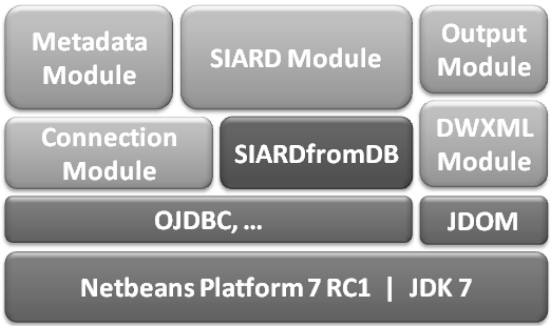

Figure 12 - DBPreserve Suite general architecture

It has been developed using the NetBeans IDE 7.0 RC1 and Netbeans Platform[vi], with support for Java 1.7[vii], using the JDOM[viii] library (Hunter, 2002) for XML processing of metadata. This application integrates a tool from the SIARD Suite that manages the migration of a relational database to the SIARD format, the *SIARDfromDB* application.

The Connection Module enables the abstraction of the data warehouse connection, using Java Database Connectivity (JDBC). The application already supports connections to Oracle database using Oracle JDBC (OJDBC), due to the DBMS used in the proposed case study. However, this

module is prepared for an easy extension to connect with other DBMS, adding just a file that rewrites some metadata retrieval methods.

The Metadata Module handles all the metadata imported from the DBMS and the metadata import process itself. The imported metadata is related with schemas, tables, views, columns, primary keys, foreign keys, dimensions, levels, level keys, hierarchies, attributes, table comments and column comments. Through the analysis of the acquired metadata, this module proposes a possible DWXML that describes the dimensional model.

The SIARD Module allows the integration of the *SIARDfromDB* tool from SIARD Suite (Thomas, 2009) that creates the SIARD format of the relational data in the data warehouse. This format still lacks the dimensional model description. It looks at the data warehouse from a relational model point of view. This module also manages the generated SIARD format, accessing the relational metadata, enabling the primary data browsing and embedding the DWXML with the dimensional model description.

The DWXML Module handles the dimensional metadata, creating the DWXML file to embed it into the SIARD format or reading it from the SIARD format if already recorded.

The Output Module manages all the graphical interfaces, such as connection management, SIARD format generation through *SIARDfromDB* integration, table and view roles visual editing, graphical representation of star or snowflake schemas and dimensions, hierarchical viewing of schemas, star and dimension, editing of the DWXML through several graphical user interfaces, viewing of the DWXML file added to the SIARD format and browsing of the primary data when selecting a star schema or dimension.

This work has been applied to a real world case study from DBPreserve project, to test the migration process, in order to validate all the features implemented in the DBPreserve Suite application.

FUTURE RESEARCH DIRECTIONS

The DBPreserve Suite application can be extended with new features, such as the implementation of the reverse migration process, i.e. starting from the XML preservation format, reactivate the data warehouse through its reconstruction in a DBMS and then loading it with the primary data. Notice that, as DWXML is meant for preservation and not as a backup utility, instrumental database objects useful in a running database like sequences or materialized views have not been included, and so they must be recreated to revive the DW. Another new feature could be the implementation of a module to generate the initial SIARD format, untying the DBPreserve Suite application from the SIARD Suite tool and making it completely autonomous. One of the most important improvements is to provide the application with methods to query the primary data XML files. To this end, it is necessary to analyze the efficiency of techniques for large XML document processing, and to choose a convenient user query language.

CONCLUSION

The proposed format for data warehouse preservation, combining the DWXML (that describes the dimensional model) with the SIARD format (for relational model description and primary data storage), proved to be a useful way to represent a data warehouse in XML-based files. In fact, the DBPreserve Suite application uses this extended SIARD format to represent star

and snowflake schemas, as well as dimension structures (hierarchies, levels and attributes), and it enables the browsing of primary data from the dimensional model perspective (through stars and dimensions). Thus, extending the SIARD format with a DWXML dimensional model metadata layer, a long-term preservation format for data warehouses is achieved.

Looking at the major implemented features of DBPreserve Suite application, the integration of the *SIARDfromDB* command line application from the SIARD Suite enables the standard SIARD format generation, migrating all relational metadata and primary data according to that format, with a total control of this process from the developed application. The reuse of this model has allowed us to concentrate on the description of the dimensional model, by importing the metadata from the data dictionary, automating the creation of DWXML after analysis of the imported metadata, providing user interfaces for a friendly DWXML editing, embedding it into the SIARD format and enabling the access to primary data through the dimensional model perspective. All the implemented features were tested and refined using the project case study.

The proposed preservation format is not in itself a guarantee of success regarding long-term digital preservation. As a preservation format for data warehouses implemented with relational databases technologies, it fulfills part of the requirements for an OAIS. The fact that it is platform independent and captures the dimensional model metadata is a contribution to that goal.

REFERENCES

Brandl, S., Keller-Marxer, P. (2007, March). Long-term Archiving of Relational Databases with Chronos. *In First International Workshop on Database Preservation - PresDB'07, 23 March 2007*.

CCSDS (2002). Reference Model for an Open Archival Information System (OAIS) - Blue *Book. Consultative Committee for Space Data Systems*. Washington: National Aeronautics and Space Administration.

Date, C. J. (2004). An Introduction to Database Systems (Eight Edition). Pearson, Addison Wesley.

Christophides, V. & Buneman, P. (2007, September). Report on the First International Workshop on Database Preservation, PresDB'07. *SIGMOD Record, Vol. 36, Nr. 3 (pp. 55–58)*.

Consortium, W. (2008, November). Extensible Markup Language (XML) 1.0 (fifth edition). *W3C Recommendation*.

Ferreira, M. (2006). Introdução à Preservação Digital - Conceitos, estratégias e actuais consensos. *Escola de Engenharia da Universidade do Minho*.

Force, A. T. (2002). Requirements for Assessing and Maintaining the Authenticity of Electronic Records (Tech. Rep.). *InterPARES Project*. Vancouver, Canada.

Hackney, D. (1997). Understanding and Implementing Successful Data Marts. *Addison-Wesley Longman Publishing Co., Inc.* Boston, MA, USA.

Hedstrom, M., & Lampe, C. (2001). Emulation vs. Migration: Do users care? *RLG DigiNews, 5 Num 6*.

Hoeven, J. (2007). Emulation for Digital Preservation in Practice: The Results. *The International Journal of Digital Curation, Issue 2, Volume 2:123132*.

Hummer, W., & Bauer, A., & Harde, G. (2003). XCube: XML for Data Warehouses. *In Proceedings of the 6th ACM International Workshop on Data Warehousing and OLAP (DOLAP '03)*. ACM, New York, USA (pp.33-40). DOI: 10.1145/956060.956067

Hunter, J. (2002). JDOM in the Real World - JDOM makes XML Manipulation in Java Easier than Ever. *Oracle Magazine, September/October 2002*.

Inmon, W. H. (1992). Building the Data Warehouse. *John John Wiley & Sons, Inc.*, New York, USA.

Jensen, M. R., & Muller, T. H., & Pedersen, T. B. (2001). Specifying OLAP Cubes on XML Data. *In Proceedings of the 13th International Conference on Scientific and Statistical Database Management, SSDBM'01*, (pp.101), Washington, DC, USA,. IEEE Computer Society.

Kimball, R., & Ross, M. (2002). The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling (2nd ed.). *John Wiley & Sons, Inc.*, New York, USA.

Lagoze, C., & Payette, S., & Shin, E., & Wilper, C. (2006). Fedora: An Architecture for Complex Objects and their Relationships. *International Journal on Digital Libraries, Vol. 6 Num. 2:124138*.

Oracle (2010). Oracle Database SQL Reference 11g Release 1 (11.1), Part Number B28286-06. Retrieved April 30, 2011, from http://docs.oracle.com/cd/B28359_01/server.111/b28286.pdf

PKWARE (2007, September). ZIP File Format Specification, Version: 6.3.2, Revised: September 28. PKWARE Inc. Retrieved April 30, 2011, from http://www.pkware.com/documents/casestudies/APPNOTE.TXT

PLANETS (2009). PLANETS: Tools and Services for Digital Preservation. *PLANETS Product Sheet.*

Pokorny, J. (2002). XML Data Warehouse: Modelling and Querying. *In Proceedings of the Baltic Conference, BalticDB&IS 2002 - Vol.1, Hele-Mai Haav and Ahto Kalja (Eds.), Vol.1.* Inst. of Cybernetics at Tallin Technical University (pp. 267-280).

Rahman, A. U., & David, G., & Ribeiro, C. (2010, June). Model Migration Approach for Database Preservation. *In The Role of Digital Libraries in a Time of Global Change, 12th International Conference on Asia-Pacific Digital Libraries, ICADL 2010*, Gold Coast, Australia, (pp. 81-90). Springer Berlin/Heidelberg.

Ramalho, J. C., & Ferreira. M., & Faria, L., & Castro, R. (2007). Relational Database Preservation through XML Modelling. *In Extreme Markup Languages 2007*.

SFA (2008). SIARD Format Description (Tech. Rep.). *Federal Department of Home Affairs. Unit Innovation and Preservation*. Berne.

Sinclair, P. (2010, March). The Digital Divide: Assessing Organizations' Preparations for Digital Preservation. *PLANETS White Paper*.

Sperberg-McQueen, C. M., & Burnard, L. (1994). A Gentle Introduction to SGML. *Guidelines for Electronic Text Encoding and Interchange. Text Encoding Initiative*. Chicago, Oxford.

Thibodeau, K. (2002). Overview of Technological Approaches to Digital Preservation and Challenges in Coming Years. *In The State of Digital Preservation: An International Perspective. Documentation Abstracts, Inc.* - Institutes for Information Science.

Thomas, H. (2009). SIARD Suite Manual. *Federal Department of Home Affairs, Swiss Federal Archives SFA Unit Innovation and Preservation*. Berne.

Zierau, E., & Wijk, C. (2008). The PLANETS Approach to Migration Tools. *In IS&T Archiving 2008*. Society for Imaging Science and Tech. Bern, Switzerland.

ADDITIONAL READING SECTION

Barbedo, F., & Corujo, L., & Faria, L., & Castro, R., Ferreira, M., & Ramalho, J. C. (2007). *RODA: Repositório de Objectos Digitais Autênticos.* Paper presented at the 9º Congresso Nacional de Bibliotecários, Arquivistas e Documentalistas, Ponta Delgada, Portugal.

Christophides, V., & Buneman, P. (2007, September). Report on the First International Workshop on Database Preservation, PresDB'07. *SIGMOD Record, Vol. 36, No.* 3 (pp. 55–58).

Committee, P. E. (2011, January). Data Dictionary for Preservation Metadata: PREMIS version 2.1, *PREMIS Editorial Committee*.

Day, M. (1998). Issues and Approaches to Preservation Metadata. *In Conf. Guidelines for Digital Imaging*, University of Warwick, Coventry, United Kingdom.

Dappert, A., & Farquhar, A. (2009, October). Implementing Metadata that Guides Digital Preservation Services. *In iPress2009*, San Francisco, California (pp. 5–6).

Farquhar, A., & Hockx-Yu, H. (2007). PLANETS: Integrated Services for Digital Preservation. *International Journal of Digital Curation*, Issue 2, Volume 2 (pp. 88–99).

Freitas, R. A. P., & Ramalho, J. C. (2011, June). Using Ontologies in Database Preservation. *In XATA 2011 - XML: Aplicações e Tecnologias Associadas*, Vila do Conde, Portugal.

Freitas, R. A. P., & Ramalho, J. C. (2011, November). New Dimension in Relational Database Preservation: rising the abstraction level". *iPRES 2011 - 8th International Conference on Preservation of Digital Objects*, Singapore.

Garrett, J., & Waters, D. (1996). Preserving Digital Information, Report of the Task Force on Archiving of Digital Information. Technical report. *The Commission on Preservation and Access and The Research Libraries Group*, Washington DC and Mountain View CA.

Hedstrom, M. (1997). Digital Preservation: A Time Bomb for Digital Libraries. *Computers and the Humanities*, 31 (pp. 189–202).

Lee, K., & Slattery, O., & Lu, R., & Tang, X., & Mc-Crary, V. (2002). The State of the Art and Practice in Digital Preservation. *Journal of Research of the National Institute of Standards and Technology, Volume 107*, Number 1 (pp. 93–106).

Lorie, R. A. (2001, June). Long-Term Archiving of Digital Information. *In Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital libraries*, Roanoke, VA, USA.

Lorie, R. A., & Diessen, R. J. van (2005). UVC: Long-Term Preservation of Complex Processes. *IS&T Archiving Conference*, Washington, DC (pp. 26–29).

Mellor, P., & Wheatley, P., & Sergeant, D. (2002, September). Migration on Request, a Practical Technique for Preservation. *In Research and Advanced Technology for Digital Libraries: 6th European. Lecture Notes in Computer Science* (pp. 516–526). Springer, Berlin/Heidelberg.

Testbed, D. P. (2003). From Digital Volatility to Digital Permanence: Preserving Databases. *Technical report, Dutch National Archives and the Dutch Ministry of the Interior and Kingdom Relations*.

Verdegem, R. (2003, April). Databases Preservation Issues. *In Erpanet workshop on Long-term Preservation of Databases*. Digital Preservation Testbed. Bern, Switzerland.

Waugh, A., & Wilkinson, R., & Hills, B., & Dell'oro, J. (2000). Preserving Digital Information Forever. *In International Conference on Digital Libraries Proceedings of the fifth ACM conference on Digital Libraries* (pp. 175–184).

KEY TERMS AND DEFINITIONS

Data warehouse – Data warehouse s are complex digital objects, based on a dimensional model, where star and snowflake schemas, facts, dimensions with levels and hierarchies, bridges tables and datamarts can be identified. Data warehouses are often implemented on relational databases (ROLAP), keeping the data in tables, views and schemas.

Digital Preservation – is a process or a set of processes that must follow a concrete plan of activities, with allocation of adequate resources and use of technologies and practices that ensure access to a digital object, in the long-term perspective.

Dimensional model metadata – includes the concepts of the dimensional model, like facts, dimensions, bridges, hierarchies, levels, data marts, star schemas or snowflake schemas, which are essential for the full description of a data warehouse.

DWXML – stands for Data Warehouse Extensible Markup Language, and it is an XML format used for characterizing the data warehouse and providing the concepts associated with the dimensional model, which are not covered by the base SIARD format. As a SIARD format extension for archiving data warehouses, the DWXML bridges the gap between the relational model description and the dimensional model description, adding a metadata file related to the dimensional model characterization and its schema definition.

OAIS – An Open Archival Information System is an organization of people and systems that has accepted the responsibility to preserve information and make it available for a designated community. The OAIS reference model introduces the appropriate terminology in the context of long-term preservation, as well as defining the functional components necessary to an archive implementation

SIARD Format – The SIARD format is a nonproprietary and published open standard, based on other open standards (e.g., ISO standard Unicode, XML, SQL1999), and the industry standard

ZIP. It was developed by the Swiss Federal Archives. In May 2008, the European PLANETS project accepted SIARD format as the official format for archiving relational databases.

XML – stands for Extensible Markup Language and it is an open standard defined by the World Wide Web Consortium (W3C). This standard is a very flexible text format derived from SGML (ISO8879), and it is widely used to structure, exchange and store data.

---

[i] Information System of Faculty of Engineering, University of Porto, Portugal

[ii] http://www.w3.org/TR/xquery

[iii] http://www.w3.org/TR/xpath

[iv] http://www.w3.org/XML/Schema

[v] https://www.fe.up.pt/si/wikis_paginas_geral.paginas_view?pct_pagina=43194

[vi] http://netbeans.org/features/platform/

[vii] http://download.java.net/jdk7/docs/api/

[viii] http://www.jdom.org/index.html