Handling Multiple Points of View in a Multimedia Data Warehouse

ANNE-MURIEL ARIGON
Laboratoire de Biométrie et Biologie Evolutive, LBBE UMR CNRS 5558, UCBL
and
ANNE TCHOUNIKINE and MARYVONNE MIQUEL
Laboratoire d'InfoRmatique en Images et Systèmes d'information, LIRIS UMR CNRS 5205 INSA

Data warehouses are dedicated to collecting heterogeneous and distributed data in order to perform decision analysis. Based on multidimensional model, OLAP commercial environments such as they are currently designed in traditional applications are used to provide means for the analysis of facts that are depicted by numeric data (e.g., sales depicted by amount or quantity sold). However, in numerous fields, like in medical or bioinformatics, multimedia data are used as valuable information in the decisional process. One of the problems when integrating multimedia data as facts in a multidimensional model is to deal with dimensions built on descriptors that can be obtained by various computation modes on raw multimedia data. Taking into account these computation modes makes possible the characterization of the data by various points of view depending on the user's profile, his best-practices, his level of expertise, and so on. We propose a new multidimensional model that integrates functional dimension versions allowing the descriptors of the multidimensional data to be computed by different functions. With this approach, the user is able to obtain and choose multiple points of view on the data he analyses. This model is used to develop an OLAP application for navigation into a hypercube integrating various functional dimension versions for the calculus of descriptors in a medical use case.

Categories and Subject Descriptors: H.1.0 [**Models and Principles**]: General

General Terms: Design

Additional Key Words and Phrases: Data warehouse, OLAP, multimedia, functional version, descriptor

1. INTRODUCTION

The most cited definition for a data warehouse is Inmon's [1996]: a data warehouse is a "subject-oriented, integrated, nonvolatile and time-variant collection of data in support of management's decisions." Data warehousing projects intend to select and extract relevant data from various production databases and to organize it so as to improve decision making. The modeling paradigm for a data warehouse must comply with requirements that are profoundly different from the data models in OLTP (On-Line Transactional Processing) environments [Kimball 1996]. The models must be easy for the end-user to

Authors' addresses: A.-M. Arigon, Laboratoire de Biométrie et Biologie Evolutive, LBBE UMR CNRS 5558, UCBL, 43 Boulevard du 11 novembre 1918, 69622 Villeurbanne Cedex, France; email: arigon@biomserv.univ-lyon1.fr; A. Tchounikine and M. Miquel, Laboratoire d'InfoRmatique en Images et Systémes d'information, LIRIS UMR CNRS 5205 INSA, 7 avenue Capelle, 69621 Villeurbanne Cedex, France; email: {anne.tchounikine, maryvonne.miquel}@insa-lyon.fr.

understand and write queries, and must maximize the efficiency of queries. Data warehouse models are called multidimensional models. The decisional analyses are based on OLAP processes (On-Line Analytical Processing) as defined in Chaudhuri and Dayal [1997] and Vassiliadis and Sellis [1999], which refer to analysis functionalities used to explore data.

Data warehouse has become a leading topic in the commercial world as well as in the research community. Until now, data warehouse technology has been mainly used in the business world, in retail or finance areas, for example. The leading motivation is then to take benefits from the enormous amount of data that relies in operational databases. Data warehouses are generally used to provide means for the analysis of facts that are depicted by numeric data (e.g. sales depicted by amount or quantity sold). However, in numerous fields, like in medical or bioinformatics, multimedia data is used as valuable information in the decision-making process. The use of multimedia data as facts in a warehouse raises numerous problems. Among these are storage of voluminous data, design of a suitable navigation interface, complex ad hoc aggregation functions and so on. Another important problem that is barely addressed relates to the semantics of multimedia data, and to the way specialists deal with it. Indeed, scientists often have to develop efficient algorithms (e.g. based on signal or image processing, pattern recognition, statistical methodologies . . .) in order to transform the initial raw data (e.g. an electrocardiogram, an X-ray . . .) into relevant information (data descriptors, risk factors, diagnosis class . . .). Beyond the difficulties encountered in extracting and modeling multimedia data, lies the difficulty in manipulating, interpreting the data and the need for transforming raw data into useful information. This expertise, which is user-dependent as well as analysis-dependent, is fully part of the decisional process.

This article is organized as follows. In Section 2 we provide some definitions for OLAP and multidimensional models, and for multimedia data warehouses. With motivating examples, we highlight the needs for integrating multiple points of view in multidimensional models, and give an outline of our contribution and related work. Section 3 details our multiversion model. The prototype, and an example of a data warehouse built for a medical study are presented in Section 4. Section 5 concludes.

2. MULTIDIMENSIONAL MODEL FOR MULMEDIA DATA

2.1 OLAP and Multidimensional Models

Data warehousing solutions are usually built following a multitier architecture (Figure 1). The first tier is a warehouse server, classically a relational DBMS. Data of interest must be extracted from operational legacy databases, cleaned and transformed by an ETL tool (extraction, transformation, and loading) before being stored in the warehouse. The purpose is to consolidate the heterogeneous schemas, to do time stamping, and to reduce the data in order to make it conform to the warehouse model. Aggregation and/or discretization functions for example, may be used to achieve reduction. This step guaranties that the warehouse contains high quality, historical, and homogeneous data. The OLAP server composes the second tier. The OLAP server models the data in a multidimensional way and precalculates aggregates in order to optimize queries. The third tier is an OLAP front-end client that provides operators to interactively explore the multidimensional data, supporting in the meantime the iterative nature of the analysis process, and allowing the decision makers to navigate across data at different levels of detail.

OLAP multidimensional models, or data cubes, have been formalized by several authors [Agrawal et al. 1995; Cabibbo and Torlone 1998; Gyssens and Lakshmanan 1997; Lehner 1998; Li and Sean Wang 1996; Vassiliadis and Sellis 1999]. They are designed to represent the analyzed fact depicted by measures, and the various dimensions that characterize the fact. As an example in a retail area, typical facts are “sales” depicted by measures such as “number of sold items” or “amount”; typical dimensions

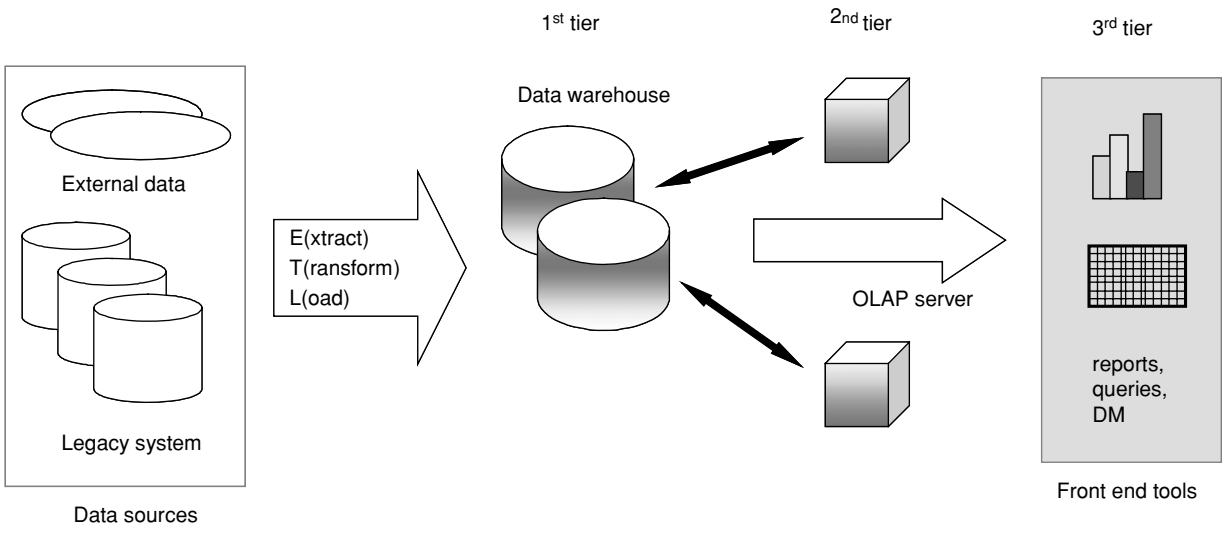


Fig. 1. Data warehousing 3-tier architecture.

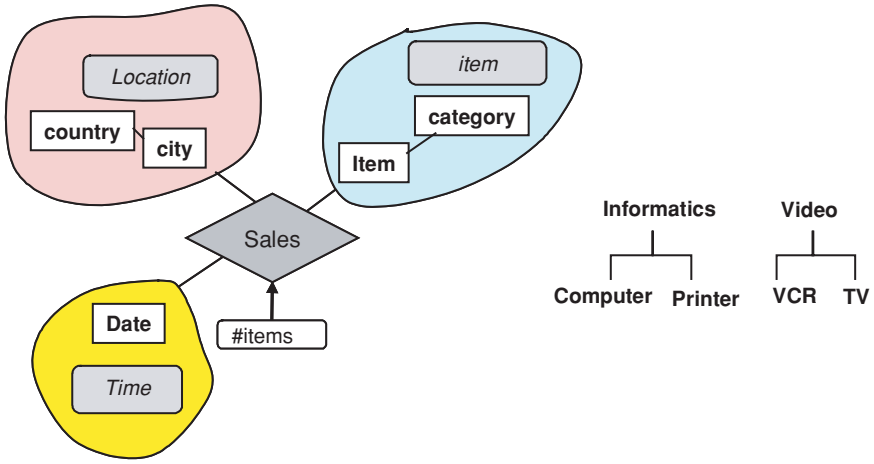


Fig. 2. Retail multidimensional model and instance of the "Product" dimension.

are "product", "location" and "time" (Figure 2). The dimensions are organized following a schema: a hierarchy representing various granularities or various degrees of precision.

For example, the hierarchy for the "location" dimension can define "city" and "country" levels. Each level of a dimension is composed of members. These members are also connected by hierarchical links; this hierarchical structure is the instance of the dimension. As an example, there are two levels, "Item" and "Category," in the "product" dimension; the "category" level is composed of the members "Informatics" and "Video" whereas the "item" level is composed of the members "Computer," "Printer," "VCR" and "TV."

In the data warehouse, the measures are stored in a fact table. The dimension data can be stored in tables following a "star model," a "snowflake model," or a "galaxy model." The "star schema" models the dimension data as a unique table, in which the hierarchical relationship of a dimension is not explicit

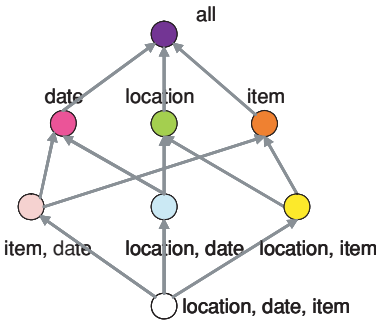


Fig. 3. Lattice of cuboids.

but is rather encapsulated in its attributes. The "snowflake schema" normalizes dimension tables, and makes it possible to explicitly represent the hierarchies by separately identifying a dimension in its various granularities. When multiple fact tables are required, the "galaxy model," or "fact constellation" models allow the design of a collection of star schemas.

In order to optimize the queries, query results are precalculated in the form of aggregates. The classical functions used to aggregate measures are "count," "sum," "min," "max," and "avg." The set of aggregates is called a hypercube and is usually represented by a lattice of cuboids [Harinarayan et al. 1996]. Figure 3 illustrates the lattice of cuboids for the example on retail area. OLAP engines are used to model and to materialize a part of the hypercube: a selected set of cuboids from the lattice. The hypercube is stored either in ROLAP (relational OLAP), MOLAP (Multidimensional OLAP) or HOLAP (Hybrid OLAP) internal format.

The hypercube maintained by the OLAP engine contains the values of the detailed and aggregated measures in its cells, and the axes of the hypercube are made from the dimension members. Common OLAP operators include roll-up, drill-down, slice and dice, rotate. These operators are performed selecting a cell or a group of cells in the hypercube maintained by the OLAP engine (Figure 4).

2.2 Multimedia Multidimensional Models

Several works on the design of multimedia data cubes were undertaken to improve multidimensional analysis of large multimedia databases. In You et al. [2001], the authors seek to extend the concept of traditional data warehouses and multimedia databases so as to store and represent multimedia data. The MultiMediaMiner analysis system uses a multimedia data cube [Zaiane et al. 1998] making it possible to store multidimensional data at different granularities. In the medical domain, the use of processes for exploration and analysis of huge multimedia data is critical. For example, let us quote a study undertaken within the domain of breast cancer detection in which the aim is the development of a data warehouse of numerical mammography for diagnostic help [Zhang et al. 2001]. Another study [Tikekar and Fotouhi 1995] deals with the problem of storage and restitution of medical image data from a warehouse by comparing the data warehouse with a pyramid of means of storage. Some other studies are made particularly on cancer by making a multidimensional analysis of an epidemiological spatio-temporal data set [Kamp and Wietek 1997].

All these works are based on multimedia database modeling and on descriptors that define the data. The design of these multimedia data cubes is copied from the design of traditional data cubes. A star or snowflake schema is used to model multimedia data warehouses: the fact table gathers multimedia data in the form of links; the dimensions represent the descriptors of this data. The descriptors that are used to characterize the multimedia data facts are of two types: content-based descriptors

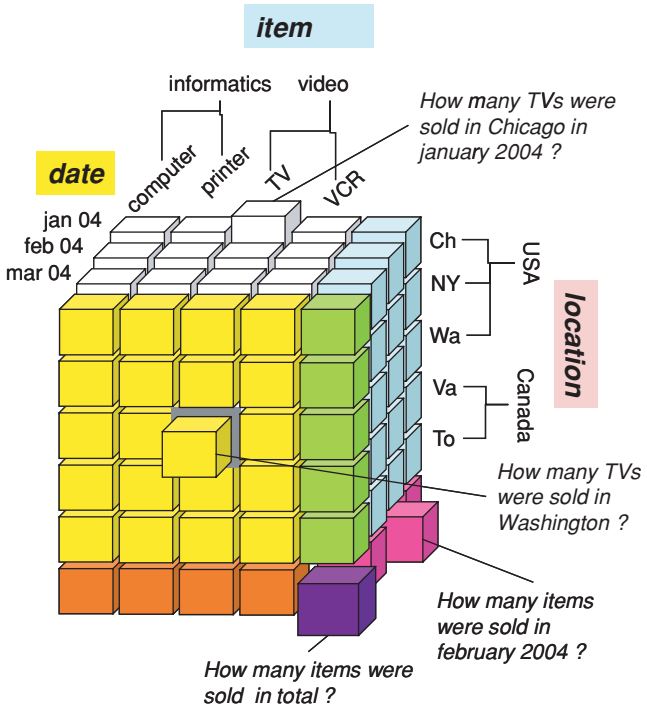


Fig. 4. A hypercube.

and description-based, or textual, descriptors. Description-based descriptors are extrinsic information (e.g., keywords, date of acquisition, author, topic . . .). Content-based descriptors represent the intrinsic content of data (e.g., color, texture, shape, . . .) and are generally automatically extracted from the data [Han and Kamber 2001; Zaiane et al. 1998]. For example, in a data warehouse with image data as fact, the dimensions for content-based descriptors can be the color, the texture, and the dimensions for description-based descriptors, the topic and the keywords. All the previously cited studies use dimensions based on data descriptors computed at loading time. The design of multimedia data cubes follows the design of traditional data cube paradigms—dimension members that correspond to values of data descriptors are computed upstream during the ETL process and remain static once stored in the dimension tables.

2.3 Motivating Examples

The first example deals with the medical area. In the case of disease studies, researchers try to extract information that best reflects the derangement from a chosen population. We work with an INSERM team (ERM 107) specialized in the methodology of cardiological information. In the EMIAT project (European Myocardial Infarct Amiodarone Trial), the aim of the ERM107 team is to extract knowledge about descriptors that can be used to evaluate a cardiac pathology evolution. As a result, the EMIAT study provides a significant amount of data to be exploited and analyzed. Among this data, is the multimedia data, electrocardiogram signals (ECGs) of the various patients who participated in the study (Figure 5).

The ECGs of the EMIAT study constitute data sources from which several descriptors or indicators are extracted and transformed by an ETL (extraction, transformation, and loading) tool before being

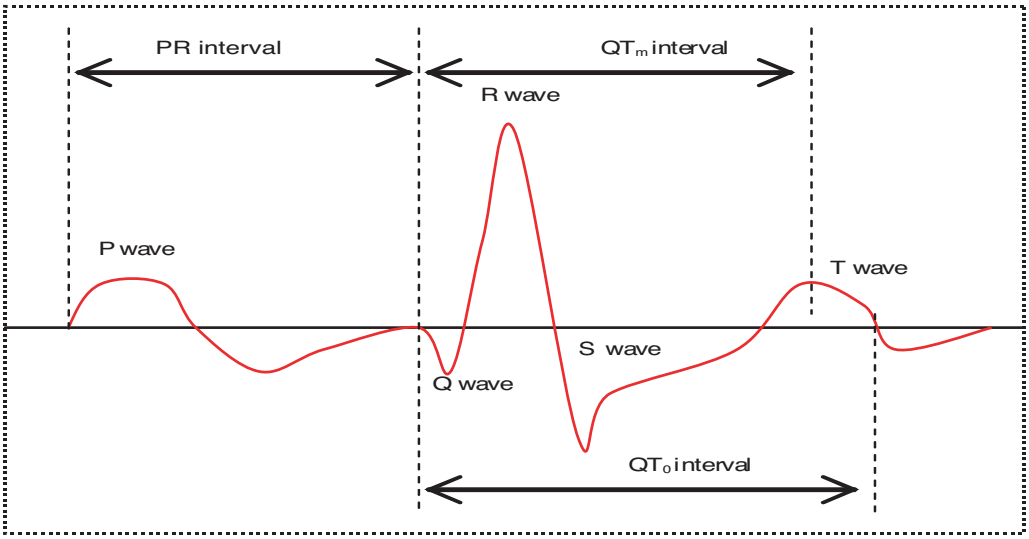


Fig. 5. ECG signal and some descriptors.

loaded in a warehouse. In this data warehouse, the facts are ECGs with various measures: list of ECGs, number of ECGs, and average ECG. This multimedia data is characterized by descriptors that correspond to the dimensions of the model. Several descriptors or indicators can be computed from an ECG to characterize the cardiac health state of a patient. The QT interval measure (time after which the ventricles are repolarized) is the most studied [Chevalier et al. 2001]. The noise level is also used as a criterion for the selection of a study population because it gives a quality indication. Other data is associated with these ECGs, such as the patient's pathology. Thus, two types of descriptors characterize ECGs and constitute the dimensions: description-based descriptors (principal pathology, age, gender of the patient, ECG acquisition slot and date, technology with which the ECG is obtained) and content-based descriptors (the QT duration and the noise level of the signal). A simple multidimensional model of the data warehouse shows the fact table (ECGs Signal) and the dimensions we use (Time, Duration of QT, gender...) (Figure 6).

It is difficult to establish before the study, the methodological approach to process these descriptors. Indeed, some indicators can be extracted by various computation modes. As an example, the QT duration can be calculated using different algorithms: three threshold methods (T1, T2, T3), and two T wave slope methods (S1, S2). The design of these methods for information extraction is often a part of the research task. When several methods compete, the aim is to evaluate them. If the purpose is to study the characteristics and the evolution of descriptors taken of a specific population, the descriptor calculation is at least as significant as the selection of the raw data. Moreover, the diversity of extraction methods enables one to take into account the expertise, the domain of specificity, and the best-practices of the physician who can choose the versions of descriptors adapted to the study. Thus, it is necessary to allow the user to choose relevant computation modes or versions of descriptors in order to select different points of view on multimedia data and to constitute several study populations for comparisons.

In the second example, data comes from the bioinformatics area. Today, sequencing techniques are routinely used and the number of new available sequences grows quickly. The sequence of a DNA fragment contains various data which have to be identified and interpreted. The best known elements

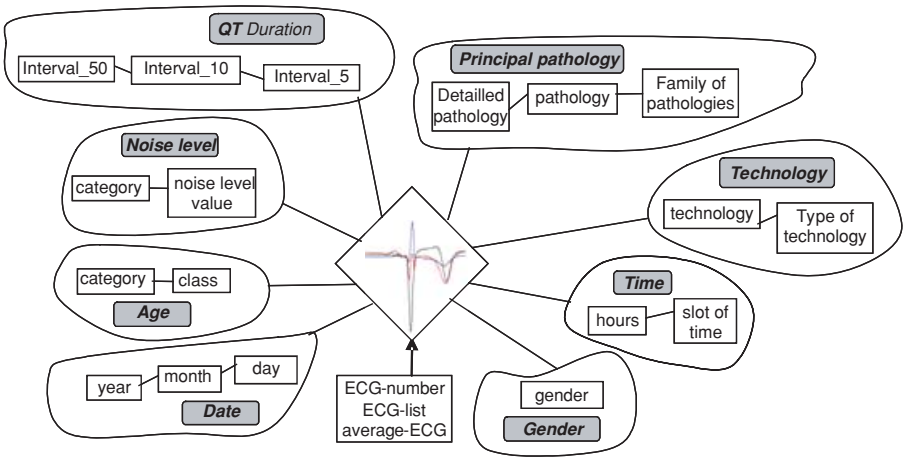


Fig. 6. Multidimensional model of the EMIAT data warehouse.

of sequences correspond to genes, delimited by beginning and end signals, and used to code proteins. The genome (i.e. the set of genetic material of an individual or a species) is identical in each cell of a given organism. However, not all genes are permanently expressed in a cell; they can have a specific expression differentiated in time (specific to a development stage), in space (proper to a cellular, tissue or organic type) and/or characteristic of a given state (normal, pathological, or in response to a particular stimulus). Thus, a gene can be characterized by various expression levels computed in different ways. A gene can also be characterized by its GC-content (guanine-cytosine content: the proportion of GC-base pairs in the sequence). This GC-content can be calculated using various methods: three methods based on the position of the GC-base pairs in the gene (GC1, GC2, GC3), and two methods based on the region of the gene where the GC-content is calculated (exon GC-content for coding region or intron GC-content for noncoding region). All this information constitutes a large amount of data that can be integrated into a data warehouse in order to analyze and cross them to better understand links between organisms and biological mechanisms. Indeed it would be interesting to be able to localize genes on chromosomes for a given species according to their expression level. For example, this would allow us to compare data between healthy tissues and cancerous tissues and to detect possible abnormality.

We use this data to design a data warehouse in which facts correspond to chromosomes with measures such as number and list of chromosomes. These chromosomes are multimedia data and more precisely, image data on which genes are localized (Figure 7). The dimensions of the model are the various descriptors characterizing the chromosomes, such as taxon (i.e., a group of organisms), genes, protein functions for description-based descriptors, and GC-content, expression level for content-based descriptors. Figure 8 shows the multidimensional model of the data warehouse.

As the GC-content and expression level descriptors can be obtained by various methods, it would be interesting to allow a user to visualize chromosomes for selected computation modes of GC-content and expression level. Therefore, a user would be able to choose various versions of descriptors in order to make comparisons, to obtain various points of view and to have better interpretations of data. The choice between the various versions of descriptors is thus necessary to have a data representation adapted to the needs and the preferences of biologists.

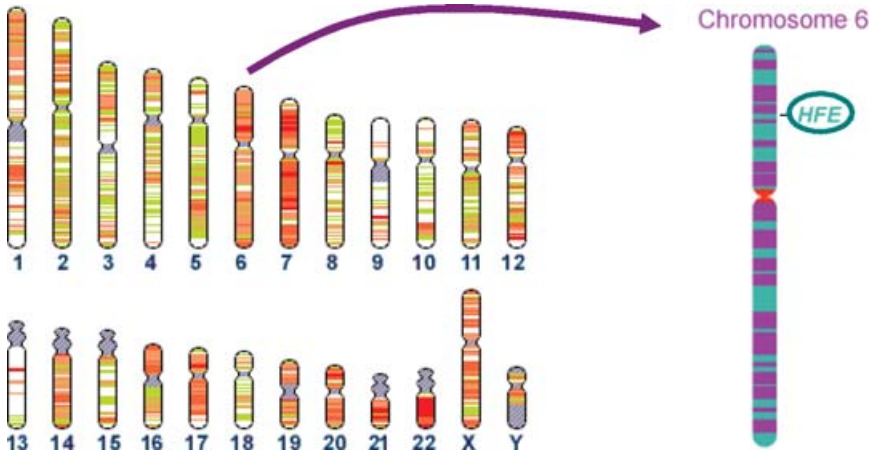


Fig. 7. Human genome and chromosome 6 on which the HFE gene is localized.

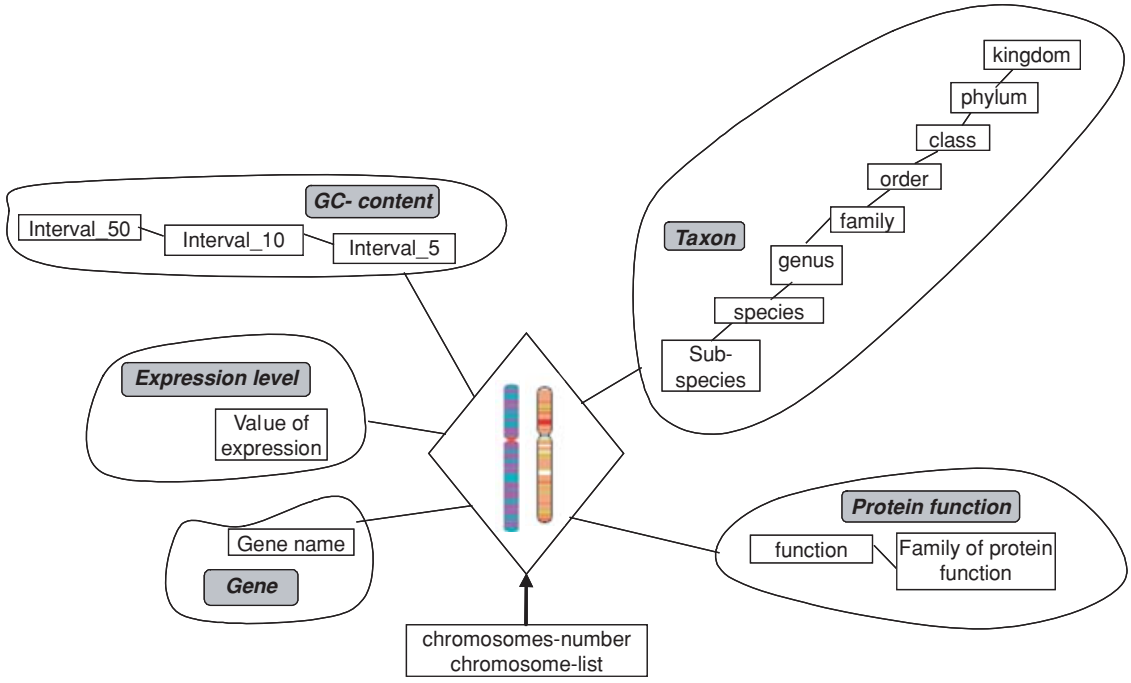


Fig. 8. Multidimensional model of the bioinformatics data warehouse.

2.4 Contribution and Related Work

As shown in the examples, candidate descriptors for multimedia data are numerous and the same descriptor can often be extracted in various ways, giving different, however still correct, values. These different computation modes can be seen as different functional versions of a descriptor. They make possible the characterization of the data by various points of view. These different points of view can

depend for instance on the user profile, his best-practices, his level of expertise, or the type of the ongoing analysis, and so on. In multidimensional models, dimensions could integrate multiple methods to represent the data according to the various functional versions of each descriptor, in order to on-the-fly "customize" the model and the decisional process. We believe that the integration of functional dimension versions into the multidimensional model, enabling the descriptors to be computed by different functions, can improve the characterization and the analysis of the data. Indeed, functional dimension versions will allow the user to obtain multiple points of view on the data he analyses. Thus, the user will be able to choose one of them in order to define the best interpretation, or his favorite representation of the multimedia data. He will also be authorized to compare the different results obtained using different computation modes for the descriptors. The goal of our proposal is to define a multidimensional conceptual model capable of managing multimedia data characterized by descriptors obtained by various computation modes. Meanwhile, we use the concept of dimension version, where the version relates to the method used to define or extract a multimedia data descriptor.

Versions or dynamic aspects in multidimensional models are carried out in studies that deal with time-related evolutions. Two types of approaches are proposed in order to take evolutions of the analysis structures into account: the first one consists in updating models [Blaschka 1999; Blaschka et al. 1999; Hurtado et al. 1999a, 1999b] that map data into the most recent version of the structure; the second is based on tracking history models [Body et al. 2003; Eder and Koncilia 2001; Mendelzon and Vaisman 2000; Pedersen et al. 2001; Bliujute et al. 1998; Chamoni and Stock 1999; Golfarelli et al. 2004] that keep track of evolutions of the system. This last type of model is particularly interesting because it makes it possible to analyze data in their various versions and evolutions. Among these tracking history approaches, some [Bliujute et al. 1998; Chamoni and Stock 1999] choose to represent data in the temporally consistent mode of presentation. They present the same limits as the "Type Two Slowly Changing Dimensions" of Kimball [1996]: they do not allow comparisons across time. Several authors have proposed models that take the handling of evolutions into account. The models of Mendelzon and Vaisman [2000], Pedersen et al. [2001], and Eder and Koncilia [2001], account for user needs in accurately tracking history and comparing data, and they provide a way of mapping data in an unchanged structure, chosen by the user. Eder and Koncilia [2001] propose mapping functions that allow conversions between structure versions. However, this solution neither takes schema evolution and time consistent presentation into account, nor considers complex dimension structures. Mendelzon and Vaisman [2000] define a temporal multidimensional model that uses valid times to build the TOLAP query language. The user can choose in his request, the way he wants the data to be aggregated—in a temporally consistent representation or in the latest version. Nonetheless, the model does not provide the means of reporting data in any other version than the latest one, and only partially takes merges and split evolutions into account. The multiversion and multidimensional model of Body et al. [2003], integrates the concept of version in dimensions and gathers the data in a fact table according to various temporal modes of presentation. It also provides an end-user interface, allowing the user to choose the temporal mode of data representation. The user chooses a temporal mode of presentation; meanwhile he selects a predetermined set of dimensions that corresponds to data versions for the temporal mode of presentation selected. Finally, Golfarelli et al. [2004] propose an approach to schema versioning in data warehouses, where the designer may decide to undertake some actions on old data aimed at increasing the flexibility in formulating cross-version queries, queries spanning multiple schema versions. The navigation process is restricted in the same way as the previous model. In conclusion, these models usually focus on temporal evolutions and mapping processes applied to facts. They do not offer navigation into functional dimension versions that could enable a comparison between extraction and interpretation.

3. THE MULTIVERSION MODEL

3.1 General Principle of Our Approach

Our approach is based on a fact table that contains the set of measures representing the data to analyze (references on multimedia data) and dimensions representing the descriptors of this multimedia data. Taking into account the problem of functional multiversion, we redefine the multidimensional structure, adding the concept of functional version. Therefore, we introduce the concept of dimension version, multiversion dimension, functional multiversion fact table, and dimension version function. A multiversion dimension is composed of several versions of a dimension, each one being a dimension for a given version with its own schema and its own instance. The functional multiversion fact table gathers all data, combining the various dimension versions of a dimension with the others. Finally, the associated function of a dimension version corresponds to the computation modes used to get the members of this version. The schemas of various dimensions are described using the hierarchical levels and the links that bind them. We also describe the instances of these dimensions with the set of members belonging to a hierarchical level and their parent-child relationships. Thus, our approach allows having explicit dimensions since the schemas of dimension are explicitly defined. Our model also supports complex hierarchies (multiple, non-onto, non-strict and non-covering hierarchies) since the dimension instances are defined from the members and the hierarchical links.

3.2 Concept Definitions

Definition 1. *Schema of a dimension version*. A schema of a dimension version is the dimension schema for a given version, that is to say a computation mode used to obtain the members of a dimension. The schema S_{VD} of the dimension version *VDid* is a tuple $<VDid, \mathcal{N}, <_{VDid}>$ where:

—*VDid* is the dimension version identifier

—$\mathcal{N} = \{n_j, j = 1, \ldots, k\}$ is the set of levels of S_{VD}. A level in S_{VD} represents a set of values with the same granularity associated with the same dimension version. A level n_j is defined by the tuple $<levelId_j, levelName_j, [\mathcal{A}_j], [description_j]>$ where:
  —$levelId_j$ is the identifier for the dimension version level,
  —$levelName_j$ is the name for the dimension version level,
  —$\mathcal{A}_j$ is an optional property representing descriptive attributes of this level,
  —$description_j$ is an optional property representing textual information on the level n_j,

—$<_{VDid}$ is a partial order on the set $\mathcal{N}$ that defines the hierarchical links between the levels of schema S_{VD} when the number of levels is more than 1.

Thus, a schema of dimension version can be seen as a directed graph in which nodes are elements of the set $\mathcal{N}$, and arcs are relations according $<_{VDid}$. This graph must be acyclic in order to enable aggregations to the least fine hierarchical levels. A level ALL is defined as the root of the hierarchy: the highest level of granularity.

Example 1. Let us suppose we want to analyze the influence of age on ECG. Age is a dimension of the data warehouse but its members can be ordered in various ways: the ages can be classified by intervals of age, for example five year, ten year or fifty year intervals.

Let $S_{agePerInterval}$ be a schema of the dimension version "agePerInterval " of which VDid = 1. The schema of this dimension version is defined by:

$S_{agePerInterval} =< 1, \{n_1, n_2, n_3\}, <_1>$ with
$n_1 =< 1,$ "Interval_5">, $n_2 =< 2,$ "Interval_10">, $n_3 =<3,$" Interval_50">
and the following order: $n_1 <_1 n_2, n_2 <_1 n_3$ and $n_3 <_1$ ALL

Fig. 9. Schema of the dimension version.

The schema of the dimension version $S_{agePerInterval}$ can be represented by the following graph (Figure 9(a)).

On the other hand, these ages can be collected in age classes (young child, child, teenager, young adult, adult, senior) that can be regrouped in categories (minor, major) allowing the possible definition of the schema of this dimension version $S_{agePerClass}$ (see Figure 9(b)).

Now let us consider the duration of the QT of these electrocardiograms as another dimension of the data warehouse. It can be computed by several algorithms: algo1 and algo2. The dimension schema characterizing the duration of the QT is hierarchically structured by the values of the duration of the QT for the finest level, gathered within intervals of 100 ms, then of 400 ms. Therefore the schemas of these two dimension versions, $S_{Qtalgo1}$ and $S_{Qtalgo2}$, can be defined.

Definition 2. *Dimension version.* A dimension version is a dimension for a given version. The dimension version *VD* of schema $S_{VD} = <VDid, \mathcal{N}, <_{VDid}>$ is a tuple $<VDid, VDname, \mathcal{M}, <_{VD}, [VDdescription]>$ where:

—*VDid* is the unique dimension version identifier,

—*VDname* is the dimension version name,

—$\mathcal{M} = \{m_j, j = 1 \ldots l\}$ is the set of members of this dimension version. A member of a dimension version is a member computed by the computation mode corresponding to the dimension version. It belongs to one of the levels of the schema S_{VD}. Thus, a level is composed of members gathered with the same granularity. A member m_j is a tuple $<id_j, val_j, [a_j], levelId_j>$ where:

  —id_j is a unique identifier for this dimension version member,

  —val_j is the value for this dimension version member,

  —a_j is an optional property that contains the set of values of the attributes related to this member (corresponding to the level). If this property is defined for the level corresponding to the member, then it must be defined for the member.

  —$levelId_j$ is the identifier for the hierarchical level to which this member of dimension version belongs.

—$<_{VD}$ is a partial order on the set $\mathcal{M}$ that defines hierarchical links between the members of the same dimension version *VD*. For each pair of levels (n_1, n_2), such as $n_1 <_{VDid} n_2$, there is at least a couple $(m_1, m_2) \in \mathcal{M} \times \mathcal{M}$, such as $m_1.levelId_1 = n_1$ and $m_2.levelId_2 = n_2$ and $m_1 <_{VD} m_2$. Thus, m_1 is said to be a member of lower level than m_2—m_1 has a finer granularity than m_2.

—*VDdescription* is an optional property containing possible comments on the dimension version.

Thus, a dimension version can be represented by an acyclic directed graph, in which nodes are elements of the set $\mathcal{M}$, and arcs are relations according to $<_{VD}$. Afterwards, we will call "leaf member"

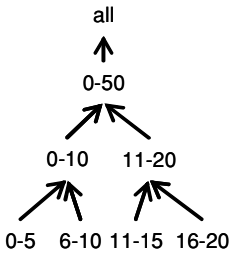


Fig. 10. Instance of the dimension version "agePerInterval."

of a dimension version, a member of a dimension version that has no child. Moreover, the member "all" is defined as the unique member contained in level "ALL." One notes $\mathcal{LM}_{VD}$ the set of leaf members of the dimension version *VD*. This set is defined as follows:

$$\mathcal{LM}_{VD} = \{m_j | m_j \in \mathcal{M} \text{ and } \neg\exists m_i \in \mathcal{M}/(i \neq j \text{ and } m_i <_{VD} m_j)\}.$$

Example 2. The dimension version "agePerInterval" with the schema $S_{agePerInterval}$ presented in the previous example is defined by the tuple:

<1,"agePerInterval", {m_1,..,m_7},$<_a$> with
m_1 = <1, 0-5,1>, m_2 = <2, 6-10,1> , m_3 = <3, 11-15,1>, m_4 = <4,16-20,1>,
m_5 = <5,0-10,2>, m_6 = <6, 11-20, 2>, m_7 = <7, 0-50, 3>

and the following order:

$$m_1 <_a m_5, m_2 <_a m_5, m_3 <_a m_6, m_4 <_a m_6, m_5 <_a m_7, m_6 <_a m_7 \text{ and } m_7 <_a \text{all}.$$

The members of level n_1 ("Interval_5") are {m_1, m_2, m_3, m_4}, the members of level n_2 ("Interval_10") are {m_5, m_6}, and the member of level n_3 ("Interval_50") is {m_7}.

The set of $\mathcal{LM}_{agePerInterval}$ is defined by: $\mathcal{LM}_{agePerInterval} = \{m_1, m_2, m_3, m_4\}$

The dimension version "agePerInterval" can be represented by a graph (Figure 10).

In the same way, one can also define the dimension versions "agePerClass", "QTalgo1" and "QTalgo2" whose schemas are $S_{agePerClass}$, $S_{QTalgo1}$, $S_{Qtalgo2}$ as well as the sets $\mathcal{LM}_{agePerClass}$, $\mathcal{LM}_{QTalgo1}$ and $\mathcal{LM}_{Qtalgo2}$ can be also defined.

Definition 3. *Multiversion dimension*. A multiversion dimension *MVD* is a dimension that contains 1 to *n* dimension versions. *MVD* is a tuple <*MVDId, MVDname,* $\mathcal{VD}$*, [MVDdescription]*> where:

—*MVDId* is the unique multiversion dimension identifier,
—*MVDname* is the multiversion dimension name,
—$\mathcal{VD} = \{VD_i, i = 1, \ldots, n\}$ is the set of dimension versions associated with this multiversion dimension,
—*MVDdescription* is an optional property containing textual information on the multiversion dimension.

One notes $\mathcal{LM}_{MVD}$ the set of leaf members of the dimension versions contained in the multiversion dimension *MVD*. This set is defined by:

$\mathcal{LM}_{MVD} = \bigcup_{i=1}^{n} \mathcal{LM}_{VD_i}$ with *n* the number of dimension versions contained in the multiversion dimension *MVD*.

Example 3. The dimension versions "agePerInterval" and "agePerClass," previously defined, belong to the multiversion dimension "Age" with the identifier 1. This multiversion dimension is defined by:

Age = <1, "Age", {"agePerInterval", "agePerClass"}>.

In the following, the names of the members of dimension versions are used in order to identify them. One defines the set $\mathcal{LM}_{\text{Age}}$ by: $\mathcal{LM}_{\text{Age}} = \{0–5, 6–10, 11–15, 16–20, young child, child, teenager, young adult, adult, senior\}$.

The dimension versions "QTalgo1" and "QTalgo2" belong to multiversion dimension "DurationQT" with the identifier 2. This multiversion dimension is defined by: Duration QT = <2, QT, {QTalgo1, QTalgo2}>.

The set $\mathcal{LM}_{\text{DurationQT}}$ is defined the same way.

Definition 4. *Functional multiversion fact table*. A functional multiversion fact table provides the measures according to various dimension versions. Let $\{\mu_i, \text{i} = 1, \ldots, \text{m}\}$ be the set of measurements, a functional multiversion fact table *ft* is defined by a function such as:

$$ft : MVD_1 \times MVD_2 \times \cdots \times MVD_n \rightarrow dom(\mu_1) \times \cdots \times dom(\mu_m)$$
$$m_1, m_2, \ldots, m_n \mapsto v_1, \ldots, v_m$$

where:

—n is the number of multiversion dimensions of the data warehouse, $m_i \in \mathcal{LM}_{MVD_i}$ with $i = 1, \ldots, n$

—$dom(\mu_k)$ is the range for the measure μ_k.

This function associates the values v_k of measures μ_k with the leaf members of the dimension versions of each multiversion dimension.

Definition 5. *Function of dimension version.* The functions of dimension version are the computation modes that enable calculating the members of a dimension version *VD* from the data of the production database. A function of dimension version f_{VD} is a tuple <$functionId_{VD}$, *VDid*, $functionName_{VD}$, $functionDefinition_{VD}$> where:

—$functionId_{VD}$ is the identifier for the function of dimension version *VD*,

—*VDid* is the identifier for the dimension version *VD* whose members are computed using this function of dimension version,

—$functionName_{VD}$ is the name for the function of dimension version,

—$functionDefinition_{VD}$ is the definition for the function of dimension version.

These functions can be formulized as in the following:

$$f_{VD} : \mathcal{DB}_f \rightarrow \mathcal{LM}_{VD}$$
$$d \mapsto m$$

where $\mathcal{DB}_f$ is the set of data of the production database restricted to f_{VD}—used to compute the members of *VD*. f_{VD} associates a value of the production database and a leaf member of the version of corresponding *VD*.

Example 4. Let the function $\text{f}_{\text{agePerClass}}$ be defined for the dimension version "agePerClass" and the defined function $\text{f}_{\text{agePerInterval}}$ for the dimension version "agePerInterval ". For a 12-year-old patient, the members of the dimension versions are respectively:

$$\text{f}_{\text{agePerClass}}(12) = \text{"young"} \text{ and } \text{f}_{\text{agePerInterval}}(12) = \text{"11–15"}$$

Let the function $\text{f}_{\text{Qtalgo1}}$ be defined for the dimension version "Qtalgo1" and the defined function $\text{f}_{\text{Qtalgo2}}$ for the dimension version "Qtalgo2". Then for an electrocardiogram "ECG5," the members of the dimension versions are respectively:

$$\text{f}_{\text{Qtalgo1}}\ (\text{ECG5}) = 100 \text{ and } \text{f}_{\text{Qtalgo2}}\ (\text{ECG5}) = 110.$$

Definition 6. *Functional multiversion multidimensional structure*. A functional multiversion multidimensional structure $F2M$ is a tuple $<\mathcal{MVD}, ft, \mathcal{F}>$ where:

—$\mathcal{MVD} = \bigcup_{i=1}^{s} MVD_i$ is the set of all the multiversion dimensions,
—ft is the functional multiversion fact table,
—$\mathcal{F} = \bigcup_{j=1}^{r} f_{VD_j}$ is the set of all the functions of the dimension versions.

Definition 7. *Aggregation of data*. Aggregation of data can be computed from the multiversion fact table and the schema of the dimension versions. Let an aggregation function ϕ_k be defined for each measure μ_k, m a non-leaf member of the dimension version VD of the multiversion dimension MVD_1 and $m_1^1, m_2^1, \ldots, m_J^1$ its leaf members such as:

$$\left(m_1^1, m_2^1, \ldots, m_J^1\right) \in \mathcal{LM}_{VD} \times \ldots \times \mathcal{LM}_{VD}.$$

For each leaf member m_p^1 and the other leaf members $m_2, \ldots, m_n$ of the relying dimensions the ft function gives the value of the measures: $\forall p \in [1, J], ft(m_p^1, m_2, \ldots, m_n) = v_1^p, \ldots, v_m^p$ with n being the number of multiversion dimensions of the data warehouse.

Thus, the aggregated values for m are obtained by:

$$ft(m, m_2, \ldots, m_n) = \phi_{1_{p=1}}^{J} v_1^p, \ldots, \phi_{m_{p=1}}^{J} v_m^p,$$

applying the associated aggregation function on each measure.

The aggregation function can be a classic function of OLAP technology (sum, min, max, avg) for numeric measures or a more specific function (statistic average ...) for signal or image type measures.

4. DESCRIPTION OF THE IMPLEMENTED PROTOTYPE

Our prototype has been implemented following a three tier architecture:

1. a functional multiversion multimedia data warehouse in which the multiversion dimensions and the functional multiversion fact table are stored,
2. an OLAP cube built from the functional multiversion multimedia data warehouse, using aggregations, which enables requests against the functional dimension versions,
3. a tool for data navigation and visualization.

The prototype is easy to use by the clinical physicians because it doesn't need any knowledge of query language nor database management system. We use Microsoft SQL Server and Analysis Services to implement the prototype. The aggregation functions are implemented in Visual Basic; an interface using Proclarity 4.0 is built for navigation (Figure 11 and detailed further). The data is loaded from the production database to the functional multiversion multimedia data warehouse using the functions of dimension versions.

The data of the EMIAT study has been integrated in the prototype. The multiversion dimension tables and functional multiversion fact tables are populated with an ETL (Extraction, Transformation, Loading) tool. The data warehouse contains a functional multiversion fact table and eight multiversion dimensions. The facts are ECG signals from the EMIAT study. Multiversion dimensions represent the description-based descriptors and the content-based descriptors of these ECGs. Among the multiversion dimensions, three dimensions are related to the patient (principal pathology, age, and gender), two to ECG acquisition (time, technology, and date) and two to the content of the ECGs (the duration of the QT, the noise level). Aggregations of data are computed from the functional multiversion fact table and the hierarchical links between the members of the dimension versions. The aggregation functions

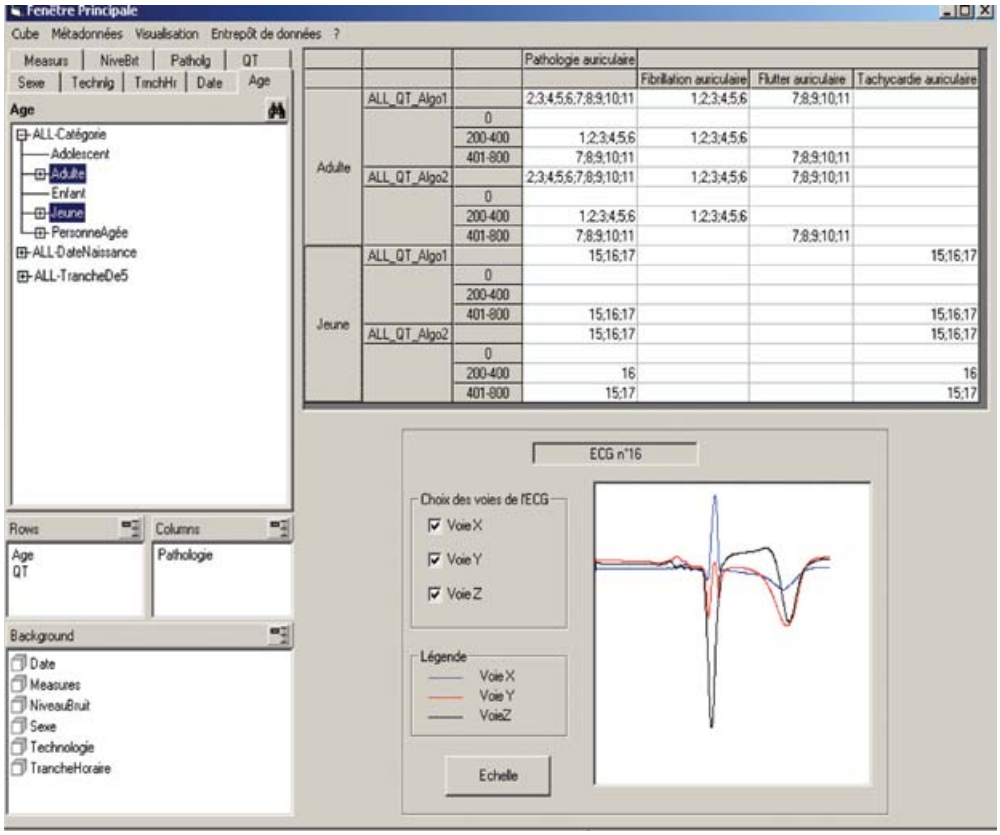


Fig. 11. Interface for the prototype.

enable computing aggregated data according to granularities of the dimension versions (the levels of the corresponding schemas). In our data warehouse, we define the following aggregation functions:

1. "ECG-count": this function counts the number of ECGs that correspond to the characteristics selected by the user.
2. "ECG-list": this function returns the list of ECGs that correspond to the characteristics selected by the user.
3. "Average-ECG": this function gives the "medium-ECG" calculated on the ECG-list.

The user can visualize in a frame, the hierarchies and the members of the dimension versions (Figure 12) and build his request choosing the appropriate data aggregation operator ("ECG-count", "ECG-list" or "average-ECG"). The user chooses the multiversion dimensions that appear in the resulting multidimensional table: he makes a "drag and drop" of the multiversion dimensions from "background" frame to "rows" or "columns" frames according to his choice. Then he can choose the granularity for each multiversion dimension in order to obtain a slice. It is also possible to visualize and to choose several dimension versions of a multiversion dimension in order to make comparisons. In this example, we can see the list of ECGs for adults and young patients with QT values obtained by two different algorithms (two dimension versions), and whose pathology is an "auricular pathology."

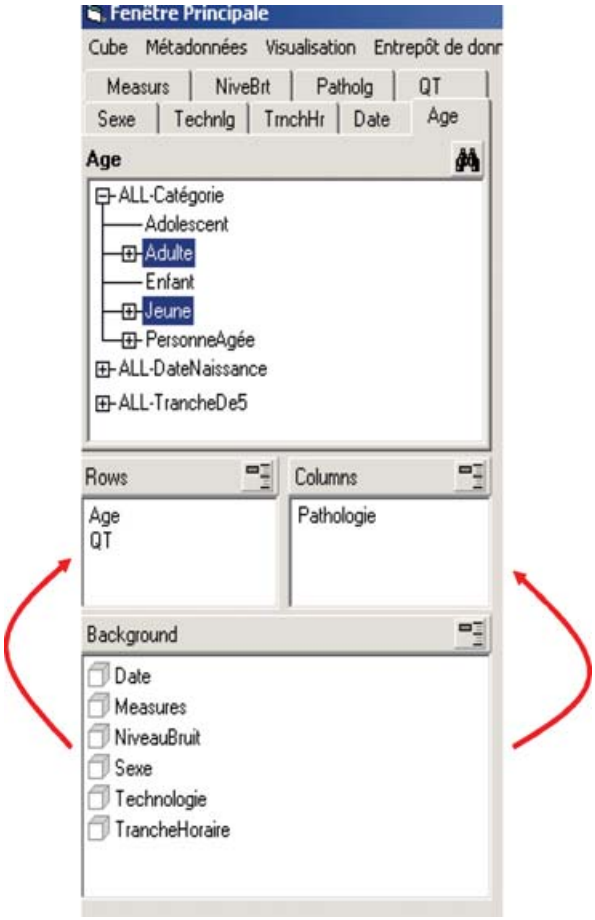
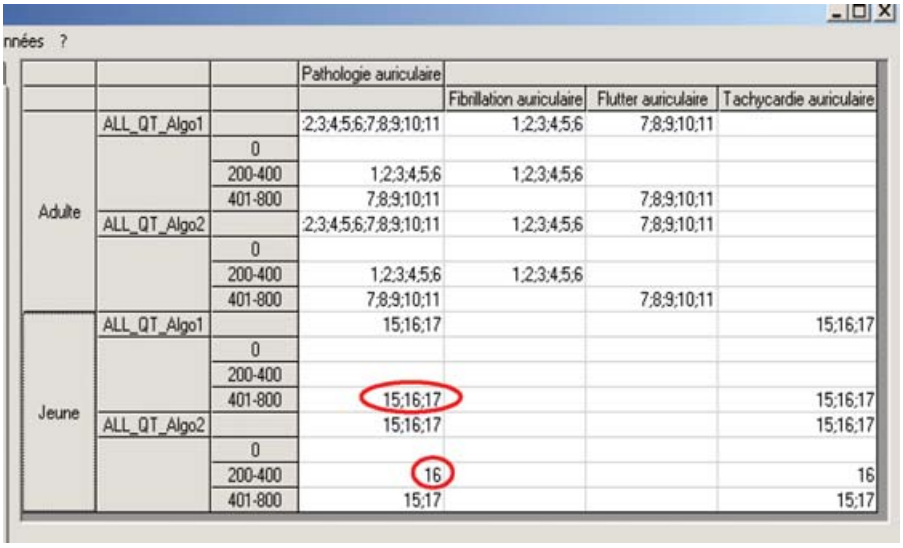


Fig. 12. Querying frame.

The result of the query is a multidimensional table (Figure 13). In this example, we have made a drill-down on the two dimension versions of QT for adult and young patients and also on the pathology dimension. We thus obtain the fact values at the finest level for pathology and QT multiversion dimensions. We can compare results between the two algorithms of QT. Indeed, we can see that the lists of ECGs are not the same according to the algorithm used. With the first algorithm, QT values are between 401 and 800 for the three young patients whereas with the second algorithm, QT values are between 200 and 400 for one patient and they are between 401 and 800 for the two other patients. This kind of comparison can allow physicians to detect some derangements in a defined population.

Moreover, the selected multimedia data can be visualized in a frame (Figure 14). The user chooses an ECG in the multidimensional table and then this ECG is drawn on the frame. In this example, we can visualize ECG number 16: it is represented by three lines (X, Y, Z). The user can select the line(s) he wants to visualize.

Finally, metadata provide details on multiversion dimensions, dimension versions, granularity, and functions of each dimension version. They can thus be visualized in order to have a global view of all dimension versions and to navigate the data cube more easily (e.g., schemas and instances of each



nnées ?

			Pathologie auriculaire			
				Fibrillation auriculaire	Flutter auriculaire	Tachycardie auriculaire
Adulte	ALL_QT_Algo1		:2;3;4;5;6;7;8;9;10;11	1;2;3;4;5;6	7;8;9;10;11	
		0				
		200-400	1;2;3;4;5;6	1;2;3;4;5;6		
		401-800	7;8;9;10;11		7;8;9;10;11	
	ALL_QT_Algo2		:2;3;4;5;6;7;8;9;10;11	1;2;3;4;5;6	7;8;9;10;11	
		0				
		200-400	1;2;3;4;5;6	1;2;3;4;5;6		
		401-800	7;8;9;10;11		7;8;9;10;11	
Jeune	ALL_QT_Algo1		15;16;17			15;16;17
		0				
		200-400				
		401-800	15;16;17			15;16;17
	ALL_QT_Algo2		15;16;17			15;16;17
		0				
		200-400	16			16
		401-800	15;17			15;17

Fig. 13. Multidimensional table.

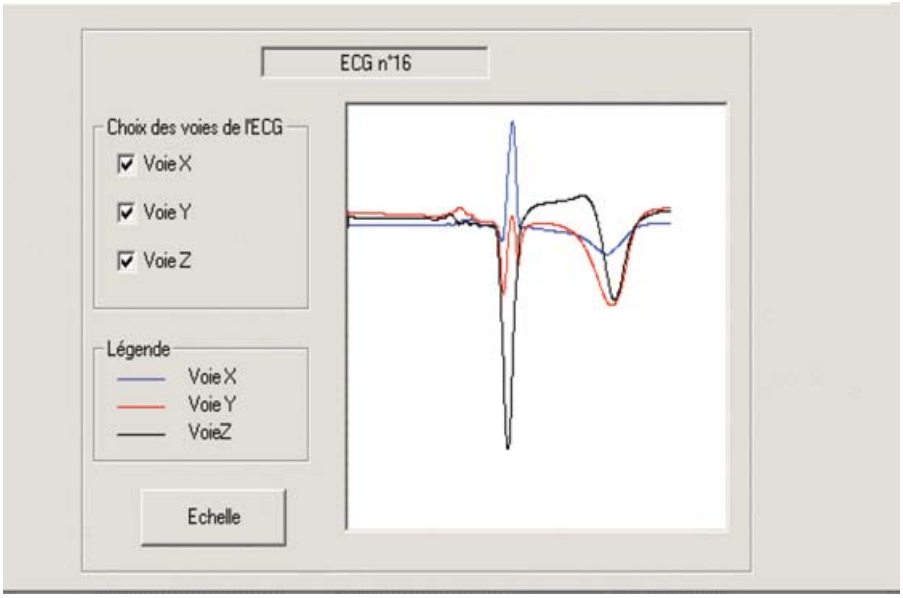
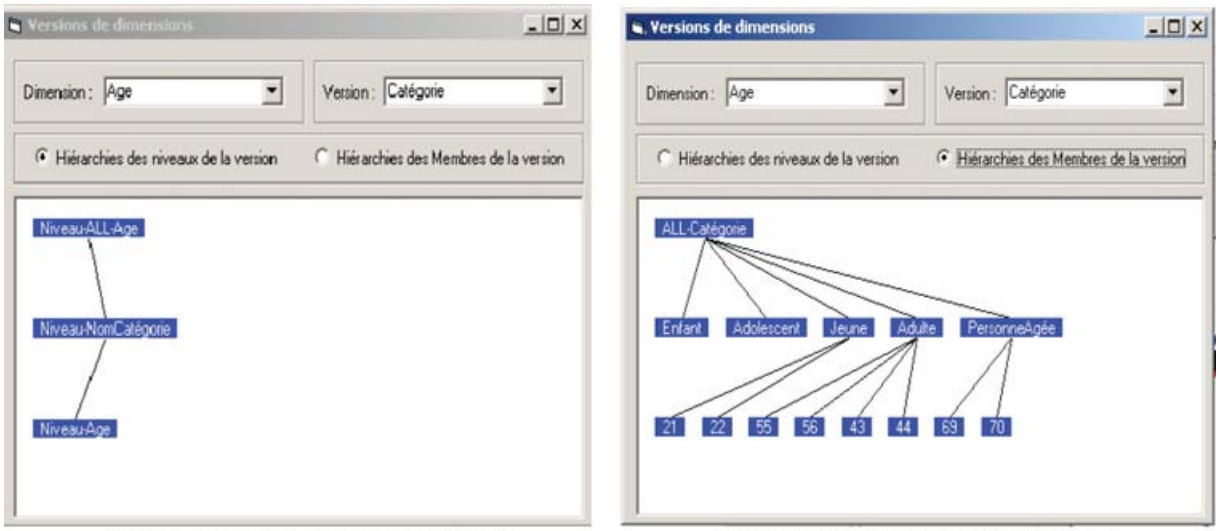


Fig. 14. Multimedia data visualization.

dimension version are represented for each multiversion dimension). The various functions of dimension versions can be represented in order to improve the analysis of obtained results. It is possible to select the multiversion dimension and the dimension version to visualize. In this example, the "Age" multiversion dimension is selected and "agePerClass" is chosen as the dimension version (Figure 15). The user can also choose the representation of the dimension version: the schema or the instance. In Figure 15(a) and 15(b), the schema and the instance of the "agePerClass" dimension version can be



(a) Schema of the "age Per Class" version

(b) Instance of the "age Per Class" version

Fig. 15. Visualization of metadata.

visualized respectively. This can help the user in his choice among different granularities and various dimension versions.

5. ONGOING WORK AND CONCLUSION

We present a model using functional versions to take into account the user's preferences according to his domain of specificity or his level of expertise. The model helps to manage multimedia data and allows the user to choose different points of view that correspond to various functional versions of the data descriptors. We define the concept of version and multiversion in dimensions in order to obtain results adapted to the user needs and to compare them. This model is particularly well suited to multimedia data because they require various computation modes for descriptor extraction. We also use specific aggregation functions for multimedia data that are integrated into the data warehouse. This model is used to develop an OLAP application for the navigation into a hypercube integrating signal data. We propose a tool to explore this complex data, which improves navigation in the multidimensional data cube. Thus, we enable the visualization of data according to several methods of analysis chosen by the user in accordance with his best-practice. We also provide the ability to visualize the representation of this multimedia data.

However, the data storage of our model could be improved. It is possible to have some redundancy in the schemas of dimension version, and the functional multiversion fact table storage is not optimized. Our model could be extended by associating the notion of functional version with the facts, in the same way that our model associates functional versions with dimensions. This could be done by adding a version of fact dimension so as to enable the user to choose the version of fact.

Finally the preaggregation stage is classically processed [Harinarayan et al. 1996]. However, in the case of multimedia data, the problem of the performance of query response time and storage limitation is definitely more crucial. In a previous study, we proposed a method [Vautrin 2004] to optimize the aggregation step in a multimedia medical use case. This method is based on the fact that the choice of the aggregated cuboids depends on the relevance of the selected population of patients. Views or

parts of views thus do not represent useful data for the user. In this case, these views should not be materialized so as to allow materialization of more useful views.

REFERENCES

AGRAWAL, R., GUPTA, A., AND SARAWAGI, S. 1995. *Modeling Multidimensional Databases*. IBM Research Report, IBM Almaden Research Center, September p. 25.

BLASCHKA, M. 1999. FIESTA: A framework for schema evolution in multidimensional information systems. In *Proceedings of 6th Doctoral Consortium.* Heidelberg, Germany, June.

BLASCHKA, M., SAPIA, C., AND HOFLING, G. 1999. On schema wvolution in multidimensional databases. In *Proceedings of Data Warehousing and Knowledge Discovery, First International Conference.* Florence, Italy, August 30–September 1, 1999, Lecture Notes in Computer Science 1676 Springer ISBN 3-540-66458-0.

BLIUJUTE, R., SALTENIS, S., SLIVINSKAS, G., AND JENSEN, C. S. 1998. Systematic change managment in dimensional data warehousing. In *Proceedings of the 3rd International Baltic Workshop on DB and IS*, pp. 27–41.

CHAMONI, P. AND STOCK, S. 1999. Temporal structures in data warehousing. In *Proceedings of Data Warehousing and Knowledge Discovery, First International Conference,* Florence, Italy, August 30–September 1, 1999.

BODY, M., MIQUEL, M., BEDARD, Y., AND TCHOUNIKINE, A. 2003. Handling Evolutions in Multidimensional structures. In *Proceedings of the 19th International Conference on Data Engineering*, March 5–8, 2003, Bangalore, India, IEEE Computer Society. ISBN 0-7803-7665-X.

CABIBBO, L. AND TORLONE, R. 1998. A logical approach to multidimensional databases. In *Proceedings of Advances in Database Technology—EDBT, 6th International Conference on Extending Database Technology*. Valencia, Spain, March 23–27, 1998, Lecture Notes in Computer Science 1377 Springer ISBN 3-540-64264-1.

CHAUDHURI, S. AND DAYAL, U. 1997. An overview of data warehousing and olap technology. *SIGMOD Record 26*, 1, (Mar.), 65–74.

CHEVALIER, P., RODRIGUEZ, C., BONTEMPS, L., MIQUEL, M., KIRKORIAN, G., ROUSSON, R., POTET, F., SCHOTT, J. J., BANO, I., AND TOUBOUL, P. 2001. Noninvasive testing of acquired long QT syndrome: Evidence for multiple arrhythmogenic substrates. *Cardiovascular Research*. *50*, 2 (May). 386–398.

EDER, J. AND KONCILIA, C. 2001. Evolution of dimension data in temporal data warehouses. In *Proceedings of the Data Warehousing and Knowledge Discovery, Third International Conference*, Munich, Germany, September 5–7, 2001, Lecture Notes in Computer Science 2114 Springer, ISBN 3-540-42553.

GOLFARELLI, M., LECHTENBÖRGER, J., RIZZI, S., AND VOSSEN, G. 2004. Schema versioning in data warehouses. *ER (Workshops)*, 415–428.

GYSSENS M. AND LAKSHMANAN, L. V. S. 1997. A Foundation for Multi-Dimensional Databases. In *Proceedings of the 23rd International Conference on Very Large Data Bases*. Athens, Greece August 25–29, 1997, Morgan Kaufmann, ISBN 1-55860-470-7.

HAN, J. AND KAMBER, M. 2001. *Data mining, concepts and techniques.* Morgan Kaufmann.

HARINARAYAN, V., RAJARAMAN, A., AND ULLMAN, J. D. 1996. Implementing data cubes efficiently. In *Proceedings of the ACM SIGMOD Conference of Management of Data*. pp. 205–216, Montreal, Quebec, June 1996.

HURTADO, C., MENDELZON, A. O., AND VAISMAN, A. 1999a. Maintaining data cubes under dimension updates. In *Proceedings of the 15th International Conference on Data Engineering*. Sydney, Australia, 23–26 March 1999, IEEE Computer Society Press.

HURTADO, C., MENDELZON, A. O., AND VAISMAN, A. 1999b. Updating OLAP dimensions. In *Proceedings of the ACM Second International Workshop on Data Warehousing and OLAP, ACM*. Kansas City, Missouri, USA, November 6, 1999.

INMON, W. H. 1996. *Building the Data Warehouse*. 3rd Edition, Wiley and Sons.

KAMP, V. AND WIETEK, F. 1997. Database system for multidimensional data analysis. In *Proceedings of the International Database Engineering and Application Symposium (IDEAS)*. Concordia University, Montreal, Candada, August 25–27, 1997.

KIMBALL, R. 1996. *The Data Warehouse Toolkit*. J. Wiley and Sons, Inc.

Lehner, W. 1998. Modeling large OLAP scenarios. In *Proceedings of the International Conference on Extending Database Technology.* Valencia, Spain.

LI, C. AND SEAN WANG, X. 1996. A data model for supporting on-line analytical processing. In *Proceedings of the Fifth International Conference on Information and Knowledge Management, ACM*. Rockville, Maryland, USA, November 12–16, 1996.

MENDELZON, A. O., AND VAISMAN, A. 2000. Temporal queries in OLAP. In *Proceedings of 26th International Conference on Very Large Data Bases*. Cairo, Egypt, September 10–14, 2000.

PEDERSEN, T. B., JENSEN, C., AND DYRESON, C. 2001. a foundation for capturing and querying complex multidimensional data. *Information Systems (special issue on data warehousing)*, *26*, 5 (July), 383–423.

TIKEKAR, R. V. AND FOTOUHI, F. 1995. Storage and retrieval of medical images from data warehouses. *Digital Image Storage and Archiving Systems*.

VASSILIADIS, P. AND SELLIS, T. 1999. A survey of logical models for OLAP databases. *SIGMOD Record 28*, 4 (Dec.), 64–69.

VAUTRIN, J. S. 2004. *Entrepôt de données multimedia*. master report.

YOU, J., DILLON, T., LIU, J., AND PISSALOUX, E. 2001. On hierarchical multimedia information retrieval. In *Proceedings of the International Conference on Image Processing*. Thessaloniki, Greece, October 7–10, 2001.

ZAIANE, O. R., HAN, J., LI, Z. H., AND HOU, J. 1998. Mining multimEdia data. In *Proceedings of CASCON, Meeting of Minds*, pp 83–96, Toronto, Canada, November 1998.

ZHANG, H., CAO, X., WONG, S. T., LOU, A. S., AND SICKLES, E. A. 2001. Developing a digital mammography data warehouse. In *Proceedings of SPIE,* Vol. 4323, pp. 308–315, Medical Imaging 2001: PACS and Integrated Medical Information Systems: Design and Evaluation, Eliot L. Siegel, H. K. Huang, Eds.

Received May 2006; accepted May 2006