Diamond multidimensional model and aggregation operators for document OLAP

Maha Azabou, Kaïs Khrouf
University of Sfax, MIR@CL Laboratory
Tunisia
Azabou.Maha@yahoo.fr, Khrouf.Kais@isecs.rnu.tn

Jamel Feki
Jeddah University, FCIT, IS dept.
King Saudi Arabia
Jamel.Feki@fsegs.rnu.tn

Chantal Soulé-Dupuy, Nathalie Vallès
IRIT, University of Toulouse 1 Capitole
France
{Chantal.Soule-Dupuy, Nathalie.Valles-Parlangeau}@ut-capitole.fr

Abstract— **On-Line Analytical Processing (OLAP) has generated methodologies for the analysis of structured data. However, they are not appropriate to handle document content analysis. Because of the fast growing of this type of data, there is a need for new approaches abling to manage textual content of data. Generally, these data exist in XML format. In this context, we propose an approach of construction of our *Diamond* multidimensional model, which includes semantic dimension to better consider the semantics of textual data In addition, we propose new aggregation operators for textual data in OLAP environment.**

Keywords—OLAP, XML documents, Diamond multidimensional model, aggregation operators.

I. Introduction

To learn from the past and forecast the future, many companies are adopting Business Intelligence (BI) tools and systems. The aim is to help companies in the processing of large amounts of heterogeneous data, from which it is difficult to manually extract useful information in order to formulate strategies and tactics for effective and profitable business. Most valuable business information is encoded in text. Companies use the advantages of the XML to analyze the explicit information of textual documents.

On-line Analytical Processing (OLAP) has been a valuable tool for analyzing trends in business information. Several studies have proposed the use of OLAP technology on textual data. The main objective of this proposal is to handle the XML documents in the same way that we manage the relational data. This implies the introduction of novel multidimensional model dedicated to the OLAP on XML documents called "diamond model".

In order to generate automatically this *Diamond* multidimensional model, we propose a set of heuristic rules aiming at determining the various components of the model (i.e., dimensions, hierarchies). Also, we provide new aggregation operators that take into consideration the specificities of this new multidimensional model. We focus in this paper on the two following operators: *List_Concept,* that extracts a list of the most used concepts and *G_Concept,* that extracts the most used generic concepts based on a semantic resource.

The rest of this paper is structured as follows. Section 2 presents the related work dealing with the multidimensional modeling and analysis of documents. Then, we describe in Section 3 the phases of construction of *diamond* models and we focus on the pretreatment phase. Section 4 details the rules and process we propose to generate *diamond* multidimensional models. We present in Section 5 the logical model as well as the rules for its derivation. We illustrate our aggregation operators in Section 6. Finally, we provide the conclusion in Section 7.

II. Related work

This section is organized in order to answer the following question: "where and how *current data analysis schemas take document semantics into account*?" Indeed textual data provide a fairly rich semantics which is important to consider when modeling and analyzing document contents.

Thus we have identified in the literature a number of studies dealing with multidimensional modeling of documents. These studies can be grouped into two categories, depending on how they treat the textual data, either as a textual dimension (mainly deduced from hierarchies of keywords or hierarchies of concepts), or as a fact with a textual dimension

- Textual dimension:
    - *Hierarchy of keywords (automatically built) summarizing a document*:

[1] proposes an indexing structure, called *D-tree*, to help organizing document contents into document cubes, which together constitute a document warehouse. As dimensional data, they use keywords extracted from document textual contents, documents' categories, and some metadata such as title, creator, date, and rights.

[2] proposes a new data cube called *Text cube* with a textual dimension represented by terms hierarchy. This

hierarchy specifies the semantic relationships between extracted terms. Each term becomes an element at base level; the parent element at upper level consists of all the children at lower level. *Text cube* involves two new OLAP operations such as pull-up (which generates a term level *L0* from a *lower* term level *L*) and push-down (which generates a term level *L0* from a *higher* term level *L*).

[3] proposes a new model called *Cube Index* based on a hierarchical description of each document. This hierarchy specifies relationships between words with respect to one document. It is used for the analysis of words in various levels of abstraction (L_i) in a document such as Document (*L5*), Paragraph (*L4*), Sentence (*L3*), Word Pair (*L2*) and Word (*L1*). Two new operations scroll up (given a level L_i and term v belonging to L_i, the result is a higher level L_{i+1} of document hierarchy) and scroll down (inverse operator of scroll up.) are discussed exclusively for the cube index. It supports term frequency and inverted document frequency to facilitate information retrieval techniques.

In all the previous approaches, the document content is lost and reduced to a small set of keywords represented by a dimension of keywords.

- *Hierarchy of concepts describing the related domain*:

In such kind of approach, the authors of [4] introduce the use of concept hierarchies in order to structure a document collection. Each concept hierarchy corresponds to a facet of the documents users can be interested in. *DocCube* model contains one dimension table per facet that describes the domain. A dimension is organized as a concept hierarchy; the different levels are depicted in a single table. The number of dimensions is not limited a priori. It depends on the application and on the type of users' needs the application answers. The fact table keeps the link between the different dimensions and the document.

[5] maps semantically text documents to an arbitrary topic hierarchy specified by an analyst. *Topic Cube* has a topic dimension which corresponds to the hierarchical topic. Drill-down and roll-up along this topic dimension will allow users to view the data from different granularities of topics.

Based on *Topic Cube* and information network analysis, the authors of [6] are interested in automatically constructing concept hierarchies by information network analysis. Such as, NetClus, dealing with multi-typed information network is used for integrated clustering, ranking, and concept hierarchy.

In the previous approaches, the hierarchies of concepts are obtained manually.

[7] proposes a contextual text cube model denoted *CXT-Cube* associated with contextual dimensions. Each dimension is related to one contextual factor corresponding to a definition of context in the data warehousing and OLAP. In *CXT-Cube* model, the contextual dimensions can be classified into two types: i) Semantic dimension: It is extracted from a domain ontology related to the dimension area as external knowledge source. ii) Metadata dimension: Metadata is external information about the documents, such as: date, title, author, etc. CXT-Cube includes a new textual analysis measure. In this measure, each document is represented by vectors of weighted concepts by using an OLAP-adapted vector space model, one vector for each dimension.

[8] proposes a new multidimensional model with textual dimensions. These textual dimensions are obtained through Text Mining techniques, from an intermediate representation form called AP-Structure. This knowledge structure can be obtained automatically and keeps the semantics of the texts.

- Fact with a textual dimension:

[9] proposes a multidimensional IR engine *MIRE* which is based on a multidimensional data model to use OLAP technology. *MIRE* is an approach to build an IR system on top of OLAP facility where fact tables contain the measure, word-appears-in document, and dimensional tables contain hierarchical structured data. *MIRE* is a new IR system integrating an inverted index for text and a multidimensional access method for dimensional data. The multidimensional access method is used for handling dimensional data, and it can offer OLAP functionalities such as drilling up and down on specific dimensions.

In *XML-OLAP* proposed by [10], it is assumed that both fact and dimension data are all represented in an XML document. *XML-OLAP* uses Text Mining operations in aggregating text contents of XML documents. In *XML-OLAP*, a text cube is returned as a query result, where each cell has a textual content such as top keywords, a summary, a set of classes, and a set of clusters.

[11] proposes data representations and algebraic operations for integrating semantic information (e.g., ontology's) into OLAP systems, which allow the authors to analyze a huge set of textual documents with their underlying semantic information. When analyzing unstructured information in a multidimensional data model, a document would be typically represented as a fact, and categories of keywords, such as protein, gene, or disease in the life science domain, would be selected as axis for the interactive analysis. It is necessary to point out that the ontology's mentioned before are external to the processed data and obtained before. This can lead the users to have no answer (no data) to their queries.

Few works deal with the *multidimensional analysis* of documents. These approaches often use knowledge models such as ontologies.

[12] proposes two aggregation functions: *AVG KW* takes as input a set of keywords extracted from documents of a corpus and returns as outputs another set of aggregated keywords; and *TOP-Keywords* aggregates keywords extracted from a corpus. They compute the frequencies of terms using the tf.idf function, then they select the first *k-most-frequent* terms. They assumed that both ontology and corpus belong to the same domain.

[7] proposes an aggregation operator *Orank (OLAP rank)* that aggregates a set of documents by ranking them in a descending order using a vector space representation.

In order to conclude this related work review, we can emphasize the fact that, to our knowledge, there is no proposal integrating a multidimensional modeling based at the same time on structures of XML documents and on their contents (from a semantic point of view). In consequence, only some parts of documents are managed in existing multidimensional models. Moreover the building of these models needs burdensome human interventions (business experts for the semantics and specialists in data management).

Thus is response our goal is to provide an OLAP framework adapted for the analysis of complete XML document contents. We introduce in [13] a new multidimensional model called *Diamond Model*, allowing us to manage both document structures and the semantics of the textual content. This model includes *standard* dimensions with factual data (such as date, author, publisher), as well as a dimension for the semantics of the textual fragment contents (such as summary, content, paragraph). Our semantic dimension deduced from one hierarchy of concepts. It is extracted from a semantic resource; this knowledge structure is obtained automatically and retains the semantics of the texts.

The *Diamond Model* allows flexibility in the specification of multidimensional analyses by not restraining the decision maker with predefined analysis subjects.

We suggest new aggregation operators. Our operators do not need the specification of the first *k-most-frequent* terms in advance. It's based on a semantic resource. They will facilitate the results interpretation of multidimensional analyses on the textual data, or at the level of documents or classes of documents.

Given a set of XML documents, the main objective of this paper is the automatic generation of a *Diamond Model* in order to facilitate the task of the designer. To do so, we propose a new process to construct *Diamond multidimensional models*. We also provide new aggregation operators to aggregate the documents during the analysis process.

III. Overview of the approach

Before introducing the different steps of our process, we describe the structure of the document collection (scientific articles) that will serve as an example throughout the article. The logical structure[1] (DTD or XSchema) of any article description called *Article* represented as a tree*,* (cf. FIG. 1) and an XML document that conforms to the logical structure *Article* (cf. FIG. 2). This *Article* is composed of one *Title*, one or more *Author(s)*, it unfolds in one *Conference*. It comprises one or more *Section(s)*, and one *Summary*.

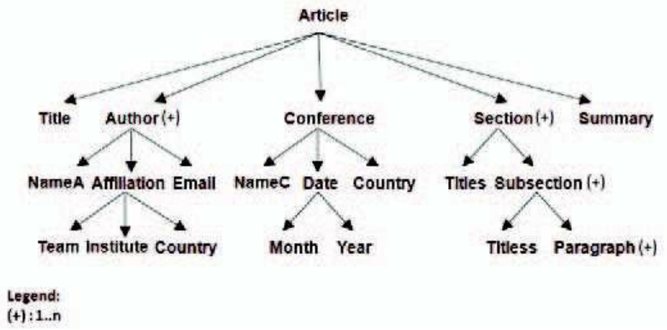


Fig. 1. *Article* DTD represented as a tree.

```
<? xml version="1.0" encoding="UTF-8"?>
<! DOCTYPE Article SYSTEM "Article.dtd">
<Article>
 <Title>XClust: Clustering XML Schemas for Effective Integration </Title>
<Author>
<NameA>Mong Li Lee</NameA>
<Affiliation>
<Team>School of Computing</Team>
<Institute>National University of Singapore 3 </Institute>
<Country>Singapore</Country>
</Affiliation>
<Email>Mong Li Lee@yahoo.com</Email>
</Author>
...
<Conference>
<NameC> Conference on Information and Knowledge Management   </NameC>
<Date>
<Month>05</Month>
<Year>2002</Year>
</Date>
<Country>USA</Country>
</Conference>
<Section>
<Titles>MODELING DTD</Titles>
<Subsection>
<Titless>DTD Trees</Titless>
<Paragraph>DTDs consist of elements and attributes. Elements…</Paragraph>
…
</Subsection>
...
</Section>
<Summary>It is increasingly important to develop scalable integration techniques...</Summary>
</Article>
```

FIG. 2. An XML document that conforms to the DTD *Article*.

This *diamond multidimensional model* consists of three layers (cf.Fig.9):

- *Standard layer* composed of *Standard* dimensions that represent the axes of analysis; these axes are constituted by the elements of the first level of the documents' structure. For each element, its descendants constitute the parameters (organized into hierarchies) or the weak attributes (affected to a parameter);
- *Semantic layer* composed of *Semantic* dimension which is a central dimension; its role is to add semantics to the textual content of documents. It is composed of the following hierarchy: *Concept* → *Semantic resource*. *Concept* parameter is connected to textual elements (like *Section*, *Paragraph*) of documents. A *semantic resource* may be ontology, taxonomy, thesaurus or any other kind of resource

[1] A logical structure is a tree of tags representing an XML document.

which can be filled and validated owing to semantic web tools and sources.

- *Document layer* concerns the document dimension, it is composed of the list of documents having identical or similar structures as well as the associated metadata, such as *Author, Date of creation,* and *Description*;

Note that the fact has been replaced by an element connecting compatible dimensions and it will be determined at the moment of interrogation. This principle has been proposed in Galaxy model [12].

In this section, we describe our process to construct *diamond multidimensional models* from several collections of XML documents conform to a logical structure (one diamond model per collection). This process involves the following three phases:

1) *Pretreatment*: it improves the conceptual legibility of the logical structure by enriching it with elements-attributes describing the type of links.
2) *Generation of a diamond multidimensional model:* we propose a set of heuristic rules to generate a *Diamond Model* in quasi-automatic way. Moreover, the Model must include all elements of the DTD or XSchema of the considered collection of documents.
3) *Instantiation of the multidimensional model:* This consists in automatically completing the various components of the *diamond multidimensional model* (fact, dimensions, parameters) from the XML documents.

Figure 3 illustrates the sequencing of these three phases of the *Diamond Model* construction.

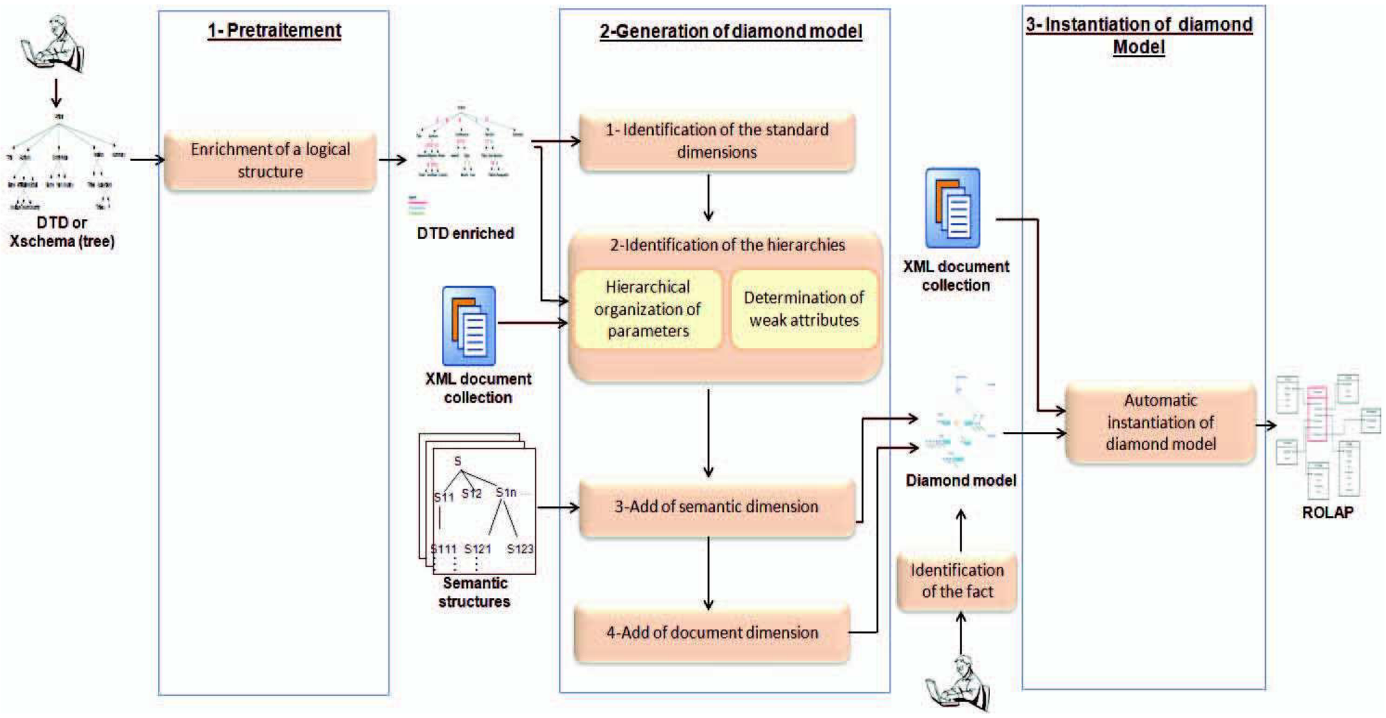

1- Pretraitement
2-Generation of diamond model
3- Instantiation of diamond Model
DTD or Xschema (tree)
Enrichment of a logical structure
DTD enriched
1- Identification of the standard dimensions
2-Identification of the hierarchies
Hierarchical organization of parameters
Determination of weak attributes
XML document collection
S
S11
S12
S1n
S111
S121
S123
Semantic structures
3-Add of semantic dimension
4-Add of document dimension
XML document collection
Diamond model
Identification of the fact
Automatic instantiation of diamond model
ROLAP

Fig. 3. Diamond multidimensional model construction process.

The pretreatment phase aims to enhance the semantics of logical structure. For that, we propose to annotate the links between the elements. We distinguish three types of links:

- *Descriptive:* the parent element is described through descendant elements, as an example, the element *Article* is described by elements *Title* and *Author(s).*
- *Structural:* the parent element is composed of descendant elements; for example, the element *Article* is organized in *Section(s)* and each *Section* is composed of *Sub-section(s).*
- *Temporal:* present all the component elements describing a date (e.g. day, month).

These elements-attributes must be carefully defined by the designer, by consulting an XML document sample, since they influence the quality of generation of a *diamond multidimensional model*.

An example of enriched DTD for *Article* DTD depicted in Figure 4.

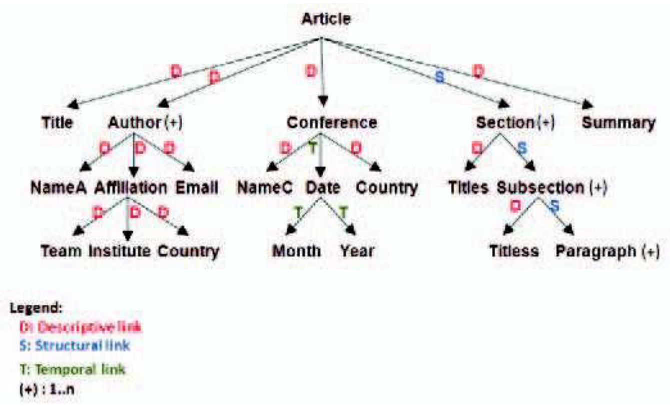


Fig. 4. *Article* DTD enriched.

In the next section, we focus on the second phase "Generation of a *diamond model*"; it is the most important phase of our approach.

IV. Diamond multidimensional model: Generation

This generation step transforms each enriched tree into a *diamond multidimensional model* in a quasi-automatic way, i.e., a way where the user's intervention is limited to the verification, rectification and approval of the generated model. Moreover, the model must include all elements of the DTD or XSchema of the considered documents collection.

A. Identification of the Standard dimensions

The heuristic rules proposed to identify the model dimensions are described in this section as follows:

Rule 1: The root *A* of a logical structure (DTD or XSchema) containing at least one leaf constitutes an element linking future dimensions (yet to be identified).

Rule 2: Each element *d* immediate descendant of the root element *A* of the logical structure becomes a dimension named *D-d*.

Rule 3: At each dimension *D* will be affected an artificial identifier (surrogate key) named *Id-D*.

Example 1: Let us consider the *DTD Article* (cf. Figure 1).

According to Rules 1, 2 and 3: The first descendants of the root *Article* become the following dimensions; *D-Title*, *D-Author*, *D-Conference*, *D-Section*, *D-Summary*, with the corresponding identifiers *Id-Title*, *Id-Author*, *Id-Conference*, *Id-Section*, *Id-Summary*.

- Case of element without leaf

Rule 4: For each element transformed into a dimension *D* that has no descendant, we shall have an attribute (parameter or weak attribute) having the same name as the dimension *D*:

- If the element contains distinct values then the attribute becomes a weak attribute, directly connected to the identifier of D (*Id-D*).
- Otherwise, it becomes a parameter of rank 2.

The intuition behind this rule is that an element having distinct values cannot be used as an aggregation criterion and, therefore could not be elected as a parameter.

Example 2: Let us take a part of the DTD Article (cf. Figure 1). According to *Rule 4*, *Title* element contains distinct values (two different articles cannot have the same title); therefore the *D-Title* dimension will have *Title* as weak attribute directly connected to the *Id-Title* identifier. The same logic applies for *Summary* element.

B. Identification of the hierarchies

In order to identify hierarchies, we have to determine the Functional Dependencies (*FD*) between the elements of the document structure. In database, a *FD* from the attribute *A* to the attribute *B*, noted *A*→*B*, expresses that each value of *A* is associated to one, and only one, value of *B*. This choice can be explained by the fact that the element *A* can play the role of parameter for a specific level and the element *B* can be a parameter of a more generic level. (Example: City→ Country).

Our objective now is to extract the parameters of rank greater than 1.

The heuristic rules are grouped into three categories; depending on the type of link added in the phase of pretreatment.

1) Rules for descriptive links

Let *N* be an element of rank *p* and *A*={a_1, a_2…a_k} immediate descendants of the element *N.*

- Hierarchical organization of parameters

For each element transformed into a dimension *D* that has descendant, we define four rules to identify these descendants belonging to the same level of the logical structure.

Rule 5: No FD between elements
If there is no FD from a_i, with i ϵ {1..k}, among each of the other descendants, then a_i becomes a parameter of rank(a_i)=p +1 for the dimension *D*.

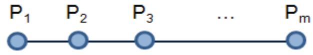
Example 3: Let us take the *Conference* sub-tree in the *DTD Article.*

Rule 5: No FD between *Name* and all descendants of *Conference* (*Date*, *Country*). *Name* is then a parameter of rank 2 connected to *Id-Conference*. The same logic applies for *Country*.

Rule 6: Non-symmetrical FD
If there is a non-symmetrical FD $a_i \to a_j$ (i.e. without $a_j \to a_i$) with *i,j* ϵ{1..k} such as i≠j then:
- If a_i is already a parameter then a_j becomes a parameter connected to a_i.

- Otherwise, a_i and a_j become two consecutive parameters for *D,* a_i is connected to the element ($rank(a_i) = p + 1$).

Rule 7: Elimination of transitive hierarchies
Let us have the following hierarchy H:



and the FD $P_i \rightarrow P_j$ with i<j≤ m then the FD $P_i \rightarrow P_j$ has to be eliminated in order to remove transitive hierarchy.

Example 4: Let us consider the *Author* sub-tree in the *DTD Article* (cf. Figure 1).

Rule 6 determines the existence of three non-symmetrical FDs *Team→Institute*; *Team→Country* and *Institute→Country*. Applying *Rule 7* allows us to determine the following hierarchy: *Team→Institute→Country* (eliminate *Team→Country*).

- Determination of weak attributes

Some dimension parameters can be characterized by so-called “weak attributes”. A weak attribute is a descriptive attribute that gives more meaning to a parameter; it is recommended especially when the parameter values are artificial data (as a Client identifier), and facilitates OLAP results understanding or interpretation.

Rule 8: symmetrical FD
If there are two symmetrical FDs $a_i \rightarrow a_j$ and $a_j \rightarrow a_i$ with i,j ϵ{1..k} such as $i{\neq}j$ then:

- If a_i containing null values, then we consider a_j as a parameter associated to the element-parameter and a_i becomes its weak attribute.
- Otherwise, a_i and a_j are considered as two parameters of the dimension *D* associated to the element-parameter.

Example 5: let us continue with the sub-tree *Author* (cf. Figure 1).

Rule 8 denotes the existence of two symmetrical FDs *Name* → *Email* and *Email* → *Name*; *Name* and *Email* do not contain null values; then we consider *Name* and *Email* as two parameters of *D-Author*.

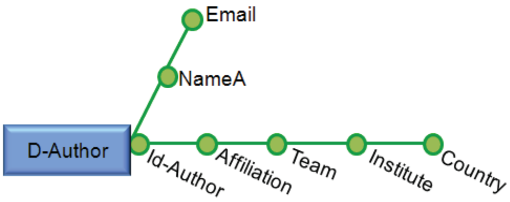


Fig. 5. Dimension D-Author.

2) Rules for structural links

In order to identify hierarchies with structural links, first we have to apply rules of descriptive links; we find that the order of hierarchy elements is erroneous. To correct this structural hierarchy, we reverse the order of the structural hierarchy elements, through the application of the hierarchical structural rule.

Let *p* be the depth of the structural hierarchy and *n* is the rank of a parameter.

Rule 9: For parameters derived from elements linked by structural relationships, we reverse the order of hierarchy. A parameter of rank *n* becomes a parameter of rank (*p-n*) +2. Where *p* is the depth of the structural hierarchy.

Example 6: Let us take the *Section* sub-tree in the *DTD Article.*

- First, we apply rules for descriptive links.

The application of these rules generates the following hierarchy.

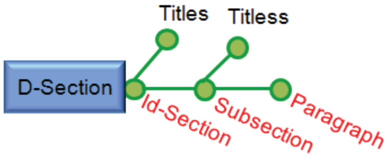


Fig. 6. Incorrect dimension D-Section.

We find that, the parameter “*Paragraph*” should constitute the parameter for a specific level, but it is generated at the end of the hierarchy as a generic parameter.

- In order to rectify this structural hierarchy, we reverse the order of the elements by applying the hierarchical structural rule.

Let us consider the parameter “*Paragraph*”: *p* = 3 (the depth of the structural hierarchy) and *n* = 3 (the rank of “*Paragraph*”).

According to *Rule 9*, the rank of “*Paragraph*” becomes

(*p* –*n*) +2= (3-3) + 2 = 2.

The same logic is applied for the element “Sub-Section”.

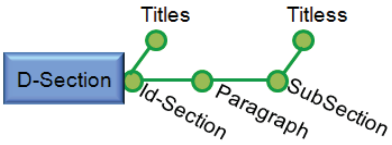


Fig. 7. Correct Dimension D-Section.

3) *Rules for temporal links*

The following rule deals with elements that describe *temporal* components.

Rule 10: Temporal rule
The set of elements of a same level that describe *temporal* components (e.g., month and day that compose a date) constitutes a temporal dimension (where each element component becomes a parameter in the hierarchy) or temporal hierarchy.

Example 7: Let us continue with the sub-tree *Conference* of the *DTD Article*.

The application of temporal rule *Rule 10* generates the following hierarchy.

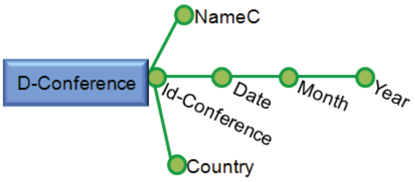


Fig. 8. Dimension D-Conference.

Note that the rules we have presented here deal only with the automatic generation of the model (standard dimensions) from the logical structure. After that we need to add the *Document* dimension which gathers information about the different versions of each document and the *Semantic* dimension which reflects the semantics of the unstructured textual elements. These dimensions are introduced in the next section.

C. *Document and Semantic dimension*

- Document dimension

Once created, documents are rarely static in time. They can be marked by an evolution in content or structure constituting several versions of the same document. These versions can be considered as different views of the same document. At this level, we add a document dimension called *D-Document,* consisting of an identifier *Id-Document*, the document Contents and two hierarchies:

One temporal hierarchy related to the creation date of the document. This hierarchy is organized as follows: *CreationDate →Month →Year*;

One hierarchy describing a set of metadata (physical document *Name*, extension *Type*, *Author*, summary *Description* of the document...).

It should be noted that each version of the document will be linked with a semantic resource [14] (cf. figure 9).

- Semantic dimension

The determination of the semantic dimension for a document is evaluated according to the approach described in [14] where authors uses a taxonomy as semantic resource: i) Extraction of significant terms from leaf elements of the document (leaves of the tree structure of the DTD or XSchema), ii) Choice of a taxonomy describing the semantics of a document, iii) Associate concepts of the selected taxonomy to the leaf elements of the document (concepts that best reflect the semantics of the terms describing leaf elements), and iv) Inference of concepts for non-leaf elements.

The Semantic dimension in the *Diamond Model* is represented by linking the parameters of the standard dimensions, rich in text, with the Concept parameter of the *Semantic Dimension*. For example, the *Titles* attribute of the *D-Section* dimension will be connected to the *Concept* parameter. Thus, we can do the analysis by the title of section or by related concepts (for example, “OLAP”, “Data Warehouse” or “OLTP” concepts).

After that, we need to identify the fact, it will be determined at the moment of interrogation (each dimension can play the role of fact, its parameters are transformed into measures).

The *diamond multidimensional model* of the logical structure *Article* (cf. Figure 1) is shown in Figure 9.

The transformation of our *Diamond Model* into a logical model is depicted in the following section.

V. Logical modeling: ROLAP

In the literature, several logical models for OLAP systems have been proposed: ROLAP (Relational OLAP), MOLAP (Multidimensional OLAP) and HOLAP (Hybrid OLAP).

We note that ROLAP systems are the most used because they are associated to the relational model which is well known by software designers. Although there are various types of R-OLAP models, we decided to detail the denormalized R-OLAP model.

The *diamond multidimensional model* is derived by applying a set of transformation rules:

- Every dimension *D* is transformed into a relational table composed a set of attributes that represent the parameters and the weak attributes of the hierarchies of *D*. The primary key of *D* is defined by the dimension identifier.
- The fact is transformed into a relational table composed by the foreign keys referencing the dimensions connected to fact. The primary key of the fact is obtained by the concatenation of attributes.

The textual elements (like Section, Paragraph) are assigned to the *Concept parameter* of the *Semantic Dimension* in order to add the semantics for the textual elements in an OLAP analysis.

The transformation of our *Diamond Model* into a logical model is depicted in Figure 10.

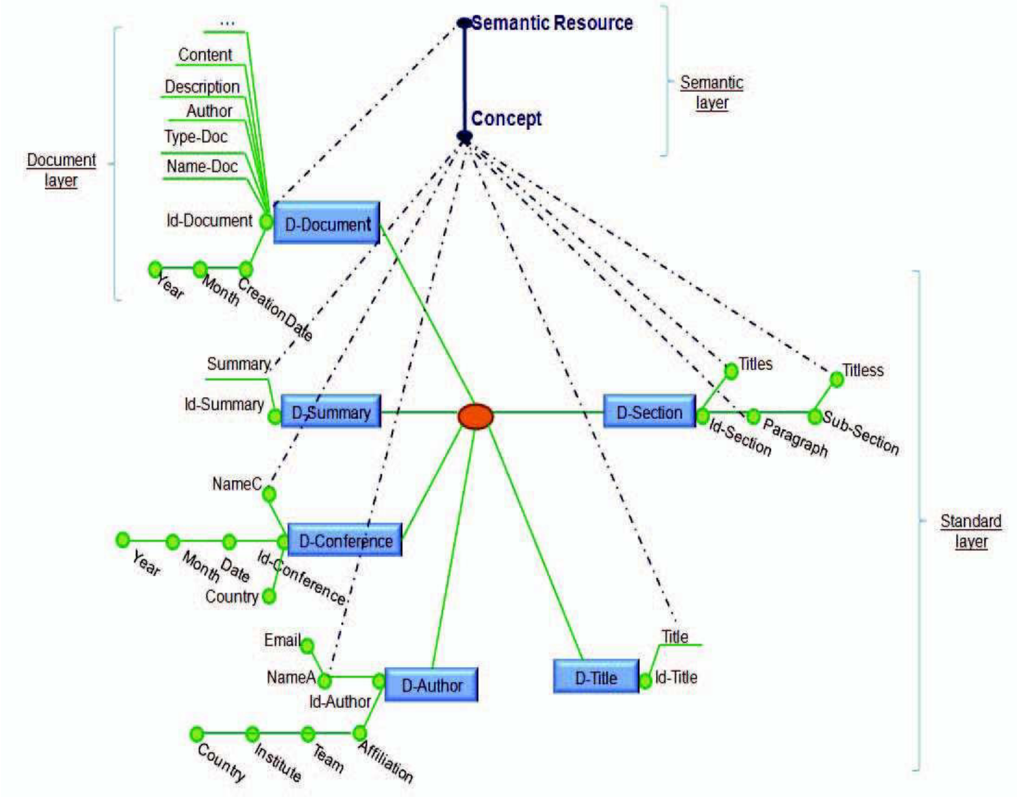


Fig. 9. Diamond multidimensional model for the Article collection.

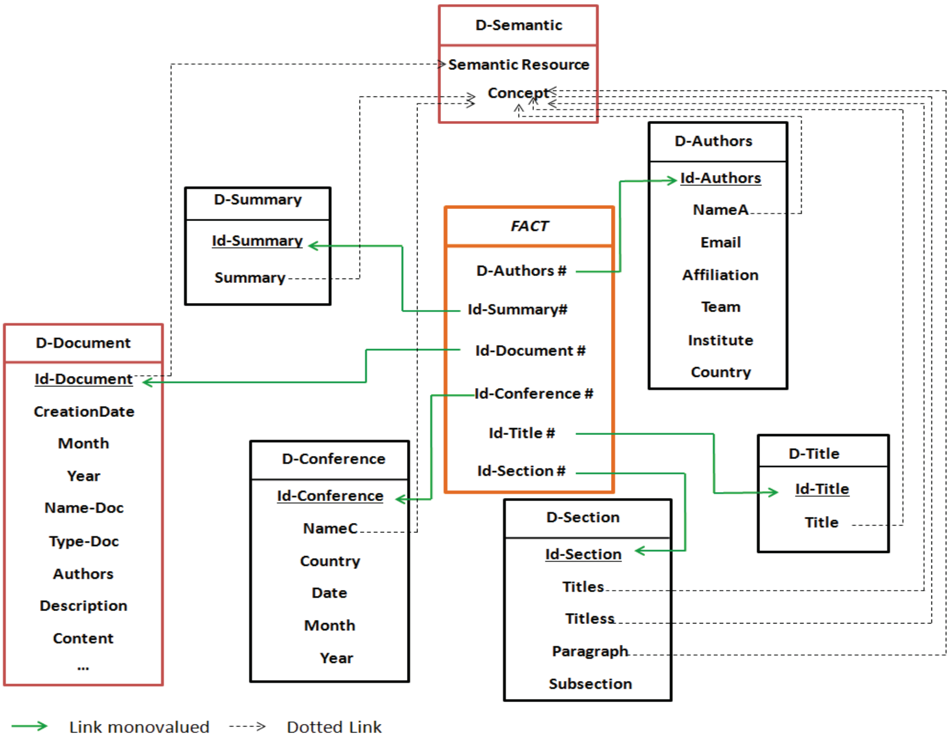


Fig. 10. Logical model (denormalized R-OLAP model).

VI. AGGREGATION OPERATORS

Our goal is to provide an OLAP framework adapted for the analysis of XML documents. This requires a new approach for aggregating XML documents.

For example, to observe the conference activities, the decision maker analyzes the concepts of scientific articles according to their author(s) and year of publication.

The results are done through multidimensional tables (cf. Figure 11). Values are placed in cells c_{ij} that are at the intersection of i^{th} line and the j^{th} column. Suppose the c_{11} cell corresponding to (Dupond, 2010) displays *9* concepts. The cube tends to be overloaded because too much information is returned to it. For this, we propose new aggregation operators that take into account the specificities of textual data from documents:

- *List_Concept* : returns a list of the most used concepts from a set of concepts in order to aggregate them into the corresponding cell c_{ij} of the multidimensional table;
- *G_Concept*: extracts the most used generic concepts;
- *S_Concept* : extracts the most used specific concepts;
- *Top_Concept:* groups the operators *List_Concept*, *G_Concept* and *S_Concept* to display the first concept of each of these operators.

The aggregation of textual data allows summarizing the volume of the data to be visualized during an analysis. So by reducing the volume of the data, the user can have a more global vision of the domain he analyzes.

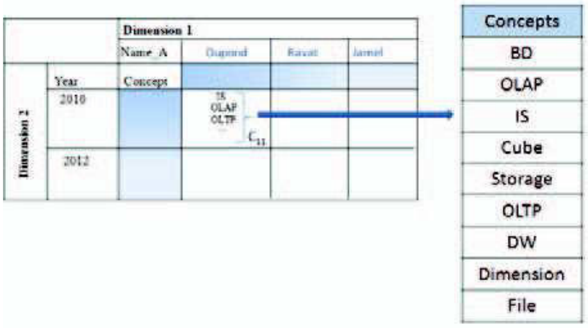


Fig. 11. Multidimensional analysis of titles concepts, by author and year.

A. *List_Concept Operator*

In order to summarize data from cells c_{ij}, there is a need for aggregating its textual data. We define an operator *List_Concept* that aggregates a set of *m* concepts in a list of *k* concepts the most used among *m* concepts. *C* represents all the displayed concepts.

A set of concepts is injected at the input to the operator *List_Concept*.

$$\text{\underline{List\_Concept:}}\ C^m \longrightarrow C^k$$
$$\{c_1,.., c_m\} \longmapsto <c_1,.., c_k>$$

- C^m is a list of m concepts,
- $C^k \subseteq C^m$ is an *ordered list* of the *k* most used concepts, such as the concepts are ordered according to levels *i* and *i+1* of a semantic resource.

Our operator *List_Concept* aggregates the semantic content of the documents, represented by concepts. Thus, such aggregation is realized owing to a semantic resource (ontology for example). We have given more importance to the father's concepts because they convey more generic and synthesized information.

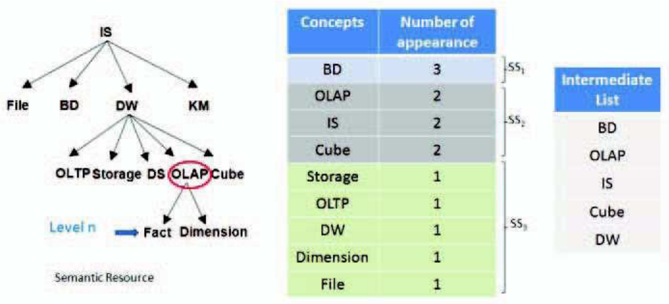
Our operator is based on three steps:

- Step1: Classify the concepts in decreasing order of their occurrences in order to group concepts into subsets.
- Step2: For each subset, browse the semantic resource from the lowest level.

For each concept $c \in C^m$:

- If its father_concept exists in the same subset or the intermediate list, its father in the intermediate list is retained.
- Otherwise the concept c is retained.

The application of these first two steps generates the intermediate list. (cf. Fig. 12).



Concepts	Number of appearance
BD	3
OLAP	2
IS	2
Cube	2
Storage	1
OLTP	1
DW	1
Dimension	1
File	1

Intermediate List
BD
OLAP
IS
Cube
DW

Fig. 12. Application of the first two steps.

For example, the father of the concept *OLAP* does not appear in the same subset or in the intermediate list thus *OLAP* will be retained (the same logic applies for *Cube*). On the other hand, the concept *Fact* will not be held because its father *(OLAP*) appears in the intermediate list.

- Step3: Retain the concepts of the first two levels. For this, we need to browse a semantic resource down from the top, in order to find one of the concepts of the intermediate list. When a concept appear, we fix the level of this concept as a level *i* and the next level *i+1*. Finally, we retain in a final list the concepts of the intermediate list that appear at level *i* and *i+1*.

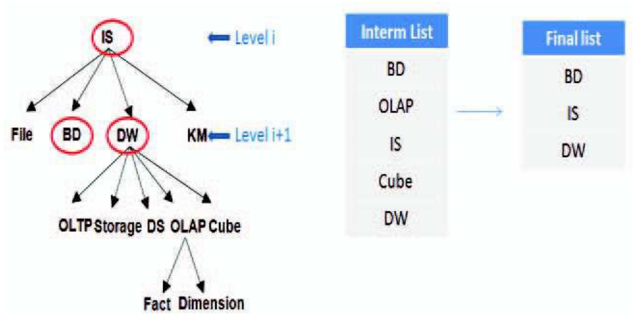


Fig. 13. Application of the third step.

In the end, the cell c_{11} contains the three most representative concepts of the set of concept aggregate. The same logic is applied for each cell of the multidimensional table.

B. *G_Concept Operator*

In order to generalized, we propose another aggregation operator *G_Concept* whose goal is to extract the most used generic concepts.

$$\underline{\text{G\_Concept}}: C^m \longrightarrow C^g$$
$$\{c_1,.., c_m\} \mapsto \{c^g_1,.., c^g_k\}$$

The input is a text represented through a set of *m* concepts and the output is a subset of C^g composed of the *k* most generic concepts.

Then, for each concept, the semantic resource is browsed from the lowest level.

- If its father_concept exists in cells c_{ij} or in final list, its father in the final list is kept.
- Otherwise the concept c is retained.

The application of *G_Concept* generates the final list. (cf. Fig. 14).

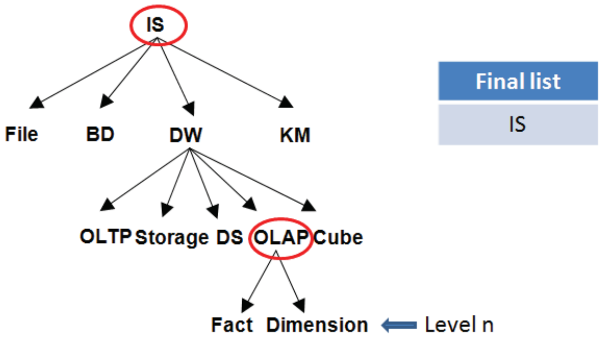


Fig. 14. Application of the operator G_Concept.

For example, for the concept *OLAP,* its father_concept appears in the cells c_{ij} (*IS*) thus *OLAP* will not be retained.

VII. Conclusions

In this paper, we proposed a new multidimensional model, called *Diamond Model*, dedicated to the design of textual data marts and On-Line Analytical Processing (OLAP) on XML documents according to their content and structure. This model mainly consists of three layers: *standard layer* (a set of *standard dimensions*) constructed from the structure of a set of documents, a *Document layer* and a *Semantic layer* (a *semantic dimension*). The main objective of the Semantic Dimension is to switch from the simple text to a semantic level.

We also proposed a specific approach to construct a *Diamond Model* starting from the XML structure (DTD or XSchema) of a collection of documents to be analyzed. Our approach is structured in three phases: *Pretreatment*, *Generation,* and *Instantiation*.

We described in this paper the first two phases: 1) *Pretreatment:* to enrich the DTD or XSchema by the addition of link type. 2) *Generation of a diamond multidimensional model:* generate a *Diamond Model* by applying different heuristic rules.

When the generation of the *Diamond Model* is achieved, the designer (assisted by the decision-maker) verifies and validates the obtained *diamond multidimensional model*. He can rename, delete the multidimensional elements and the links between dimensions, or reorganize the parameters and so on.

To illustrate our generation approach, we have applied the phases and rules on a collection of *Article* descriptions (Article are described by the DTD *Article* of Figure 1) to obtain the *Diamond Model* shown in Figure 9. We also provided aggregation operators based on semantic resource. They will facilitate the interpretation of the results of the multidimensional analyses on the textual data, or at the documents level or documents classes (according to the analysis aggregates).

Several perspectives for this work are possible. It would be interesting to define an instantiation process of these diamond models from the documents contents. It might also be useful to visualize analyses' results in cubes' form or multidimensional tables. We also intend to exploit text mining techniques to extract knowledge from documents so as to enhance the semantic dimension in the Diamond Model. Finally, we plan to evaluate the scalability of our approach.

References

[1] F. S. C. Tseng, and W.-P. Lin., "D-Tree: a multidimensional indexing structure for constructing document warehouses", Journal of Information Science and Engineering, vol. 22, pp. 819-841,2006.

[2] C. X. Lin, B. Ding, J. Han, F. Zhu, and B. Zhao, "Text cube: Computing in measures for multidimensional text database analysis". Eighth IEEE International Conference on Data Mining 54, pp 905–910, 2008.

[3] B. Janet, and A.V. Reddy, "Cube Index for Unstructured Text Analysis and Mining". ICCC'11 Proceedings of the 2011 International

Conference on Communication, Computing & Security. pp 397-402, 2011.

[4] J. Mothe, B. Chrisment, C. Dousset, and J. Alaux, “DocCube: Multi-dimensional visualisation and exploration of large document sets”. Journal of the American Society for Information Science and Technology, 54, pp 650–659, 2003.

[5] D. Zhang, C. Zhai, and J. Han.D., “Topic cube: Topic modeling for olap on multidimensional text databases”. SDM ’09: Proceedings of the 2009 SIAM International Conference on Data Mining, Sparks, NV, USA, pp 1124–1135, 2009.

[6] Y. Yu, C. Lin, Y. Sun, C. Chen, J. Han, B. Liao, T. Wu, C. Zhai, D. Zhang, and B. Zhao, “iNextCube: Information network-enhanced text cube”. VLDB '09: Proceedings of the 35th International Conference on very Large Data Bases, Lyon, France, 2009.

[7] L. Oukid, O. Asfari, F. Bentayeb, N. Benblidia, and O.Boussaid, “CXT-cube: contextual text cube model and aggregation operator for text OLAP”, DOLAP, San Francisco USA, 2013.

[8] M. Bautista, C. Molina, E. Tejeda, and A. Vila, “A new multidimensional model with text dimensions: definition and implementation”. International Conference, IPMU, Dortmund, Germany, pp 158–167, 2013.

[9] J. Lee, D. Grossman, and R. Orlandic, “MIRE: A multidimensional information retrieval engine for structured data and text”. Proceedings of the International Conference on Information Technology: Coding and Computing, Las Vegas, NV, USA, pp 224-229, 2002.

[10] B-K. Park, H.Han, and I. Y. Song, “XML-OLAP: A multidimensional analysis framework for XML warehouses”. DaWaK’05: Proceedings of Data Warehousing and Knowledge Discovery, LNCS 3589, Springer, pp 32-42, 2005.

[11] A. Inokuchi and K. Takeda, A method for online analytical processing of text data, in Proceedings of the sixteenth Association Computing Machinery (ACM) conference on information and knowledge Management (CIKM ’07). New York, NY, USA: Association Computing Machinery (ACM) pp 455–464,2005.

[12] F. Ravat, O. Teste, R. Tournier, and G. Zuruh. “Top keyword: an aggregation function for textual document olap”. Intl Journal of data Warehousing and Mining, pp 55-64, 2008.

[13] M. Azabou, K. Khrouf, J. Feki , and C. Soulé-Dupuy ,and N. Vallès, “A Novel Multidimensional Model for the OLAP on Documents: Modeling, Generation and Implementation”. International Conference on Model & Data Engineering MEDI’2014, 24-26 Septembre, 2014, Larnaca, Cyprus, pp 258-272, 2014.

[14] S. Ben Mefteh, K. Khrouf, J. Feki, and C. Soulé-Dupuy, “Semantic Structure for XML Documents: Structuring and pruning”. Journal of Information Organization, Volume 3, Number 1, p. 36-46, March, 2013.