A Requirement Engineering Approach for Designing XML-View Driven, XML Document Warehouses

Vicky Nassis[1], Rajugan, R.[2], Tharam S Dillon[2] and Wenny Rahayu[1]
[1] *Department of CS-CE, La Trobe University, Melbourne, Australia*
vnassis@students.latrobe.edu.au, w.rahayu@latrobe.edu.au
[2] *eXel Lab, Faculty of Information Technology, University of Technology, Sydney, Australia*
{tharam, rajugan}@it.uts.edu.au

Abstract

The EXtensible Markup Language (XML) has emerged as the dominant standard in describing and exchanging data among heterogeneous data sources. The increasing presence of large volumes of data appearing in enterprise settings creates the need to investigate XML Document Warehouses (XDW) as a means of handling and analysing XML data for business intelligence. In our previous work, we proposed a conceptual modelling approach for the design and development of XDWs, with emphasis on capturing data warehouse requirements early in the design stage. To address this issue, in this paper, we explore a Requirement Engineering (RE) approach, namely the Goal-Oriented approach. We adopt and extend the notion of this approach and introduce the XDW Requirement Model. This focuses on deriving dimensions, as opposed to associating organizational objectives to the system functions, which is carried out by the traditional requirement engineering process.

Keywords: Data/Document Warehouse, OO Conceptual Model, XML, XML Views, UML, Requirement Engineering, Requirement Definition.

1. Introduction

Enterprise Content Management (ECM) is the integration and utilization of one or more technologies, tools, and methods to capture, manage, store, preserve, and deliver content across an enterprise [1]. Since its introduction in 1996, eXtensible Markup Language (XML) [2] has become the *defacto* standard for storing and manipulating self-describing information (meta-data), which creates vocabularies to assist with information exchange between heterogeneous enterprise data sources over the web [3]. With enterprise content (EC) moving rapidly towards web based, e-Commerce/ e-Business and information systems, XML contents add complexity to the ECM equation.

At the most basic level, data warehousing is an approach adopted for management of large volumes of historical data for detailed analysis and to provide crucial business intelligence (BI) to enterprises [4]. With heterogeneous EC and XML, there is considerable work to be achieved in order to allow electronic document handling, electronic storage, retrieval and exchange. It is also possible to logically encode documents using XML for multi-domain enterprises. Hence it is likely that a large number of XML documents will populate the would-be repository and several disparate transactional databases.

The need for managing large amounts of XML document data raises the necessity to explore the data warehouse approach through the use of XML document marts and XDWs. Our purpose is to capture and represent native XML type semantics. The discussion of existing work outlined in the following section on dimensional modelling of data warehouses, indicates that the issue is not completely addressed. In relation to Requirement Engineering (RE) the elicitation (discovery) of requirements and their use to design the data warehouse is a significant and as yet an undealt with issue, hence in this paper we focus on this aspect.

As an example to illustrate our XDW Requirement Model introduced in this paper, we investigate a possible XDW of a simple *Conference Publishing System (CPS)*, for managing and distributing conference proceedings of various international conferences held in different cities throughout the year. The main component of a conference

Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05)
0730-3157/05 $20.00 © 2005 **IEEE**

publication is comprised of a collection of papers (past and present), stored in various geographically distributed conference databases/systems, in varying proceedings format such as ACM, LNCS, IEEE. The system is similar to that of existing systems such as ACM Portal [5], SpringerLink [6] or IEEE Xplore® [7]. Logically, we treat all the different conferences and their proceedings as one big (logical) conference proceeding on the web (similar to the concept of a "global view" in enterprise systems).

1.1 Related Work

Since the introduction of dimensional modelling which, revolves around facts and dimensions, several design techniques have been proposed to capture multidimensional data (MD) at the conceptual level. These include Ralph Kimball's Star Schema [4] and derived extensions of this model (SnowFlake, StarFlake) where facts and dimensions are connected in a certain way. Also Object-Oriented (OO) approaches where data is handled in the structure of n-dimensional cubes. [8-10]. The Object-Relational Star schema (O-R Star) model [11] is used to provide support for the representation of hierarchical relationship types along a dimension which are inheritance and aggregation.

These models, both object and relational, have limitations if one wishes to use them for XML document warehouses. The two major reasons are: (a) they lack the ability to utilise or represent XML design level constructs in a well-defined abstract and implementation-independent form, and (b) there is insufficient emphasis or capturing requirements early in the design stage.

The concept of RE explores the objectives of different stakeholders and the activities carried out by the system. Based on numerous studies [12] [13] it is stated that a system's functionality fails due to insufficient understanding of user requirements, hence RE is concerned with overcoming this inadequacy. Nowadays it is becoming evident that organisational change is rapid hence the reflection on the evolution of user requirements. An unresolved issue is actually understanding and recording the impact of business changes on user requirements [13].

Existing work and literature in RE regarding system objectives and system functionalities has raised the incentive to explore one of its existing approaches, namely the *goal-oriented approach* as illustrated in [14]. It is important to recognise that the goal-oriented approach has been largely targeted at the development of software systems rather than focused on document data warehouses involving embedded XML structures. Given this we extract and extend the notion of this goal modelling approach to an XDW, particularly in deriving requirements. This would then help in guiding the formation of the structural design by identifying data warehouse dimensions as well as the nature of the document fact repository.

1.2. Contribution of the Paper

In our previous work [15,16], we proposed a model and a conceptual design methodology of an XDW model using XML-views [17], Unlike other data warehouse models, we regarded capturing of the data warehouse requirements early in the design stage highly important and a must for a well-defined warehouse model [18].

Our proposed XDW Requirement Model focuses on capturing and eliciting requirements by taking into consideration organisational objectives as well as user viewpoints. Furthermore these are related to the XDW particularly focussing on deriving dimensions, as opposed to associating organisational objectives to the system functions, which is traditionally carried out in RE. The key issue is the principal of correspondence between the real world representation and its domain. This involves the mapping of real world entities to clearly define the corresponding entities in the system, which tends to facilitate a system's evaluation. We employ a graphical representation for our XDW Requirement Model to initiate a dialogue between the requirement engineer and the domain expert. Our approach is distinctive as up to now there have been no attempts to capture requirements and the entirety of their semantic nature.

2. Brief Overview of Our XDW Model

The main aspects of our methodology [15,16] are as follows: (1) *Requirement Level (RL):* assist in determining different dimensions (perspectives) of the document warehouse, (2) *XML Document structure*: using XML document capability in accommodating and explicitly describing heterogeneous data along with their relationship semantics (unlike flat-relational data), (3) *XML Schema* [2]: describes, validates and provides semantics for its corresponding instance document (XML document). Also, it has the capability to capture all OO concepts and relationships as well as intuitive XML specific constructs such as ordering

Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC'05)
0730-3157/05 $20.00 © 2005 IEEE

and homogeneous relationships of components, and (4) *Conceptual Views*: the role of conceptual views is to provide perspectives of the document hierarchy stored in the XML FACT (xFACT) document repository. A *context* is more than a measure but instead is an item that is of interest for the organization as a whole [19]. The xFACT is a snapshot of the underlying transactional system for a given *context.*

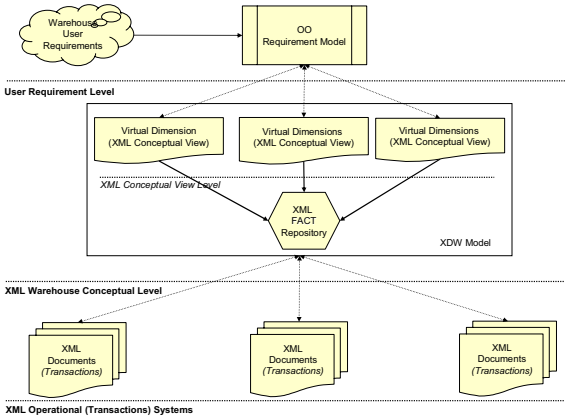

Figure 1: Our XDW Model (Context Diagram)

To our knowledge, this is unique in that it utilizes XML itself (together with XML Schema) to provide: structural constructs, metadata, validity and expressiveness (via refined granularity and class decompositions). The proposed model is composed of three levels**: (1) the Requirement Level (RL):** This includes: (a) *Warehouse Requirement Document*: corresponds to the non-technically written outline of the XDW document warehouse, and (b) *OO Requirement Model*: expresses all non-technical requirements into technical terms and software specific concepts using UML. **(2) XML Warehouse Conceptual Level:** Composed of an *xFACT Document Repository* and a collection of logically grouped *Conceptual Views* which translate as the document warehouse dimensions, corresponding to satisfied captured warehouse requirements. This logical grouping of *conceptual views* generates what is referred to as one or more *Virtual Dimension(s) (VDim)*. **(3) Logical Level:** Involves the transformation of the entire document warehouse conceptual model, which includes the xFACT document repository and virtual dimensions (VDims), into XML schema.

3. The XDW Requirement Model

We adopted the central notion of the *goal-oriented approach* as illustrated [14] and with additional components we built the XDW Requirement Model as shown in Fig. 2. Fig. 3 provides an interpretation of the graphical aspect of our model including the symbols and notations used. The main themes of this methodology are to: (1) capture requirements early in the design process, (2) understand the current requirements and further elicit these to promote new requirements (3) illustrate the approach of how each stated requirement is mapped to a corresponding dimension of the XDW, (4) ensure that the information necessary to construct a dimension is available or can be obtained from the xFACT document repository and (5) ensure the information for the requirements of the xFACT document repository can be assembled from the information available in the XDW repository.

The XDW Requirement model (Fig. 2) is composed of three main levels**: (1) Dimension Level:** this has two sub-levels namely the *High Dimension Level* and the *Aggregated Dimension Level*. **(2) Document Level:** includes all XML documents involved in a requirement's search query and, **(3) Constraint Level:** parameter values that change the presentation of a query's result according to the corresponding requirement's specification. What follows is a detailed discussion of each level.

3.1. Dimension Level

This level represents the different dimensions of the data warehouse. There are two sub-levels as shown in Fig. 2: the *High Dimension Level* and the *Aggregated Dimension Level.* A high level or an abstract requirement indicates its wide-ranging content type. This shows that the corresponding query's result is retrieved from the entire content of one or more XML document(s) as opposed to extracting specific XML document(s) elements. In some cases an abstract type requirement might not be sufficient to satisfy a user need and therefore, it becomes necessary to aggregate the high level requirement and form several sub-requirements. This leads to the creation of the second sub-level of our model namely the *Aggregated Dimension Level.* Commonly a requirement ranked at the high level of the hierarchy can be further decomposed to several sub-requirements. Each one of the elicited sub-requirements represents an additional dimension of the XDW, which is shown by the cardinality (1 : +)

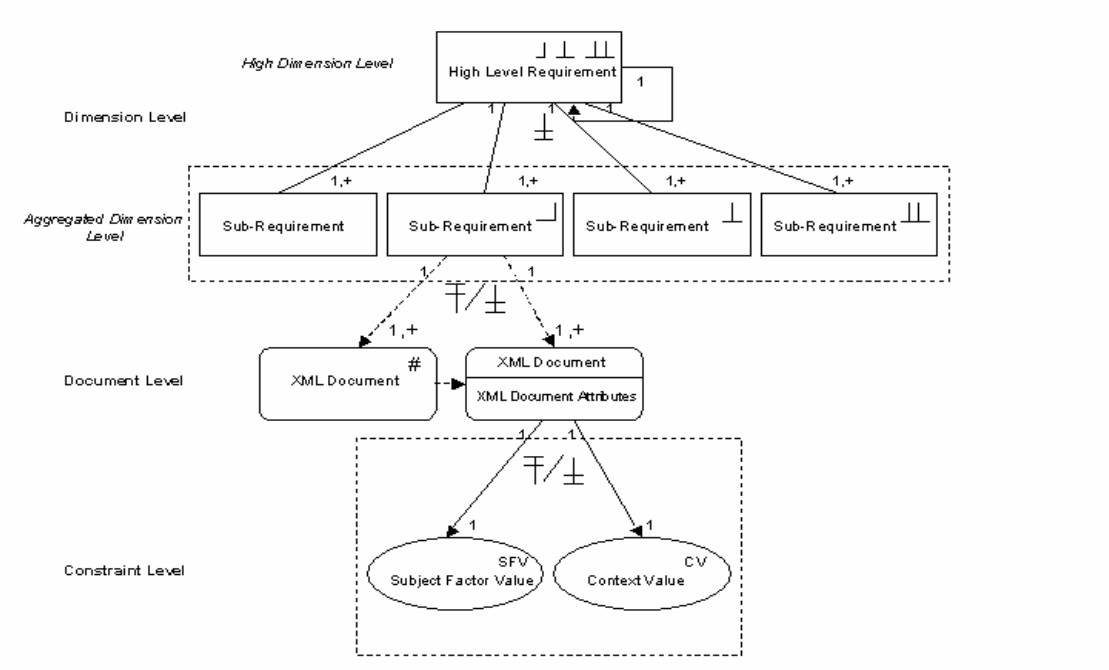

Figure 2: XDW Requirement Model

between the two components (Fig. 2). This, in other words means that, when a query requires specific attribute(s) value(s) and more than one XML documents, then it itself becomes very complex. Therefore decomposition helps in understanding the current requirement and defines new avenues for fulfilment. In our XDW Requirement model (Fig. 2) where the high level requirement has an arrow pointing to itself, depicts that a requirement does not necessarily need to be decomposed but instead can remain in its current form and considered as one of the possible dimensions. This case justifies that a direct link may exist from the *High Level Dimension* to the subsequent *Document Level.* This is shown in Fig. 2 where the *Aggregated Dimension* component is surrounded by a dashed line indicating that it is not valid in all cases.

3.2. Document Level

As discussed previously, a direct connection may exist amongst the *Document Level* and any of the two preceding levels, meaning that it can be immediately related either to: a requirement or a sub-requirement. Regarding the *Document Level*, the related XML document(s) to be queried depends entirely on the specifications of the requirement in question. A search query may require information that exists within one or more XML documents. It is important to note that there are two cases likely to occur in relation to the XML documents and a query outcome which are: (1) the entire XML document(s) attributes values are required and extracted and, (2) only specific attribute(s) value(s) from the document(s) is/are needed. In Fig. 2 at the *Document Level* there are two types of documents to conform to each one of these cases. In the first document type, the # symbol represents that the entire document(s) properties are required. The second document type is divided into two sections of which, the lower fragment is of significance as it shows the specific attribute(s) required to generate a query outcome. These two cases may be applied simultaneously as shown in Fig. 2 where the two document types are linked by a dashed arrow (dash indicates optional) pointing towards the direction of the independent/stronger component of the two. The classification of the stronger or weaker component is determined based on the developer's judgment in conjunction with the requirement specification.

3.3. Constraint Level

This level applies in two cases: Firstly, to a result that needs to be arranged in a specific order (i.e. alphabetically) and which is denoted by the **Subject Factor Value (SFV).** Secondly, to conduct a search and extract the necessary data based on a given value, which is indicated by the term **Context Value (CV).** These two parameter values have the ability to alter

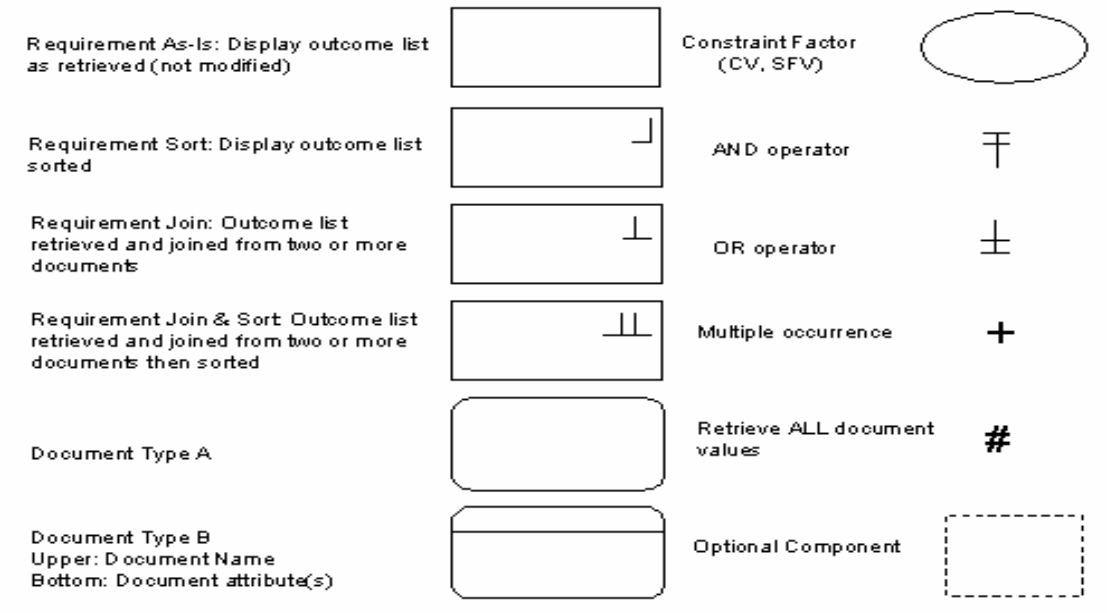

Figure 3: Interpretation of XDW Requirement Model

the presentation of a query's end result list. Depending on the query context, both cases can appear and be applied simultaneously as shown in Fig. 2. To define more broadly, a **CV** is the specified value within a document, which can take the form of: a precise word, phrase, number or a group of values. A **SFV** can include: a name, a number or it can be of any value provided that it exists within the document(s) attributes involved in the query.

3.4. Components of the XDW Requirement Model

In the previous section we explained the different levels of our proposed model. Before we proceed into the specific and detailed terminology it is important to first briefly state the main aspects that comprise the requirement model.

A component **Comp** in the XDW requirement model is a set of elements, which include: a requirement **R**, a sub-requirement **SR**, an XML document **D,** a constraint **C** and an attribute **A**. In other words, $\mathbf{Comp}_1 \ldots \mathbf{Comp}_N$ aids in the formulation of the entire requirement model. There can be relationships that exist amongst these components. A relationship **Rel,** links together each of the components in the model considering the relevant purpose and cardinality amongst them. A requirement **R** expresses what we aim to achieve, meaning the expectant outcome of the data warehouse. As discussed in the *Document Level* section, in order to fulfill the conditions of each stated requirement, the required XML document(s) **D** is/are queried to allow one to retrieve the information needed. The query structure depends entirely on the requirement's degree of complexity. This means that a query tends to get equally compound the more precise and demanding a requirement is. In some cases a requirement can be further decomposed to form several sub-requirements **SRs**. An XML document contains a set of elements where each individually has a set of attributes **A.** These are single valued attributes from the XML schema built-in data types (eg. string, integer). Finally the constraint **C** is a parameter of which its value can modify a query's result presentation according to the requirement's specification.

3.5. Concepts and Terminology of the XDW Requirement Model

This section provides an illustration of the previously outlined components of our XDW Requirement Model in greater detail through demonstrations and formal definitions.

We can now formalize the earlier definition of a requirement as follows:

Definition 1: Requirement = the formal definition terms of a requirement are:

$$R = (I)[(C)]^*$$

Where **I** = Intention and **C** = Constraint and the content within **[ ]** (square brackets) shows that it is optional whereas ***** (star) indicates that this component may occur multiple times.

Definition 2: Intention (I) = Phrase that states what is aimed to be achieved; the expectant outcome/end result from the data warehouse. It is expressed in the form of natural language and can be extracted from a text-based document. This will be illustrated in detail, as a sample requirement is formed in section 4.

Definition 3: Constraint (C) = A parameter used in order to vary the way a query's result is displayed considering the need to conform with the user's preference and requirement needs. The two major constraints, which will be used, are: *Context Value (CV)* and *Subject Factor Value (SFV),* which have been discussed in section 3.3. A *constraint* can also be expressed as:

$$C = (CV, SFV)$$

In the case where there is no **CV** (Context Value), meaning that the query search is conducted with no values being specified, the sign # is displayed within the document type component, as shown in the requirement model (Fig. 2), to indicate that the search will extract all the values within that concerned document.

Definition 4: Sub-Requirement = A high level requirement can often be decomposed to one or more sub requirements (**SR**). Therefore the sub-requirement set of system **S** is defined by the following function:

$$R : S \rightarrow SR$$

It is important to note that the principal definition of a **Requirement** stated previously, also applies when forming a Sub-Requirement.

Definition 5: Relationship = A **n**-ary relationship **Rel** amongst the components $\mathbf{Comp_1}$ … $\mathbf{Comp_N}$ existing within the XDW Requirement Model is defined as:

$$Rel = (R, SR) \mid (R, D, [C]) \mid (SR, D, [C])$$

The above denotes that each instance of **Rel** is equivalent to each of the above combination instances. For example a relationship can exist, between a requirement and a sub-requirement (**R, SR**), or amongst a requirement, document and constraint (**R, D, C**).

The cardinality of the relationship **Rel** instances among the component instances $\mathbf{Comp_N}$ includes the sets **(1:1)** and **(1:+)** which express the minimum and maximum amount of possible relationships respectively. In the latter set the **+** sign indicates a multiple connection with other components. For example a requirement may require data from multiple documents hence the corresponding query will involve a search across several XML documents.

Definition 6: AND/OR Relationships = Based on the AND/OR graph structures introduced by [20] we implement this concept into the XDW Requirement Model and note that it provides a different meaning to the components at each level. For instance the relationship between the *Requirement* and *Dimension levels* shows that a high level requirement can be further refined into several alternative combinations of sub-requirements where each one can be further aggregated to smaller requirements. A requirement/sub-requirement gathers its source of data through several combinations of XML documents of which, the outcome's presentation varies depending on the presence of constraint value(s). An AND/OR relationship **Rel** on the components $\mathbf{Comp_1}$ and $\mathbf{Comp_2}$ is defined as follows [14]:

$$Rel \in S$$

where $S = \{AndRel, OrRel\}$

The AND relationship is presented by the symbol $\mp$ and the OR relationship by the symbol $\pm$.

This means that any instance of **Rel** can take the form of an **AND** or **OR** type when linking two components. Referring to the XDW Requirement Model (Fig. 2), the **AndRel** signifies that a combination of sub-requirements as a whole can achieve a high-level requirement. Alternatively the **OrRel** shows different ways a requirement can be achieved. Similarly between the *Dimension* and *Document* levels, depending on the query the OR operator means that the query search can include either one of the document types. Lastly amongst the *Document* and *Constraint* levels, both of **CV** and **SFV** can be applied simultaneously or independently depending on the requirement's specification. In order to enable tracking of the alternatives selected, each constraint has an attribute with a true of false value, which must be updated every time if we wish to examine another alternative [14]. Having the

system context of our case study, we are able to formulate the necessary requirements by applying the terminology principle of a **Requirement**. A more detailed discussion on this section of the paper can be found in our referenced work [18]

4. Practical Walk-Through of the XDW Requirement Model

Based on the *CPS* example, we can begin to extract a one of the numerous possible requirements due to page limitations. What follows is an illustration of the stated requirement, including modelling and implementation on actual real-world requirement values. The proposed XDW Requirement Model (Fig. 2) is the foundation for building smaller requirement modelling segments to suit the structure of each likely to occur requirement ranging from straightforward to more complex types. This enables full capturing of the current requirements and encourages further elicitation of new requirements.

Before we progress to the actual implementation of the requirement it is important to note that, based on the general characteristics that emerge from various requirements, we are able to form five main categories that portray the nature of each likely to occur requirement specification.

(1) Retrieve values of an entire document.
(2) Retrieve values of specified document element(s).
(3) Retrieve values of a document based on a particular *context value*. The CV in some cases may exist in a different document, which has no direct link with the immediately related documents involved in the query.
(4) Sorting an already generated list of values based on a subject factor value.
(5) Retrieve values by merging two or more documents.

The different types of requirements featured above are identified solely but can be combined with one another particularly when a requirement's specification acquires a highly complex structure.

Example Requirement: *"List of conferences names sorted chronologically with cross-reference to authors' details."*

This requirement can be broken down further to simplify the process of gathering the required information. Firstly, the list of conferences names is a straightforward search, which evolves within the `Publ_Conference` document. It is then required for the list to be displayed chronologically, which indicates that the *`Subject Value Factor (SVF)`* applies in this case and takes the value "`Year`". A third component to this requirement is to also include cross-referencing from the document `Publ_Authors`. This indicates that the search is not bound to be only within documents with direct connections amongst them. Instead at times it may be necessary to go through several intra-connections with external documents such as `Publ_Authors` to obtain the required values. In Fig. 4a in the high level requirement, the symbol of sort and merge appears, meaning that the outcome is a result of merging two requirements where one or both has/have been

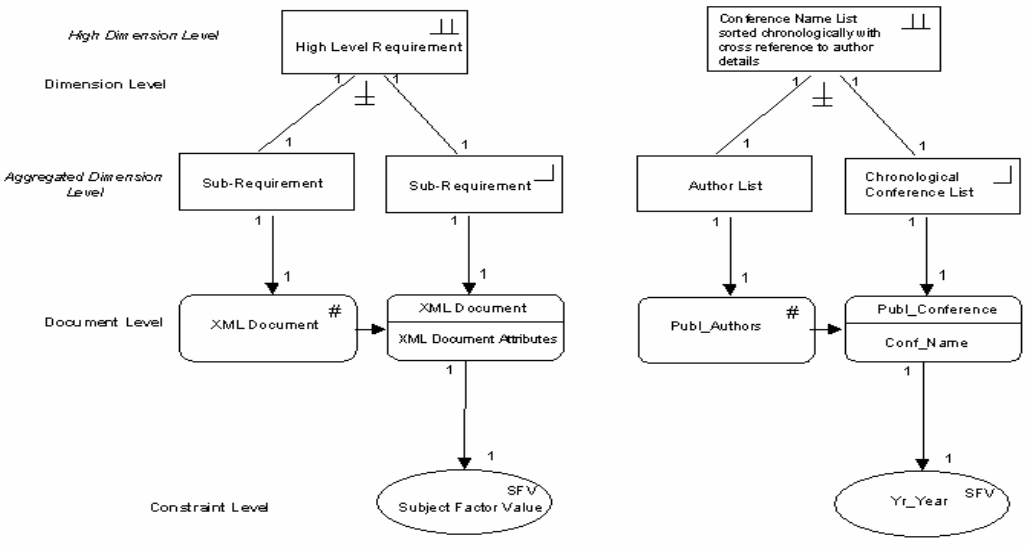

Figure 4(a) and 4(b): Chronological Listing of Conferences Names With cross-reference to Authors' Details

sorted. Fig. 4b also shows how a document list is retrieved based on a *selected* attribute then sorted based on a *subject factor value* and finally combined with another document list. The arrow between the documents is again present, as the extracted authors' details must correspond to the related conference.

The concept of requirement elicitation is performed in this case. We are able to decompose the high level requirement into several smaller sub-requirements, which may also prove valuable information when considered either solely or in combination with several other requirements. The approach of decomposing components aids in better understanding of the context at hand and discovering important factors, which may have not been initially exhibited.

5. Conclusion and Future Work

In this paper, we concentrated on the aspect of document warehouse structure and design. We carried out a formal, goal-driven approach to model warehouse requirements by introducing the XDW Requirement Model. In doing so, we have highlighted some of the challenges faced and the benefits of using such a requirement driven, conceptual design methodology for document warehouse design.

For future work, many issues deserve investigation. Firstly, to develop a plan of empirical studies to validate both our newly proposed requirement model and the already introduced conceptual model. Also a significant issue is checking for requirement consistency. Secondly we need to investigate feasible technologies to automate the mapping between the XML content and the XDW repositories with their semantics (conceptual and operational) intact. The performance issues associated with such a demanding task require examination.

6. References

[1] "The ECM Association (http://www.aiim.org/index.asp)," AIIM, 2005.

[2] W3C-XML, "Extensible Markup Language (XML) 1.0, (http://www.w3.org/XML/)," 3 ed: The World Wide Web Consortium (W3C), 2004.

[3] J. Pokorny, "XML Data Warehouse: Modelling and Querying," in *Proc. of the Baltic Conf. (BalticDB-IS '02)*, vol. 1, A. Kalja, Ed.: Institute of Cybernetics at Tallin Technical University, 2002.

[4] R. Kimball and J. Caserta, "The data warehouse ETL toolkit : practical techniques for extracting, cleaning, conforming, and delivering data." Hoboken, NJ: Wiley, 2004.

[5] ACM, "ACM Portal, (http://portal.acm.org/)," ACM, 2005.

[6] Springer, "SpringerLink: http://www.springerlink.com," Springer, 2005.

[7] IEEE, "IEEE Xplore®: http://ieeexplore.ieee.org," Rel 1.8 ed: IEEE, 2004.

[8] J. Trujillo, M. Palomar, J. Gomez, and I.-Y. Song, "Designing Data Warehouses with OO Conceptual Models," in *IEEE Computer Society, "Computer"*, 2001, pp. 66-75.

[9] S. Lujan-Mora, J. Trujillo, and I.-Y. Song, "Multidimensional Modeling with UML Package Diagrams," in *Proc. of the 21st Int. Conf. on Conceptual Modeling (ER '02)*, *Lecture Notes In Computer Science*: Springer-Verlag London, UK, 2002, pp. 199-213.

[10] A. Abelló, J. Samos, and F. Saltor, "Understanding facts in a multidimensional object-oriented model," in *4th Int. Workshop on Data Warehousing and OLAP (DOLAP '01)*, 2001.

[11] S. Mohammed, "Object-Relational Data Warehouse," in *Department of Computer Science & Computer Engineering*: La Trobe University, Melbourne, Australia, 2001.

[12] M. Lubars, C. Potts, C. Richer. "A review of the state of the practice in requirements modelling" *Proc. IEEE Symp. Requirements Engineering, San Diego*, 1993.

[13] K. McGraw, K. Harbison."User Centered Requirements, The Scenario-Based Engineering Process*". Lawrence Erlbaum Associates Publishers*, 1997.

[14] A. Dardenne, A. Van-Lamsweerde, and S. Fickas, "Goal-Directed Requirements Acquisition," in *Science of Computer Programming*, vol. 20, 1993, pp. 3-50.

[15] V. Nassis, Rajugan R., T. S. Dillon, and W. Rahayu, "XML Document Warehouse Design," in *Data Warehousing and Knowledge Discovery, 6th Int. Conf. (DaWaK '04)*, vol. 3181, *Lecture Notes in Computer Science*, M. K. M. Yahiko Kambayashi, Wolfram Wöß, Ed. Zaragoza, Spain: Springer, 2004, pp. 1-14.

[16] V. Nassis, Rajugan R., T. S. Dillon, and W. Rahayu, "Conceptual and Systematic Design Approach for XML Document Warehouses," in *Int. Journal of Data Warehousing and Mining*, vol. 1, No 3, 2005.

[17] Rajugan R., E. Chang, T. S. Dillon, and F. Ling, "XML Views: Part 1," in *14th Int. Conf. on Database and Expert Systems Applications (DEXA '03)*,: Springer 2003, pp. 148-159.

[18] V. Nassis, T. S. Dillon, W. Rahayu, and Rajugan R., "Goal-Oriented Requirement Engineering for XML Document Warehouses," in *Processing and Managing Complex Data for Decision Support*, O. Boussaid, Ed.: Idea Group Publishing (to appear), 2005.

[19] M. Golfarelli, D. Maio & S. Rizzi, "The dimensional Fact model: A conceptual model for data warehouses", *Int. Journal of Cooperative Information Systems* (Invited Proceeding), 7-2-3, 1998.

[20] N.J. Nilsson, "Problem solving methods in AI", McGraw Hill, 1971.

[21] V. Nassis, Rajugan R., T. S. Dillon, and W. Rahayu, *"*A Systematic Design Approach for XML-View Driven Web Document Warehouses". *Int. Workshop on Ubiquitous Web Systems and Intelligence (UWSI '05)*, Singapore, pp. 914-924, 2005.

[22] V. Nassis, T. S. Dillon, Rajugan R. and W. Rahayu, "An XML Document Warehouse Model," in *Data Warehousing and Knowledge Discovery, 7th Int. Conf. (DaWaK '05)*, Copenhagen, Denmark (submitted paper).