Improving the Maintainability of Data Warehouse Designs: Modeling Relationships between Sources and User Concepts

Alejandro Maté
Lucentia Research Group
University of Alicante
Alicante, Spain
amate@dlsi.ua.es

Juan Trujillo
Lucentia Research Group
University of Alicante
Alicante, Spain
jtrujillo@dlsi.ua.es

Elisa de Gregorio
Lucentia Research Group
University of Alicante
Alicante, Spain
edg12@dlsi.ua.es

Il-Yeol Song
College of Information Science
and Technology
Drexel University
USA
song@drexel.edu

ABSTRACT

In data warehouse (DW) development, a series of mappings must be specified between user concepts and data source elements, in order to identify which sources must undergo an integration process. Until now, these mappings are either assumed to be implied by name matching or identified according to the designer's experience. Then, the result is implemented as Extraction/Transformation/Loading (ETL) processes. Since ETL processes relate elements at the logical level, designers cannot adequately analyze how a change in requirements or in the data sources affects the analysis capabilities. Furthermore, this approach makes it difficult to perform incremental changes in DW design, requiring in some cases to perform the whole analysis again. In this paper we present a set of semantic mappings that relate user concepts specified by requirements to those obtained from data sources. In turn, this allows us to accurately identify how any potential change affects the different structures and ETL processes. As a DW evolves over time, our approach easily allows us to incorporate new concepts, as well as any change introduced at requirements or data sources into the DW repository with no need to redesign the whole DW. In order to show the application of our proposal, we show a real case study focusing on the Digital library of the University of Alicante.

Categories and Subject Descriptors

H.2.7 [**Database Management**]: Database administration—*Data warehouse and repository*

DOLAP'12, November 2, 2012, Maui, Hawaii, USA.
Copyright 2012 ACM 978-1-4503-1721-4/12/11 ...$15.00.

General Terms

Design, Theory

Keywords

Data warehouses, reconciliation, evolution, maintainability.

1. INTRODUCTION

Data warehouse (DW) development requires to extract information from multiple heterogeneus sources in order to support the decision making process [9]. In DW development, user requirements specify the information needs [7, 14] that must be met and determine the structure of the target DW. Once the structure has been defined, the DW is populated with information coming from different data sources, which are exploited during decision making process.

Nevertheless, the naming conventions and structures of data sources may not always match the structure specified by user requirements since (i) they are designed with different objectives than Online-Analytical Processing (OLAP), and (ii) the target structure is the (expected) result of integrating the different data sources each following its own conventions [3]. Therefore, in order to evaluate the viability of the DW, a series of mappings must be identified between user concepts and data source elements. After these mappings have been identified and the DW has been implemented, the integration process is implemented as Extraction/Transformation/Loading (ETL) processes in their corresponding files [4].

However, ETL processes only capture the results of these mappings at the logical level, while mappings themselves are not recorded in any diagram. In turn, analyzing ETL processes alone ignores key aspects related to the analysis needs. For example, consider Figure 1. An ETL process extracts information from "Document_TD" table and loads it into "Publication" and "Time" DW tables. Compare the information provided in this ETL flow with the one captured in Figure 2. In this Figure, a conceptual MD model of the data sources shows column "publicationMentio" related with user concepts by means of a Solvable Conflict relationship

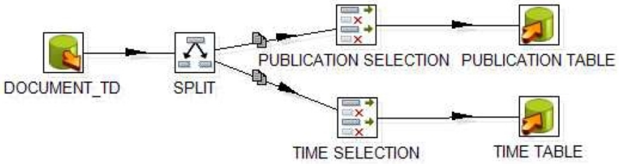

Figure 1: A single data source column determines the viability of multiple levels. ETL view

(SC, see Section 3.2). This Figure shows that (i) this single data source column "publicationMentio" actually provides the necessary information to identify three different analysis levels "Place", "Province" and "Time", and (ii) "Country" information is missing thus we require to gather it from another source. Therefore, if any change is performed on the source column or if additional information is provided, the designer can quickly identify the impact of these changes.

As shown, by following a traditional approach, designers cannot adequately document the correspondences between user concepts and data sources. In turn, designers cannot accurately assess how a change in the concepts required by decision makers or in the data sources will affect the DW. In some cases, changes may require to exhaustively analyze all the documentation available, such as when new data is available and some requirements had not been previously met. Thus, with current approaches, the maintainability of the whole system is impacted negatively [4].

In our previous work [11, 12, 13, 14, 16], we defined a hybrid DW development approach in the context of the Model Driven Architecture (MDA) [10]. In our approach, requirements are specified in a Computation Independent Model (CIM) by means of a goal-based UML profile [14] extending the i* framework [21]. Then, they are automatically derived into a conceptual DW model [11], and reconciliated with the data sources by using reverse engineering [15].

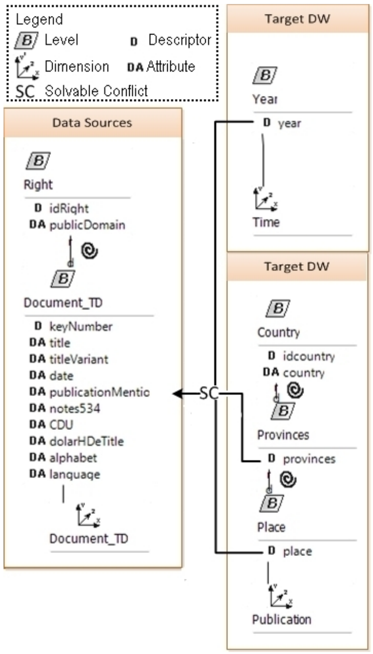

Figure 2: A single source column determines the viability of multiple levels. Conceptual view

In this paper, we present a set of semantic mappings specifying the existing relationships between user concepts specified by requirements and concepts derived from data sources. Our proposal allows us to (i) identify which DW structures and ETL processes will be affected by a change either in user concepts or in the data sources, (ii) identify which user requirements can be satisfied, as well as calculate different quality metrics over them [19], and (iii) preserve the mapping between the target DW and the data sources, thus allowing us to instantiate the mappings in the form of conceptual ETL processes [6, 17].

The remainder of this paper is structured as follows, Section 2 briefly presents the Related Work in ETL processes, DWs, and data provenance. Section 3 describes semantics of the relationships between conceptual DW elements and data source elements. Section 4 describes the application of the proposal to a case study from the digital library of the University of Alicante. Finally, Section 5 summarizes the conclusions and sketches the future work.

2. RELATED WORK

The matching and reconciliation process between requirements and data sources has been tackled in different DW design approaches [7, 8, 15]. Our research shows that we can differentiate two basic aspects in this process: how the matching is performed and how DW structures can be traced to their original counterparts.

On one hand, in order to perform the matching, hybrid DW approaches such as [1, 7, 15] first obtain a reverse engineering model of the data sources. Then, a name matching is performed, fusing together structures derived from requirements and those obtained from the data sources. However, as described in [5], the language employed by decision makers to describe DW requirements is different than the one used by engineers to design an OLTP schema or other sources such as plain text. Therefore, the name matching process may not obtain the expected result. Furthermore, even if a partial solution such as previously defining a common terminology is applied [1, 18], existing structural differences still require to perform a manual reconciliation process. In addition, each time a new data source is added, this process will have to be repeated in order to integrate new and previously existing data.

On the other hand, another important aspect is how structures and data are traced to their original counterparts. As both data sources and requirements evolve over time, this evolution influences the DW. Data sources, such as OLTP databases, are updated as business processes change. In turn, at least some of this changes must be propagated to the DW supporting the organization's Business Intelligence system in order to provide correct data and being able to take adequate decisions. In a similar way, requirements also evolve as the business strategy adapts to constant changes in the environment, such as new competitors appearing, sales decreasing due to an economic crisis, etc. This leads decision makers to require additional information that may not have been previously considered, thus requiring to modify not only the target DW but also the ETL processes.

Traditionally, this aspect is tackled by means of traceability [20] and data provenance [2] approaches. On one hand, traceability is focused on tracking the relationships between different elements involved in the design process as they evolve through a set of operations. However, most DW modeling approaches rely on ETL processes to trace DW structures and data lineage back to their origins. As ETL processes do not model DW structures at the conceptual level, traceability between the multidimensional concepts and the data sources is lost. In addition, user concepts which were not incoporated in the target DW are removed from the final diagram in the reconciliation process. Therefore, when incorporating new data they may be overlooked, thus losing analysis capabilities. On the other hand, data provenance is focused on identifying where the data stored came from and which operations were performed to obtain its current value. However, implicit data provenace provides no mechanisms to record the mappings and transformations performed to user concepts during the reconciliation process. As a result, implicit data provenance serves to trace back to its origins the data used but cannot be applied to user concepts as they include no data.

Summarizing, current DW development approaches do not provide any mechanism to record the mappings involved in the reconciliation process, thus decreasing the maintainability of the system. This process is typically approached by name matching, despite the different name conventions employed in requirements and data sources. Moreover, any structural differences are to be solved by the designer using his own methods. Finally, analysis of the impact of changes cannot be adequately performed since (i) there is no documentation specified recording the structural changes, and (ii) ETL processes relate sources and DW structures at the logical level. Our approach solves these drawbacks by providing a set of concepts to model these mappings. In turn, traceability towards requirements and data sources is also preserved as described in [13]. In this way, we can easily identify which elements should be integrated when new data sources are added or previously existing data sources and requirements are updated.

3. RELATIONSHIPS BETWEEN USER CONCEPTS AND DATA SOURCES

In order to address the maintainability problem in DW design, we propose to model the relationships established in the reconciliation process explictly, by comparing user expectations with the information provided by the data sources. This comparison is performed by relating the conceptual DW model obtained from requirements [11] with the different conceptual models obtained from data sources by means of reverse engineering [15], which capture real data characteristics and structure.

3.1 Overview of the Approach

In our approach, we make explicit the implicit knowledge in the reconciliation process, avoiding information losses and improving the maintainability of the system. This is performed by introducing a set of semantic mappings. In our development process, shown in Figure 3, we start by obtaining a conceptual model of the DW from user requirements (1). Afterwards, by means of reverse engineering, we obtain a multidimensional model of the data sources (2). Once we have both models, we apply our proposal to capture the semantic relationships between them (3), and finally derive the target DW according to the relationships modeled.

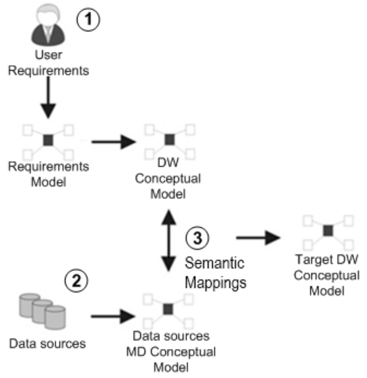

Figure 3: Steps in the development process

In the following, we focus on the definition of these relationships in order improve the maintainability of the DW. Then, we show the applicability by means of a case study.

3.2 Formal Definition

In order to adequately relate elements and analyze how a change affects the DW, we must be able to capture the semantic of the relationship between user concepts and data. In this section, we present the formal definitions involved in our approach. First, we start by defining the basic elements. Then, we define the different relationships which can be identified according to the set theory by analyzing how user expectations compare to data provided.

Conceptual DW models are defined by means of multidimensional modeling, specifying sets of Facts (center of the analysis) and Dimensions (context of analysis). According to the definitions from [16] D is a set of dimension schemata such that two distinct dimension schemata $D_1, D_2 \in D$ only have dimension level l_{All}. Each Dimension Schema is a quadruple $D =(N, L, \preceq, C)$ where N is the dimension name, L is a finite set of dimension levels, $\preceq$ is a lattice, specifying a dimension hierarchy, such that (a) $infL$ is not null, and (b) $supL = l_{All}$; where $infL$ represents the root level and $supL$ represents the highest level; and C is a (possibly empty) set of context dependencies.

In addition to these definitions, we formally define Levels and Attributes. A Level L_i is a pair $L = (N, A)$ such that N is the name of the level, and A is a set of attributes over this level. One of these attributes is the identifier of the level and satisfies $Des(a_i)$. Finally, an attribute a_i is an atomic element, which takes values from a given domain D_a.

After providing the formal definitions for the different elements composing a multidimensional schema, we focus on specifying the correspondences between user concepts obtained from requirements and conceptual elements obtained from data sources. On one hand, user concepts define the space of possible values for each element by describing their characteristics. As such, domains corresponding to requirements will be defined by intension, i.e. $D_1 = \{$“all provinces in the country”$\}$. On the other hand, data sources are characterized by providing specific sets of instances, thus domains corresponding to data sources will be defined by extension, i.e. $D_2 = \{$“Alicante”,“Valencia”,“Castellón”,...$\}$.

Analyzing the existing similarities and differences between both sets will allow us to categorize their relationships and provide accurate information which can be used by the designer when incorporating changes.

3.2.1 Attribute Analysis

Now we proceed to analyze the different kinds of relationships that may be established between conceptual elements. We start by analyzing the most basic element: attributes. Attributes can be compared by analyzing their domains. The first aspect to consider is if the domains of the different attributes being related are the same or not. For example, consider the code for a document in the Digital Library. The decision maker specifies that each document has a code, expected to be composed by a sequence of numbers. Indeed this code is a sequence. In this case both domains would be the same, thus no transformation would be necessary to go from one domain to the other. We categorize this situation as an *Overlap* (O). However, this is not always the case. Consider the language of each document. The decision maker expects to analyze a set of language names, such as "English", "French", etc. Instead, the data sources store languages recording their language code. While both attributes refer to languages, codes cannot be used directly in the DW and require to be transformed in order to obtain the expected language names. This difference between the domains of both attributes is categorized as a *Conflict* (C).

$$\left.\begin{array}{l} a_1 = \mathit{idDocument} = \{\text{document codes}\} \\ a_2 = \mathit{keyNumber} = \{11111, 22222, ...\} \end{array}\right\} \implies \mathrm{O}(a_1,a_2)$$

$$\left.\begin{array}{l} a_3 = \mathit{name} = \{\text{names of languages}\} \\ a_4 = \mathit{language} = \{aar, abk, afk, ...\} \end{array}\right\} \implies \mathrm{C}(a_3,a_4)$$

By performing a thorough analysis w.r.t. the set theory we can differentiate six different categories. These categories allow the designer to identify mismatches between the information expected and the information provided:

1. An *Equivalent Overlap* (EO) relationship holds between two attributes when both domains define the same set. For example, we have "idDocument" and "keyNumber" as previously shown.

2. A *Subset Overlap* (UO) relationship holds between two attributes when the second attribute has a restricted domain compared to the first one. Thus, the second domain is strictly included in the first one. For example, if we expected "idDocument" to include the identifiers of documents from all the different libraries being integrated, but instead "keyNumber" did not have the identifiers of one or more libraries available.

3. A *Superset Overlap* (SO) relationship holds between two attributes when the second attribute has a more general domain compared to the first one. Thus, the second domain strictly includes the first one.

4. A *Complementary Overlap* (CO) relationship holds between two attributes when both domains share a common part but none of them is fully included in the other. For example, if we expected "type" in "Type" level to include "handwritten" and "digital" formats and, instead, it includes "handwritten", "music composition", and "theater".

5. A *Solvable Conflict* (SC) relationship holds between two attributes when a function F is necessary to project elements from the second domain into the first one, and such function exists. For example, "name" expecting a language name and instead we have "language" codes, from which we can obtain the expected set of names.

6. An *Irresolvable Conflict* (IC) relationship holds between two attributes when a function F is necessary but this function does not exist. For example, if we expected the "idDocument" to be the name of the document and we had "keyNumber" as the only element provided which stores only sequences of numbers.

7. Finally, *No Relationship* (NR) is the absence of a correspondence under any of the previously-defined relationships, such as when two attributes do not have anything in common.

A summary of the previous relationships is shown in Figure 4 in the form of trace metaclasses, where the four Overlap relationships described are a specialization of the Overlap concept, while the two Conflict relationships described are a specialization of the Conflict concept from [12]. All the relationships can have n source target elements.

We specify subset and superset relationships by means of proper inclusion relationships between sets. This is because we wish to identify the cases when there is less information than required and when there is more information than required. For example, imagine that the decision maker wishes to include only english authors in the DW, such that D_1=$\{$*ids of English authors*$\}$. However, the data source we are using include information from all the authors available (SO), thus we have additional authors. This data may be useful information, or, alternatively, it can turn into noise which should be filtered out. Conversely, the lack of information (UO) can derive into erroneous decisions, which are counterproductive for the organizations. For example, considering an author is not important because he created a few documents while lacking information of certain books. These scenarios cannot be detected by means of constraints over the summarizability of measures, but they can be analyzed thanks to the categorization proposed.

3.2.2 Level Analysis

Once we have defined the different kinds of relationships between attributes, which are considered atomic, we can proceed to define the relationships between coarser-grain elements in the multidimensional model. The second element we analyze is levels. Levels, $L = (N, A)$, are combinations of attributes (A) paired with a semantic name (N), giving the whole set of attributes a meaning. The most important attribute in a level is its descriptor, $Des_{L1}(a_1)$, since it identifies the instances of that level. For example, compare a "Document" level (Figure 5), expected to be identified by an id, with a "Document_TD" level from the data sources (Figure 6), also identified by an id. If both identifiers are sequence of numbers, they will be overlapping with each other, thus the DW will be able to provide each instance of "Document" as was expected. As such, both levels would be overlapping, and the amount of information provided by each one would depend on the rest of attributes. However, if both levels used different types of identifiers, i.e. expecting to identify a "Document" by its title, then we would obtain

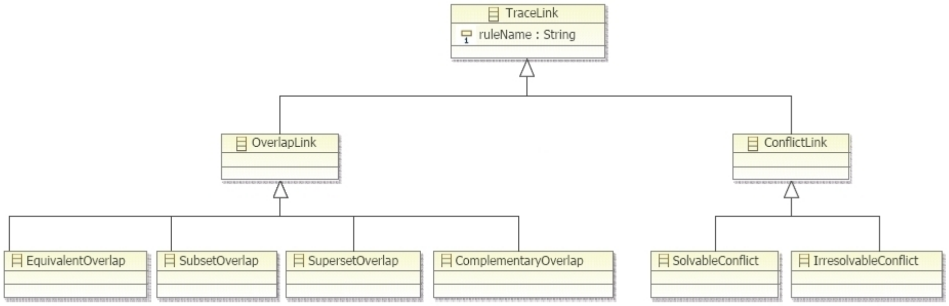

Figure 4: Categorization of Overlap and Conflict relationships between requirements and data sources

a different result than we expected since a title can be repeated for multiple codes. Therefore, both levels would be conflicting and would require a transformation in order to obtain the correct data.

We now present the definitions used to relate levels, allowing the designer to identify mismatches between how concepts are identified, and how much information is provided:

1. An *Equivalent Overlap* (EO) relationship between two levels holds when both levels have all their attributes are related such as in the case of "Author". Furthermore, in order to guarantee the compatibility, an overlap between descriptor attributes, $O(Des(l_1), Des(l_2))$, must hold in every overlap relationship. For example, "Author" (data source) presents exactly the expected set of attributes in "Author" (requirements), thus they have an EO relationship.

2. A *Subset Overlap* (UO) relationship between two levels holds when the second level provides less information than the first one, thus its set of attributes is a subset of l_1. For example, if we had a "Country" (data source) level providing only the descriptor.

3. A *Complementary Overlap* (CO) relationship between two levels holds when both levels share a set of common attributes in addition to other attributes that are exclusive to each of them. For example, "Document_TD" is missing the expected "uuid" attribute, but instead provides additional information such as "CDU", "Notes534", or "Date".

4. A *Solvable Conflict* (SC) relationship between two levels holds when their descriptors are conflicting or they are not related to each other but, instead, are related to other dimension attributes. Therefore, the way of identifying each level is different. For example, we can obtain "Alphabet" from "Document_TD" by extracting the data from the "alphabet" column. However, these two levels have a different meaning (identifier), thus we are actually transforming one into the other.

5. An *Irresolvable Conflict* (IC) relationship between two levels holds when their descriptors are not related or they present an IC between them, thus one level cannot be converted into the other. For example, "Language" cannot be obtained from "Document_TD" by means of a transformation since we are missing the required descriptor for the level.

Complementary Overlaps are common when the user requires to analyze information from concepts which are entities in the data sources. Usually, the identifier employed is the same. However, it is rare that all the required attributes are stored directly in the data source nor every attribute stored is explicitly enumerated in the requirements.

Conflicts between levels involve a mismatch between the semantics of levels, and they are common when the user requires to analyze in detail concepts which are not entities on their own in the data sources. This situation can be seen in Figures 2 and 7 where we require to extract the "Year", the "Place" of publication of a given document, and the "Language" of the document.

3.2.3 Dimension Analysis

The last element we analyze are dimensions. A dimension is a quadruple $D = (N, L, \preceq, C)$. Dimensions are named (N) sets of levels (L) which satisfy a partial order relationship $(\preceq)$ which may depend on context dependencies (C). For example, in Figure 5, dimension "Document" includes a base level "Document" which is previous to every other level. On the other hand, "SupportForm" and "Volume" cannot be ordered as they are aggregated in parallel paths.

Typically, the name of the dimension is the same as the name of the lowest level in the Lattice, since this level identifies the tuples of the dimension in a similar way as *Descriptor* attributes for levels. Thus the lowest level is one of the critical elements when comparing two dimensions. If the lowest level is overlapping or its conflict can be solved, then, the relationship between dimensions will depend on how their partial order relationships behave. However, if the lowest level is not related or presents no solution, then, it will not be possible to obtain the required dimension. For example, in Figure 7, the expected "Language" descriptor could not be related with any other element. Therefore, we cannot obtain the basic set of instances for the "Language" dimension.

Additionally, in the case of dimensions, the Lattice constitutes another critical element, since it defines the possible aggregation paths. If the expected $\preceq$ relationship is altered, and two levels interchange their order, then, both dimensions will be conflicting. Altering the partial order implies that the actual aggregation paths are contrary to what was expected, thus a transformation will be required in order to provide the order expected by the user. For example, an advanced case is shown in Figure 8, where "Format" corresponds with "SupportForm". However, "Format" is not aggregated by any level, thus a transformation will be required in order to combine "Document_TD" instances with their corresponding "Format", thus obtaining the expected "Document" dimension. In the following we present the categorization used to relate dimensions, allowing the designer to identify structural differences in the dimensions between the expected analysis hierarchies and the data retrieved:

1. *Equivalent Overlap* (EO) relationship holds between two dimensions when both dimension present the same set of levels as well as the same Lattice. As with levels, in every overlap relationship it is required that the lowest level is related by either an Overlap or Solvable Conflict relationship, in order to guarantee that we can analyze the data at the finest aggregation level specified by the dimension. For example, the "Author" (data source) dimension provides exactly the expected hierarchy of levels.

2. *Subset Overlap* (UO) relationship holds between two dimensions when the expected dimension presents more levels or relationships while maintaining the partial order between levels related by Conflict or Overlap relationships. For example, if we expected a "User" dimension representing the users of the Digital Library and instead of two levels "User" and "Category", data sources provided only "User" level.

3. *Complementary Overlap* (CO) relationship holds between two dimensions when each dimension presents levels or relationships that the other dimension does not include, while also maintaining the partial order between levels related by Conflict or Overlap relationships. For example, "Document_TD" dimension includes the "Right" level, which was not expected, but lacks the "SupportForm" and "Type" levels.

4. *Solvable Conflict* (SC) relationship holds between two dimensions when the partial order relationship is altered. In other words, levels in each dimension Lattice are crossed in the other Lattice. For example, "Format" dimension presents the "Format" level as the first level, instead of aggregated from individual documents as expected in "Document".

5. *Irresolvable Conflict* (IC) relationship holds between two dimensions when the lowest level in the first dimension is not related or presents an IC with the other level. Therefore, we cannot obtain a transformation to relate the first dimension with its corresponding fact. The solution for an IC is to look for a common higher aggregation level. For example, we cannot obtain the required instances of the "Language" dimension since the lowest level is lacking its identifier but we can correctly obtain "Alphabet" instances.

4. CASE STUDY

In this section we will present the application of our conceptualization to a real case study: the integration process in the Digital library of the University of Alicante.

Recently, the digital library in the University of Alicante performed an integration process by combining different data sources. In order to analyze the information related to documents, the Digital library included sources modelled according to different standards for digital works, such as FOAF or MARC21 ontologies among others. Each of these standards specifies its own way to name and structure information. As such, certain fields contain multiple information, such as the title, the name of the author and the date published, together in a single field, separated by special characters ("$"). Furthermore, some of this information was optional, and did not always appear in the same field. Moreover, tracking all this information was a complex task, since there was duplicated information present, such as variations including alternative titles in different columns or multiple indexes relating the same concept.

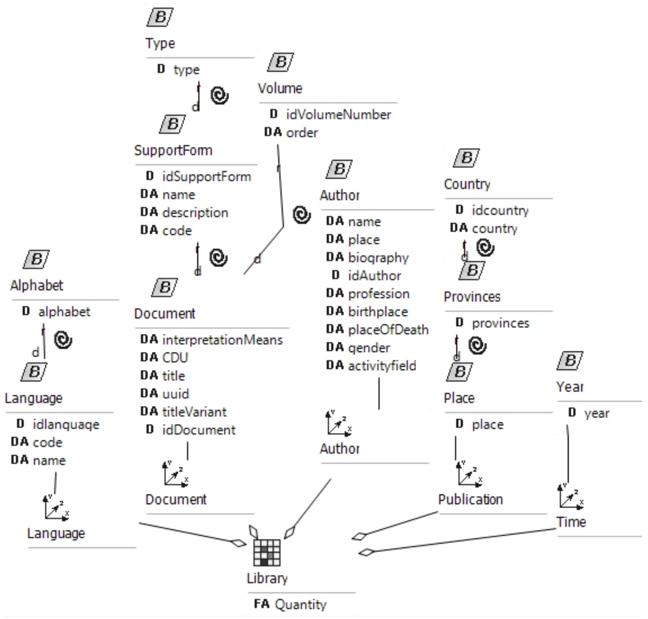

Figure 5: Conceptual model satisfying user requirements for the DW

Although each source was designed according to a standard, the structural differences and naming conventions made impossible to successfully apply a name matching approach. Therefore, in order to analyze which requirements could be satisfied, identify critical attributes, and analyze the impact of changes, we applied our approach to relate the data sources with the expected model in one data mart.

Our approach starts by obtaining the expected DW model from user requirements [13]. The resulting model is shown in Figure 5. In this model, we analyze the digital documents in the library. In this model, the most important dimension is "Document", which represents the different documents in the library. From each document we need to know its "title", the Universal Decimal Classification ("CDU") which classifies the document and the universal unique identifier ("uuid"). This dimension has three additional aggregation levels: the format it is stored ("SupportForm"), the "Type" of document, and additionally its "Volume". There are also other dimensions involved in the analysis: the "Author" of the document, its "Language", where it was published ("Publication"), and the date when it was published ("Time").

Once the requirements model has been obtained, the next step is to obtain a multidimensional model from the sources. This step can be performed either manually or by means of

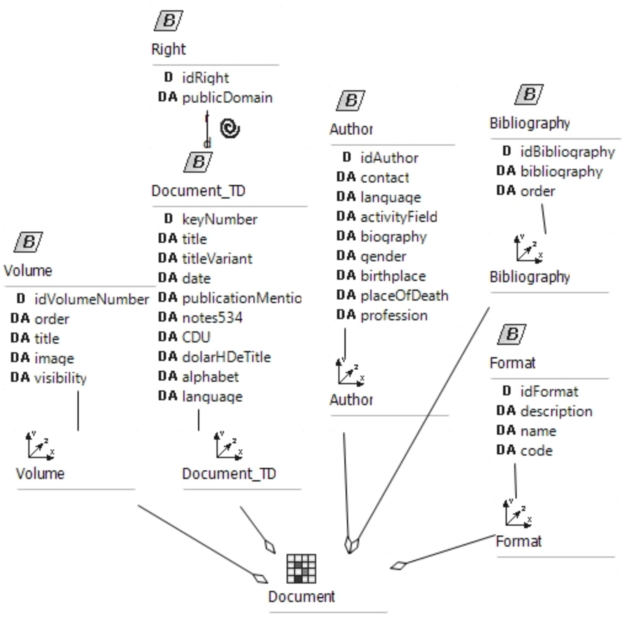

Figure 6: Conceptual model obtained from data sources

algorithms which apply some of the existing heuristics to obtain a multidimensional model from relational sources [16]. For the sake of simplicity, we only show a handful attributes in the conceptual model although some tables presented over 40 attributes. The result of the modeling step is shown in Figure 6. In this figure we can observe some differences w.r.t. the requirements model previously described. Some dimensions are initially missing like “Publication”, “Time”, and “Language”, while others, such as “Bibliography” appear. Moreover, some requirements levels like “Volume” and “SupportForm” appear as dimensions “Volume”, “Format”. Finally, some attributes are missing, such as “activityField” in “Author”, while other new ones appear, like “notes534”.

However, these differences do not mean that the necessary data do not exists in the data sources. Instead, important structural differences must also be considered. In order to capture this differences and be able to analyze the correspondences between requirements and sources, we apply our proposal to the mapping between both schemata[1]. By applying our approach, we are able to identify some attributes in the source model acting as dimension attributes, e.g. “publicationMentio”, which actually provide data for multiple levels and dimensions in the requirements model shown in Figure 2. These attributes pack all the necessary information in a single database column, and must be transformed (Solvable Conflict) in order to obtain the necessary data for the DW. On the other hand, some dimensions like “Document” require combining multiple dimensions from the sources into a single one, as can be seen in Figure 8. In this case, we must combine the “Document_TD” and “Format” dimensions to obtain the two first levels in the “Document” dimension. As a result of the mapping, we identify the existence of an additional level, “Right”, not considered in the requirements model which could provide useful information for the analysis.

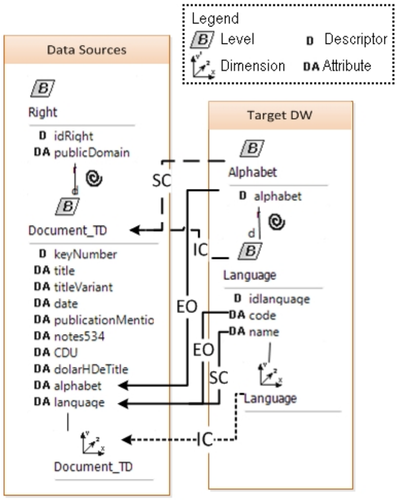

Figure 7: Irresolvable conflict between Document_TD and Language. We can only obtain alphabets from the data sources.

Furthermore, some attributes in the sources are empty, and the information is located in another member of the dimension. Finally, some attributes cannot obtain their value directly from the sources, but could be obtained by means of an external source, such as the country codes (“idCountry’).

In addition to solve the name matching and structural problem, our approach allowed us to preserve traceability and perform requirements validation as well as to analyze the impact of changes. For example, the analysis of the relationships captured shows that, out of 32 attributes in the requirements model, only 23 attributes required a cleaning process, 8 attributes required a transformation to calculate their value, and 1 attribute was completely missing. More-

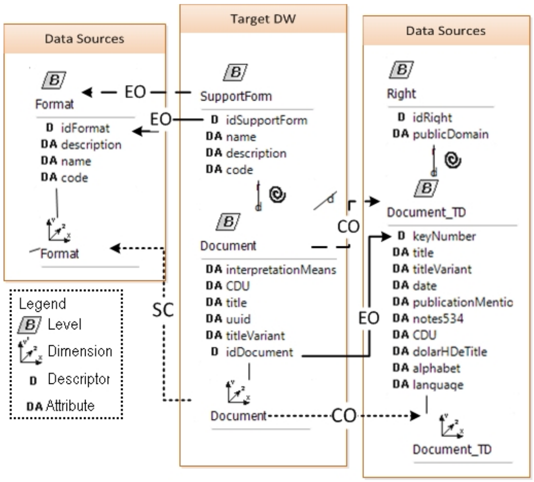

Figure 8: Relationships between pieces of work and their format

[1] In order to focus on the main problem tackled in this work, we omit the relationships between attributes which have the same name in both schemata

over, a critical attribute was identified in the data sources, providing information to 3 different descriptors in the requirements. Finally, one requirements level was not viable and its descriptor attribute had to be changed.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a formal approach that relates user concepts obtained from requirements to those coming from data sources, in order to better support the reconciliation process. Our approach is complementary to ETL processes, since it allows to easily identify and keep track of each element affected by a change both at data sources level as well as at requirements level. Thus the designer can easily identify how changes affect the different elements involved in the DW. Moreover, our approach also provides the necessary scaffolding to calculate a series of measures which may help in the analysis of alternative implementations, including but not limited to: (i) number of different sources that must be integrated to satisfy a given requirement, (ii) number of elements for which no information can be retrieved, and (iii) number of requirements supported by a given data source. Finally, we have shown the applicability of our approach by means of a real case study involving the integration process in the digital library of the University of Alicante. Thanks to our approach we were able to perform the analysis of the three aspects.

Our plans for the immediate future involve providing improved tool support for the traces. In the medium-long term, we plan to elaborate a series of metrics which are automatically calculated in order to evaluate the quality of the DW and the impact of changes.

6. ACKNOWLEDGMENTS

This work has been supported by the MESOLAP (TIN2010-14860) and SERENIDAD (PEII-11-0327-7035) projects from the Spanish Ministry of Education and the Junta de Comunidades de Castilla La Mancha. Alejandro Maté is funded by the Generalitat Valenciana (ACIF/2010/298).

7. REFERENCES

[1] Bonifati, A., Cattaneo, F., Ceri, S., Fuggetta, A., Paraboschi, S., et al. Designing data marts for data warehouses. *ACM Transactions on Software Engineering and Methodology 10*, 4 (2001), 452–483.

[2] Buneman, P., Khanna, S., and Wang-Chiew, T. Why and where: A characterization of data provenance. *Database Theory - ICDT 2001* (2001), 316–330.

[3] Chaudhuri, S., and Dayal, U. An overview of data warehousing and OLAP technology. *ACM Sigmod record 26*, 1 (1997), 65–74.

[4] Dayal, U., Castellanos, M., Simitsis, A., and Wilkinson, K. Data integration flows for business intelligence. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology* (2009), ACM, pp. 1–11.

[5] Eckerson, W. *Performance dashboards: measuring, monitoring, and managing your business.* Wiley, 2010.

[6] El Akkaoui, Z., and Zimányi, E. Defining ETL worfklows using BPMN and BPEL. In *Proceeding of the ACM twelfth international workshop on Data warehousing and OLAP* (2009), ACM, pp. 41–48.

[7] Giorgini, P., Rizzi, S., and Garzetti, M. GRAnD: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems 45*, 1 (2008), 4–21.

[8] Inmon, W. *Building the data warehouse.* Wiley-India, 2005.

[9] Kimball, R., and Ross, M. *The data warehouse toolkit: the complete guide to dimensional modeling.* Wiley, 2011.

[10] Kleppe, A., Warmer, J., and Bast, W. *MDA explained: the model driven architecture: practice and promise.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[11] Luján-Mora, S., Trujillo, J., and Song, I.-Y. A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering 59*, 3 (2006), 725–769.

[12] Maté, A., and Trujillo, J. Incorporating traceability in conceptual models for data warehouses by using MDA. *Conceptual Modeling–ER 2011* (2011), 459–466.

[13] Maté, A., and Trujillo, J. A trace metamodel proposal based on the model driven architecture framework for the traceability of user requirements in data warehouses. *Information Systems 37*, 8 (2012), 753 – 766.

[14] Mazón, J.-N., Pardillo, J., Soler, E., Glorio, O., and Trujillo, J. Applying the i* Framework to the Development of Data Warehouses. In *iStar* (2008), pp. 79–82.

[15] Mazón, J.-N., and Trujillo, J. A hybrid model driven development framework for the multidimensional modeling of data warehouses. *SIGMOD Record 38*, 2 (2009), 12–17.

[16] Mazón, J.-N., Trujillo, J., and Lechtenbörger, J. Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms. *Data & Knowledge Engineering (DKE) 63*, 3 (2007), 725–751.

[17] Muñoz, L., Mazón, J.-N., Pardillo, J., and Trujillo, J. Modelling ETL Processes of Data Warehouses with UML Activity Diagrams. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops* (2008), Springer, pp. 44–53.

[18] Romero, O., and Abelló, A. A framework for multidimensional design of data warehouses from ontologies. *Data & Knowledge Engineering 69*, 11 (2010), 1138–1157.

[19] Vassiliadis, P. *Data Warehouse Modeling and Quality Issues.* PhD thesis, Athens, 2000.

[20] Winkler, S., and von Pilgrim, J. A survey of traceability in requirements engineering and model-driven development. *Software and Systems Modeling 9* (2010), 529–565.

[21] Yu, E. *Modelling strategic relationships for process reengineering.* PhD thesis, Toronto, Ont., Canada, Canada, 1995.