A Methodology for the Conceptual Modeling of ETL Processes

Alkis Simitsis[1], Panos Vassiliadis[2]

[1] National Technical University of Athens, Dept. of Electrical and Computer Eng., Computer Science Division, Iroon Polytechniou 9, 157 73, Athens, Greece
asimi@dbnet.ece.ntua.gr
[2] University of Ioannina,
Dept. of Computer Science, 45110, Ioannina, Greece
pvassil@cs.uoi.gr

Abstract. Extraction-Transformation-Loading (ETL) tools are pieces of software responsible for the extraction of data from several sources, their cleansing, customization and insertion into a data warehouse. In this paper, we propose a methodology for the earliest stages of the data warehouse design, with the goal of tracing the analysis of the structure and content of the existing data sources and their intentional mapping to the common conceptual data warehouse model. The methodology comprises a set of steps that can be summarized as follows: (a) identification of the proper data stores; (b) candidates and active candidates for the involved data stores; (c) attribute mapping between the providers and the consumers, and (d) annotation of the diagram with runtime constraints.

1 Introduction

In order to facilitate and manage the data warehouse operational processes, specialized tools are already available in the market, under the general title *Extraction-Transformation-Loading* (ETL) tools. To give a general idea of the functionality of these tools we mention their most prominent tasks, which include: (a) the identification of relevant information at the source side; (b) the extraction of this information; (c) the customization and integration of the information coming from multiple sources into a common format; (d) the cleaning of the resulting data set, on the basis of database and business rules, and (e) the propagation of the data to the data warehouse and/or data marts.

In Fig.1 we abstractly describe the general framework for ETL processes. In the bottom layer we depict the data stores that are involved in the overall process. On the left side, we can observe the original data providers. Typically, data providers are relational databases and files. The data from these sources are extracted (as shown in the upper left part of Fig.1) by extraction routines, which provide either complete snapshots or differentials of the data sources. Then, these data are propagated to the Data Staging Area (DSA) where they are transformed and cleaned before being loaded to the data warehouse. The data warehouse is depicted in the right part of the data store layer and comprises the target data stores, i.e., fact tables for the storage of information and dimension tables with the description and the multidimensional, roll-up hierarchies of the stored facts. The loading of the central warehouse is performed from the loading activities depicted on the upper right part of the figure.

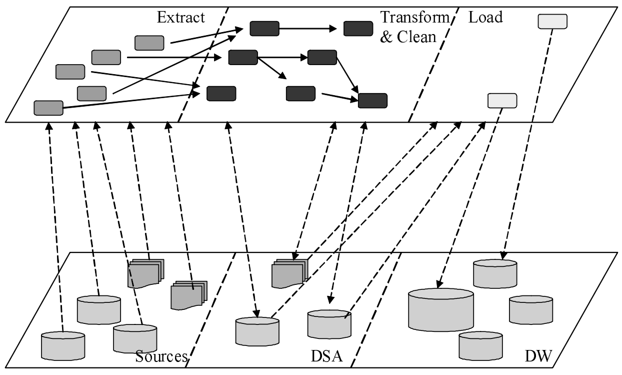

Fig. 1. The environment of Extract-Transform-Load processes

In this paper, we are dealing with the earliest stages of the data warehouse design. During this period, the data warehouse designer is concerned with two tasks that are practically executed in parallel. The first of these tasks involves the *collection of requirements* from the part of the users. The second task, which is of equal importance for the success of the data warehousing project, involves *the analysis of the structure and content of the existing data sources* and their *intentional mapping to the common data warehouse model*. Related literature [14, 27] and personal experience suggest that the design of an ETL process aims towards the production of a crucial deliverable: the mapping of the attributes of the data sources to the attributes of the data warehouse tables. The production of this deliverable involves several interviews that result in the revision and redefinition of original assumptions and mappings; thus it is imperative that a simple conceptual model is employed in order to (a) facilitate the smooth redefinition and revision efforts and (b) serve as the means of communication with the rest of the involved parties.

In a previous line of work [29], we have proposed a conceptual model for ETL processes. In this paper, we complement this model in a set of design steps, which lead to the basic target, i.e., the attribute interrelationships. These steps constitute the methodology for the design of the conceptual part of the overall ETL process and could be summarized as follows: (a) identification of the proper data stores; (b) candidates and active candidates for the involved data stores; (c) attribute mapping between the providers and the consumers, and (d) annotating the diagram with runtime constraints (e.g., time/event based scheduling, monitoring, logging, exception handling, and error handling).

This paper is organized as follows. In Section 2, we give a motivating example, over which our discussion will be based. Section 3 shortly presents an overview of the conceptual model for ETL processes. In Section 4, we demonstrate the methodology for the usage of the conceptual model. Finally, in Section 5 we present related work and in Section 6 we conclude our results.

2 Motivating example

To motivate our discussion we will introduce an example involving two source databases S_1 and S_2 as well as a central data warehouse DW. The scenario involves the propagation of data from the concept PARTSUPP(PKEY,SUPPKEY,QTY,COST) of source S_1 as well as from the concept PARTSUPP(PKEY,DEPARTMENT,SUPPKEY, DATE,QTY,COST) of source S_2 to the data warehouse. In the data warehouse, DW.PARTSUPP(PKEY,SUPPKEY,DATE,QTY,COST) stores daily (DATE) information for the available quantity (QTY) and cost (COST) of parts (PKEY) per supplier (SUPPKEY). We assume that the first supplier is European and the second is American, thus the data coming from the second source need to be converted to European values and formats. For the first supplier, we need to combine information from two different tables in the source database, which is achieved through an outer join of the concepts PS_1 and PS_2 respectively. In Fig. 2 we depict the full-fledged diagram of our motivating example. Throughout all the paper, we will clarify the introduced concepts through their application to our motivating example.

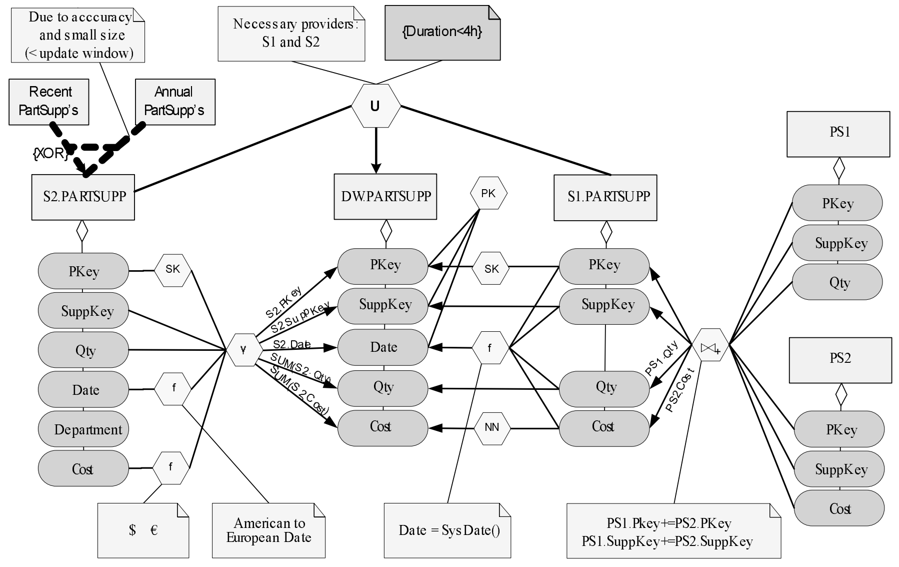

Fig. 2. The diagram of the conceptual model for our motivating example

3 Conceptual Model for ETL processes

In this section, we focus on the conceptual part of the definition of the ETL process. For a detailed presentation of our conceptual model and formal foundations for the representation of ETL processes, we refer the interested reader to [29]. This model has a particular focus on (a) the interrelationships of attributes and concepts, and (b) the necessary transformations that need to take place during the loading of the warehouse. The latter part is directly captured in the proposed metamodel as a first class citizen; we employ *transformations* as a generic term for the restructuring of schema and values or for the selection and even transformation of data. Attribute interrelationships are captured through *provider* relationships that map data provider attributes at the sources to data consumers in the warehouse. Apart from these fundamental relationships, the proposed model is able to capture constraints and transformation composition, too.

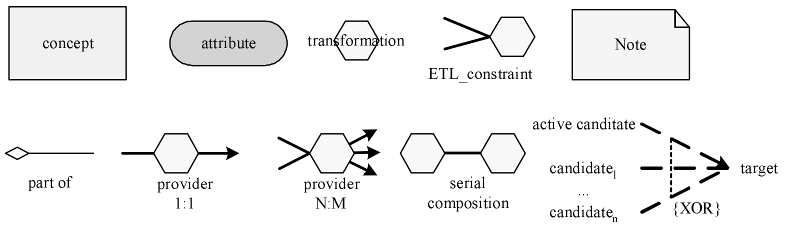

Fig. 3. Notation for the conceptual modeling for ETL activities

In Fig. 3 we graphically depict the different entities of the proposed model. We do not employ standard UML notation for concepts and attributes, for the simple reason that we need to treat attributes as first class citizens of our model. We try to be orthogonal to the conceptual models which are available for the modeling of data warehouse star schemata; in fact, any of the proposals for the data warehouse front end can be combined with our approach, which is specifically tailored for the back end of the warehouse.

Attributes. A granular module of information. The role of attributes is the same as in the standard ER/dimensional models (e.g., PKEY, DATE, SUPPKEY, etc.).

Concepts. A concept represents an entity in a source database or in the data warehouse (e.g., S_1.PARTSUPP, S_2.PARTSUPP, DW.PARTSUPP). Concept instances are the files in the source databases, the data warehouse fact and dimension tables and so on. A concept is formally defined by a name and a finite set of attributes.

Transformations. Transformations are abstractions that represent parts, or full modules of code, executing a single task and include two large categories: (a) filtering or data cleaning operations, and (b) transformation operations, during which the schema of the incoming data is transformed (a surrogate key assignment transformation (SK), a function application (f), a not null (NN) check, etc.). Formally, a transformation is defined by (a) a finite set of input attributes; (b) a finite set of output attributes, and (c) a symbol that graphically characterizes the nature of the transformation. A transformation is graphically depicted as a hexagon tagged with its corresponding symbol.

ETL Constraints. ETL constraints are used in several occasions when the designer wants to express the fact that the data of a certain concept fulfill several requirements (e.g., to impose a PK constraint to DW.PARTSUPP for the attributes PKEY, SUPPKEY, DATE). This is achieved through the application of ETL constraints, which are formally defined as follows: (a) a finite set of attributes, over which the constraint is imposed, and (b) a single transformation, which implements the enforcement of the constraint. Note, that despite the similarity in the name, ETL constraints are different modeling elements from the well-known UML constraints. An ETL constraint is graphically depicted as a set of solid edges starting from the involved attributes and targeting the facilitator transformation.

Notes. Exactly as in UML modeling, notes are informal tags to capture extra comments that the designer wishes to make during the design phase or render UML constraints attached to an element or set of elements [3] (e.g., a runtime constraint specifying that the overall execution time for the loading of `DW.PARTSUPP` cannot take longer than `4 hours`).

Part-of Relationships. We bring up part-of relationships, not to redefine UML part-of relationships, but rather to emphasize the fact that a concept is composed of a set of attributes, since we need attributes as first class citizens in the inter-attribute mappings. In general, standard ER modeling does not treat this kind of relationship as a first-class citizen of the metamodel; UML modeling on the other hand, hides attributes inside classes and treats part-of relationships with a much broader meaning. Naturally, we do not preclude the usage of the part-of relationship for other purposes, as in standard UML modeling.

Candidate relationships. A set of candidate relationships captures the fact that a certain data warehouse concept (e.g., S_2`.PARTSUPP`) can be populated by more than one candidate source concepts (e.g., `AnnualPartSupp`, `RecentPartSupp`). This information is normally discovered quite early in the data warehouse project and should be traced for the possibility of redesign or evolution of the warehouse.

Active candidate relationships. An active candidate relationship denotes the fact that out of a set of candidates, a certain one (in our case, `RecentPartSupp`) has been selected for the population of the target concept.

Provider relationships. A provider relationship can be used at two abstraction levels. At the concept level, it maps provider to target concepts. At the attribute level, it maps a set of input attributes to a set of output attributes through a relevant transformation. Practically, a provider relationship at the concept level implies that the target concept will be populated from the provider one. This kind of provider relationship at the concept level can be further refined to provider relationships at the attribute level (through simple part-of relationships). In the simple `1:1` case, provider relationships capture the fact that an input attribute in the source side populates an output attribute in the data warehouse side. If the attributes are semantically and physically compatible, no transformation is required. If this is not the case though, we pass this mapping through the appropriate transformation (e.g., European to American data format, not null check, etc.). In general, it is possible that some form of schema restructuring takes place; thus, the formal definition of provider relationships comprises (a) a finite set of input attributes; (b) a finite set of output attributes, and (c) an appropriate transformation (i.e., one whose input and output attributes can be mapped one to one to the respective attributes of the relationship). In the case of `N:M` relationships the graphical representation obscures the linkage between provider and target attributes. To compensate for this shortcoming, we annotate the link of a provider relationship with each of the involved attributes with a tag, so that there is no disambiguate for the actual provider of a target attribute.

Transformation Serial Composition. It is used when we need to combine several transformations in a single provider relationship (e.g., the combination of SK and γ in our motivating example).

4 Methodology for the usage of the conceptual model

In this section, we will present the proposed methodology, by giving the sequence of steps for a designer to follow, during the construction of the data warehouse. Each step of this methodology will be presented in terms of our motivating example (also hoping that it clarifies the nature of the employed modeling entities). As already mentioned, the ultimate goal of this design process is the production of inter-attribute mappings, along with any relevant auxiliary information.

Step 1: Identification of the proper data stores. The first thing that a designer faces during the requirements and analysis period of the data warehouse process is the identification of relative data sources. Assume that for a particular subset of the data warehouse, we have identified the concept DW.PARTSUPP which is a fact table of how parts where distributed according to their respective suppliers. Assume also that we have decided that for completeness reasons, we need to populate the target table with data from both sources S_1 and S_2, i.e., we need the union of the data of these two sources. Moreover, in order to populate the concept S_1.PARTSUPP, we need an outer join of the data of the concepts PS_1 and PS_2. Then, the diagram for the identified data sources and their interrelationships is depicted in Fig. 4.

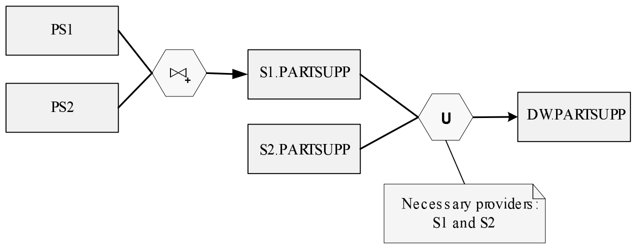

Fig. 4. Identification of the proper data stores

Step 2: Candidates and active candidates for the involved data stores. As already mentioned before, during the requirements and analysis stage the designer can possibly find out that more than one data stores could be candidates to populate a certain concept. The way the decision is taken cannot be fully automated; for example, our personal experience suggests that, at least the size, the data quality and the availability of the sources play a major role in this kind of decision. For the purpose of our motivating example, let us assume that source S_2 has more than one production systems (e.g., COBOL files), which are candidates for concept S_2.PARTSUPP. Assume that the available sources are:

– A concept AnnualPartSupp's (practically representing a file F1), that contains the full annual history about part suppliers; it is used basically for reporting purposes and contains a superset of fields than the ones required for the purpose of the data warehouse.

– A concept RecentPartSupp's (practically representing a file F2), containing only the data of the last month; it used on-line by end-users for the insertion or update of data as well as for some reporting applications.

The candidate concepts for concept S_2.PARTSUPP and the (eventually selected) active candidate `RecentPartSupp's`, along with the tracing of the decision for the choice, are depicted in Fig. 5.

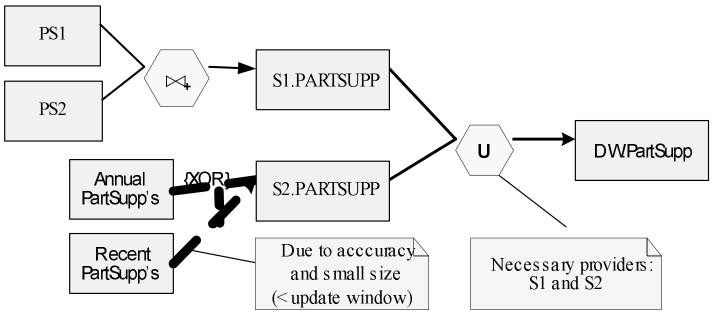
PS1
PS2
S1.PARTSUPP
S2.PARTSUPP
U
DW.PartSupp
Annual PartSupp's
{XOR}
Recent PartSupp's
Due to acccuracy and small size (< update window)
Necessary providers: S1 and S2

Fig. 5. Candidates and active candidates for the motivating example

Once a decision on the active candidate has been reached, a simplified 'working copy' of the scenario that hides all other candidates can be produced. There are two ways to achieve this (a) by ignoring all the information on candidates for the target concept (Fig. 6), or (b) by replacing the target with the active candidate (Fig. 7). In other words, in order to avoid cluttering the diagram with information not directly useful, the designer can choose to work in a simplified version. The hidden candidates, are not eliminated; on the contrary, they remain part of the conceptual model of the data warehouse, to be exploited in later stages of the warehouse lifecycle (e.g., in the case of changes to the active candidate that may possibly lead to the reconsideration of this choice). The only difference is that they are not shown to the designer.

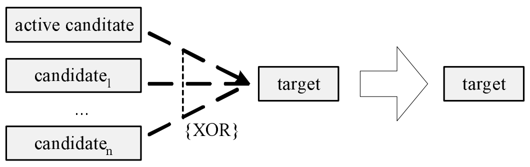

Fig. 6. Working copy transformation: ignoring candidates

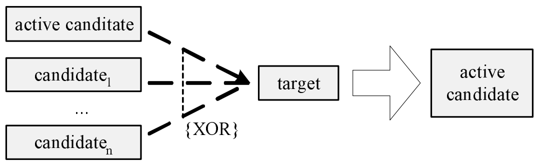

Fig. 7. Working copy transformation: replace targets with active candidates

Step 3: Attribute mapping between the providers and the consumers. The most difficult task of the data warehouse designer is to determine the mapping of the attributes of the sources to the ones of the data warehouse. This task involves several discussions with the source administrators to explain the codes and implicit rules or values, which are hidden in the data and the source programs. Moreover, it involves quite a few 'data preview' attempts (in the form of sampling, or simple counting queries) to discover the possible problems of the provided data. Naturally, this process is interactive and error-prone. The support that a tool can provide the designer with lies mainly in the area of inter-attribute mapping, with focus on the tracing of transformation and cleaning operations for this mapping.

For each target attribute, a set of provider relationships must be defined. In simple cases, the provider relationships are defined directly among source and target attributes. In the cases where a transformation is required, we pass the mapping through the appropriate transformation. In the cases where more than one transformations are required, we can insert a sequence of transformations between the involved attributes, all linked through composition relationships. ETL constraints are also specified in this part of the design process. An example of inter-attribute relationship for the concepts $\mathtt{PS}_1$, $\mathtt{PS}_2$, $\mathtt{S}_1$`.PARTSUPP`, $\mathtt{S}_2$`.PARTSUPP` and `DW.PARTSUPP`, is depicted in Fig. 8.

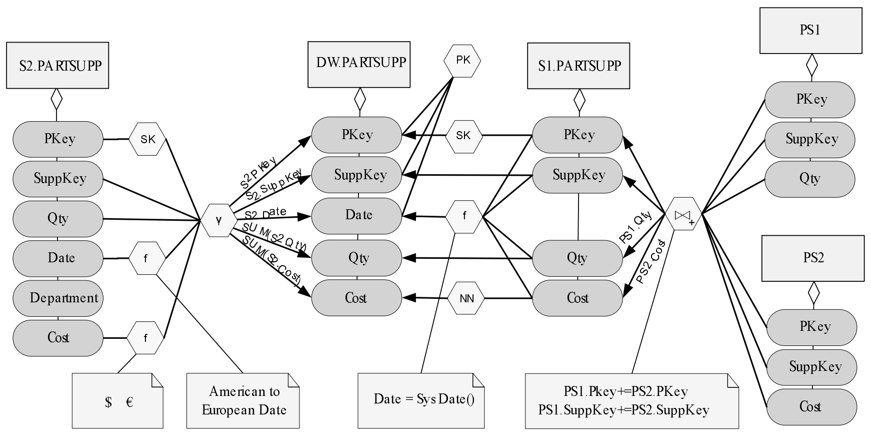

Fig. 8. Attribute mapping for the population of recordset `DW.PARTSUPP`

Step 4: Annotating the diagram with runtime constraints. Apart for the job definition for an ETL scenario, which specifies how the mapping from sources to the data warehouse is performed, along with the appropriate transformations, there are several other parameters that possibly need to be specified for the runtime environment. This kind of *runtime constraints* include:

– *Time/Event based scheduling*. The designer needs to determine the frequency of the ETL process, so that data are fresh and the overall process fits within the refreshment time window.
– *Monitoring*. On-line information about the progress/status of the process is necessary, so that the administrator can be aware of what step the load is on, its start time, duration, etc. File dumps, notification messages on the console, e-mail, printed pages, or visual demonstration can be employed for this purpose.
– *Logging*. Off-line information, presented at the end of the execution of the scenario, on the outcome of the overall process. All the relevant information (e.g., the previous tracing stuff) is shown, by employing the aforementioned techniques.
– *Exception handling*. The row-wise treatment of database/business rules violation is necessary for the proper function of the ETL process. Information like which rows are problematic, or how many rows are acceptable (acceptance rate, that is), characterizes this aspect of the ETL process.
– *Error handling*. Crash recovery and stop/start capabilities (e.g., committing a transaction every few rows) are absolutely necessary for the robustness and the efficiency of the process.

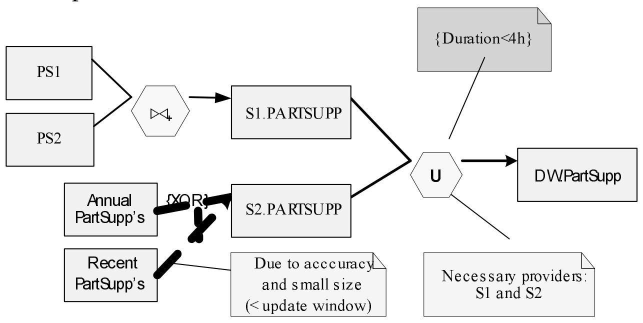

Fig. 9. Annotation of Fig. 5 with constraints on the runtime execution of the ETL scenario

To capture all this information, we adopt *notes*, tagged to the appropriate concepts, transformations or relationships. In Fig. 9 we depict the diagram of Fig. 5, annotated with a runtime constraint specifying that the overall execution time for the loading of `DW.PARTSUPP` (that involves the loading of S_1.`PARTSUPP` and S_2.`PARTSUPP`) cannot take longer than `4 hours`.

5 Related Work

There have been quite a few attempts around the conceptual modeling of data warehouses. Most of them are focused around the front-end of the data warehouse [4, 5, 8, 9, 10, 13, 14, 17, 19, 23, 24, 25, 26]. A variety of commercial ETL tools exist in the market [11, 12, 16, 20] (see also a recent review [6] for more details). Naturally, there also exist research efforts including [1, 2, 7, 15, 18, 21, 22, 28, 30] mostly targeting logical modeling and implementation issues of the ETL process (like duplicate elimination, logical modeling of ETL workflows, etc.).

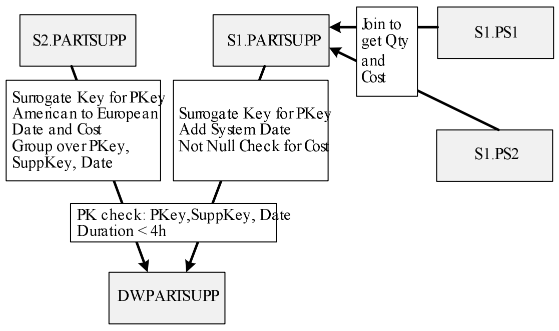

Fig. 10. Informal conceptual model following the [14] notation

The only methodology that we know of is found in [14], mainly grouped as a set of tips and technical guidelines; on the contrary, in our approach we try to give a more disciplined methodology. The major deficiency of the [14] methodology is the lack of conceptual support for the overall process. According to the [14] proposal (e.g., p.613), the closest thing to a conceptual model for our motivating example is the informal description depicted in Fig. 10. As far as inter-attribute mappings are concerned, tables of matching between source and warehouse attributes are suggested. The spirit of [14] is more focused towards the logical design of ETL workflows; thus, rather than systematically tracing system and user requirements, transformations and constraints, the proposed methodology quickly turns into a description of data flows for common cases of ETL transformations (e.g., surrogate keys, duplicate elimination and so on).

6 Conclusions

In this paper, we have proposed a methodology for the earliest stages of the data warehouse design, with the goal of tracing the analysis of the structure and content of the existing data sources and their intentional mapping to the common data warehouse model. As already suggested in previous research efforts [29] the major deliverable of this effort is a model of the attribute interrelationships. In this paper, we present a set of design steps that constitute the methodology for the design of the conceptual part of the overall ETL process. These steps could be summarized as follows: (a) identification of the proper data stores; (b) candidates and active candidates for the involved data stores; (c) attribute mapping between the providers and the consumers, and (d) annotating the diagram with runtime constraints.

Clearly, a lot of work remains to be done for the completion of our research approach. The main challenge is the practical application of this disciplined approach in real world cases and its further tuning to accommodate extra practical problems.

Moreover, a second challenge involves the linkage of our conceptual model and methodology to a logical model for ETL workflows.

References

[1] M. Bouzeghoub, F. Fabret, M. Matulovic. Modeling Data Warehouse Refreshment Process as a Workflow Application. In Proc. Intl. Workshop on Design and Management of Data Warehouses (DMDW'99), Heidelberg, Germany, (1999).

[2] V. Borkar, K. Deshmuk, S. Sarawagi. Automatically Extracting Structure from Free Text Addresses. Bulletin of the Technical Committee on Data Engineering, 23(4), (2000).

[3] G. Booch, I. Jacobson, J. Rumbaugh. The Unified Modeling Language User Guide. Addison-Wesley Pub Co; ISBN: 0201571684; 1st edition, October 1998.

[4] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Information integration: Conceptual modeling and reasoning support. In Proc. Proceedings of the International Conference on Cooperative Information Systems (COOPIS), New York, USA, pp. 280-291, (August 1998)

[5] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, R. Rosati. A principled approach to data integration and reconciliation in data warehousing. In Proc. Intl. Workshop on Design and Management of Data Warehouses (DMDW'99), Heidelberg, Germany, (1999).

[6] Gartner. ETL Magic Quadrant Update: Market Pressure Increases. Available at http://www.gartner.com/reprints/informatica/112769.html

[7] H. Galhardas, D. Florescu, D. Shasha and E. Simon. Ajax: An Extensible Data Cleaning Tool. In Proc. ACM SIGMOD Intl. Conf. On the Management of Data, pp. 590, Dallas, Texas, (2000).

[8] M. Golfarelli, D. Maio, S. Rizzi. The Dimensional Fact Model: a Conceptual Model for Data Warehouses. Invited Paper, International Journal of Cooperative Information Systems, vol. 7, n. 2&3, 1998.

[9] M. Golfarelli, S. Rizzi: Methodological Framework for Data Warehouse Design. In ACM First International Workshop on Data Warehousing and OLAP (DOLAP '98), pp. 3-9, November 1998, Bethesda, Maryland, USA.

[10] B. Husemann, J. Lechtenborger, G. Vossen. Conceptual data warehouse modeling. In Proc. 2nd Intl. Workshop on Design and Management of Data Warehouses (DMDW), pp. 6.1 –6.11, Stockholm, Sweden (2000).

[11] IBM. IBM Data Warehouse Manager. Available at http://www-3.ibm.com/software/data/db2/datawarehouse/

[12] Informatica. PowerCenter 6. Available at: http://www.informatica.com/products/data+integration/powercenter/default.htm

[13] R. Kimball. A Dimensional Modeling Manifesto. DBMS Magazine August 1997

[14] R. Kimbal, L. Reeves, M. Ross, W. Thornthwaite. The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses. John Wiley & Sons, February 1998.

[15] W. Labio, J.L. Wiener, H. Garcia-Molina, V. Gorelik. Efficient Resumption of Interrupted Warehouse Loads. In Proc. of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 46-57, Dallas, Texas, USA (2000).

[16] Microsoft Corp. MS Data Transformation Services. Available at www.microsoft.com/sq

[17] D.L. Moody, M.A.R. Kortink: From enterprise models to dimensional models: a methodology for data warehouse and data mart design. Proceedings of the Second Intl.

Workshop on Design and Management of Data Warehouses, DMDW 2000, Stockholm, Sweden, June 5-6, 2000. Available at http: //sunsite.informatik.rwthaachen.de/Publications/ CEUR-WS/Vol-28/

[18] A. Monge. Matching Algorithms Within a Duplicate Detection System. Bulletin of the Technical Committee on Data Engineering, 23(4), (2000).

[19] T. B. Nguyen, A Min Tjoa, R. R. Wagner. An Object Oriented Multidimensional Data Model for OLAP. Proc. of the First International Conference on Web-Age Information Management (WAIM-00), Shanghai, China, June 2000.

[20] Oracle Corp. Oracle9i™ Warehouse Builder User’s Guide, Release 9.0.2. November 2001. Product web page available at http://otn.oracle.com/products/warehouse/ content.html

[21] E. Rahm, H. Do. Data Cleaning: Problems and Current Approaches. Bulletin of the Technical Committee on Data Engineering, 23(4), (2000).

[22] V. Raman, J. Hellerstein. Potter's Wheel: An Interactive Data Cleaning System. In Proceedings of 27th International Conference on Very Large Data Bases (VLDB), pp. 381-390, Roma, Italy, (2001).

[23] C. Sapia, M. Blaschka, G. Höfling, B. Dinter: Extending the E/R Model for the Multidimensional Paradigm. In ER Workshops 1998, pp. 105-116. Lecture Notes in Computer Science 1552 Springer 1999

[24] J. Trujillo, M. Palomar, J. Gómez. Applying Object-Oriented Conceptual Modeling Techniques to the Design of Multidimensional Databases and OLAP Applications. Proc. of the First International Conference on Web-Age Information Management (WAIM-00) pp. 83-94, Shanghai, China, June 2000.

[25] J. Trujillo, M. Palomar, J. Gómez, I. Song. Designing Data Warehouses with OO Conceptual Models. IEEE Computer 34(12), pp. 66-75, 2001.

[26] N. Tryfona, F. Busborg, J.G.B. Christiansen. starER: A Conceptual Model for Data Warehouse Design. In ACM Second International Workshop on Data Warehousing and OLAP (DOLAP’99), pp.3-8, November 1999, Missouri, USA.

[27] P. Vassiliadis. Gulliver in the land of data warehousing: practical experiences and observations of a researcher. In Proc. 2nd Intl. Workshop on Design and Management of Data Warehouses (DMDW), pp. 12.1 –12.16, Sweden (2000).

[28] P. Vassiliadis, A. Simitsis, S. Skiadopoulos. Modeling ETL activities as graphs. In Proc. of Design and Management of Data Warehouses (DMDW'2002) 4th Intl Workshop in conjunction with CAiSE’02, pp. 52-61, Toronto, Canada, 2002.

[29] P. Vassiliadis, A. Simitsis, S. Skiadopoulos. Conceptual Modeling for ETL processes. In Proc. of Data Warehousing and OLAP (DOLAP'2002) ACM 5th Intl Workshop in conjunction with CIKM’02, McLean, USA, November 8, 2002.

[30] P. Vassiliadis, Z. Vagena, S. Skiadopoulos, N. Karayannidis, T. Sellis. ARKTOS: Towards the modeling, design, control and execution of ETL processes. Information Systems, 26(8), pp. 537-561, December 2001, Elsevier Science Ltd.