

[Journal of King Saud University – Computer and Information Sciences (2011) 23, 91](http://dx.doi.org/10.1016/j.jksuci.2011.05.005)–[104](http://dx.doi.org/10.1016/j.jksuci.2011.05.005)

King Saud University

Journal of King Saud University – Computer and Information Sciences

[www.ksu.edu.sa](http://www.ksu.edu.sa/) [www.sciencedirect.com](http://www.sciencedirect.com/science/journal/13191578)

ORIGINAL ARTICLE

A proposed model for data warehouse ETL processes

Shaker H. Ali El-Sappagh [a](#_bookmark0),[\*](#_bookmark2), Abdeltawab M. Ahmed Hendawi [b](#_bookmark1),

Ali Hamed El Bastawissy [b](#_bookmark1)

a *Mathematics Department, College of Science, King Saud University, Saudi Arabia*

b *Information Systems Department, Faculty of Computers and Information, Cairo University, Cairo, Egypt*

Received 19 September 2010; accepted 23 February 2011

Available online 8 May 2011

Abstract Extraction–transformation–loading (ETL) tools are pieces of software responsible for the extraction of data from several sources, its cleansing, customization, reformatting, integration, and insertion into a data warehouse. Building the ETL process is potentially one of the biggest tasks of building a warehouse; it is complex, time consuming, and consume most of data warehouse pro- ject’s implementation efforts, costs, and resources. Building a data warehouse requires focusing clo- sely on understanding three main areas: the source area, the destination area, and the mapping area (ETL processes). The source area has standard models such as entity relationship diagram, and the destination area has standard models such as star schema, but the mapping area has not a standard model till now. In spite of the importance of ETL processes, little research has been done in this area due to its complexity. There is a clear lack of a standard model that can be used to represent the ETL scenarios. In this paper we will try to navigate through the efforts done to conceptualize

KEYWORDS

Data warehouse; ETL processes; Database;

Data mart; OLAP;

Conceptual modeling

*Abbreviations*: ETL, extraction–transformation–loading; DW, data warehouse; DM, data mart; OLAP, on-line analytical processing; DS, data sources; ODS, operational data store; DSA, data staging area; DBMS, database management system; OLTP, on-line transaction processing; CDC, change data capture; SCD, slowly changing dimension; FCME, ﬁrst-class modeling elements; EMD, entity mapping diagram; DSA, data storage area.

* Corresponding author.

E-mail addresses: [Sker@ksu.edu.sa](mailto:Sker@ksu.edu.sa) (S.H. Ali El-Sappagh), [Abdeltawab\_fci@yahoo.com](mailto:Abdeltawab_fci@yahoo.com) (A.M. Ahmed Hendawi), [aelbastawesy@](mailto:aelbastawesy@fci-cu.edu.eg) [fci-cu.edu.eg](mailto:aelbastawesy@fci-cu.edu.eg) (A.H. El Bastawissy).

1319-1578 ª 2011 King Saud University. Production and hosting by Elsevier B.V. All rights reserved.

Peer review under responsibility of King Saud University. doi:[10.1016/j.jksuci.2011.05.005](http://dx.doi.org/10.1016/j.jksuci.2011.05.005)



Production and hosting by Elsevier

the ETL processes. Research in the ﬁeld of modeling ETL processes can be categorized into three main approaches: Modeling based on mapping expressions and guidelines, modeling based on con- ceptual constructs, and modeling based on UML environment. These projects try to represent the main mapping activities at the conceptual level. Due to the variation and differences between the proposed solutions for the conceptual design of ETL processes and due to their limitations, this paper also will propose a model for conceptual design of ETL processes. The proposed model is built upon the enhancement of the models in the previous models to support some missing mapping features.

ª 2011 King Saud University. Production and hosting by Elsevier B.V. All rights reserved.

1. Introduction

A data warehouse (DW) is a collection of technologies aimed at enabling the decision maker to make better and faster deci- sions. Data warehouses differ from operational databases in that they are subject oriented, integrated, time variant, non volatile, summarized, larger, not normalized, and perform OLAP. The generic data warehouse architecture consists of three layers (data sources, DSA, and primary data warehouse) ([Inmon, 2002; Vassiliadis, 2000](#_bookmark25)). Although ETL processes area is very important, it has little research. This is because of its difﬁculty and lack of formal model for representing ETL activ- ities that map the incoming data from different DSs to be in a suitable format for loading to the target DW or DM ([Kimball](#_bookmark25) [and Caserta, 2004; Demarest, 1997; Oracle Corp., 2001; In-](#_bookmark25) [mon, 1997](#_bookmark25)). To build a DW we must run the ETL tool which has three tasks: (1) data is extracted from different data sources, (2) propagated to the data staging area where it is transformed and cleansed, and then (3) loaded to the data warehouse. ETL tools are a category of specialized tools with the task of dealing with data warehouse homogeneity, clean- ing, transforming, and loading problems ([Shilakes and Tyl-](#_bookmark26) [man, 1998](#_bookmark26)). This research will try to ﬁnd a formal representation model for capturing the ETL processes that map the incoming data from different DSs to be in a suitable format for loading to the target DW or DM. Many research projects try to represent the main mapping activities at the conceptual level. Our objective is to propose conceptual model to be used in modeling various ETL processes and cover the limitations of the previous research projects. The proposed model will be used to design ETL scenarios, and document, customize, and simplify the tracing of the mapping between the data source attributes and its corresponding in data ware- house. The proposed model has the following characteristics:

* + *Simple*: to be understood by the DW designer.
  + *Complete*: to represent all activities of the ETL processes.
  + *Customizable*: to be used in different DW environments.

We call the proposed model entity mapping diagram (EMD). Also, the paper will make a survey of the previous work done in this area. The paper will be organized as follows: Section 2 will discuss the ETL modeling concepts. The ETL processes related or previous work is discussed in Section 3. We will discuss the proposed framework in Section 4. The

1. ETL modeling concepts

The general framework for ETL processes is shown in [Fig. 1](#_bookmark3). Data is *extracted* from different data sources, and then prop- agated to the DSA where it is *transformed* and cleansed be- fore being *loaded* to the data warehouse. Source, staging area, and target environments may have many different data structure formats as ﬂat ﬁles, XML data sets, relational tables, non-relational sources, web log sources, legacy sys- tems, and spreadsheets.

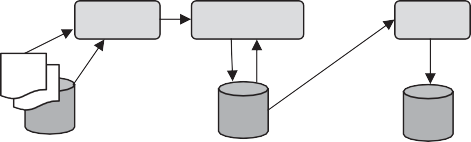
* 1. *The ETL phases*

During the ETL process, data is extracted from an OLTP dat- abases, transformed to match the data warehouse schema, and loaded into the data warehouse database ([Berson and Smith,](#_bookmark22) [1997; Moss, 2005](#_bookmark22)). Many data warehouses also incorporate data from non-OLTP systems, such as text ﬁles, legacy sys- tems, and spreadsheets. ETL is often a complex combination of process and technology that consumes a signiﬁcant portion of the data warehouse development efforts and requires the skills of business analysts, database designers, and application developers. The ETL process is not a one-time event. As data sources change the data warehouse will periodically updated. Also, as business changes the DW system needs to change – in order to maintain its value as a tool for decision makers, as a result of that the ETL also changes and evolves. The ETL processes must be designed for ease of modiﬁcation. A solid, well-designed, and documented ETL system is necessary for the success of a data warehouse project.

An ETL system consists of three consecutive functional steps: extraction, transformation, and loading:

* + 1. *Extraction*

The ﬁrst step in any ETL scenario is data extraction. The ETL extraction step is responsible for extracting data from the source systems. Each data source has its distinct set of charac- teristics that need to be managed in order to effectively extract data for the ETL process. The process needs to effectively inte- grate systems that have different platforms, such as different

Extract Transform Load

comparison between the previous model and proposed one is discussed in Section 5. Next, other related works will be shown in Section 6. Finally, Section 7 shows the conclusion and fu-

Data Sources

DSA DW

ture work. Figure 1 A general framework for ETL processes.

database management systems, different operating systems, and different communications protocols.

During extracting data from different data sources, the ETL team should be aware of (a) using ODBC JDBC drivers con- nect to database sources, (b) understand the data structure of sources, and (c) know how to handle the sources with different nature such as mainframes. The extraction process consists of two phases, initial extraction, and changed data extraction. In *the initial extraction* ([Kimball et al., 1998](#_bookmark25)), it is the ﬁrst time to get the data from the different operational sources to be loaded into the data warehouse. This process is done only one time after building the DW to populate it with a huge amount of data from source systems. *The incremental extraction* is called changed data capture (CDC) where the ETL processes refresh the DW with the modiﬁed and added data in the source systems since the last extraction. This process is periodic according to the refresh cycle and business needs. It also captures only chan- ged data since the last extraction by using many techniques as audit columns, database log, system date, or delta technique.

n

* + 1. *Transformation*

The second step in any ETL scenario is data transformation. The transformation step tends to make some cleaning and con- forming on the incoming data to gain accurate data which is correct, complete, consistent, and unambiguous. This process includes data cleaning, transformation, and integration. It de- ﬁnes the granularity of fact tables, the dimension tables, DW schema (stare or snowﬂake), derived facts, slowly changing dimensions, factless fact tables. All transformation rules and the resulting schemas are described in the metadata repository.

* + 1. *Loading*

Loading data to the target multidimensional structure is the ﬁ- nal ETL step. In this step, extracted and transformed data is written into the dimensional structures actually accessed by the end users and application systems. Loading step includes both loading dimension tables and loading fact tables.

1. Models of ETL processes

This section will navigate through the efforts done to concep- tualize the ETL processes. Although the ETL processes are critical in building and maintaining the DW systems, there is a clear lack of a standard model that can be used to represent the ETL scenarios. After we build our model, we will make a comparison between this model and models discussed in this section. Research in the ﬁeld of modeling ETL processes can be categorized into three main approaches:

1. Modeling based on mapping expressions and guidelines.
2. Modeling based on conceptual constructs.
3. Modeling based on UML environment.

In the following, a brief description of each approach is presented.

* 1. *Modeling ETL process using mapping expressions*

[Rifaieh and Benharkat (2002)](#_bookmark26) have deﬁned a model covering different types of mapping expressions. They used this model to create an active ETL tool. In their approach, queries are used

to achieve the warehousing process. Queries will be used to rep- resent the mapping between the source and the target data; thus, allowing DBMS to play an expanded role as a data trans- formation engine as well as a data store. This approach enables a complete interaction between mapping metadata and the warehousing tool. In addition, it addresses the efﬁciency of a query-based data warehousing ETL tool without suggesting any graphical models. It describes a query generator for reus- able and more efﬁcient data warehouse (DW) processing.

* + 1. *Mapping guideline*

Mapping guideline means the set of information deﬁned by the developers in order to achieve the mapping between the attri- butes of two schemas. Actually, different kinds of mapping guidelines are used for many applications. Traditionally, these guidelines are deﬁned manually during the system implementa- tion. In the best case, they are saved as paper documents. These guidelines are used as references each time there is a need to understand how an attribute of a target schema has been gener- ated from the sources attributes. This method is very weak in the maintenance and evolution of the system. To keep updating these guidelines is a very hard task, especially with different ver- sions of guidelines. To update the mapping of an attribute in the system, one should include an update for the paper document guideline as well. Thus, it is extremely difﬁcult to maintain such tasks especially with simultaneous updates by different users.

* + 1. *Mapping expressions*

Mapping expression of an attribute is the information needed to recognize how a target attribute is created from the sources attributes. Examples of the applications where mapping expressions are used are listed as follows:

*Schema mapping* ([Madhavan et al., 2001](#_bookmark27)): for database schema mapping, the mapping expression is needed to deﬁne the correspondence between matched elements.

●

*Data warehousing tool* (*ETL*) ([Staudt et al., 1999](#_bookmark28)): includes a transformation process where the correspondence between the sources data and the target DW data is deﬁned. *EDI message mapping*: the need of a complex message trans- lation is required for EDI, where data must be transformed from one EDI message format into another.

●

●

*EAI* (*enterprise application integration*): the integration of information systems and applications needs a middleware to manage this process ([Stonebraker and Hellerstein,](#_bookmark29) [2001](#_bookmark29)). It includes management rules of an enterprise’s appli- cations, data spread rules for concerned applications, and data conversion rules. Indeed, data conversion rules deﬁne the mapping expression of integrated data.

●

* + 1. *Mapping expression examples*

Some examples of the mapping expressions identiﬁed from dif- ferent type of applications are shown as follows:

*Break-down*/*concatenation*: in this example the value of a ﬁeld is established by breaking down the value of a source and by concatenating it with another value, as shown in [Fig. 2](#_bookmark4).

●

*Conditional mapping*: sometimes the value of a target attri- bute depends on the value of another attribute. In the exam- ple, if X = 1 then Y = A else Y = B, as shown in [Fig. 3](#_bookmark5). More about mapping expression rules and notation are found in [Jarke et al. (2003) and Miller et al. (2000)](#_bookmark25).

●

Figure 2 Example 1: Break-down/concatenation ([Jarke et al.,](#_bookmark25) [2003](#_bookmark25)).

|  |
| --- |
|  |
| DD\MM\AA |
| 1234 - XYZ |

|  |
| --- |
| XYZ |
|  |
| DD\MM\AA |
|  |
| 1234 |

|  |
| --- |
|  |
| AA |
| DDMM |

* 1. *Modeling ETL processes using conceptual constructs*

In [Vassiliadis et al. (2002a, 2003, 2005)](#_bookmark30) the authors attempt to provide a ﬁrst model towards the conceptual modeling of the data warehousing ETL processes. They introduce a framework for the modeling of ETL activities. Their framework contains three layers, as shown in [Fig. 4](#_bookmark7).

*The lower layer* namely; schema layer, involves a speciﬁc ETL scenario. All the entities of the schema layer are *instances* of the classes data type, function type, elementary activity, recordset and relationship.

*The higher layer* namely; metamodel layer involves the aforementioned classes. The linkage between the metamodel and the schema layers is achieved through instantiation (‘‘instanceOf’’) relationships. The metamodel layer implements the aforementioned generality: the ﬁve classes which are in- volved in the metamodel layer are generic enough to model any ETL scenario, through the appropriate instantiation.

*The middle layer* is the template layer. The constructs in the template layer are also meta-classes, but they are quite custom-

ized for the regular cases of ETL processes. Thus, the classes of the template layer represent specializations (i.e., subclasses) of the generic classes of the metamodel layer (depicted as ‘‘IsA’’ relationships). After deﬁning the previous framework, the authors present the graphical notation and the metamodel of their proposed graphical model as shown in [Fig. 5](#_bookmark6). Then, they detail and formally deﬁne all the entities of the metamodel:

* *Data types*. Each data type T is characterized by a name and a domain which is a countable set of values. The values of the domains are also referred to as constants.
* *Recordsets*. A recordset is characterized by its name, its log- ical schema (structure of the recordset) and its physical extension (i.e., a ﬁnite set of records under the recordset schema) which is the actual records values. Any data struc- ture can be treated as a ‘‘record set’’ provided that there are the means to logically restructure it into a ﬂat, typed record schema. The two most popular types of recordsets are namely relational tables and record ﬁles.
* *Functions*. A function type comprises a name, a ﬁnite list of parameter data types, and a single return data type. A func- tion is an instance of a function type.
* *Elementary activities*. Activities are logical abstractions rep- resenting parts, or full modules of code. An abstraction of the source code of an activity is employed, in the form of a LDL (logic-programming, declarative language) state- ment, in order to avoid dealing with the peculiarities of a particular programming language ([Naqvi and Tsur, 1989](#_bookmark26)).

Attribute



Note

Concept



Transformation ETL\_Constraint

Part Of Provider 1:1 Provider N: M Serial



A

B

X **?**

67899

2

67899

12345

Y Composition

Active Candidate Candidate Candidate

Figure 3 Example 2: Conditional mapping ([Jarke et al., 2003](#_bookmark25)).



Figure 5 Notations for the conceptual modeling of ETL activities ([Vassiliadis et al., 2002a](#_bookmark30)).



Data types

Functions

Metamodel Layer

Elementary Activity

RecordSet

Relationships

IsA

NotNull

Domain Mismatch

Source Table

Template Layer

SK Assignment

Fact Table

Provider Rel

InstanceOf

S1.PW

NN1

DM1

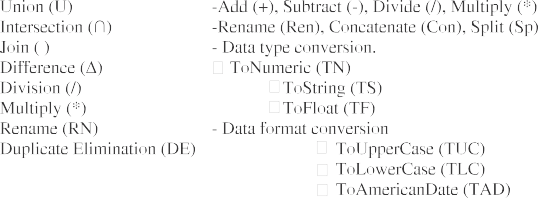
SK1

DW.PS

Schema Layer

Figure 4 The metamodel for the logical entities of the ETL environment ([Vassiliadis et al., 2003](#_bookmark33)).

– *Relationships*. Depict the follow of data from the sources to the target.



Then the authors use their graphical model to represent ETL processes in a motivating example. As shown in [Fig. 6](#_bookmark9), two data sources (S1.partsupp and S2.partsupp) are used to build the target data warehouse (DW.partsupp). The concep- tual model of [Vassiliadis et al. (2002a)](#_bookmark30) is complemented in [Vassiliadis et al. (2002b, 2003) and Simitsis (2003)](#_bookmark31) with the logical design of ETL processes as data-centric workﬂows. In [Vassiliadis et al. (2003)](#_bookmark33) the authors describe a framework for the declarative speciﬁcation of ETL scenarios. They discuss the implementation issues and they present a graphical tool ‘ARKTOS II’ that facilitates the design of ETL scenarios, based on their model. In [Vassiliadis et al. (2002b)](#_bookmark31) the authors model an ETL scenario as a graph which they call architectural graph and they introduce some notations for this graph. They introduce importance metrics to measure the degree to which entities are bound to each other. In [Simitsis (2003)](#_bookmark26) the author focuses on the optimization of the ETL processes, in order to minimize the execution time of an ETL process. Regarding data mapping, in [Dobre et al. (2003)](#_bookmark23) authors discuss issues re- lated to the data mapping in the integration of data, and a set of mapping operators is introduced and a classiﬁcation of pos- sible mapping cases is presented, as shown in [Fig. 7](#_bookmark8). However, no graphical representation of data mapping scenarios is pro- vided, hence, it is difﬁcult to be used in real world projects. In [Bernstein and Rahm (2000)](#_bookmark21) a framework for mapping between models (objects) is proposed.

Models are manipulated by a role of high-level operations including:

Match – create a mapping between two models.

●

Apply Function – apply a given function to all objects in a model.

●

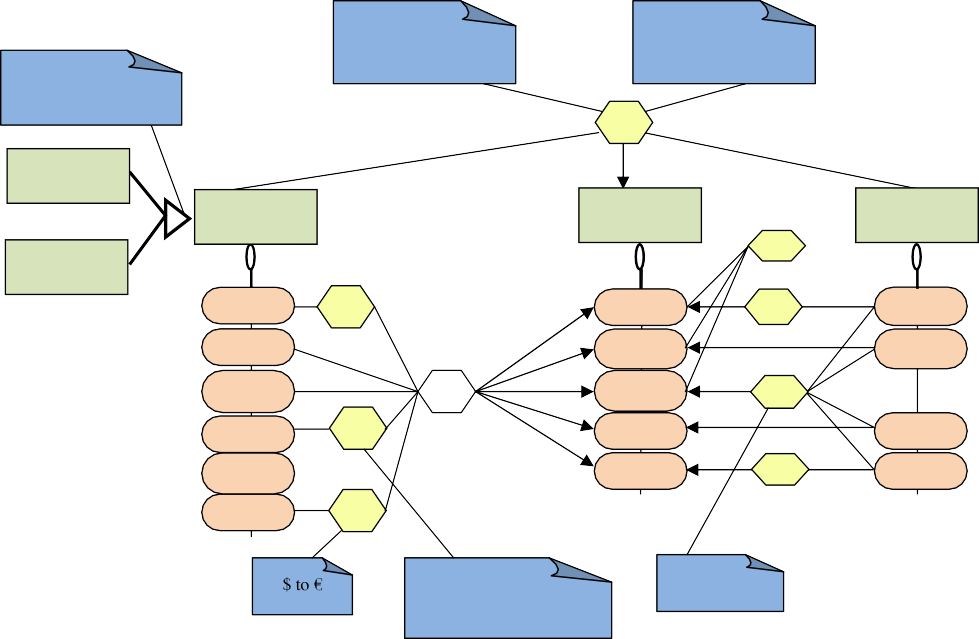
Figure 7 Sample mapping operators.

* Union, Intersection, Difference – applied to a set of objects.
* Delete – delete all objects in a model.
* Insert, Update – applied to individual objects in models.
  1. *Modeling based on UML environment*

In [Lujan-Mora et al. (2004)](#_bookmark25) the authors introduce their model that is based on the UML (uniﬁed modeling language) nota- tions. It is known that UML does not contain a direct relation- ship between attributes in different classes, but the relationship is established between the classes itself, so the authors extend UML to model attributes as ﬁrst-class citizens. In their attempt to provide complementary views of the design artifacts in dif- ferent levels of detail, the framework is based on a principled approach in the usage of UML packages, to allow zooming in and out the design of a scenario.

* + 1. *Framework*

The architecture of a data warehouse is usually depicted as various layers of data in which data from one layer is derived from the data of the previous layer ([Lujan-Mora and Trujillo,](#_bookmark25) [2003](#_bookmark25)). Following this consideration, the development of a DW can be structured into an integrated framework with ﬁve stages



Necessary Provider

Duration < 4h

and small size

Due to accuracy

S1 and S2

U

Annual

PartSupp's

S2.Partsupp

DW.Partsupp

S2.Partsupp

Partsupp's

Recent

PPKKeeyy SupKey Qty

Date

Dept

Cost

SK

PK

SK

V

F

PKey

SupKey Qty Date

Cost

Pkey

Supkey

F

NN

Date

Cost

F

American to European

SysDate

date

Figure 6 Motivating example for conceptual model in [Vassiliadis et al. (2002a)](#_bookmark30).

and three levels that deﬁne different diagrams for the DW model, as explained below:

– Phases: there are ﬁve stages in the deﬁnition of a DW: *Source*: it deﬁnes the data sources of the DW, such as OLTP systems, external data sources.

●

*Integration*: it deﬁnes the mapping between the data sources and the data warehouse.

●

*Data warehouse*: it deﬁnes the structure of the data warehouse.

●

*Customization*: it deﬁnes the mapping between the data warehouse and the clients’ structures.

●

*Client*: it deﬁnes special structures that are used by the cli- ents to access the data warehouse, such as data marts or OLAP applications.

●

– Levels: each stage can be analyzed at three levels or perspectives:

*Conceptual*: it deﬁnes the data warehouse from a conceptual point of view.

●

*Logical*: it addresses logical aspects of the DW design, as the deﬁnition of the ETL processes.

●

*Physical*: it deﬁnes physical aspects of the DW, such as the storage of the logical structures in different disks, or the conﬁguration of the database servers that support the DW.

●

* + 1. *Attributes as ﬁrst-class modeling elements (FCME)* Both in ERD model and in UML, attributes are embedded in the deﬁnition of their comprising ‘‘element’’ (an entity in the ER or a class in UML), and it is not possible to create a rela- tionship between two attributes. In order to allow attributes to play the same role in certain cases, the authors propose the representation of attributes as FCME in UML. In a UML class diagram, two kinds of modeling elements are treated as FCME. Classes, as abstract representations of real-world enti- ties are naturally found in the center of the modeling effort. Being FCME, classes acting as attribute containers. The rela- tionships between classes are captured by associations. Associ- ations can also be FCME, called association classes. An association class can contain attributes or can be connected to other classes. However, the same is not possible with attri- butes. They refer to the class that contains the attributes as the container class and the class that represents an attribute as the

attribute class. The authors formally deﬁne attribute/class dia- grams, along with the new stereotypes, ÆÆAttributeææ and ÆÆContainææ, deﬁned as follows:

*Attribute classes* are materializations of the ÆÆAttributeææ

stereotype, introduced speciﬁcally for representing the attri- butes of a class. The following constraints apply for the correct deﬁnition of an attribute class as a materialization of an ÆÆAttributeææ stereotype:

* *Naming convention:* the name of the attribute class is the name of the corresponding container class, followed by a dot and the name of the attribute.
* *Features*: an attribute class can contain neither attributes nor methods.

*A contain relationship* is a composite aggregation between a container class and its corresponding attribute classes, origi- nated at the end near the container class and highlighted with the ÆÆContainææ stereotype.

*An attribute/class diagram* is a regular UML class diagram extended with ÆÆAttributeææ classes and ÆÆContainææ relationships.

In the data warehouse context, the relationship, involves three logical parties: (a) the provider entity (schema, table, or attribute), responsible for generating the data to be further propagated, (b) the consumer, that receives the data from the provider and (c) their intermediate matching that involves the way the mapping is done, along with any transformation and ﬁltering. Since a mapping diagram can be very complex, this approach offers the possibility to organize it in different levels thanks to the use of UML packages.

Their layered proposal consists of four levels as shown in [Fig. 8](#_bookmark10):

* *Database level (or level 0*). In this level, each schema of the DW environment (e.g., data sources at the conceptual level in the SCS ‘source conceptual schema’, conceptual schema of the DW in the DWCS ‘data warehouse conceptual schema’, etc.) is represented as a package ([Lujan-Mora and Trujillo, 2003; Trujillo and Lujan-Mora, 2003](#_bookmark25)). The mappings among the different schemata are modeled in a single mapping package, encapsulating all the lower-level mappings among different schemata.

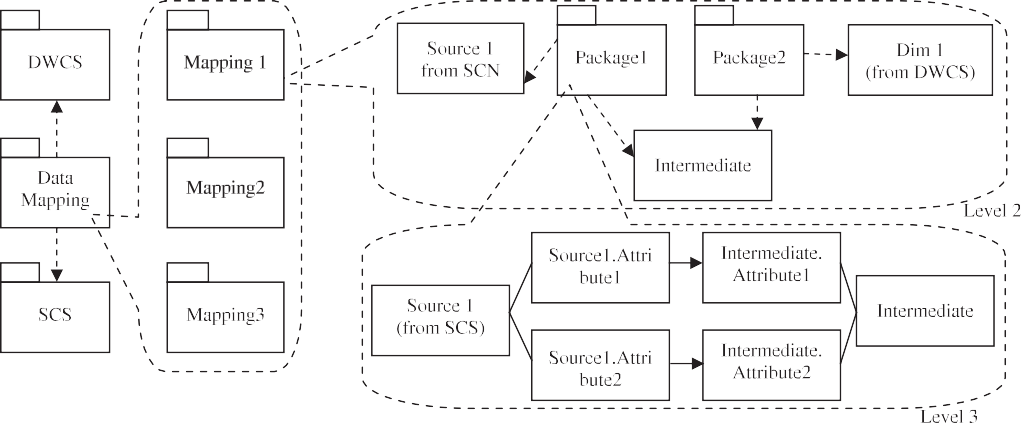


Figure 8 Data mapping levels ([Lujan-Mora et al., 2004](#_bookmark25)).

* *Dataﬂowlevel (or level 1*). This level describes the data rela- tionship among the individual source tables of the involved schemata towards the respective targets in the DW. Practi- cally, a mapping diagram at the database level is zoomed- into a set of more detailed mapping diagrams, each captur- ing how a target table is related to source tables in terms of data.
* *Table level (or level 2*). Whereas the mapping diagram of the dataﬂow level describes the data relationships among sources and targets using a single package, the mapping dia- gram at the table level, details all the intermediate transfor- mations and checks that take place during this ﬂow. Practically, if a mapping is simple, a single package that represents the mapping can be used at this level; otherwise, a set of packages is used to segment complex data mappings in sequential steps.
* *Attributelevel (or level 3*). In this level, the mapping diagram involves the capturing of inter-attribute mappings. Practi- cally, this means that the diagram of the table is zoomed- in and the mapping of provider to consumer attributes is traced, along with any intermediate transformation and cleaning.

At the leftmost part of [Fig. 8](#_bookmark10), a simple relationship among the DWCS and the SCS exists: this is captured by a single data mapping package and these three design elements constitute the data mapping diagram of the database level (or level 0). Assuming that there are three particular tables in the DW that we would like to populate, this particular data mapping pack- age abstracts the fact that there are three main scenarios for the population of the DW, one for each of these tables. In the dataﬂow level (or level 1) of our framework, the data rela- tionships among the sources and the targets in the context of each of the scenarios, is practically modeled by the respective package. If we zoom in one of these scenarios, e.g., mapping 1, we can observe its particularities in terms of data transfor- mation and cleaning: the data of source 1 are transformed in two steps (i.e., they have undergone two different transforma- tions), as shown in [Fig. 8](#_bookmark10). Observe also that there is an inter- mediate data store employed, to hold the output of the ﬁrst transformation (Step 1), before passed onto the second one (Step 2). Finally, at the right lower part of [Fig. 8](#_bookmark10), the way the attributes are mapped to each other for the data stores source 1 and intermediate is depicted. Let us point out that in case we are modeling a complex and huge data warehouse, the attribute transformation modeled at level 3 is hidden with- in a package deﬁnition.

1. The proposed ETL processes model (EMD)

To conceptualize the ETL processes used to map data from sources to the target data warehouse schema, we studied the previous research projects, made some integration, and add some extensions to the approaches mentioned above. We pro- pose entity mapping diagram (EMD) as a new conceptual model for modeling ETL processes scenarios. Our proposed model mainly follows the approach of modeling based on con- ceptual constructs. The proposed model will fulﬁll six require- ments ([El Bastawesy et al., 2005; Maier, 2004; Arya et al.,](#_bookmark24) [2006](#_bookmark24)):

1. Supports the integration of multiple data sources.
2. Is robust in view of changing data sources.
3. Supports ﬂexible transformations.
4. Can be easily deployed in a suitable implementation environment.
5. Is complete enough to handle the various extraction, trans- formation, and loading operations.
6. Is simple in creating and maintaining.

In this section, we will describe EMD framework, EMD metamodel, primitives of EMD constructs, and ﬁnally we will provide a demonstration example. A comparison and evalua- tion of the previous approaches against our proposed model will be presented in Section 5.

* 1. *EMD framework*

[Fig. 9](#_bookmark11) shows the general framework of the proposed entity mapping diagram.

* In the data source(s) part: the participated data sources are drawn. The data sources may be structured databases or non-structured sources. In case of structured sources; the participated databases and their participated tables and attributes are used directly as the base source, and in case of non-structured sources; a conversion step should be applied ﬁrst to convert the non-structured source into struc- tured one (tables and its attributes). From the design view, there is one conversion construct that can convert any non- structured source into structured (relational) database, but from the implementation view, each type of non-structured source will have its own conversion module which is called wrapper. Wrappers are specialized program routines that automatically extract data from different data sources with

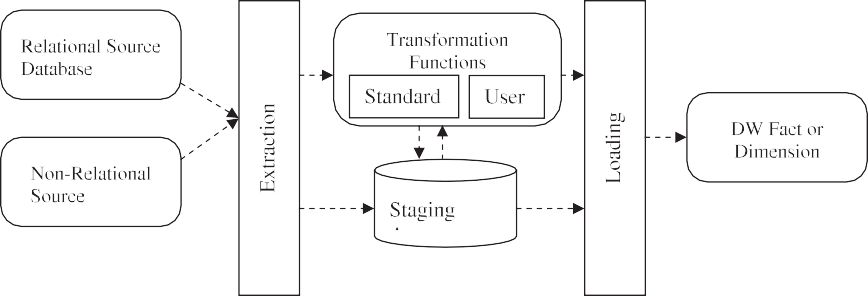


Figure 9 A general framework of EMD.

different formats and convert the information into a struc- tured format. The typical tasks of a wrapper are: (a) fetch- ing data from a remote resource, (b) searching for, recognizing and extracting speciﬁed data, and (c) saving this data in a suitable structured format to enable further manipulation ([Vassiliadis et al., 2005](#_bookmark35)).

* + Extraction: during the extraction process some temporary tables may be created to hold the result of converting non-structured sources into databases. The extraction pro- cess includes initial extraction and refresh. The initial extraction takes place when the ETL scenario executed for the ﬁrst time while there is no data in the destination data warehouse. The refresh extraction takes place to cap- ture the delta data (difference between old data in the DW and updated data in the data sources). It is preferred to separate the ETL scenario with initial extraction from the ETL scenario with refresh extraction. *This means that the user may need to build two EMD models for the same ETL scenario*; one for the initial extraction, and the other for the refresh extraction using the old data in the temp tables found in the staging area.
  + In the DW schema part: the data warehouse schema table (fact or dimension) is drawn. In spite of that the fact table and the dimension table are clearly different in their func- tionalities and features but all of them are data containers. Basically the data warehouse is stored as relational struc- ture not as multidimensional structure. The multidimen- sionality occurs in the online analytical processing (OLAP) engines.
  + In the mapping part: the required transformation functions are drawn. The transformation operations take place on the incoming data from both the base source and/or the tempo- rary source in the staging area. Some transformation oper- ations lead to temporary results which are saved in temporary tables in the staging area.
  + The staging area: a physical container that contains all tem- porary tables created during the extraction process or resulted from the applied transformation functions.
  + Loading: as the data reaches the ﬁnal appropriate format, it is loaded to the corresponding data element in the destina-

tion DW schema. The data may be loaded directly as a result of certain transformation function or captured from the desired temporary tables in the staging area.

Notice that both data sources and data warehouse schemas should be deﬁned clearly before starting to draw EMD. Also the arrows’ directions show that ﬁrst, the data sources are drawn, after that a set of transformation are applied, and then the data are loaded to the destination data warehouse schema.

* 1. *EMD metamodel*

EMD is a proposed conceptual model for modeling the ETL processes which are needed to map data from sources to the target data warehouse schema. [Fig. 10](#_bookmark12) shows the metamodel architecture for the proposed conceptual model EMD. The metamodel of the proposed EMD is composed of two layers; the ﬁrst layer is abstraction layer in which ﬁve objects (func- tion, data container, entity, relationship, and attribute) are clearly deﬁned. The objects in the abstraction layer are a high level view of the parts or objects that can be used to draw an EMD scenario.

The second layer is the template layer which is an expansion to the abstraction layer.

The link between the abstraction layer and the template layer may be considered as an aggregation relationship. A function may be an attribute transformation, an entity trans- formation, a UDF (user deﬁned function), or convert into structure (relation). [Fig. 11](#_bookmark13) shows the types of transformation functions that can be applied to sources in the proposed EMD.

An entity transformation is a function that can be applied to a source table (e.g. duplicate elimination, union, etc.). An attribute transformation function can be applied to a source attribute (e.g. to upper case, to String, etc.). A user deﬁned function (UDF) is any function that may be added by the user who is the creator of the ETL scenario (e.g. uniﬁcation be- tween different types of units). Convert into structure is a func- tion that can be applied to the non-structured (semi-structured and unstructured) data sources so that it can be converted into structured source to enable the other transformation functions

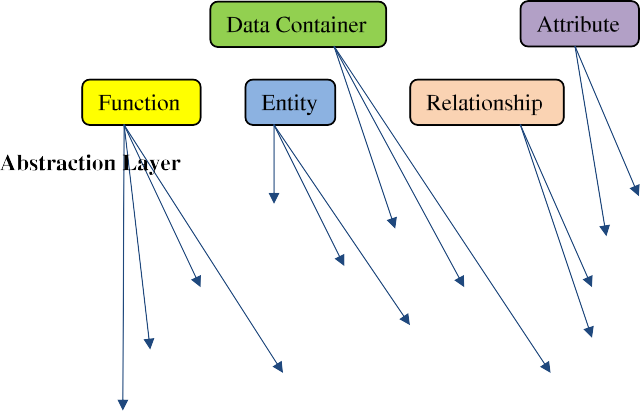
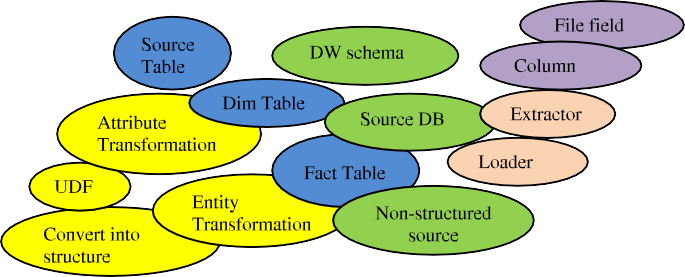


Figure 10 EMD metamodel.

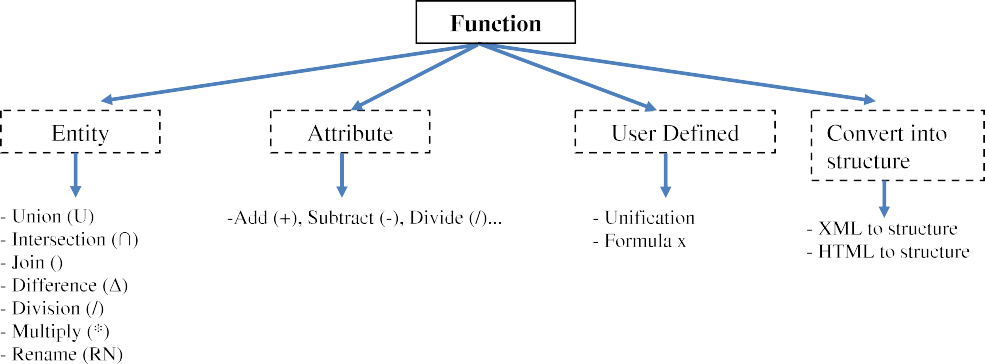
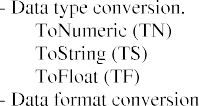


Figure 11 Types of transformations in EMD.

to be applied on it. A data container may be a source database, a target data warehouse or data mart, or non-structured source. An entity may be a source table, a dimension table, or a fact table. A relationship may be an extractor or a loader. The extractor expresses the data extraction process from the source and the loader expresses the data loading process to the ﬁnal destination. The attribute may be a table column or a non-structured ﬁle ﬁeld.

The metamodel can be expanded to include any extra ob- jects that may be required in the future. The user can use in- stances of the template layer to create his model to build the desired ETL scenario. It should be mentioned here that the user of EMD is the data warehouse designer or the ETL de- signer; this means that some primitive rules, limitations, and constrains are kept in his mind during the usage of different parts of EMD, i.e., union operation will be applied successfully when the participated tables have the same number of attributes with the same data types for the corresponding attributes.

* 1. *Primitives of EMD constructs*

The basic set of constructs that is used in the proposed entity mapping diagram are shown in [Fig. 12](#_bookmark14). In this section, some explanation about the usage of the constructs of the proposed entity mapping diagram will be given, as follows:

*Loader relationship*: is used when the data are moved directly from the last source element (the actual source or the temporary one) to the target data element. The actual source; is the base source from which the data are extracted, on the other hand, the temporary source; is the one that is resulted during the transformation operations.

●

*Optional loader relationship*: is used to show that the loaded data to the output attribute could be extracted from candi- date source element x or candidate source element y.

●

*Convert into structure*: represents the conversion operations required to restructure the non-structured base source into structured one (relations as tables and attributes). The



●

|  |  |  |
| --- | --- | --- |
|  | |  |
|  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | |
|  |  | |
|  |  | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Figure 12 Graphical constructs for the proposed EMD.



conversion operation saves its result into temporary tables, so the transformation operation can be applied to the new temporary source.

*Entity transformation operation*: this kind of transforma- tions usually results in a temporary entity. There are stan- dard operators that are used inside this construct, [Fig. 11](#_bookmark13)(a) shows some of these operators.

●

*Attribute transformation operation*: standard operations are used with this construct, [Fig. 11](#_bookmark13)(b) shows sample of these operators.

●

*User deﬁned function* (*UDF*) *as a transformation operation*: user can use his deﬁned operations, so any kind of transfor- mation can be added, such as currency conversion func-

●

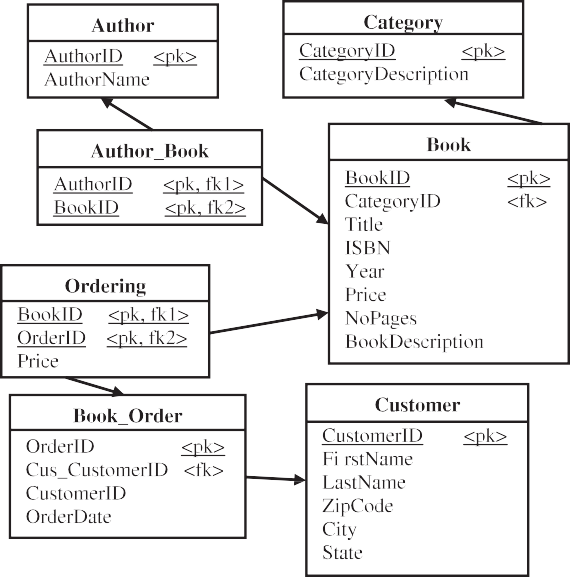


Figure 13 Relational schema DS1 for books-orders database.

tions, packages (units) conversions, and so on, as shown in [Fig. 11](#_bookmark13)(c).

*Non-structured source*: represents any source that is not in the relational structure. The non-structured source may be semi-structured or unstructured source such as XML ﬁles, web logs, excel workbook, object oriented database, etc., as shown in [Fig. 11](#_bookmark13)(d).

●

Notice that a symbol or shorthand of the operation is put inside the entity or the attribute transformations construct. The transformation functions that take place in the proposed model EMD are classiﬁed into built-in or standard functions, such as join, union, and rename, and user deﬁned functions as mentioned above, like any formula deﬁned by the user. An- other classiﬁcation for the transformation functions according to the level of transformation is entity transformations func- tions, and attribute transformations functions.

* 1. *Demonstration example*

To illustrate the usage of our proposed graphical model, we introduce a simple example. A company wants to build a data warehouse for monitoring the sales processes in its two branches. It has a relational data source described by schema DS1 for selling books, shown in [Fig. 13](#_bookmark15), another relational data source described by schema DS2 for selling general prod- ucts, shown in [Fig. 14](#_bookmark16). A relational data warehouse is designed to capture sales data from the two predeﬁned data sources. The star schema in [Fig. 15](#_bookmark17) shows the design of the proposed data warehouse which consists of one fact table and four dimensions tables.

[Fig. 16](#_bookmark18) depicts the entity mapping diagram for building the products dimension from the desired data sources, passing through the required ETL activities. The explanation of this diagram is as follows:

* DS1: refers to the ﬁrst data source (books-orders database).

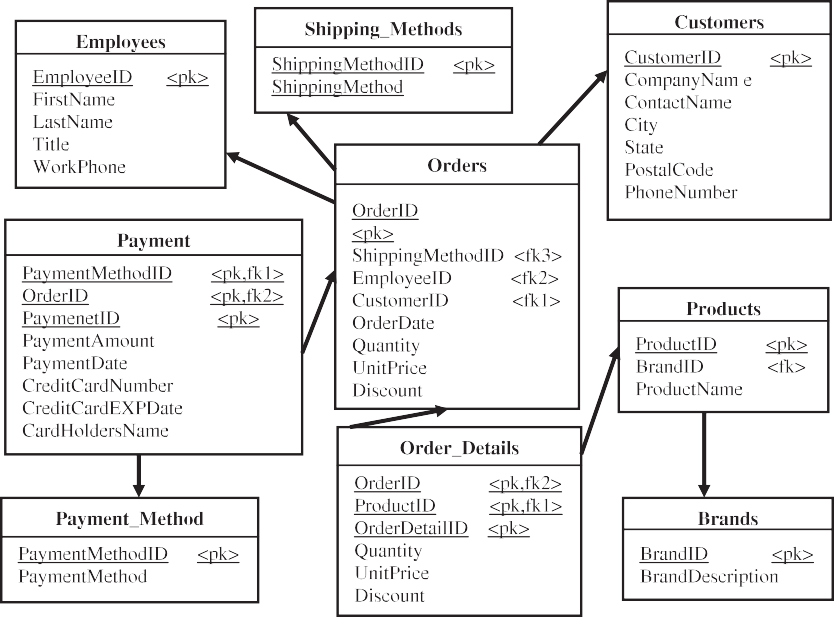


Figure 14 Relational schema DS2 for products-orders database.



Figure 15 Star schema for the proposed data warehouse.

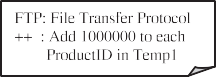
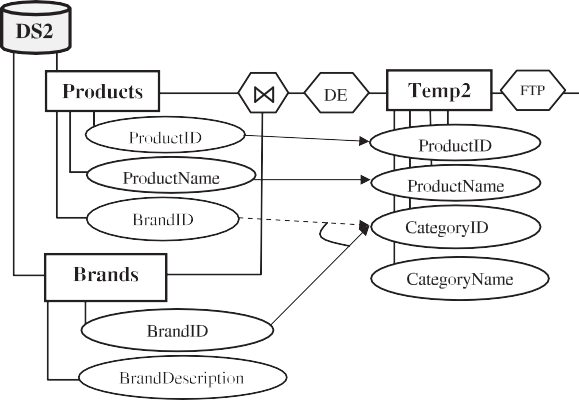
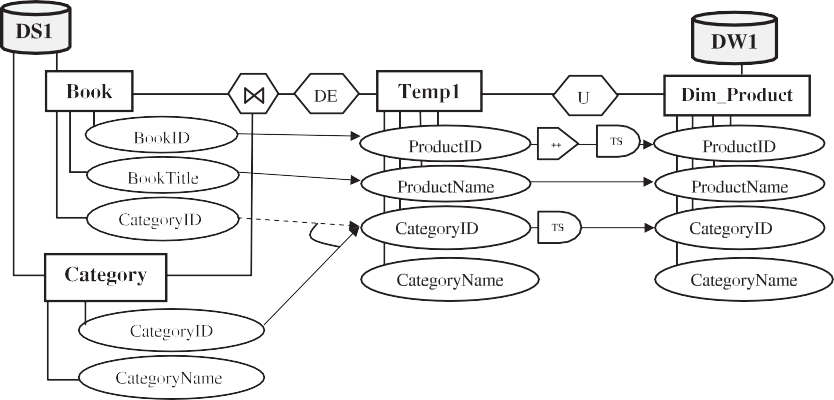


Figure 16 EMD scenario for building products dimension.

DS2: refers to the second data source (products-orders database).

●

There are two entities from each data source that partici- pate in this diagram: Book (BookID, BookTitle, CategoryID) and Category (CategoryID, CategoryName) from the ﬁrst data source, and Products (ProductID, ProductName, Bran- dID) and Brands (BrandID, CategoryName) from the second data source.

DW1: refers to the data warehouse schema to which the data will be moved, we may have one or more DW schemas, one or more data mart (DM) schemas, or a combination of DW and DM. Dim\_Products is a dimension entity found in DW1. In the middle of the diagram, mapping processes are

represented using a set of transformation steps; starting with join operation between Book and Category tables, then remov- ing the redundant records by applying the duplicate elimina- tion operation.

Temporary entity (Temp1) is created to capture the inter- mediate data that result from the previous operations. Notice that data of attribute Temp1.CategoryID could be loaded optionally from DS1.Book.CategoryID or DS1.Category.Cat- egoryID. The same activities take place in the other site that contains DS2 to result Temp2 table.

After that, some attribute transformation operations take place before loading data to the target data warehouse, some of them are described as follows: (++) is a user deﬁned trans- formation operation applied to Temp1.ProductID to add

10,00,000 to each product code number as a user requirement. ProductID and CategoryID data types are transformed to string data type by using ToString (TS) operation. Temp2 ta- ble is transferred to the site of DS1 using ﬁle transfer protocol (FTP) operation, then a union operation (U) runs to combine the two tables. The loader relationships connected to Product- Name and CategoryName attributes mean that data is loaded from these two attributes to their corresponding attributes in the DW without any transformation.

We can now develop a prototype tool (named EMD Builder) to achieve the following tasks:

* Introducing a tool for drawing the entity mapping diagram scenarios using a pallet of graphical controls.
* Implementing a set of transformation operations.
* Transforming the graphical model to a code by generating SQL script.
* Generating the mapping document according to Kimball’s standards ([Kimball and Caserta, 2004](#_bookmark25)).
* Executing the EMD scenario on the data sources to apply the extraction, and transformation operations, then loading data to the target DW schema.
* The code of may be written in C# or JAVA object-oriented programming languages and a rational database manage- ment system as Oracle or Microsoft SQL Server.

We propose the architecture in [Fig. 17](#_bookmark19) for the model, and in the future work we will implement and test this model.

The ﬁrst module checks the connection to the database management system installed on the machine on which the source databases exist. If the connection failed, an error mes-

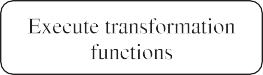
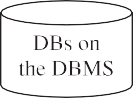
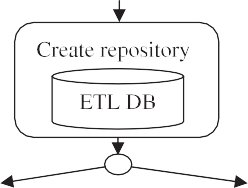
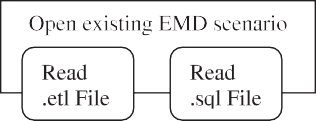


Figure 17 Basic modules of ‘‘EMD Builder’’.

sage will appear to alert the user and the application will halt. If the connection succeeded, new database ‘‘ETL’’ will be cre- ated. ‘‘ETL’’ plays the role of repository in which the metadata about the EMD scenarios will be stored. The metadata in the repository will be used to generate the mapping document. After creating ‘‘ETL’’ database the user may either create new EMD scenario or open existing one to complete it. In case of creating new scenario, new building area will appear to en- able the user to draw and build his new model, and in case of opening an existing EMD scenario, two ﬁles will be read, the ﬁrst one is ‘‘.etl’’ ﬁle from which the old scenario will be loaded to the drawing area to enable the user to complete it, and the second ﬁle is ‘‘.sql’’ in which the SQL script of the old part of the existing scenario were written and will be complete as the user completes his model. The next module loads both the metadata about the databases found on the database manage- ment system and ‘‘EMD Builder’’ interface icons. The metada- ta includes the databases names, tables, attributes, and so on. The interface icons will be loaded from our icon gallery, the interface elements will be shown in next sections. The next module facilitates the drawing process by which the user can use our palette of controls to draw and build his EMD sce- nario. By using the execution module, the EMD model will be translated into SQL script then executed on the incoming data from the source databases, so the extraction, transforma- tion, and loading processes can be applied and the desired re- cords will be transferred to the target DW schema in the required format. The last module is responsible for saving the user’s EMD model. During the save operation, three ﬁles are generates; the ﬁrst one contains the user EMD model in a binary format, so the user can open it at any time to update in its drawing, the second contains the generated SQL script, and the third generated ﬁle is the mapping document which is considered as dictionary and catalog for the ETL operations found in the user EMD scenario. The user can specify the folder in which the generated ﬁles will be saved. The generated ﬁles can be transferred from one machine to be used on an- other one that contains the same data sources and the same target data warehouse schema; this means that the generated ﬁles from our tool are machine independent, however they are data source and destination schema dependent. It is clear that the destination is a whole schema (data warehouse or data mart), but each part of this schema (fact or dimension) is han- dled as a standalone destination in a single EMD scenario.

* 1. Models evaluation and comparison

[Table 1](#_bookmark20) contains the matrix that is used to compare the differ- ent ETL modeling approaches and evaluates our proposed model against the other models. The letter P in the matrix means that this model had partially supported the correspond- ing criteria.

* 1. Other related work

The ETL process, in data warehouse, is a hot point of research because of its importance and cost in data warehouse project building and maintenance. The method of systematic review to identify, extract and analyze the main proposals on model- ing conceptual ETL processes for DWs ([Mun˜oz et al., 2010a](#_bookmark36)). Generating ETL processes for incremental loading ([Jo¨ rg and](#_bookmark25)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 1 Comparison and evaluation matrix. | | | | |
| Criteria | Model |  |  |  |
|  | Mapping expressions | Conceptual constructs | UML environment | EMD |
| *Design aspects*  Complete graphical model | No | Yes | Yes | Yes |
| New constructs | No | Yes | No | Yes |
| (OO) concept independent | Yes | P | No | Yes |
| DBMS independent | Yes | Yes | Yes | Yes |
| Mapping operations | Yes | Yes | Yes | Yes |
| User deﬁned transformation | No | No | No | Yes |
| Mapping relationship | Yes | Yes | Yes | Yes |
| Source independent (non-relational) | No | No | No | Yes |
| Source converting | No | No | No | Yes |
| Flat model | Yes | Yes | No | Yes |
| *Implementation aspects*  Develop a tool | Yes | Yes | No | Yes |
| Generate SQL | Yes | No | No | Yes |
| Generate mapping document | No | No | No | Yes |
| Non-relational handling | No | No | No | No |
| Evaluation | 7 | 7.5 | 4 | 13 |
| Yes = 1; No = 0; P: partial = 0.5; total = 14. | | | | |

[Deßloch, 2008](#_bookmark25)). A simulation model for secure data extraction in ETL processes ([Mrunalini et al., 2009](#_bookmark32)). A set of measures with which to evaluate the structural complexity of ETL pro- cess models at the conceptual level discussed in [Mun˜oz et al.](#_bookmark26) [(2010b)](#_bookmark26). In [Simitsis and Vassiliadis (2008)](#_bookmark26) the author discusses the mapping of the conceptual model to the logical model. Generating an incremental ETL process automatically from the full ETL process is discussed in [Zhang et al. (2008)](#_bookmark37). In [Sim-](#_bookmark26) [itsis et al. (2008)](#_bookmark26) the author discusses the application of natural language generation techniques to the ETL environment. Mea- sures the ETL processes models in data warehouses are dis- cussed in [Mun˜oz et al. (2009)](#_bookmark34).

* 1. Conclusion and future work

ETL processes are very important problem in the current re- search of data warehousing. In this paper, we have investigated a very important problem in the current research of data ware- housing. This problem represents a real need to ﬁnd a standard conceptual model for representing in simpliﬁed way the extrac- tion, transformation, and loading (ETL) processes. Some ap- proaches have been introduced to handle this problem. We have classiﬁed these approaches into three categories; ﬁst, is modeling based on mapping expressions and guidelines, sec- ond, is modeling based on conceptual constructs, and the ﬁnal category, is modeling based on UML environment. We have explained each model in some detail.

What is more, we proposed a novel conceptual model entity mapping diagram (EMD) as a simpliﬁed model for represent- ing extraction, transformation, and loading processes in data warehousing projects. In order to explain our proposed model; we deﬁned a metamodel for the entity mapping diagram. In the metamodel we deﬁned two layers; the ﬁrst is the abstraction layer in which ﬁve objects (function, data container, entity, relationship, and attribute) are clearly deﬁned. The objects in the abstraction layer are a high level view of the parts or ob- jects that can be used to draw an EMD scenario. The second

is the template layer which is an expansion to the abstraction layer. The user can add his own layer in which the ETL de- signer draws his EMD scenario. We also set a framework for using this model. The framework consists of data sources part, data warehouse schema part, and mapping part. Both data sources and data warehouse schemas should be deﬁned clearly before starting to draw EMD scenario. By comparing the pro- posed model to the previous research projects using the evalu- ation matrix, the proposed model handle may weak points that appear in the previous work. In the future work to this paper, we will develop and test a prototype tool call it ‘EMD Builder’ to achieve the following tasks: introducing a tool for drawing the entity mapping diagram scenarios using a pallet of graph- ical constructs, implementing a set of transformation opera- tions, transforming the graphical model to a code by generating SQL script, and generating the mapping document according to Kimball’s standards.

References

Arya, P., Slany, W., Schindler, C., 2006. Enhancing Wrapper Usability through Ontology Sharing and Large Scale Cooperation.

<[www.ru5.cti.gr/HT05/ﬁles/andreas\_rath.ppt](http://www.ru5.cti.gr/HT05/files/andreas_rath.ppt)> (accessed 2006).

Bernstein, P., Rahm, E., 2000. Data warehouse scenarios for model management. In: Proceedings of the 19th International Conference on Conceptual Modeling (ER’00), LNCS, vol. 1920, Salt Lake City, USA, pp. 1–15.

Berson, A., Smith, S.J., 1997. Data Warehousing, Data Mining, and OLAP. McGraw-Hill.

Demarest, M., 1997. The Politics of Data Warehousing. <[http://](http://www.hevanet.com/demarest/marc/dwpol.html) [www.hevanet.com/demarest/marc/dwpol.html](http://www.hevanet.com/demarest/marc/dwpol.html)>.

Dobre, A., Hakimpour, F., Dittrich, K.R., 2003. Operators and classiﬁcation for data mapping in semantic integration. In: Proceedings of the 22nd International Conference on Conceptual Modeling (ER’03), LNCS, vol. 2813, Chicago, USA, pp. 534–547.

El Bastawesy, A., Boshra, M., Hendawi, A., 2005. Entity mapping diagram for modeling ETL processes. In: Proceedings of the Third International Conference on Informatics and Systems (INFOS), Cairo.

Inmon, B., 1997. The Data Warehouse Budget. DM Review Magazine, January 1997. <[www.dmreview.com/master.cfm?NavID=55&](http://www.dmreview.com/master.cfm?NavID=55&amp;amp%3BEdID=1315) [EdID=1315](http://www.dmreview.com/master.cfm?NavID=55&amp;amp%3BEdID=1315)>.

Inmon, W.H., 2002. Building the Data Warehouse, third ed. John Wiley and Sons, USA.

Jarke, M., Lenzerini, M., Vassiliou, Y., Vassiliadis, P., 2003. Funda- mentals of Data Warehouses, second ed. Springer-Verlag.

Jo¨ rg, Thomas, Deßloch, Stefan, 2008. Towards generating ETL processes for incremental loading. In: ACM Proceedings of the 2008 International Symposium on Database Engineering and Applications.

Kimball, R., Caserta, J., 2004. The Data Warehouse ETL Toolkit. Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data. Wiley.

Kimball, R., Reeves, L., Ross, M., Thornthwaite, W., 1998. The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing, and Deploying Data Warehouses. John Wiley and Sons.

Lujan-Mora, S., Trujillo, J., 2003. A comprehensive method for data warehouse design. In: Proceedings of the Fifth International Workshop on Design and Management of Data Warehouses (DMDW’03), Berlin, Germany.

Lujan-Mora, S., Vassiliadis, P., Trujillo, J., 2004. Data mapping diagram for data warehouse design with UML. In: International Conference on Conceptual Modeling, Shanghai, China, November 2004.

Madhavan, J., Bernstein, P.A., Rahm, E., 2001. Generic schema matching with cupid. In: Proceedings of the 27th International Conferences on Very Large Databases, pp. 49–58.

Maier, T., 2004. A formal model of the ETL process for OLAP-based web usage analysis. In: Proceedings of the Sixth WEBKDD Workshop: Webmining and Web Usage Analysis (WEBKDD’04), in conjunction with the 10th ACM SIGKDD Conference (KDD’04), Seattle, Washington, USA, August 22, 2004 (accessed 2006).

Miller, R.J., Haas, L.M., Hernandez, M.A., 2000. Schema mapping as query discovery. In: Proceedings of the 26th VLDB Conference, Cairo.

Moss, L.T., 2005. Moving Your ETL Process into Primetime. <[http://](http://www.businessintelligence.com//ex/asp/code.44/xe/article.htm) [www.businessintelligence.com//ex/asp/code.44/xe/article.htm](http://www.businessintelligence.com//ex/asp/code.44/xe/article.htm)> (visited June 2005).

Mrunalini, M., Kumar, T.V.S., Kanth, K.R., 2009. Simulating secure data extraction in extraction transformation loading (ETL) pro- cesses. In: IEEE Computer Modeling and Simulation Conference. EMS’09. Third UKSim European Symposium, November 2009, pp. 142–147. ISBN: 978-1-4244-5345-0.

Mun˜oz, Lilia, Mazo´ n, Jose-Norberto, Trujillo, Juan, 2009. Measures for ETL processes models in data warehouses. In: ACM Proceed- ing of the First International Workshop on Model Driven Service Engineering and Data Quality and Security, November 2009.

Mun˜oz, Lilia, Mazon, Jose-Norberto, Trujillo, Juan, 2010. Systematic review and comparison of modeling ETL processes in data warehouse. In: Proceedings of the Fifth Iberian Conference on IEEE Information Systems and Technologies (CISTI), August 2010, pp. 1–6. ISBN: 978-1-4244-7227-7.

Mun˜oz, Lilia, Mazo´ nand, Jose-Norberto, Trujillo, Juan, 2010b. A family of experiments to validate measures for UML activity diagrams of ETL processes in data warehouses. Information and Software Technology 52 (11), 1188–1203.

Naqvi, S., Tsur, S., 1989. A Logical Language for Data and Knowledge Bases. Computer Science Press.

Oracle Corp., 2001. Oracle9i™ Warehouse Builder User’s Guide,

Release 9.0.2, November 2001. <[http://www.otn.oracle.com/prod-](http://www.otn.oracle.com/products/warehouse/content.html) [ucts/warehouse/content.html](http://www.otn.oracle.com/products/warehouse/content.html)>.

Rifaieh, R., Benharkat, N.A., 2002. Query-based data warehousing tool. In: Proceedings of the Fifth ACM International Workshop on Data Warehousing and OLAP, November 2002.

Shilakes, C., Tylman, J., 1998. Enterprise Information Portals. Enterprise Software Team. <[http://www.sagemaker.com/com-](http://www.sagemaker.com/company/downloads/eip/indepth.pdf) [pany/downloads/eip/indepth.pdf](http://www.sagemaker.com/company/downloads/eip/indepth.pdf)>.

Simitsis, A., 2003. Modeling and Managing ETL Processes. VLDB Ph.D. Workshop.

Simitsis, Alkis, Vassiliadis, Panos, 2008. A method for the mapping of conceptual designs to logical blueprints for ETL processes. Decision Support Systems, Data Warehousing and OLAP 45 (1), 22–40.

Simitsis, Alkis, Skoutas, Dimitrios, Castellanos, Malu´ , 2008. Natural language reporting for ETL processes. In: Proceeding of the ACM 11th International Workshop on Data Warehousing and OLAP, pp. 65–72. ISBN: 978-1-60558-250-4.

Staudt, M., Vaduva, A., Vetterli, T., 1999. Metadata Management and Data Warehousing. Technical Report, The Department of Infor- mation Technology (IFI) at the University of Zurich.

Stonebraker, M., Hellerstein, J., 2001. Content integration for e- business. In: Proceedings of the ACM SIGMOD/PODS 2001, Santa Barbara, CA, May 21–24, 2001.

Trujillo, J., Lujan-Mora, S., 2003. A UML based approach for modeling ETL processes in data warehouses. In: Proceedings of the 22nd International Conference on Conceptual Modeling. LNCS, Chicago, USA.

Vassiliadis, P., 2000. Data Warehouse Modeling and Quality Issues. Ph.D. Thesis, Department of Electrical and Computer Engineering, National Technical University of Athens (Greece).

Vassiliadis, P., Simitsis, A., Skiadopoulos, S., 2002. Conceptual modeling for ETL processes. In: Proceedings of the Fifth ACM International Workshop on Data Warehousing and OLAP, pp. 14– 21.

Vassiliadis, P., Simitsis, A., Skiadopoulos, S., 2002. Modeling ETL activities as graphs. In: Proceedings of the Fourth International Workshop on the Design and Management of Data Warehouses (DMDW’02), Toronto, Canada, pp. 52–61.

Vassiliadis, P., Simitsis, A., Georgantas, P., Terrovitis, M., 2003. A framework for the design of ETL scenarios. In: Proceedings of the 15th CAiSE, Velden, Austria, June 16, 2003.

Vassiliadis, P., Simitsis, A., Georgantas, P., Terrovitis, M., Skiadopo- ulos, S., 2005. A generic and customizable framework for the design of ETL scenarios. Information Systems Journal.

Zhang, Xufeng, Sun, Weiwei, Wang, Wei, Feng, Yahui, Shi, Baile, 2008. Generating incremental ETL processes automatically. In: IEEE Computer and Computational Sciences, pp. 516–521.