

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/262356199>

EMD: Entity mapping diagram for automated extraction, transformation, and loading processes in data warehousing

Article in *International Journal of Intelligent Information and Database Systems* · May 2012

DOI: 10.1504/IJIDS.2012.047003

CITATIONS

2

READS

759

2 authors:



Abdeltawab M. Ahmed Hendawi
University of Virginia

46 PUBLICATIONS 234 CITATIONS

[SEE PROFILE](#)



Shaker El-Sappagh
Benha University

43 PUBLICATIONS 290 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Predictive Spatial Queries with Uncertainty [View project](#)



Liver Fibrosis Diagnosis [View project](#)

EMD: entity mapping diagram for automated extraction, transformation, and loading processes in data warehousing

Abdeltawab M.A. Hendawi* and
Shaker H. Ali El-Sappagh

Information Systems Department,
Faculty of Computers and Information,
Minia University, El-Minia, Egypt
E-mail: Abdeltawab_fci@yahoo.com
E-mail: a.hendawi@fci-cu.edu.eg
E-mail: Shaker_elsappagh@yahoo.com
E-mail: Sker@ksu.edu.sa

*Corresponding author

Abstract: During the last few years, researchers and developers had proposed various trials to put a standard conceptual design of ETL processes in data warehouse. These trials try to represent the main mapping activities at the conceptual level. Due to limitations of the previous trials, in this paper

- 1 We propose a model for conceptual design of ETL processes and we call it entity mapping diagram (EMD). The proposed model is built upon the enhancement of the models in the previous work to support some missing mapping features.
- 2 We implemented the proposed conceptual model in a prototype called EMD Builder and use it in an illustration scenario.

Keywords: modelling ETL processes; database; data mart; DM; online analytical processing; OLAP; conceptual modelling; intelligent information systems; entity mapping diagram; EMD; database systems; data warehouse; DW.

Reference to this paper should be made as follows: Hendawi, A.M.A. and Ali El-Sappagh, S.H. (2012) 'EMD: entity mapping diagram for automated extraction, transformation, and loading processes in data warehousing', *Int. J. Intelligent Information and Database Systems*, Vol. 6, No. 3, pp.255–272.

Biographical notes: Abdeltawab M.A. Hendawi received his MSc in Information Systems from the Information Systems Department, Faculty of Computers and Information, Cairo University. His research interests are in the areas of data warehousing, spatio-temporal database and data mining. Currently, he is a Teaching Assistant in the Information Systems Department, Faculty of Computers and Information, Cairo University.

Shaker H. Ali El-Sappagh received his MSc in Information Systems from Information Systems Department, Faculty of Computers and Information, Cairo University. He has been working as a Teaching Assistant in the Information Systems Department, Faculty of Computers and Information, Minia University, El-Minia since 2003. He is interested in data warehousing technology, data mining, business intelligence and health informatics.

1 Introduction

A data warehouse (DW) is a collection of technologies aimed at enabling the decision maker to make better and faster decisions. The DW has a generic architecture which consists of three layers, data sources, ODS, and primary DW (Inmon, 2002; Vassiliadis, 2000). 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 DW.

ETL tools are a category of specialised tools with the task of dealing with DW homogeneity, cleaning, transforming, and loading problems (Shilakes and Tylman, 1998). Although ETL processes are very important (Kimball and Caserta, 2004; Demarest, 1997; Oracle Corp., 2001; Inmon, 1997), but it has little researches, because of its difficulty and lacking of formal model for representing ETL activities. The elementary problem that this research turns around is the lack of a formal representation model for capturing the ETL processes that maps the incoming data from different DSs to be in a suitable format for loading to the target DW or data mart (DM).

Our objective is to propose conceptual model to be used in modelling various ETL processes and cover the limitations of the previous trials. The proposed model will be used to design ETL scenarios, and document, customise, and simplify the tracing of the mapping between the Data Source attributes and its corresponding in DW. The proposed model has the following characteristics:

- *simple*: to be understood by the DW designer
- *complete*: to represent all activities of the ETL processes
- *customisable*: to be used in different DW environments.

We call the proposed model entity mapping diagram (EMD). In Section 2, we briefly list the related work. In Section 3, we explain the research problem. In Section 4, we introduce our model. Section 5 shows the implemented prototype EMD builder. Section 6 is models comparison. Finally, we conclude and hint about the future work in Section 7.

2 Related work

The problem of finding a standard conceptual model for representing in simplified way the extraction, transformation, and loading (ETL) processes has been discussed by many approaches. These approaches have been classified into three categories:

- 1 Modelling based on mapping expressions and guidelines (Rifaieh and Benharkat, 2002; Madhavan et al., 2001). This approach has defined a model for different types of mapping expressions.
- 2 Modelling based on conceptual constructs (Vassiliadis et al., 2003, 2005, 2002). This model tries to introduce a framework for the modelling of ETL activities.

- 3 Modelling based on UML environment (Lujan-Mora et al., 2004; Lujan-Mora and Trujillo, 2003). This framework is based on a principled approach in the usage of UML packages to allow zooming in and out the design of a scenario.

The method of systematic review to identify, extract and analyse the main proposals on modelling conceptual ETL processes for DWs is covered in Munoz et al. (2010a). Generating ETL processes for incremental loading is explained in Jörg and Deßloch (2008). A simulation model for secure data extraction in ETL processes is founded in Mrunalini et al. (2009). A set of measures with which to evaluate the structural complexity of ETL process models at the conceptual level is discussed in Muñoz et al. (2010b). In Simitsis and Vassiliadis (2008), 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 addressed in Zhang et al. (2008). In Simitsis et al. (2008), the author shows the application of natural language generation techniques to the ETL environment. Measures the ETL processes models in DWs are introduced in Muñoz et al. (2009).

3 Research problem and objective

The elementary problem that this research turns around is the lack of 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. In Kimball and Caserta (2004), Kimball and Caserta consider ETL processes as the back room or the kitchen of the DW, in which data is prepared to be loaded to the multidimensional schema.

Our objective is to propose conceptual model to be used in modelling various ETL processes. The proposed model will be used to design ETL scenarios, and document, customise, and simplify the tracing of the mapping between the data source attributes and its corresponding in DW.

Shilakes and Tylman (1998) mention that ETL and data cleaning tools are estimated to cost at least one-third of effort and expenses in the budget of the DW while Demarest (1997) and Oracle Corp. (2001) mention that 70% to 80% of development time and effort in DW projects are devoted to the ETL process. Inmon (1997) mentions that the ETL process costs 55% of the total costs of DW runtime. Kimball and Caserta (2004) decide that ETL system consumes 70% of the resources needed for implementation and maintenance of a typical DW.

We covered the previous work in details in another per.

4 EMD, the proposed model

To conceptualise the ETL processes used to map data from sources to the target DW schema, we studied the previous trials, made some integration, and added some extensions to the approaches mentioned above. We propose EMD as a new conceptual model for modelling ETL processes scenarios (El Bastawesy et al., 2005). Our proposed model mainly follows the approach of modelling based on conceptual constructs. The proposed model will fulfil six requirements (Maier, 2004):

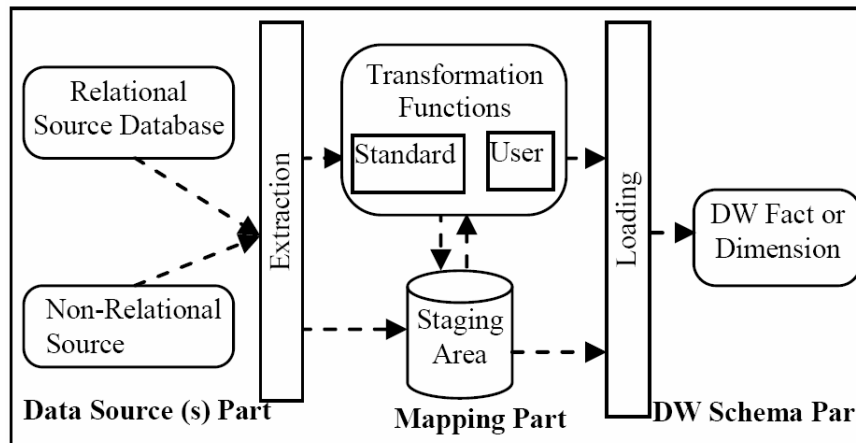
- 1 supports the integration of multiple data sources
- 2 is robust in view of changing data sources
- 3 supports flexible transformations
- 4 can be easily deployed in a suitable implementation environment
- 5 is complete enough to handle the various ETL operations
- 6 is simple in creating and maintaining.

We will describe the meta-model for our proposed model EMD, explain its framework and constructs, list the types of transformations handled in EMD. A comparison and evaluation of the previous approaches against our proposed model will be presented at the end of this paper.

4.1 EMD framework

Figure 1 shows the general framework of the proposed EDM.

Figure 1 A general framework of EMD



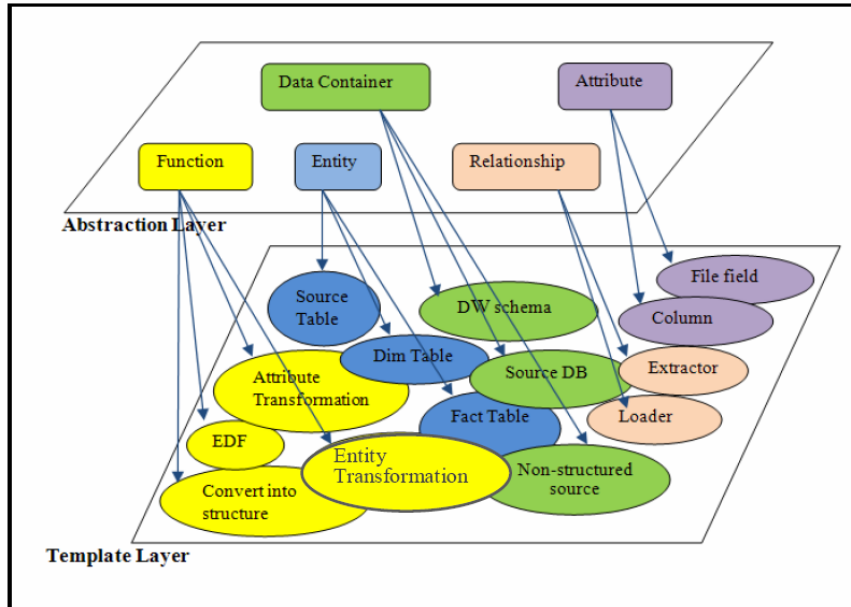
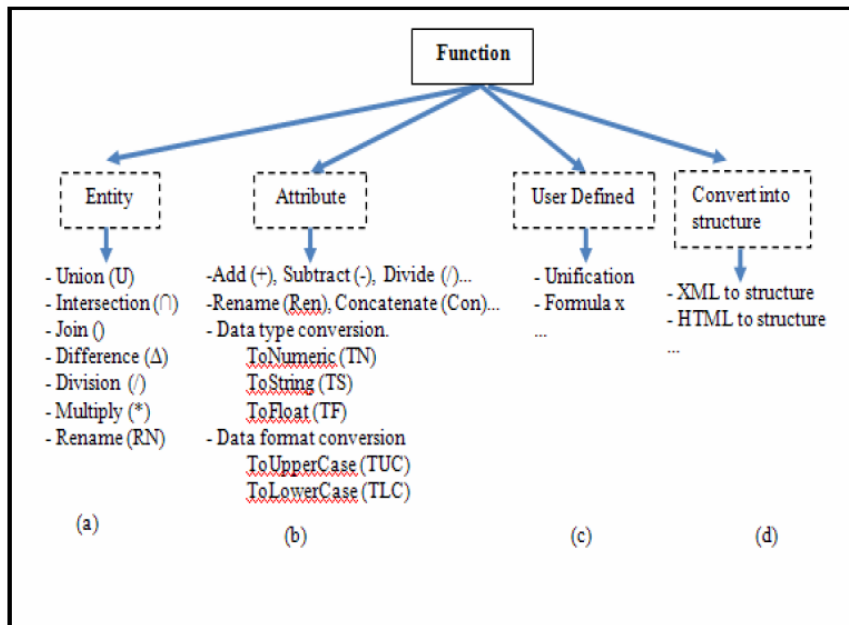
- *In the data source(s) part:* the participated data sources are drawn. The data sources may be structured databases or non-structured sources. In case of structured sources; the participated databases and their participated tables and attributes are used directly as the base source, and in case of non-structured sources; a conversion step should be applied first to convert the non-structured source into structured 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 specialised programme routines that automatically extract data from different data sources with different formats and convert the information into a structured format. The typical tasks of a wrapper are:

- a fetching data from a remote resource
- b searching for, recognising and extracting specified data
- c saving this data in a suitable structured format to enable further manipulation (Munoz et al., 2010a; Arya et al., 2006).
- *Extraction:* during the extraction process, some temporary tables may be created to hold the result of converting non-structured sources into databases. The extraction process 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 DW. The refresh extraction takes place to capture 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 DW schema table (fact or dimension) is drawn. In spite of that the fact table and the dimension table are clearly different in their functionalities and features but all of them are data containers. Basically, the DW is stored as relational structure not as multidimensional structure. The multidimensionality 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 temporary source in the staging area. Some transformation operations lead to temporary results which are saved in temporary tables in the staging area
- *The staging area:* a physical container that contains all temporary 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 destination 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 DW 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 DW schema.

4.2 EMD meta-model

EMD is a proposed conceptual model for modelling the ETL processes which are needed to map data from sources to the target DW schema. Figure 2 shows the meta-model architecture for the proposed conceptual model EMD. The meta-model of the proposed EMD is composed of two layers; the first layer is 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 objects that can be used to draw an EMD scenario.

Figure 2 EMD meta-model

Figure 3 Types of transformation in EMD


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

transformation, a user defined function (UDF), or convert into structure (relation). Figure 3 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 is function that can be applied to a source attribute (e.g., to upper case, to string, etc). A UDF is any function that may be added by the user who is the creator of the ETL scenario (e.g., unification between different types of units). Convert into structure is a function 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 to be applied on it. A data container may be a source database, a target DW or DM, 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 meta-model can be expanded to include any extra objects that may be required in the future. The user can use instances 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 DW designer or the ETL designer; 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.

4.3 Primitives of EMD constructs












The basic set of constructs that is used in the proposed EDM are shown in Figure 4. In this section, some explanation about the usage of the constructs of the proposed EDM 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 candidate 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 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 transformations usually results in a temporary entity. There are standard operators that are used inside this construct, Figure 3(a) shows some of these operators.
- *Attribute transformation operation*: standard operations are used with this construct, Figure 3(b) shows sample of these operators.

- *UDF as a transformation operation*: user can use his defined operations, so any kind of transformation can be added, such as currency conversion functions, packages (units) conversions, and so on, as shown in Figure 3(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 Figure 3(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 UDFs as mentioned above, like any formula defined by the user. Another classification for the transformation functions according to the level of transformation is entity transformations functions, and attributes transformations functions.

Figure 4 Graphical constructs for the proposed EMD

<i>Mapping construct</i>		<i>To represent</i>
<i>Name</i>	<i>Shape</i>	
Cylinder		Schema
Rectangle		Entity
Oval		Attribute
Diamond with rounded arrow		Convert into structure
Solid arrow		Loader relationship
Connected arrows		Optional loader relationship
Square with rounded edge		Attribute transformation
Square with triangle edge		UDF
Hexagon		Entity transformation operation
Document		Non-structured source
Rectangle with folded corner		User note

4.4 Demonstration example

To illustrate the usage of our proposed graphical model, we introduce a simple example. A company wants to build a DW for monitoring the sales processes in its two branches. It has a relational data source described by schema DS1 for selling books, shown in

Figure 5, another relational data source described by schema DS2 for selling general products, shown in Figure 6.

Figure 5 Relational schema DS1 for books-orders database

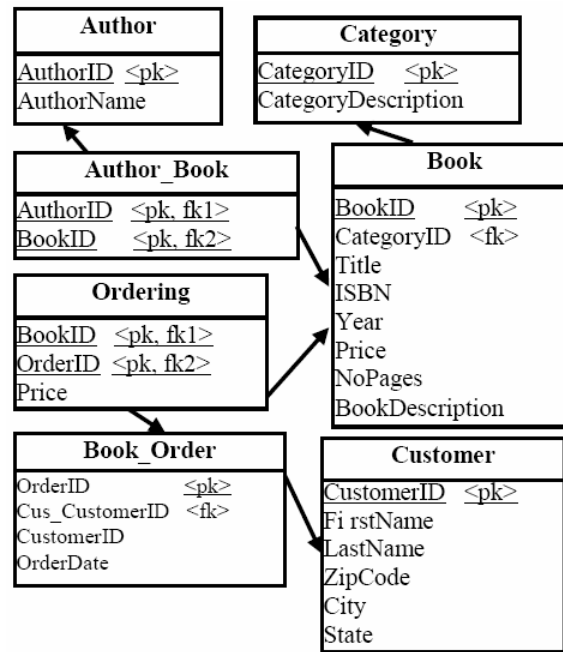
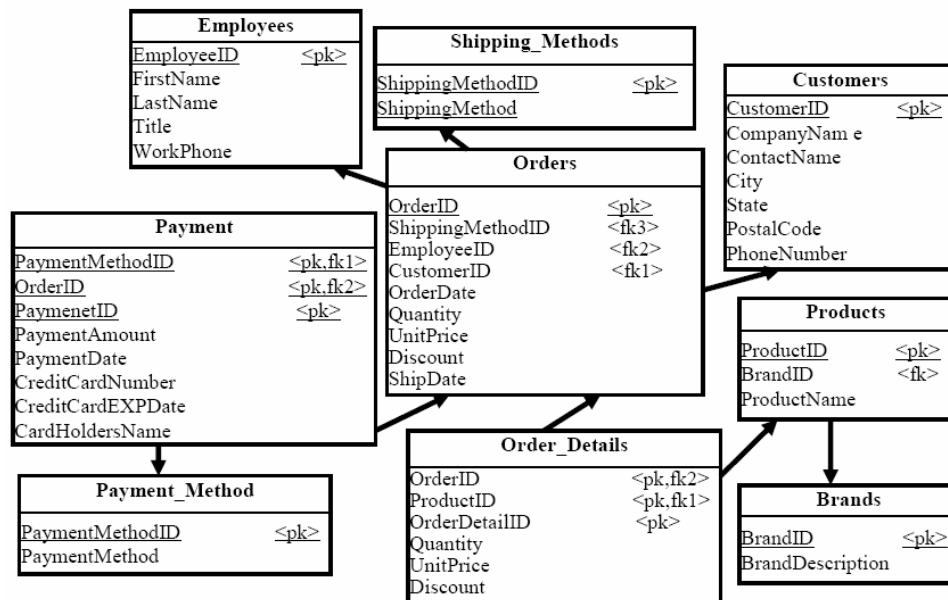


Figure 6 Relational schema DS2 for products-orders database



A relational DW is designed to capture sales data from the two predefined data sources. The star schema in Figure 7 shows the design of the proposed DW which consists of one fact table and four dimensions tables.

Figure 7 Star schema for the proposed DW

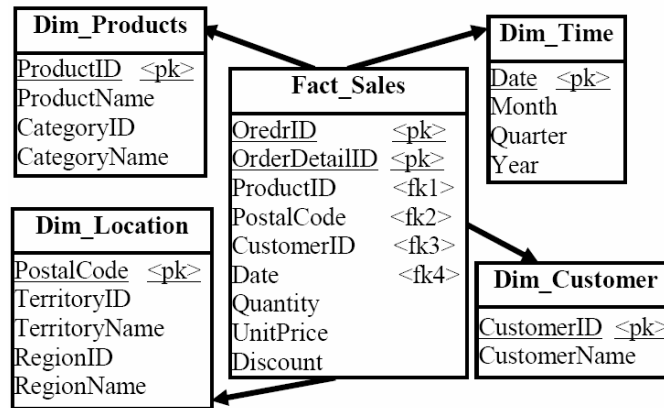


Figure 8 EMD scenario for building products dimension

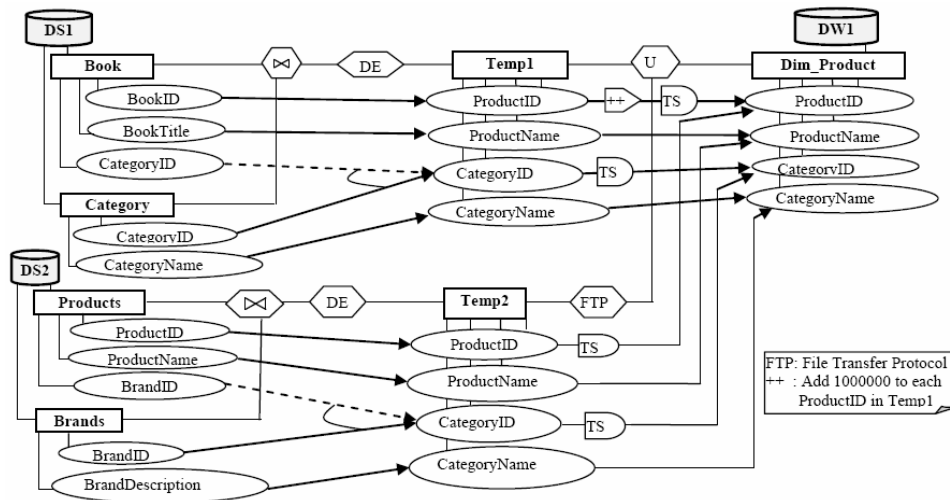


Figure 8 depicts the EDM for building the products dimension from the desired data sources, passing through the required ETL activities. The explanation of this diagram is as follows:

- *DS1*: refers to the first data source (books-orders database).
- *DS2*: refers to the second data source (products-orders database).

There are two entities from each data source that participate in this diagram; book (BookID, BookTitle, CategoryID) and category (CategoryID, CategoryName) from the

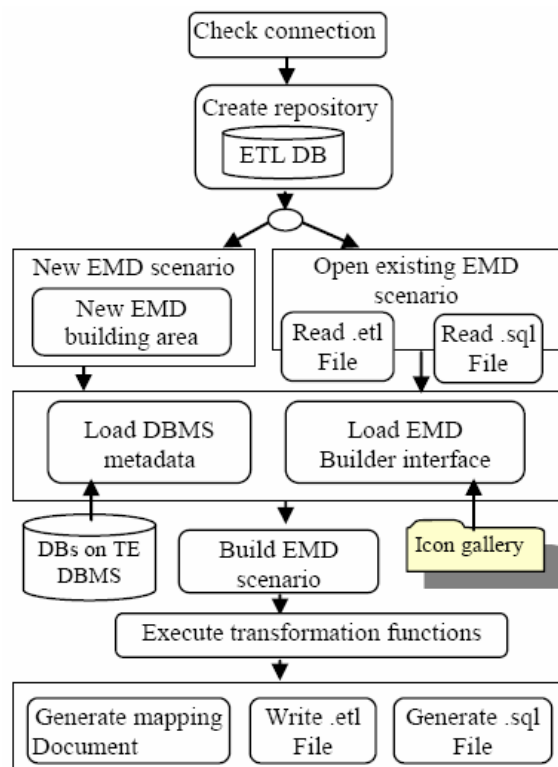
first data source, and products (ProductID, ProductName, BrandID) and Brands (BrandID, CategoryName) from the second data source.

DW1 refers to the DW schema to which the data will be moved, we may have one or more DW schemas, one or more 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 removing the redundant records by applying the duplicate elimination operation.

Temporary entity (Temp1) is created to capture the intermediate 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.CategoryID. 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 DW, some of them are described as follows: (++) is a user defined transformation (UDF) operation applied to Temp1.ProductID to add 1,000,000 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 table 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 ProductName and CategoryName attributes mean that data is loaded from these two attributes to their corresponding attributes in the DW without any transformation.

Figure 9 Basic modules of 'EMD builder' (see online version for colours)



We developed a prototype tool (named EMD builder) to achieve the following tasks:

- introducing a tool for drawing the EDM 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 (MD) according to Kimball's standards (Kimball and Caserta, 2004)
- executing the EMD scenario on the data sources to apply the extraction, and transformation operations, then loading data to the target DW schema
- EMD prototype was implemented in C# and use Microsoft SQL Server as a database/DW server to store both DB tables and DW fact and dimension tables in addition to ETL processes intermediate results.

Figure 9 shows the general architecture for EMD tool.

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 message will appear to alert the user and the application will halt. If the connection succeeded, new database 'ETL' will be created. '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 MD. 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 enable 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 management system and 'EMD builder' interface icons. The metadata 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 scenario. 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 ETL processes can be applied and the desired records 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 generated; 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 MD which is considered as dictionary and catalogue 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 another one that contains the same data sources and the same target DW 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 (DW or DM), but each part of this schema (fact or dimension) is handled as a standalone destination in a single EMD scenario.

5 EMD builder interface

In this section, we show the basic interface elements of our prototype tool 'EMD builder'.

Figure 10 'EMD builder' interface part 1

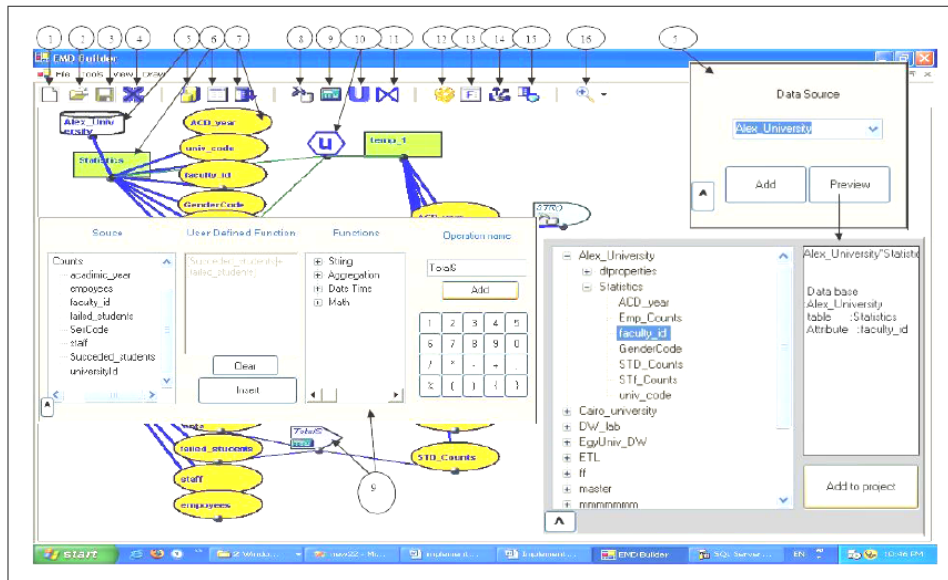


Figure 11 'EMD builder' interface part 2

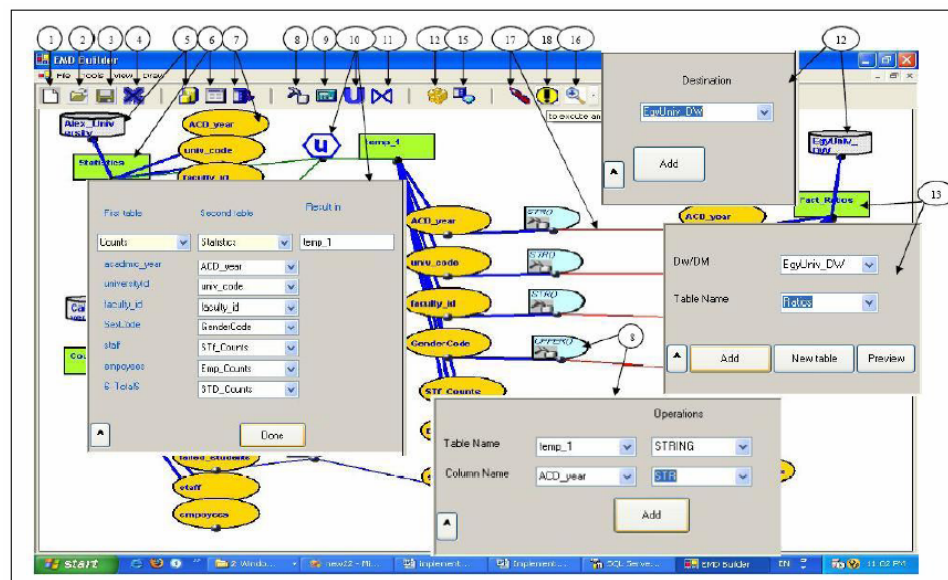


Figure 10 and Figure 11 show the basic components of 'EMD builder' GUI. Each number inside white circle points to a specific component or set of components with the same function.

The indication of each number shown in Figure 10 is described as follows:

- 1 To open a new project to draw new EMD model for user's ETL scenario.
- 2 To open an existing project and load both '.etl' file and '.sql' file.
- 3 To generate the '.etl' file, the '.sql' file and the MD then save them into location specified by the user.
- 4 To exit from the application environment and back to the operating system.
- 5 To select the data source database then dropping its construct (cylinder) on the drawing area. The data source database can be selected from the data source panel or clicking 'preview' button and selecting multiple databases from preview panel.
- 6 To select the data source table, draw its construct (rectangle) on the drawing area and connect it to its source database.
- 7 To select the data source attributes, draw its construct (oval) on the drawing area and connect it to its source table.
- 9 To build new UDF to be applied to an attribute or set of attributes. Clicking on this control opens UDF panel and draw and link the UDF construct (Square with triangle edge) to the participated attributes.

In Figure 11, the indication of each number is described as follows:

- 8 To perform attribute transformation operation such as (convert to string, to capital letter, to small, concatenating ... etc). After selecting the required operation to be applied to a specified attribute, the attribute operation construct (square with triangle edge) will be linked to the participated attribute.
- 10 To perform union operation by specifying the participated tables and the corresponding attributes, then create temporary table to capture the result. The union construct (hexagon with U letter) will be linked to both the participated tables and the result table.
- 11 To joining tables, create temporary table to put the result in it, and add its construct (Hexagon with join symbol) to the drawing area.
- 12 To select the destination DW or DM and add its construct (cylinder) to the drawing area.
- 13 To select the fact table from the destination DW/DM.
- 14 To select the dimension table from the destination DW/DM.
- 15 To select the attributes of either the fact table or the dimension table.
- 16 To zoom in and zoom out the diagram in the drawing area.
- 17 To draw the loader that maps each element in the final temporary source to its corresponding element in the target schema.

- 18 To execute the EMD model, and apply ETL processes that found in that model starting with extracting data from different sources, then performing the transformation operations, and loading data to the right elements in the destination. Clicking on this control will not add any graphical construct to the drawing area but the actual data will be transferred to the destination, so a message box will appear to indicate the number of rows affected by executing that EMD model.

As we mentioned before, saving the EMD scenario will generate '.etl' file which contains the graphical model, and '.sql' file which contains the generated SQL script that resulted from executing the scenario and MD. Figure 12 shows sample SQL script generated by the 'EMD builder' as result from running some scenario.

Figure 12 Sample of the SQL script generated by 'EMD builder'

```

/*this script is to Extract, Transfer and Load data
////////////////////////////////////*/

/*the following DDL to Create temporary table called (temp_1) to catch results from union
operations*/
if exists (select * from dbo.sysobjects where id = object_id(N'[temp_1]') and
OBJECTPROPERTY(id, N'IsUserTable') = 1)drop table [temp_1]
CREATE TABLE [temp_1]
(
[ACD_year] [int] NULL,
[faculty_id] [int] NULL,
[univ_code] [int] NULL,
[GenderCode] [char] (1) NULL,
[STf_Counts] [int] NULL,
[Emp_Counts] [int] NULL,
[STD_Counts] [int] NULL
)
/*the following code is used to make union between two tables
(Counts and Statistics. result will inserted in table called (temp_1) */
Insert into [Cairo_university].[dbo].[temp_1] select [academic_year], [faculty_id], [universityId],
[SexCode], [staff], [employees], ([failed_students] + [Succeded_students]) from
[Cairo_university].[dbo].[Counts] union select [ACD_year], [faculty_id], [univ_code],
[GenderCode],
[STf_Counts], [Emp_Counts], [STD_Counts] from [Alex_University].[dbo].[Statistics]

```

6 Models evaluation and comparison

Table 1 contains the matrix that is used to compare the different ETL modelling approaches and evaluates our proposed model against the other models. The letter P in the matrix means that this model had partially supported the corresponding criteria. The used criteria are classified two groups:

- 1 *Design aspects*: these criteria are related to the design ability of the tool, and it contains: *complete graphical model* which specifies if the tool gives its user with a graphical notations or not, *object-oriented concept* which show if the tool uses object-oriented concepts or not, *DBMS independent* which determine if the tool can interface with different DBMSs as Oracle Microsoft SQL Server, Sybase, etc., *mapping operations* which says if the tool gives user of transformer functions to perform mapping from sources to destination, *UDF* which determines if the tool support user-defined mapping functions or not, *mapping relationships* which determines if the tool support source to target mapping, *source independence* which explains if the tool can interface with databases not relational, and *source converting* that determines if the tool convert from one source type to another type.
- 2 *Implementation aspects*: these aspects compare the implementation characteristics of the deferent models. *Develop a tool* determine if these models are implemented and its authors provide tools, *generate SQL* which determine if the tool generate SQL code for the processes performed, *generate mapping documentation* which determines if the tool generate metadata about what done, and *non-relational handling* which determine if the tool can handle non-relational sources.

Table 1 Comparison and evaluation matrix

Criteria \ Model	Model	Mapping expressions	Conceptual constructs	UML environment	EMD
Design aspects					
Complete graphical model		No	Yes	Yes	Yes
(OO) concept		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
Implementation aspects					
Develop a tool		Yes	Yes	No	Yes
Generate SQL		Yes	No	No	Yes
Generate MD		No	No	No	Yes
Non-relational handling		No	No	No	No
Evaluation		6	5.5	4	11

Notes: Yes = 1, No = 0, P: partial = 0.5, and total = 14.

7 Conclusions and future work

ETL processes are very important problem in the current research of data warehousing. We proposed a novel conceptual model EDM as a simplified model for representing ETL processes in data warehousing projects. The proposed model is mainly based on the

second approach, as we used some constructs from the previous researches and introduced some extra constructs such as UDF. In order to explain our proposed model; we defined a meta-model for the EDM. In the meta-model 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 objects 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 designer draws his EMD scenario. To show the way of using the proposed EDM, we set a framework for using this model. The framework consists of data sources part, DW schema part, and mapping part. Both data sources and DW schemas should be defined clearly before starting to draw EMD scenario. When we compare the proposed model to the previous trials we find it better than all of the previous trials.

References

- Arya, P., Slany, W. and Schindler, C. (2006) *Enhancing Wrapper Usability through Ontology Sharing and Large Scale Cooperation*, available at http://www.ru5.cti.gr/HT05/files/andreas_rath.ppt.
- Demarest, M. (1997) *The Politics of Data Warehousing*, available at <http://www.noumenal.com/marc/dwpoly.html>.
- El Bastawesy, A., Boshra, M. and Hendawi, A. (2005) 'Entity mapping diagram for modeling ETL processes', *The Third International Conference on Informatics and Systems (INFOS)*, Cairo.
- Inmon, B. (1997) 'The data warehouse budget', *DM Review Magazine*, January, available at <http://www.datawarehouse.inf.br/Papers/inmonbudget-1.pdf>.
- Inmon, W.H. (2002) *Building the Data Warehouse*, 3rd ed., John Wiley & Sons, USA.
- Jörg, T. and DeBloch, S. (2008) 'Towards generating ETL processes for incremental loading', *ACM Proceedings of the 2008 International Symposium on Database Engineering & Applications*.
- Kimball, R. and Caserta, J. (2004) 'The data warehouse ETL toolkit', *Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data*, Wiley, ISBN: 978-0-7645-6757-5.
- Lujan-Mora, S. and Trujillo, J. (2003) 'A comprehensive method for data warehouse design', *Proc. of the 5th Intl. Workshop on Design and Management of Data Warehouses (DMDW'03)*, Berlin, Germany.
- Lujan-Mora, S., Vassiliadis, P. and Trujillo, J. (2004) 'Data mapping diagram for data warehouse design with UML', *International Conference on Conceptual Modeling*, Shanghai, China, November.
- Madhavan, J., Bernstein, P.A. and Rahm, E. (2001) 'Generic schema matching with cupid', *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', *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, 22 August (accessed on 2006).
- Mrunalini, M., Kumar, T.V.S. and Kanth, K.R. (2009) 'Simulating secure data extraction in extraction transformation loading (ETL) processes', *IEEE Computer Modeling and Simulation Conference, EMS'09, Third UKSim European Symposium*, November, pp.142–147, ISBN: 978-1-4244-5345-0.
- Muñoz, L., Mazón, J-N. and Trujillo, J. (2009) 'Measures for ETL processes models in data warehouses', *ACM Proceeding of the First International Workshop on Model Driven Service Engineering and Data Quality and Security*, November.

- Munoz, L., Mazon, J.-N. and Trujillo, J. (2010a) 'Systematic review and comparison of modeling ETL processes in data warehouse', *2010 5th Iberian Conference on IEEE Information Systems and Technologies (CISTI)*, August 2010, pp.1–6, ISBN: 978-1-4244-7227-7.
- Muñoz, L., Mazónand, J.-N. and Trujillo, J. (2010b) 'A family of experiments to validate measures for UML activity diagrams of ETL processes in data warehouses', *Science Direct Information and Software Technology*, November, Vol. 52, No. 11, pp.1188–1203.
- Oracle Corp. (2001) *Oracle9i™ Warehouse Builder User's Guide*, Release 9.0.2, November, available at http://download.oracle.com/docs/html/B10996_01/toc.htm.
- Rifaieh, R. and Benharkat, N.A. (2002) 'Query-based data warehousing tool', *Proc. of the 5th ACM International Workshop on Data Warehousing and OLAP*, November.
- Shilakes, C. and Tylman, J. (1998) 'Enterprise information portals', Enterprise Software Team, available at http://ikt.hia.no/perrep/eip_ind.pdf.
- Simitsis, A. and Vassiliadis, P. (2008) 'A method for the mapping of conceptual designs to logical blueprints for ETL processes', *Decision Support Systems, Data Warehousing and OLAP*, April, Vol. 45, No. 1, pp.22–40.
- Simitsis, A., Skoutas, D. and Castellanos, M. (2008) 'Natural language reporting for ETL processes', *Proceeding of the ACM 11th international workshop on Data warehousing and OLAP*, pp.65–72, ISBN:978-1-60558-250-4.
- Vassiliadis, P. (2000) *Data Warehouse Modeling and Quality Issues*, PhD thesis, Department of Electrical and Computer Engineering, National Technical University of Athens, Greece.
- Vassiliadis, P., Simitsis, A. and Skiadopoulos, S. (2002) 'Conceptual modeling for ETL processes', *Proceedings of the Fifth ACM International Workshop on Data Warehousing and OLAP.*, pp.14–21.
- Vassiliadis, P., Simitsis, A., Georgantas, P. and Terrovitis, M. (2003) 'A framework for the design of ETL scenarios', *Proceedings of the 15th CAiSE*, 16 June, Velden, Austria.
- Vassiliadis, P., Simitsis, A., Georgantas, P., Terrovitis, M. and Skiadopoulos, S. (2005) 'A generic and customizable framework for the design of ETL scenarios', *Information Systems Journal*, Vol. 30, No. 7, pp.492–525.
- Zhang, X., Sun, W., Wang, W., Feng, Y. and Shi, B. (2008) 'Generating incremental ETL processes automatically', *IEEE Computer and Computational Sciences*, November, pp.516–521.