Reconstruction of the Conceptual Model from the Implemented Database

Bogdan Walek and Cyril Klimeš

***Abstract*—The paper deals with reconstructing the conceptual model from the implemented database. Sometimes the conceptual model of the implemented database is missing and is appropriate to reconstruct the conceptual model from the implemented database to understand the content of the database and for effective additional development or maintance. We propose algorithm for reconstruction the conceptual model from the implemented database using JDBC methods. Reconstructed conceptual model is represented by XML document and then is transformed to CDM file supported by CASE tool called Sybase PowerDesigner. Finally, reconstructed conceptual model imported to Sybase PowerDesigner can be used for vizualization, transformation to logical and physical model, and in other various ways. Proposed algorithm is verified on specific implemented database and the reconstructed conceptual model is shown in CASE tool Sybase PowerDesigner.**

***Index Terms*—Conceptual model, database, relational database, reconstruction.**

I. Introduction

In present days there are a lot of databases which are implemented in a relational database [1]. The databases are used for storing data in information systems, web applications, e-shops, etc. Each of the implemented databases can be designed in different ways, but the result is an implemented database in specific relational database management system [further in text referred as RDBMS].

During the creation and data modeling of the database there are three phases [5], [6], [7]:

- Conceptual model design (Platform Independent Model).
- Logical model design (Platform Specific Model).
- Physical model design (Implementation Specific Model).

In first phase we define the content of the future database and result of this phase is a conceptual model of the future database. Conceptual model consists of entities, attributes and relationships and represents suitable visual model to describe future database [2], [3]. Conceptual model also helps for better understanding of database which will be implemented. Conceptual model belongs to model called Platform Independent Model (PIM) and it can be transformed to the logical model of the future database or to another models.

In second phase we define primary and foreign keys for relationships between entities, and general data types of attributes for specifying type of data which will be stored in attributes. The result of this phase is a logical model of the future database and because we specify that we constructing a database, this model is more detailed than conceptual model and belongs to Platform Specific Model (PSM).

In third phase we specify RDBMS which will be used for implementing the logical model of the future database. Then we define specific data types for attributes supported by selected RDBMS and generate physical model of database. The result of this phase is the resulting database implemented in selected RDBMS and the physical model of database belongs to Implementation Specific Model.

Process of database modeling is shown in the Fig. 1:

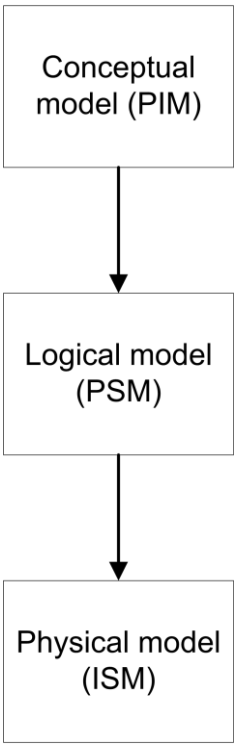

Fig. 1. Process of database modeling.

II. Problem Formulation

Sometimes the conceptual model in visual form is missing and we have only the implemented database. Or we assume the information system with the implemented database without documentation and manuals (with conceptual model). Then it is appropriate to reconstruct conceptual model from the implemented database to understand relationships and the

Manuscript received May 30, 2012, 2012; revised June 9, 2012. This work was supported in part by the University of Ostrava under internal grant SGS10/PřF/2012, called Fuzzy modeling tools for analysis and design of information systems.

The authors are with the Department of Informatics and Computers, University of Ostrava, Ostrava, Czech Republic (email: bogdan.walek@osu.cz, cyril.klimes@osu.cz).

content of the database for effective additional development or maintance [8].

Reconstructing the conceptual model from the implemented database is appropriate in these cases:

- Conceptual model of database is missing and we need to make changes in the implemented database.
- Database specialist which implemented the database is out of the project or company and we need to understand dependences in the implemented database.
- Customer needs a visual form of the implemented database.
- We assume the information system or project with implemented database, but without documentation and conceptual model, and we need to rebuild or upgrade the implemented database.

For these reasons we propose algorithm for reconstructing the conceptual model from the implemented database. The output of the algorithm will be the conceptual model generated from the implemented database and transformed to the selected modeling tool.

III. Problem Solution

Proposed algorithm consists of few steps which are visually shown in the Fig. 2:

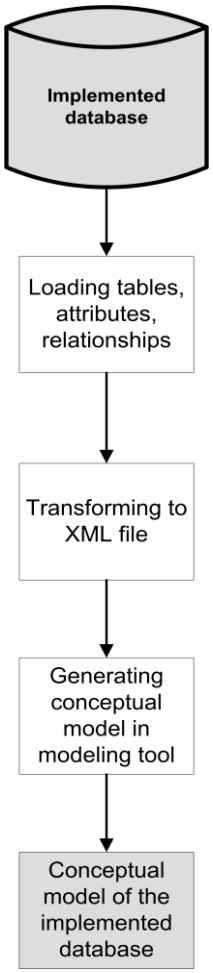

Fig. 2. An algorithm for reconstructing the conceptual model from the implemented database.

A. Connecting to the Implemented Database

First, we need to connect to the implemented database for loading their logical model. There are several ways for connecting to the database implemented in specific RDBMS, we perform connection via JDBC driver which is selected for specific RDBMS. Advantage of connecting via JDBC driver is that JDBC driver provides methods for loading all database tables with their attributes and relationships between specified database tables.

B. Loading Database Tables, Attributes and Relationships

In this step we need to load all database tables, their attributes and all relationships in the implemented database. This step can be performed by general methods provided by JDBC driver which can be generally used in most of RDBMS.

The outputs of this step are three lists:

- List of database tables.
- List of attributes divided according to database tables.
- List of relationships between database tables.

C. Transforming Loaded Data to the XML Document

Based on previous step we propose transforming loaded lists from the implemented database to the XML document. We choose XML document as appropriate format for storing information about database because a lot of data modeling tools can import XML document as input for generating conceptual model. We propose general structure of XML document which can be with suitable transformation used in selected data modeling tools [4]:

- Elements *entity* contains names of loaded database tables.
- Elements *attribute* contains names of loaded attributes of database tables.
- Elements *relationship* consists of subelements for specifying database tables ingoing to relationship with description of relationship.

Example of XML document contains information about sample database is shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<database>
   <entity name="Lecturer">
   <attribute name="name"/>
   <attribute name="surname"/>
   <attribute name="address"/>
   <attribute name="phone"/>
   <attribute name="login"/>
   <attribute name="password"/>
</entity>
<entity name="Course">
   <attribute name="course_id"/>
   <attribute name="name"/>
   <attribute name="location"/>
   <attribute name="start_date"/>
   <attribute name="end_date "/>
   <attribute name="description"/>
</entity>
<relationship>
   <startentity>Lecturer</startentity>
   <relation>lecturer_to_course
   </relation>
   <endentity>Course</endentity>
```

</relationship>
</database>

D. Generating Conceptual Model in the Modeling Tool

Finally we transform XML document to structure supported by selected data modeling tool and generate conceptual model of the implemented database. In specified modeling tool the resulting conceptual model can be exported to other file formats and can be used for possible transforming to logical or physical model in the same or another RDBMS.

Based on information of implemented database in XML document shown above we transform the XML document to CDM file which is supported by modeling CASE tool called Sybase PowerDesigner. The resulting conceptual model is shown in the following Fig. 3:

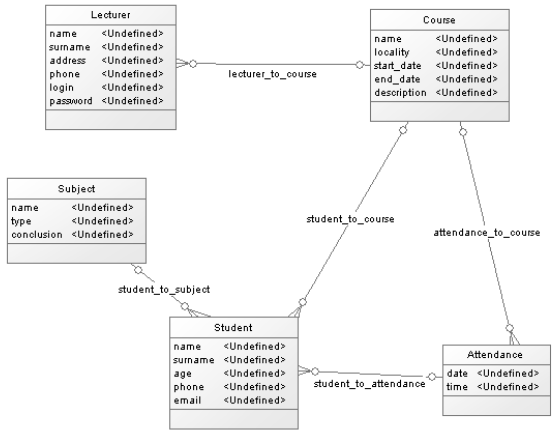

Fig. 3. The resulting conceptual model generated in data modeling tool sybase powerdesigner.

The resulting conceptual model is fully flexible and valid for Sybase PowerDesigner modeling tool. Generated conceptual model can be used for subsequent transforming to logical and physical model of the database, generating SQL dump file, etc.

Proposed algortithm for reconstructing the conceptual model from the implemented database can be also used in other data modeling tools with suitable transformation of the XML document described in this article.

IV. Conclusion

In this article we proposed general algorithm for reconstructing the conceptual model from the implemented database. Algorithm can be used in any RDBMS for generating and showing conceptual model of the specific database. We also propose general structure of XML document which contains information about specific database and can be transformed to various file formats supported by different data modeling tools. At the end of article main steps of the proposed algorithm was described and the outputs of proposed algorithm was verified on selected database and modeling CASE tool SybasePowerDesigner.

Acknowledgment

The presented topic is also a part of the internal grant SGS10/PřF/2012, called Fuzzy modeling tools for analysis and design of information systems, at the Department of Infomatics and Computers, University of Ostrava.

References

[1] J. L. Harrington, *Relational database design and implementation*, Burlington: Elsevier Inc., 2009, ch. 1.
[2] V. Řepa, *Analýza a návrh informačních systémů*, Praha: EKOPRESS, 1999, pp. 146-159.
[3] C. C. Batini, S. Ceri, and S. Navathe, *Conceptual database design: an entity-relationship approach*, Redwood City: The Benjamin/Cummings Publishing Company, Inc, 1992, ch. 1,2,3,4.
[4] B. Walek and C. Klimes, "Fuzzy tool for conceptual modeling under uncertainty," in *Proceedings of SPIE*, Bellingham, vol. 8349, 2012, pp. 83492Z-83492Z-5.
[5] Data *Modeling 101*, [Online]. Available: http://www.agiledata.org/essays/dataModeling101.html, 2011.
[6] J. L. Harrington, *Relational database design and implementation*, Burlington: Elsevier Inc., 2009, ch. 4.
[7] T. Teorey, S. Lightstone, and T. Nadeau, *Database Modeling and Design: Logical Design*, Burlington: Elsevier Inc., 2006, ch.4, 5.
[8] A. Olivé, *Conceptual Modeling of Information Systems*, New York: Springer Berlin Heidelberg, 2007, ch.1.

Bogdan Walek was borned on July 9, 1985 in Karvina, Czech Republic. In 2004 he began studying informatics at University of Ostrava, Czech Republic and in 2007 he earned a bachelor ś degree. In 2009 he received a master ś degree in information systems at University of Ostrava. Since 2006 he works in IT Company, located in Ostrava and called Your System as IBM Lotus Domino Developer. Since 2010 he studies at University of Ostrava as PhD. student. He published several papers focused on fuzzy modeling systems and relational databases at international conferences, such as International Conference on Soft Computing MENDEL, International Conference on Fuzzy Information and Engineering, International Conference on Fuzzy Systems and Neural Computing, etc. His current research interests are fuzzy modeling tools, expert systems, relational databases.