World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:1, No:9, 2007

# University of Jordan Case Tool (Uj-Case-Tool) for Database Reverse Engineering

Fawaz A. Masoud, Heba_tallah Khattab, and Mahmoud Al-Karazoon



***Abstract*—**The database reverse engineering problems and solving processes are getting mature, even though, the academic community is facing the complex problem of knowledge transfer, both in university and industrial contexts. This paper presents a new CASE tool developed at the University of Jordan which addresses an efficient support of this transfer, namely UJ-CASE-TOOL. It is a small and self-contained application exhibiting representative problems and appropriate solutions that can be understood in a limited time. It presents an algorithm that describes the developed academic CASE tool which has been used for several years both as an illustration of the principles of database reverse engineering and as an exercise aimed at academic and industrial students.

***Keywords*—**Reverse engineering, ERD, DBRE, case tools.

## I. Introduction

THE college of the King Abdullah II School for Information Technology (KASIT) at the University of Jordan is devoted for teaching and research. There are many licensed tools used in the college for teaching support such as rational rose, power designer and CaseStudio. The college has eleven PCs labs. The labs will be increased to twenty two in near future. Two labs are devoted to the development of tools, techniques, models and method to support all the engineering activities related to database, software engineering, and their applications. Materials and information systems are developed to link database knowledge with industries. Every information system that has been developed and maintained is assumed to have a complete and up to date technical and functional documentation associated with it. The rest of this paper is organized as follows, we first describe the database reverse engineering in section 2. Then, we describe the database reverse engineering objectives in section 3. In section 4, we briefly describe the transformational technology. In section 5, we talk about the database reverse engineering methodology. In section 6, we discuss the advantages and disadvantages of the available CASE Tools in our college. In section 7, we describe and present the UJ-CASE Tools and the suggested algorithm. We finally present our conclusion and future works in section 8.

F. A. Masoud is with the University of Jordan, Amman-Jordan, Vice Dean of KASIT (e-mail: fawaz@ju.edu.jo).

H. Khattab and M. Al-Karazoon are with the University of Jordan, Graduate students (e-mails: heba_tallah@Yahoo.com, mk80_bt@yahoo.co.uk).

## II. Database Reverse Engineering (DBRE)

Reverse engineering (RE) is defined as a piece of software consists of recovering or reconstructing its functional and technical specifications, starting mainly from the source text of the programs [1]. Database Reverse engineering (DBRE) is a software engineering process through which the analyst tries to understand and document the files and/or the database of an application. More precisely, this process is to generate the complete logical schema and the conceptual schema of this database [1].

The problem is particularly complex with old and bad designed applications. This is because, not only the insufficient documentation (if exists) that can be relied on, but the lack of systematic methodologies for designing and maintaining of these applications. These problems have led to misleading and obscure code. Therefore, reverse engineering has long been recognized as a complex, painful activity, in such a way that it is simply not undertaken most of the time, leaving huge amounts of invaluable knowledge buried in the programs, and therefore definitively lost.

In information systems, or in applications whose central component is a database or a set of permanent files, the complexity can be broken down by considering that the files or databases can be reverse engineered independently of the procedural parts.

The permanent data structures are generally the most stable part of applications; even in very old applications, the semantic structures that underlie the file structures are mainly procedureindependent, though their physical structure is highly proceduredependent; reverse engineering the procedural part of an application is much easier when the semantic structure of the data has been elicited.

Therefore, concentrating on reverse engineering the data components of the application first can be much more efficient than trying to cope with the whole application.

## III. Database Reverse Engineering Objectives

Reverse engineering is just one step in the Information System life cycle. It is generally intended to document, convert, restructure, maintain or extend legacy applications. Here follow some of database reverse engineering objectives [2].

1. *System maintenance*. Fixing bugs and modifying system functions require understanding the concerned component.

2. *System reengineering*. Reengineering a system is changing its internal architecture or rewriting the code of some components without modifying the external specifications. The overall goal is to restart with a cleaner implementation that should make further maintenance and evolution easier.
3. *System integration*. Integrating two or more systems to produce a unique system that includes the functions and the data of the former systems.
4. *Quality assessment*. Analyzing the code and the data structures of a system in some detail can bring useful hints about the quality of this system, and about the way it was developed.
5. *Data administration*. DBRE is also required when developing a data administration function that has to know and record the description of all the information resources of the organization.
6. *Component reuse*. In emerging system architectures, reverse engineering allows developers to identify, extract and wrap legacy functional and data components in order to integrate them in new systems.

## IV. Transformational Technology

Most database engineering processes can be modeled by a chain of schema rewriting operators called transformations [4]. For example, mapping a relational schema from an entity relationship diagram can be carried out by replacing each entity type with a table, each single-valued attribute with a column, each one-to-many relationship type with a foreign key, each multi-valued attribute with a dependent table, etc [1].

## V. Database Reverse Engineering Methodology

In database development methodologies, the complexity has three abstraction levels, conceptual level, logical level, and physical level. In the conceptual level the designer produces a technology independent specification of the information, expressed as a conceptual schema. At the logical level, the information is expressed in a model for which a technology exists, where the required information is organized into a relational or object-oriented logical schema [1]. DBRE methodology consists of three phases which are Project Preparation where system components are evaluated, Data Structure Extraction where logical schema are rebuild, and Data Structure Conceptualization where semantic structures are conceptualized, as shown in Fig. 1.

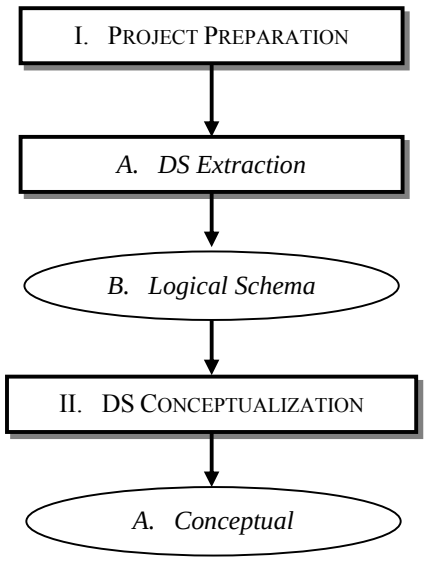


Fig. 1 Database Reverse Engineering phases

The first phase is called project preparation which aims to:

1. Identify and evaluate system components through which the files, programs, screens, reports, forms, data dictionaries, repositories, program sources and documentation.
2. Recover the system architecture that consists in drawing the main procedural and data components of the system and their relationships.
3. Identify system resources, such as skills, calendar, machine, tools and budget.

The second phase is called data structure extraction process aims to rebuilding a complete logical schema in which all the explicit and implicit structures and properties are documented. The third phase is called the data structure conceptualization process which tries to specify the semantic structures of this logical schema as a conceptual schema. While some constructs are fairly easy to interpret (e.g., a standard foreign key generally is the implementation of a one-to-many relationship type), others pose complex problems due to the use of tricky implementation and optimization techniques.

## VI. Case Tools Support

There are many licensed CASE tools at KASIST that are used to perform database reverse engineering such as power designer, Case-Studio and rational rose. The rational rose is mainly used for the course systems development and CASE Tools carried out by the department of the computer information systems in the college. The other two tools are mainly used for database application developments. Power Designer consists of set of tools, each of them used to do certain job, which gives Power Designer many advantages. For example, Power Designer has DataArchitect provides database designers and database administrators bi-level conceptual and physical data modeling for design, generation,

reverse engineering, maintenance, and documentation for several DBMSs. The Case-Studio is a database design tool, which can be used in reverse engineering of the database structures (tables), generate SQL scripts, generate very detailed HTML reports, generate ERM diagram, and Data Flow diagram.

Our experience in using the above CASE Tools reveals many disadvantages and bugs found in such tools. Some of these disadvantages are given below:

- The Power Designer gives incorrect results for producing the ERD from the database tables. Moreover, when we build the ERD from a written SQL statements, it doesn't make a check if there is/are error(s) in the code and it gives a message "The database has been successfully reverse-engineered" in all cases, although the code has error(s), and can not determine where is/are this/these error(s).
- The Case Studio tool has a weak point that it doesn't perform reverse engineering process from a written SQL.

The above mentioned problems have compelled us to generate a new modified tool called University of Jordan Case Tool (UJ-CASE-TOOL). This tool can solve some of the bugs that available in the Power Designer and the weak point of the Case Studio. The UJ-CASE-TOOL and its algorithm are described in section 7.

## VII. University of Jordan Case Tool (UJ-CASE-TOOL)

The UJ-CASE-TOOL tool can conduct a reverse engineering process on the existing Access database tables and generate Entity Relationship Diagram (ERD) and specify the relationship cardinalities (one-to-one, one-to-many, and many-to-many), and display the data type for each attribute (number, varchar, text, …, etc). Further more, our tool can generate ERD from DDL code (SQL statements). Help for users is provided as part of the tool. The main algorithm of our tool is described in Fig. 2.

```
Procedure generateERD ( )
Input: existing database or SQL commands.
Output: Entity-Relationship Diagram
Begin
 Case 1: existing DB.
    Read database;
   Fetch table names.
   For each table do
      Fetch the attributes and their data types
      Determine the Primary key(s) "if exist"
      Determine the foreign key(s) "if exist"
   End for
   For each table do
      Determine that tables which are related to each other.
      Determine the cardinality of this relationship.
   End for
 End case 1
 Case 2: SQL statements/code
   For each SQL statement does
      Generate tokens
      Check the syntax of the statement
      If the syntax is correct
        For each table do
          Fetch the attributes and their data types
            Determine the Primary key(s) "if exist"
            Determine the foreign key(s) "if exist"
        End for
        For each table do
          Determine that tables which are related to each other.
            Determine the cardinality of this relationship.
          End for
        Else
          Inform the user where the error in the SQL statement is
        End if
   End for
 End case 2
 Draw ERD.
End procedure
```

Fig. 2 the main algorithm of UJ-CASE-TOOL

## VIII. Conclusion and Future Works

This paper has pointed out the limits of the power designer and Case-Studio tools that are used in our IT College. This paper has presented a new CASE tool developed at the University of Jordan which addresses an efficient support of knowledge transfer, both in university and industrial contexts. It is a small and self-contained application exhibiting representative problems and appropriate solutions that can be understood in a limited time. It presents an algorithm that describes the developed academic CASE tool which has been used for several years both as an illustration of the principles of database reverse engineering and as an exercise aimed at academic and industrial students. However, our tool is not comprehensive one for other properties of database system. Therefore, for future work, we suggest the following additional capabilities to be added to our tool to perform additional functions including:

- Increase the inputs as much as we can to enable the reverse engineering of more database types, like ORACLE database, SQL Server database, and others.
- To provide more properties for the relationships, such as relationship integrity.

- To modify the generated ERD and create corresponding database. Such changes are required based on the functional or business requirements.

## REFERENCES

[1] Abraham Silberschatz, Henry F. Korth and S. Sudarshan, "Database System Concepts", McGraw-Hill, 5th edition, 2006.
[2] Ian Somerville," Software Engineering", 7th edition, Addison Wesley, 2004.
[3] Case Studio, http://www.casestudio.com.
[4] Jean-Luc Hainaut, "Database Reverse Engineering, University of Namur-Institute d'Informatique", Belgium, 1998.
[5] Power Designer http://is.twi.tudelft.nl.