A Tool for Database Testing and Optimization

Bogdan Walek and Cyril Klimeš

***Abstract*—The article deals with testing database against original requirements for its creation. Currently there are several tools for unit database testing, testing stored procedures and functions in database or performance and load testing of relational database. But there is no tool for testing logical model of the relational database against requirements for its creation. In the paper we propose a tool for testing logical model of the implemented database against original requirements for its creation. Proposed tool uses XML document for representation of the requiremensts and JDBC methods for loading logical model of the implemented database. Outputs of the proposed tool are tables with comparison of entities and tables, attributes and relationships. Finally, the tool generates possible database optimization proposals based on the tool ´s outputs. At the end of the article proposed tool is shown on testing database of the university information system and possible optimization proposals for this database are suggested.**

***Index Terms*—Relational database, testing, optimization, database testing.**

I. Introduction

Today, in the area of information systems development is appropriate to monitor, if customer´s requirements are understood and implemented correctly. This activity should be provided continously during the implementation to ensure higher quality of the resulting information system (further in text referred as IS). With the continous testing of specific parts of the IS, the possible errors can be detected and fixed even during the implementation of the IS. The emphasis on the correct understanding and implementing of the customer´s requirements is also based on iterative approach to the design and development of IS [1].

Information systems are mostly connected to the database, so it is suitable to focus also on testing database due to the original requirements for its creation which are defined in the analysis of the IS. For the proper functioning of an IS is needed to ensure correct implementation of the database, so it´s appropriate to test whether the database is designed and implemented correctly.

This paper is a continuation of articles [2], [3], [5]. The type of database which is used in this article is a relational database [6].

II. Problem Formulation

From the survey called The Current State of Data Management Survey intiated between developers, IT management and data professionals it can be concluded that 95.7% of respondents believe that data is a corporate asset, but only 40.3% had a database test suite to validate the data and of those without a test suite only 31.6% had even discussed the concept. 63.7% of respondents indicated that they implemented mission-critical functionality in the database, but only 46% had a regression tests in place to validate the logic of this functionality [4].

Currently, there are tools for database refactoring, testing data consistency, stored procedures, triggers, data validity [4]. Selected tools are described below [4]:

SQL Refactor – the tool allows finding invalid objects in database, finding unused parameters and variables, renaming objects, splitting tables, etc. The result is more readable and clear database schema, and database optimization.

DTM Data Generator **–** with this tool we can generate test database tables and large amount of data for performance testing, loading tests and usability testing.

DbFit – the tool allows testing SQL queries, stored procedures and functions.

From this short description of current tools for database testing we can conclude that current tools are not able for testing logical structure of the implemented database against original requirements for its creation.

But inappropriate or poor design of database logical model can cause these problems [7]:

1) Data redundancy.
2) Problems with data consistency.
3) Meaningful identifiers.
4) Unnecessary database tables or relationships which can store useless data are linked to the implemented classes of methods.

Based on these facts we propose a tool for testing logical model of the database against original requirements for its creation. The outputs of the tool will be differences between requirements and implementation of the database and proposals for optimizing the implemented database.

III. Problem Solution

The main goal of this article is to propose and describe a tool for testing logical model of the database against original requirements for its creation. Proposed tool is shown in the Fig. 1:

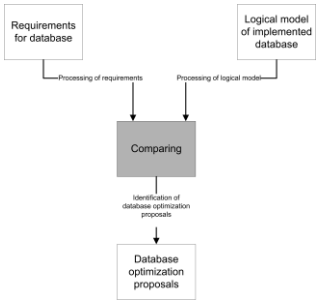

Fig. 1. Proposed tool for database testing.

Manuscript received May 30, 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).

Now we can describe specific parts of the proposed tool.

A. Loading Requirements for Creating a Database into the XML Document

In the first step we need to load and understand the original requirements for database creation. Requirements for creating database can be stored in various ways, such as vision document, notation from the meeting with customer, communication with customer. After loading requirements we need to transform requirements to appropriate format which can be easily processed by the tool and is understandable for customer. Suitable format for storing and processing requirements is an XML document. We propose the stucture of XML document described on the part of requirements for database of the university information system:

```
<?xml version="1.0" encoding="UTF-8" ?>
<requirements>
<entity name="Department">
   <attribute name="name"/>
   <attribute name="shortcut"/>
   <attribute name="address"/>
   <attribute name="email"/>
   <attribute name="phone"/>
</entity>
<entity name="Faculty">
   <attribute name="name"/>
   <attribute name="shortcut"/>
   <attribute name="head"/>
   <attribute name="staff"/>
   <attribute name="email"/>
   <attribute name="phone"/>
</entity>
<relationship>
   <startentity>Department</startentity>
   <relation>is_a_part_of</relation>
   <endentity>Faculty</endentity>
</relationship>
</requirements>
```

In this example of XML document we can see two entities *Department* and *Faculty*, their attributes and relationships between these entities.

B. Loading Logical Model of the Implemented Database

In this step we need to load a logical model (database tables, attributes, relationships) of the database which was created and implemented based on the requirements shown in previous example. Firstly we have to connect to the implemented database and their RDBMS (relational database management system). One of the possible solutions is to connect to database via JDBC driver (because it is easy and there are default and important methods included in the JDBC driver for specific RDBMS. Next, we need to load database tables, attributes and relationships - logical model of the implemented database. Loaded logical model should be also transformed to the proposed XML document structure. Structure of the XML document is very similar to the XML document for requirements:

```
<?xml version="1.0" encoding="UTF-8" ?>
<database>
<entity name="Department">
   <attribute name="name"/>
   <attribute name="shortcut"/>
   <attribute name="address"/>
</entity>
<entity name="Faculty">
   <attribute name="name"/>
   <attribute name="head"/>
   <attribute name="address"/>
   <attribute name="email"/>
   <attribute name="phone"/>
</entity>
<relationship>
   <startentity>Department</startentity>
   <relation>department_to_faculty
   </relation>
   <endentity>Faculty</endentity>
</relationship>
</database>
```

We can see two database tables *Department* and *Faculty*, their attributes (columns) and relationship between these database tables in the XML document.

C. Comparing the XML Documents

In this step we propose algorithm for comparing the XML files for requirements and the implemented database. The proposed algorithm consists of these parts:

1) Forall entities in XML requirements find specific entity in the implemented database.
2) Write possible differences.
3) Repeat steps 1. and 2. for attributes and relationships.
4) Show differences between requirements and the implemented database.

Now we will show three tables with comparing bettween requirements and the implemented database based on the database of the university information system:

TABLE I: COMPARING ENTITIES.

Entity in requirements	Entity in database	Differences
University	University	
Faculty	Faculty	
Department	Department	
Student	Student	
Subject	Subject	
Subject_type		Missing database table Subject_type in database
Conclusion_type		Missing database table Conclusion_type in database
Registered subjects	Registered subjects	

TABLE II: COMPARING ATTRIBUTES.

Attribute in requirements	Attribute in database	Differences
University.name	University.name	
University.address	University.address	
University.phone	University.phone	
University.email	University.email	
Faculty.name	Faculty.name	
Faculty.shortcut		Missing attribute Faculty.shortcut in database

Faculty.head	Faculty.head	
Faculty.address	Faculty.address	
Faculty.email	Faculty.email	
Faculty.phone	Faculty.phone	
Department.name	Department.name	
Department.shortcut	Department.shortcut	
Department.address	Department.address	
Student.name	Student.name	
Student.surname	Student.surname	
Student.email	Student.email	
Student.number	Student.number	
Student.phone		Missing attribute Student.phone in database
Subject.name	Subject.name	
Subject.credits	Subject.credits	
Subject.description	Subject.description	
Subject_type.name		Missing attribute Subject_type.name in database
Conclusion_type.name		Missing attribute Conclusion_type.name in database
	Subject.type	Unnecessary attribute Subject.type in database
	Subject.conclusion	Unnecessary attribute Subject.conclusion in database
Registered_subjects.student	Registered_subjects.student	
Registered_subjects.subject	Registered_subjects.subject	

TABLE III: COMPARING RELATIONSHIPS.

Start entity in require- ments	End entity in requirements	Start entity in database	End entity in database	Differences
Faculty	University	Faculty	University	
Department	Faculty	Department	Faculty	
Student	Department	Student	Department	
Subject	Department	Subject	Department	
Subject_type	Subject			Missing relationship in database
Conclusion_type	Subject			Missing relationship in database
Subject	Registered_subjects	Subject	Registered_subjects	

We can conclude that in the implemented database are missing database tables *Subject_type* and *Conclusion_type*, few attributes are missing and few attributes are unnecessary.

D. Optimization Proposals for the Implemented Database

From the previous step and tables with differences between requirements and the implemented database we can conclude optimization proposals which can be applied for the implemented database:

- Add database tables *Subject_type* and *Conclusion_type*.
- Add attributes *shortcut* to database table *Faculty* and *phone* to database table *Student*.
- Add attributes *name* to database table *Subject_type* and *name* to database table *Conclusion_type*.
- Remove attributes *type* and *conclusion* from database table *Subject*.
- Add relationships between database tables *Subject_type* and *Subject*, and *Conclusion_type* and *Subject*.

In Fig. 2 and Fig. 3 are shown requirements for database creation and the implemented database:

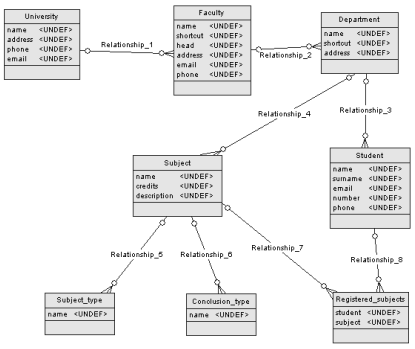

Fig. 2. Requirements for database of the university information system.

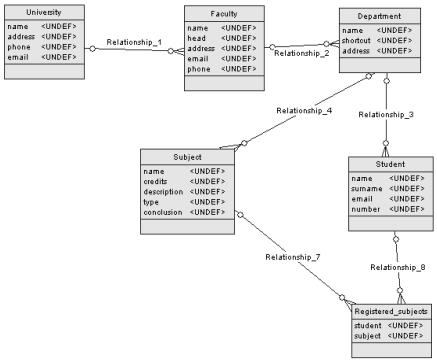

Fig. 3. Implemented database of the university information system.

Based on optimization proposals the database specialist should modify logical model of the implemented database.

IV. CONCLUSION

In this paper the current state in the area of database testing was analyzed. Then we proposed a tool for testing logical model of the implemented database against original requirements for its creation. Then, the specific parts of the proposed tool were described. Finally, we show proposed tool on testing database of the university information system and we suggest possible optimization proposals for this database. Proposed tool can be used by database specialist to analyze differences between requirements and implemented database, and then the implemented database should be optimized.

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] *Rational Unified Process*, [Online]. Available: http://www-01.ibm.com/software/awdtools/rup/, 2012.
[2] B. Walek, "Testování databáze informačního systému", in *Proc. of Studentská vědecká konference 2011*, Ostrava, 2011, pp. 218-221.
[3] B. Walek and C. Klimeš, "Testing logical structure of the information system database," *International Conference on Business Intelligence and Financial Engineering*, Hong Kong, 2011, to be published.
[4] *How to Regression Test a Relational Database*, [Online]. Available: http://www.agiledata.org/essays/databaseTesting.html, 2010.
[5] B. Walek and C. Klimeš, "Testing database of information system using conceptual modeling," in *Proc. of International Conference on Fuzzy Systems and Neural Computing*, Paris, 2012, pp. 736-742.
[6] J. L. Harrington, *Relational database design and implementation*. Burlington: Elsevier Inc., 2009, ch. 1.
[7] J. L. Harrington, *Relational Database Design Clearly Explained, Second Edition*, San Francisco: Elsevier Inc., 2002, ch. 1, pp. 4-9.

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.