A Framework for Test Case Impact Analysis of Database Schema Changes Using Use Cases

Jiratchaya Jainae and Taratip Suwannasart

***Abstract*—Software testing is an important activity in software development. Software testing requires design and creation of test cases for testing the system; therefore, test cases are important for software testing. Moreover, database applications become an important part of software and are increasingly complex. If the database schema is changed, database schema can affect test cases which are principal of software testing. It is not easy to specify affected test cases that are still usable, or unusable. So, if a software tester uses unusable test cases, this can lead to various troubles such as wasted effort, as well as wasted time and cost to find affected test cases for the generation of new test cases. In this paper, we have presented a framework that is planned to implement tool. Furthermore, this framework have displayed the method to support finding impacts on test cases from database schema changes. The approach of finding impacts on test cases based on black-box testing by using use cases.**

***Index Terms*—Software testing, test case, use case, database schema change**

I. Introduction

Software testing [1] is an important activity in the software development life cycle. It requires a high cost and effort to perform, because it requires generation of test cases, which have to show correctness of a system. Therefore, test cases are one of significant factors in software testing. Moreover, database applications become popular and an important part of software, especially database schema, which is a main component of database. Database scheme changes can happen all the time. For instance, some attributes in a table are dropped, or some tables are added into the schema. They affect other software components, including test cases. Some affected test cases are still usable, but others are not. If a software tester uses an unusable test case to test the software, this can lead to many problems such as wasted effort for the software testing process, as well as wasted time and cost to find affected test cases for the generation of new test cases.

From many studies [2]-[4] approaches analysis of impacts from database schema changes, most analysis processes are based on source code or white-box testing [5], [6] techniques. Moreover, they focus on increasing performance and accuracy of database schema impact analysis. Therefore, this paper presents a framework of test case impact analysis for database schema changes, and focuses on black-box testing [5], [6] techniques by using use cases.

Manuscript received June 14, 2013; revised August 7, 2013.

The authors are with the Software Engineering Laboratory Center of Excellence in Software Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand (e-mail: jiratchaya.jinae@gmail.com; Taratip.S@chula.ac.th).

The paper is organized as follows. Section II reviews related works. Section III describes use case description, Relational database constraints is defined in Section IV, while Section V explains the approach of this framework. Finally, Section VI summarizes conclusions and future work.

II. Related Work

A. Karahasanovic [2] developed a tool called "Schema Evolution Management Tool" (SEMT) for finding the impact of schema changes upon object-oriented applications. This tool displays the components of the database schema as a graph. Furthermore, this tool identifies impact at the field and method levels.

A. Maule, W. Emmerich and D. Rosenblum [3] presented a static program analysis technique to identify the impact of relational database schema changes on object-oriented applications. This research developed the Schema Update Impact Tool (SUITE). The tool uses dataflow analysis to extract all possible database interactions. The tool then uses the information to predict the effects of database schema changes. Furthermore, the program slicing is used for reducing the size of the original program required for analysis.

S. Gardikiotis and N. Malevris [4] presented a twofold database schema change impact analysis, in terms of source code and in terms of test suite. Also, a software tool called "DATA" was developed for impact analysis of both the source code and test cases. The program slicing technique is used to analyze the source code by displaying it as a control flow graph, and then test cases are analyzed.

III. Use Case Description

Cockburn [7], [8] has presented many forms of use case descriptions, such as the one-column table, which is recommended for being easy to understand. This use case description form has a successful scenario and may have alternative scenarios for explaining unexpected events which make the use case unsuccessful and not reach the goal.

Moreover, S. Leeraharattanarak [7] created a use case description form, of which the main content is based on Cockburn's form, and is defined more necessary details as shown in Table I.

IV. Relational Database Constraints

Relational database constraints are identified on database by user. This paper is interested in entity integrity and domain constraints [9].

A. Entity Integrity

Entity integrity is an integrity rule, where every table must have a primary key and that the column or columns chosen to be the primary key should be unique and not null.

B. Domain Constraint

Domain Constraint is an integrity rule, where a domain of possible values should be associated with every attribute. These domain constraints are the most basic form of integrity constraint. The domain constraints consist of data type (Character, Integer, Double, Boolean etc.), format and range.

TABLE I: USE CASE DESCRIPTION [7]

Use case no:	1			
Use case name:	Add contact			
Description:	Add new contact to list			
Actor:	Client			
Pre-condition:	Client enters the contact information: name, telephone, address and zip code.			
Required items				
Item name	Item type	Item size	Max value	Min value
name	String	20	-	-
lastname	String	10	-	-
address	String	35	-	-
zip code	Integer	5	10000	99999
Is abstract:	0			
Success scenario				
Condition no:	1	(name.length> 0)		
Step	Action			
1 2 3	System submits the contact information from client. System saves the contact information into database. System shows a message. Saving new contact complete.			
Alternative scenario				
Condition no:	2.1	(name.length<= 0)		
Step	Action			
2.1.1	System shows an error message. Please enter name.			
Post-condition :	0	System saves new contact and shows a message. Saving new contact complete.		
	2.1	System shows an error message. Please enter name.		

V. APPROACH OF OUR FRAMEWORK

This paper presents a framework of impact analysis on test cases affected by database schema changes using use cases. The framework is shown in Fig. 1.

Our framework consists of 4 major steps as follows:

A. Analyzing Database Schema File

There are 2 input files for analysis in this framework. These files are SQL scripts, consisting of create-scripts and alter-scripts. An example of alter-scripts shown in Fig. 2.

1) Create-script

Create-script provides data for creating a database schema or a database schema structure that consists of the schema name, table name, attribute name, type of attribute, size of attribute, null, unique, and primary key. An example is shown in details in Table II.

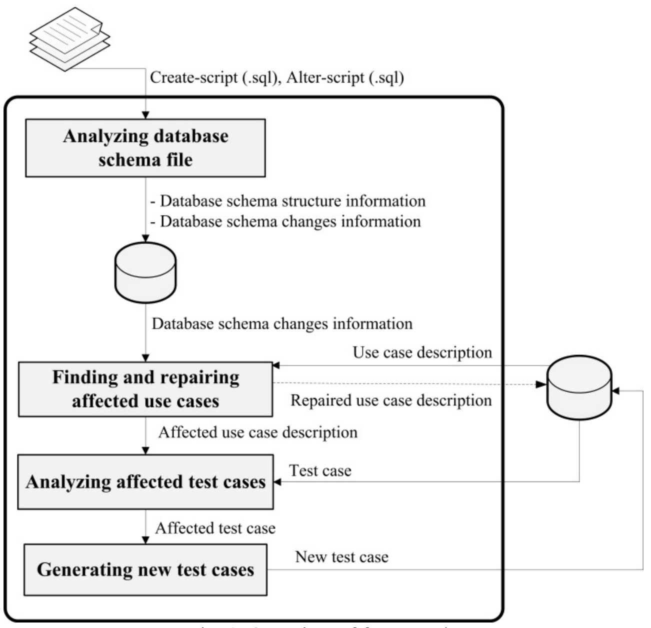
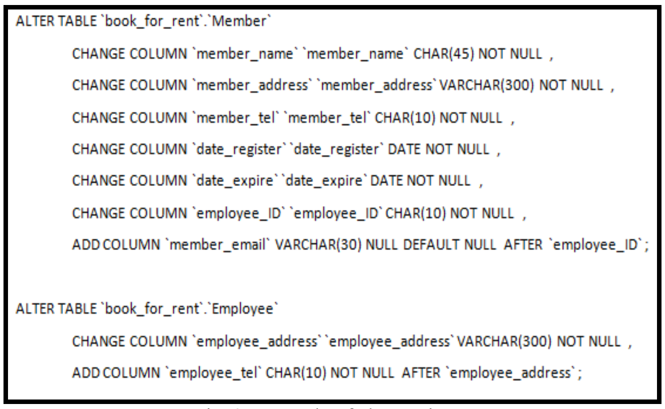

Create-script (.sql), Alter-script (.sql)
Analyzing database schema file
- Database schema structure information
- Database schema changes information
Database schema changes information
Use case description
Finding and repairing affected use cases
Repaired use case description
Affected use case description
Analyzing affected test cases
Test case
Affected test case
Generating new test cases
New test case

Fig. 1. Overview of framework.



```
ALTER TABLE `book_for_rent`.`Member`
        CHANGE COLUMN `member_name` `member_name` CHAR(45) NOT NULL ,
        CHANGE COLUMN `member_address` `member_address` VARCHAR(300) NOT NULL ,
        CHANGE COLUMN `member_tel` `member_tel` CHAR(10) NOT NULL ,
        CHANGE COLUMN `date_register` `date_register` DATE NOT NULL ,
        CHANGE COLUMN `date_expire` `date_expire` DATE NOT NULL ,
        CHANGE COLUMN `employee_ID` `employee_ID` CHAR(10) NOT NULL ,
        ADD COLUMN `member_email` VARCHAR(30) NULL DEFAULT NULL  AFTER `employee_ID`;

ALTER TABLE `book_for_rent`.`Employee`
        CHANGE COLUMN `employee_address` `employee_address` VARCHAR(300) NOT NULL ,
        ADD COLUMN `employee_tel` CHAR(10) NOT NULL  AFTER `employee_address`;
```

Fig. 2. Example of alter-scripts.

TABLE II: CREATE-SCRIPT INFORMATION

Table name	Attribute name	Attribute type	Attribute size	Null	Unique	Primary key
Customer	ID	CHAR	10	Not null	Unique	PK
	name	VARCHAR	50	Not null	-	-
	address	VARCHAR	300	Null	-	-
	telephone	CHAR	15	Null	-	-
Employee	register	DATE	-	Null	-	-
	em_ID	CHAR	5	Not null	Unique	PK

2) Alter-script

Alter-script provides data for database schema changes. The type of changes depends on SQL command, namely, ADD, DROP and CHANGE. Changes cover entity integrity rules (primary key, unique, and not null), and domain constraints (table name, SQL command, attribute name, new attribute name, attribute type and attribute size). An example is shown in Table III.

This framework extracts create-scripts to get schema name, table name, attribute name, type of attribute, size of attribute, null, unique, and primary key, for recording information about database schema structure. Furthermore, this framework extracts alter-scripts to get table name, SQL command, attribute name, new attribute name (if any), attribute type, and attribute size, for recording information about database schema changes.

B. Finding and Repairing Affected Use Cases

This step finds use cases that are affected by database schema changes. Affected use cases are then repaired according to the latest database schema that has been changed. There are 2 sub steps explained as follows:

1) Finding affected use cases

This step uses the existing use case descriptions of the system and information in Table II. The framework takes each attribute in Table II to check for consistency with each required item in the use case description. Consistency analysis depends on the type of SQL command that is queried, between DROP, ADD and CHANGE as shown as an activity diagram in Fig. 3.

2) Repairing affected use cases

This step takes affected use cases found in 2.1 to be repaired according to the latest database schema. The framework uses the changed data to repair the use case description in terms of required items and conditions of each scenario. Moreover, the repair approach depends on the SQL command as shown in Table IV.

C. Analyzing Affected Test Cases

Existing test cases, which were generated from the use case description [7] are taken to be analyzed. An example of a test case is shown in Table V. Therefore, affected test cases are only generated from affected use cases.

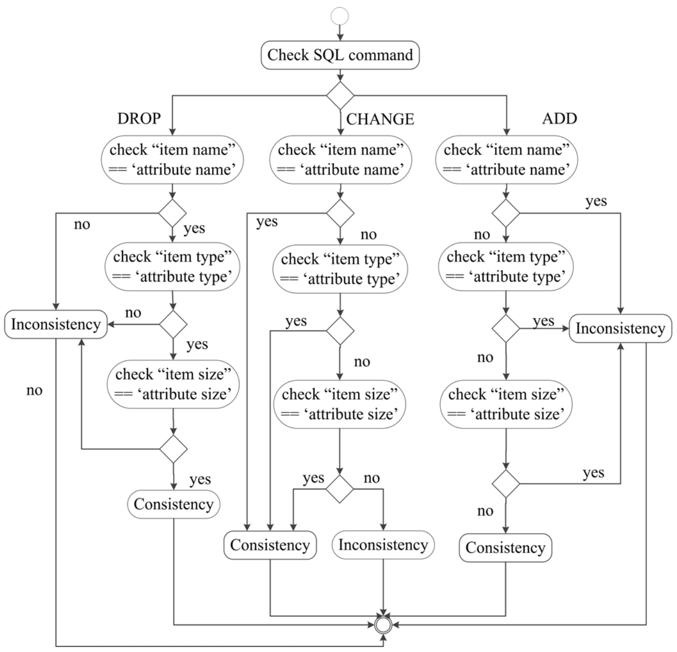

Check SQL command
DROP
CHANGE
ADD
check "item name" == 'attribute name'
check "item type" == 'attribute type'
check "item size" == 'attribute size'
yes
no
Inconsistency
Consistency

Fig. 3. Consistency analysis for finding impact test cases.

TABLE IV: AFFECTED USE CASE REPAIR APPROACH

Type of SQL command	Description
DROP	Data using DROP commands are deleted in the required items and scenarios according to the new database schema.
ADD	Data using ADD command is added, user has to add more use case description in the required items and scenarios according to the new database schema.
CHANGE	Data using CHANGE command is changed in the part of required items and scenarios according to the new database schema.

TABLE III: ALTER-SCRIPT INFORMATION

		SQL command	Attribute name	Attribute type	Attribute size	Null	Unique	Primary key
Table name	Customer	CHANGE	name	VARCHAR	35	Not null	-	-
		DROP	address	VARCHAR	300	Not null	-	-
	Employee	CHANGE	register	DATE	-	Not null	-	-
		CHANGE	em_ID	CHAR	10	Not null	Unique	PK
		ADD	email	VARCHAR	30	Null	-	-

TABLE V: EXAMPLE OF A TEST CASE

Use case no:	1
Test case no:	1.1
Version no:	1.0
Description:	Success scenario
	(member_ID.length > 0)&&(member_name.length >0)&&(member_ID>="0000000000")&& (member_ID =< "9999999999")
Type:	Valid
Input	
Name	Value
member_ID	4536731356
member_name	nerslkd
member_address	234dhsKr3
member_tel	2314
date_register	11-23-2012
date_expire	09-13-2014
Expected output:	System shows a message. "Saving new member complete."

This step compares the input value of the test case with the constraints of required items in the repaired use case description. The analysis process is shown in Fig. 4.

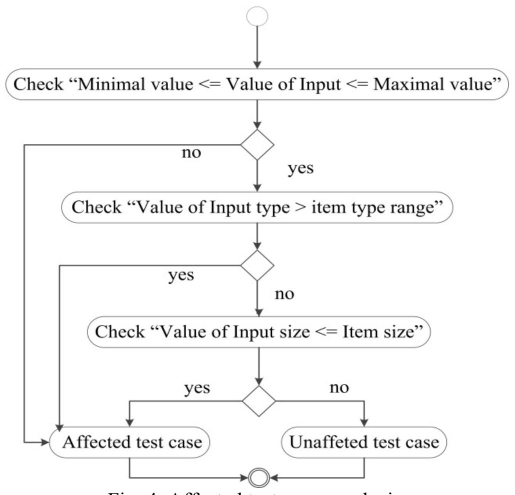

Check "Minimal value <= Value of Input <= Maximal value"
no
yes
Check "Value of Input type > item type range"
yes
no
Check "Value of Input size <= Item size"
yes
no
Affected test case
Unaffeted test case

Fig. 4. Affected test cases analysis.

In summary, there are 2 types of test cases that are analyzed, shown in Table VI.

TABLE VI: TYPES OF TEST CASE THAT WERE ANALYZED

Type of test cases	Description
Unaffected test cases	This type is not affected by database schema changes. The input values of these test cases are still useable, because their values follow the constraints of required items in the use case description. So, new test cases do not need to be generated.
Affected test cases	This type of test case is affected by database schema changes. The values of these test cases are unusable, because their values do not follow the constraints of required items in the use case description. So, new test cases need to be generated instead.

D. Generating New Test Cases

This step generates new test cases to replace only affected test cases. This paper applies the approach [7] to generate new test cases for replacing affected test cases and we divide the test cases that are generated into 2 types: valid and invalid. An example of a valid test case is shown in Table VI.

VI. CONCLUSION AND FUTURE WORK

This paper presents a framework for test case analysis of database schema changes. This framework focuses on black box techniques by using use case description. First, we find use cases affected by database schema changes. If affected test cases are unusable, we will generate new test cases according to the new database schema.

In the future, we plan to implement a tool analyzing of database schema changes using use cases.

REFERENCES

[1] C. Doungsa-ard, K. Dahal, A. Hossain, and T. Suwannasart, "Test data generation from uml state machine diagrams using Gas," in *Proc. International Conference on Software Engineering Advances-ICSEA*, 2007, pp. 47.

[2] A. Karahasanovic, "Identify impact of database schema on application," *Industrial Systems Development Group,* Department of Informatics, University of Oslo, 2001.

[3] A. Maule, W. Emmerich and D. S. Rosenblum, "Impact analysis of database schema changes," in *Proc. International Conference on Software Engineering - ICSE*, 2008, pp. 451-460.

[4] S. K. Gardikiotis and N. Malevris, "A two-folded impact analysis of schema changes on database applications," *International Journal of Automation and Computing*, 2009, pp. 109-123.

[5] S. Leeraharattanarak, "Approach for automatically generating test cases from use cases," M.S. thesis, Dept. Com. Eng., Chulalongkorn Univ., Bangkok, Thailand, 2005.

[6] A. Cockburn, *Writing Effective Use cases*, U.S.A: Addison-Wesley, 2000.

[7] T. Murnane, and K. Reed, "On the effectiveness of mutation analysis as a black box testing technique," in *Proc. Software Engineering Conference*, 2001, pp. 12-20.

[8] M. Chan and S. Cheung, "Applying white box testing to database applications," CSTR, Hong Kong University of ER-Models to Generate Test Cases, Science and Technology, 1999.

[9] P. Tongruk and T. Suwannasart, "A tool for generating test case from relational database constraints testing," in *Proc. International Conference on Computer Science and Information Technology -ICCSIT*, pp. 435-439, 2009.

Jiratchaya Jainae has got B.Sc. in Computer Science, Thammasat University, BKK, Thailand, 2011

She is a young researcher from Thailand. She specializes in software engineering. Currently, she is a master student in computer engineering, Chulalongkorn University. Her main research interests include software testing and database.

Taratip Suwannasart has got her Ph.D in Computer Science, llinois Institute of Technology, CHI, U.S.A, 1996 and M.Sc. in Computer Science, Chulalongkorn University, BKK, Thiland, 1991.

She currently is associate professor at Chulalongkorn University. Her areas of interest are Test Process Improvement, Test-Related Metrics, Software Testing and Quality Assurance, Software Process Engineering, Software Testing Techniques and Methods, and Test Management. She has several papers on this subject, especially Software Testing and Quality Assurance, and Software Engineering.