Impact Analysis on Database Instance from Functional Requirements' Input Changes

Kritsada Kaenchaliao and Taratip Suwannsart

Abstract—**Requirement changes always occur in the software development process. Functional requirements are used for designing software functions. Some functions are associated with databases, such as insert or update function. Thus, functional requirement inputs are associated with database schema. If inputs of functional requirements are changed, the database schema must be affected. Database instances which are generated from database schema must be affected as well. To test software functions, test cases are created by using the functional requirement inputs and database instances. Therefore, when inputs of functional requirement are changed, database instances must be updated. This paper proposes an approach to analyze the impact to database instances from functional requirements' input changes, which can affect database schema, database instances, other functional requirements and test cases.**

Index Terms—**Database Schema, Database Instance, Test Case, Impact Analysis, Functional Requirements**

I. Introduction

IN general, software development has 4 main processes: Requirements Gathering, Analysis & Design, Testing, and Deployment & Maintenance. Requirement changes always occur in software development. When changes occur, they affect many components of the software.

Functional requirement is [1], [2] a requirement that specifies a function that a system or system component must be able to perform to help users perform particular tasks, which is used to design software functions. Some functions are associated with databases. These consist of [3] user-defined operations (or transactions) that will be applied to the database, including both retrievals and updates. Therefore, the structure of functional requirements input is associated with a database schema. If functional requirements inputs are changed, they may affect the database schema. When database schema is affected, database instances will be affected as well.

In software testing, test cases [4] are used for testing the behavior of the program. Test cases consist of an ID, input or test data and expected output. To test a function associated with a database, test cases are created based on inputs of functional requirement and database instance. Therefore, it is necessary to analyze the impact to database instances.

Manuscript received January 8, 2017; revised January 23, 2017. K. Kaenchaliao and T. Suwannasart are with the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand e-mail: k.kanchalew.b@gmail.com, Taratip.S@chula.ac.th

This paper proposes an approach to analyze the impact on database schemas, database instances, functional requirements, and test cases when inputs of functional requirements are changed. Database schemas and database instances are analyzed to determine impact and then updated. Finally, test cases are analyzed then updated if necessary, using a Requirement Traceability Matrix (RTM). [5] This matrix gives a list of requirements and their corresponding test cases. Mapping from requirements to test cases can be one to many.

The rest of the paper is organized as follows. Section II describes related work. Section III introduces the background of database instance, database schema, and schema-based constraints. For Section IV presents an approach for impact analysis on database instances. Lastly, Section V concludes all the contents provided in this paper, altogether with future work.

II. Related work

A. Kampeera and T. Suwannasart [6] proposed an approach to analyze impact on database schema and test cases from changed inputs of functional requirements by comparing 2 versions of functional requirements with different inputs. A functional requirement consists of a function number, a function name, an objective, a function version, and input specifications which include an input name, data type, length, and constraints. An SQL script is then generated to update the database schema and the requirement traceability matrix is used for tracking impacted test cases. This research has not concerned on analyzing of the impact on database instances.

C. Sriarpanon and T. Suwannasart [7] proposed a tool for impact analysis on source code and test cases for database schema changes, using SQL statements for creating a table, and log files that contain SQL statements for altering the table structure. This tool allow 5 types of changes (add new field, drop field, change name, change type, and change size).

P. Tongrak and T. Suwannasart [8] proposed a tool for relational database constraints testing. Relational database constraints consist of entity integrity constraints, referential integrity constraints, and domain constraints. The tool retrieves database schemas to generate test cases. These test cases are generated in the form of SQL statements, including insert, update, and delete statements.

J. Jainae and T. Suwannasart [9] proposed a framework to find the impact on test cases from database schemas by using use cases. First, database schema is analyzed using SQL scripts. These consist of create-script and alter-script.

ISBN: 978-988-14047-3-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Create-script is a SQL script for creating a database schema. Alter-script is a SQL script for altering a database schema. This script provides data of database schema changes. The type of change depends on the SQL command in alter-script (ADD, DROP, and CHANGE). Second, use cases are analyzed and repaired by checking consistency between conditions in use case descriptions and database schema changes information. Third, test cases which are generated from use case descriptions of affected use cases are repaired.

Gregory M. Kapfhammer and et al. [10] introduced a test data generation technique for relational database integrity constraints testing. In this research, integrity constraints are divided into 5 types (primary keys, unique constraints, foreign keys, not null constraints, and check constraints). Test data are generated in the form of SQL insert statements. These statements contain data values that exercise each of the schema's integrity constraints according to original database schema. Test data are evaluated using mutation testing. These mutants are database schemas which are changed from original database schema.

Andy Maule and et al. [11] proposed an approach to analyze impact from relational database schema changes upon object-oriented applications. This approach consists of 3 major steps. First, 'Program Slicing'. Source code of application that can affect or be affected by these database calls is extracted. Second, 'Dataflow Analysis'. Source code from the previous step is analyzed to extract all possible database interactions that an application may make. Third, 'Impact Calculation'. All possible effects are predicted, at this step.

The results show that the tool is implemented by using this approach provides a speedup over analysis of the whole program.

III. Background

A. Database Instance

A database instance [3] is the data in database at a particular moment in time. It is also called a set of records in database. If a record is deleted or the data or value of field in a record is updated, the database instances are changed.

B. Database Schema

A database schema is [3] the description of the database, which describes the structure of the whole database for a community of users. In the conceptual level, the database schema describes entities, data types, relationships, user operations, and constraints.

C. Schema-based Constraints

Schema-based constraints [3], [8], [12] include domain constraints, key constraints, constraints on NULLs, entity integrity constraints, and referential integrity constraints.

1) Domain constraints: Domain constraints specify that within each tuple, the value of each attribute must be an atomic value from the domain. The domain constraints consist of data type (Character, Integer, Float, Datetime, and etc.), format, and range.
2) Key constraints and constraints on NULL Value: The value of a key attribute can be used to uniquely identify each tuple in the relation. Another constraint on attributes specifies whether NULL values are or are not permitted.
3) Entity integrity constraints: Every table must have a primary key (PK) and the column or columns chosen to be the primary key should be unique and not null.
4) Referential integrity constraints: The attributes in foreign key (FK) have the same domain(s) as the primary key attributes in the table referenced table. Each foreign key value in a table exists as either a primary key in the referenced table or NULL.

IV. Approach

This paper presents an approach for impact analysis on database instances. The approach is shown in Fig. 1.

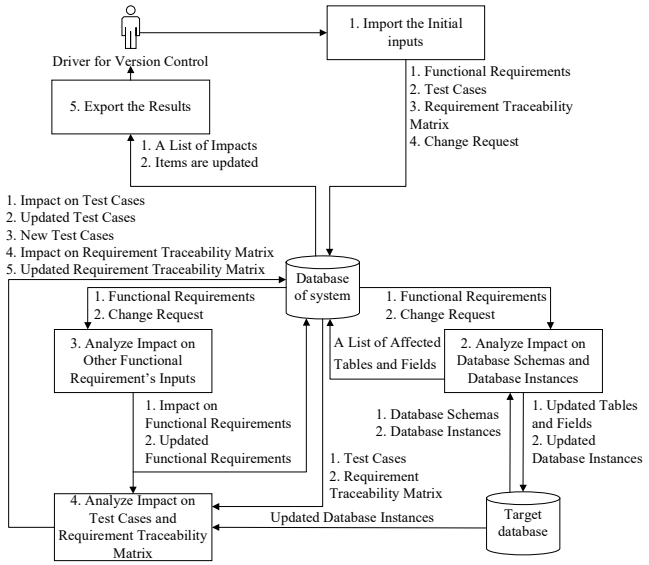

Fig. 1. Overview of Approach

Our approach consists of 5 major steps as follows:

A. Import the Initial inputs

There are 4 initial inputs for analysis in this approach as follows:

1) Functional Requirements. A functional requirement consists of a functional requirement number, a description, a version, and inputs, which include an input name, data type, length, constraints (Default, Null, Unique, Min and Max), and relation to database (Table and Field). An example of a functional requirement is shown in TABLE I.

TABLE I
FUNCTIONAL REQUIREMENT

FR No.			FR01						
Description			Search student information by stdId						
Version			1.0						
List of Inputs								**Relation**	
Input Name	**Data Type**	**Length**	**Constraints**					**Table Name**	**Field Name**
			Default	Null	Unique	Min	Max		
StdId	CHAR	10	-	N	Y	-	-	STUDENT	id

2) Test Cases. A test case consists of a test case number, a description and test data, which include the name and

value. Test data are extracted from inputs of the functional requirement. Values of test data are database instances in the target database. An example of a test case is shown in TABLE II.

TABLE II
TEST CASE

Test Case No.	TC01
Description	For testing function number FR01
Version	1.0
Test Data	
Name	**Value**
stdId	5870111621

3) Requirement Traceability Matrix. A requirement traceability matrix consists of the version, as well as relation between functional requirements and test cases. An example of a requirement traceability matrix is shown in TABLE III.

TABLE III
REQUIREMENT TRACEABILITY MATRIX

Version	1.0			
Functional Requirements No.	**Test Cases No.**			
	TC01	**TC02**	**TC03**	**TC04**
FR01	x			

4) Change Request. A change request consists of the functional requirement number and a list of changed requirement inputs, which include input name, change detail and type of changes. (ADD, DELETE, EDIT). An edit means data type, length, and constraints are changed. In our approach, change can occur only in one functional requirement at a given time. An example of a change request is shown in TABLE IV.

TABLE IV
CHANGE REQUEST

Functional Requirement No.	FR01	
Input Name	**Change Detail**	**Change Type**
stdId	change length from 10 to 8.	EDIT

B. Analyze Impact on Database Schemas and Database Instances

In this step, a functional requirement and change request are used for analyzing the impact on database schemas and database instances of the target database.

First, the impact depending on type of changes in the change request is analyzed as follows:

1) ADD. If a field is added and has not been in the target database, the target database will be updated by inserting a new field and generate a new database instance.
2) DELETE. A field in the target database is deleted. Our approach can't delete a field that is the primary key.
3) EDIT. The database schema of the target database is definintely affected, but in some cases the database instances are not affected. Impact to the database instances according to the change detail is shown in TABLE V.

TABLE V
IMPACT ON DATABASE INSTANCES

Change Detail	**Impact on Database Instances**
Change data type	may be affected
Increase length	not affected
Decrease length	Affected
Change default value	not affected
Change null value from YES to NO	Affected
Change null value from NO to YES	not affected
Change unique value from YES to NO	not affected
Change unique value from NO to YES	Affected
Increase min value	Affected
Decrease min value	not affected
Increase max value	not affected
Decrease max value	Affected

Our approach supports 4 main data types as follows:

1) Integer number (INT)
2) Float number (FLOAT and DECIMAL)
3) Character string (CHAR and VARCHAR)
4) Date and time (DATE and DATETIME)

If the data type is changed from CHAR to VARCHAR or CHAR to VARCHAR, the database instances are not affected.

Second, other affected fields are found using primary key and foreign key.

For example: Functional requirement number FR01 has an input. The input name 'stdId' is related to the field 'id' in the table 'STUDENT' of the database (TABLE I). This input is required to change length from 10 to 8 (TABLE IV). Thus, the field 'id' in the table 'STUDENT' is affected. This field is referred to by the field 'stdId' in the table 'PAYMENT' shown in Fig 2. Therefore, the field 'stdId' in the table 'PAYMENT' is also affected. The result is shown in TABLE VI.

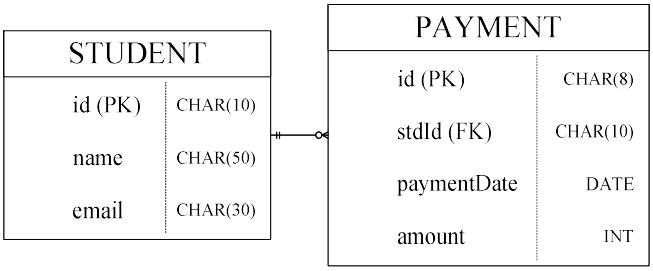

Fig. 2. ER-Diagram

TABLE VI
A LIST OF AFFECTED TABLES AND FIELDS

Table	**Field**	**Impact**	
		Database schema	**Database instance**
Student	id	Decrease length from 10 to 8	Generate new database instance
Payment	stdId		

Third, the affected fields and database instances are updated. Affected fields in the database are updated according to the impact on the database schema. The database instances are updated by randomly generating new data according to the data type, length, and constraints of the updated fields. If the affected field is a foreign key, the data of this field will be generated according to the data in reference field.

ISBN: 978-988-14047-3-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

For example, The field 'id' in the table 'STUDENT' and the field 'stdId' in the table 'PAYMENT' are affected by the decrease in length from 10 to 8 (TABLE VI). Then the field 'id' in the table 'STUDENT' and the field 'stdId' in the table 'PAYMENT' have their lengths updated from 10 to 8. In the target database, the database instance of the table 'STUDENT' is shown in VII. Because the 'id' field is affected, the data of this field are updated from '5870111621' to 'Asdw21', and '5870000521' to '6dscsqs'. These are shown in TABLE VIII. The database instances of the 'PAYMENT' table is shown in TABLE IX. The 'stdId' field is affected, but this field is a foreign key that refers to the 'id' field in the 'STUDENT' table. Therefore, the data of the 'stdId' field are updated by using data according to the 'id' field in the 'STUDENT' table. These are shown in TABLE X.

TABLE VII
DATABASE INSTANCES OF 'STUDENT' TABLE

Id	name	Email
5870111621	Kritsada	k.kanchalew@gmail.com
5870000521	Surasak	Surasak@gmail.com

TABLE VIII
UPDATED DATABASE INSTANCES OF 'STUDENT' TABLE

Id	name	Email
Asdw21	Kritsada	k.kanchalew@gmail.com
6dscsqs	Surasak	Surasak@gmail.com

TABLE IX
DATABASE INSTANCES OF 'PAYMENT' TABLE

Id	stdId	paymentDate	Amount
ATH03	5870111621	26-10-2015	31000
ATH04	5870000521	12-10-2015	31000
ATH05	5870000521	05-12-2015	31000
ATH06	5870111621	12-10-2015	31000

TABLE X
UPDATED DATABASE INSTANCES OF 'PAYMENT' TABLE

Id	stdId	paymentDate	amount
ATH03	**Asdw21**	26-10-2015	31000
ATH04	**6dscsqs**	12-10-2015	31000
ATH05	**6dscsqs**	05-12-2015	31000
ATH06	**Asdw21**	12-10-2015	31000

C. Analyze Impact to Other Functional Requirements' Inputs

This step analyzes the impact according to the input's name of a functional requirement. Our approach assumes that inputs with the same name are related. When a functional requirement's inputs are changed, other inputs with the same name will be affected too. Afterwards, the affected functional requirements' inputs are updated.

For example, A functional requirement's input named 'stdId' is required to change length from 10 to 8 (TABLE IV). This input is only contained in the functional requirement number FR01 (TABLE I). Thus, the input name 'stdId' of functional requirement number FR01 is affected. Afterwards, the functional requirement number FR01 is updated by editing length of the input 'stdId' from 10 to 8. This update is shown in TABLE XI.

TABLE XI
UPDATED FUNCTIONAL REQUIREMENT

FR No.	FR01								
Description	Search student information by stdId								
Version	1.0								
List of Inputs								**Relation**	
Input Name	**Data Type**	**Length**	**Constraints**					**Table Name**	**Field Name**
			Default	Null	Unique	Min	Max		
StdId	CHAR	**8**	-	N	Y	-	-	STUDENT	id

D. Analyze Impact on Test Cases and Requirement Traceability Matrix

First, the impact on test cases is analyzed using the requirement traceability matrix and the impact of functional requirements. If functional requirement inputs are added or deleted, test cases are related to affected function requirement will be deleted and generated new test cases. If inputs of a functional requirements are edited. Test cases are related to the affected function requirement will be updated by changing their test data. To generate or update test cases, test data are generated/updated according to the updated inputs of functional requirement, and values of test data are updated by the updated database instances.

Second, the requirement traceability matrix is analyzed. If a test case is generated. The relation between test case and the function requirement will be added in the requirement traceability matrix. If a test case is deleted, the relation between the test case and the function requirement will be deleted from the requirement traceability matrix.

For example, the functional requirement number FR01 is affected by input name 'stdId' changing length from 10 to 8. In the requirement traceability matrix (TABLE III), functional requirement number FR01 is related to test case number TC01 (TABLE II). Thus, test case number TC01 is affected. This impact is shown in TABLE XII . In the updated functional requirement number FR01 (TABLE XI), the input name 'stdId' is related to the field 'id' in the table 'STUDENT'. Therefore, the value of test data named 'stdId' in test case number TC01 is updated from '5870111621' to 'Asdw21'. This is shown in TABLE XIII.

TABLE XII
IMPACT ON TEST CASE

Test Case No.	Impact Detail
TC01	Edit value of test data name 'stdId'

TABLE XIII
UPDATED TEST CASE

Test Case No.	TC01
Description	For testing function number FR01
Version	1.0
Test Data	
Name	**Value**
stdId	**Asdw21**

E. Export the Results

In this step, results of impact analysis and updated items are exported to a software configuration management system.

ISBN: 978-988-14047-3-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

V. Conclusion and Future Work

This paper presents an approach for impact analysis on database instances. This approach can analyze the impact on database schemas, database instances, functional requirements and test cases, and update them. First, database schemas and database instances have their impact analyzed. Second, other function requirements are analyzed whether the changes have an affect on them or not. Afterwards, test cases are analyzed using the requirement traceability matrix.

In the future, we plan to implement a tool for analyzing the impact on database instances according to this approach.

References

[1] "Systems and software engineering -- Vocabulary," in ISO/IEC/IEEE 24765:2010(E), ed, 2010, pp. 1-418.
[2] K. E. Wiegers, Software Requirements, 1999.
[3] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, Seven Edition: Pearson, 2016.
[4] P. C. Jorgensen, Software Testing A Craftsman's Approach Third Edition: NewYork: Auerbach Publications, 2008.
[5] S. M. Ooi, R. Lim, and C. C. Lim, "An integrated system for end-to-end traceability and requirements test coverage," in IEEE 5th International Conference on Software Engineering and Service Science, 2014, pp. 45-48.
[6] A. Kampeera and T. Suwannasart, "Impact analysis to database schema and test cases from inputs of functional requirements changes," in The International MultiConference of Engineers and Computer Scientists 2016, 2016, pp. 449-453.
[7] C. Sriarpanon and T. Suwannasart, "A source code and test cases impact analysis tool for database schema changes," in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2015, Hong Kong, 2015, pp. 466-469.
[8] P. Tongrak and T. Suwannasart, "A tool for generating test case from relational database constraints testing," in 2nd IEEE International Conference on Computer Science and Information Technology, 2009, pp. 435-439.
[9] J. Jainae and T. Suwannasart, "A Framework for Test Case Impact Analysis of Database Schema Changes Using Use Cases," International Journal of Engineering and Technology, vol. 6, pp. 186-189, 2014.
[10] G. M. Kapfhammer, P. McMinn, and C. J. Wright, “Search-based testing of relational schema integrity constraints across multiple database management systems,” in Proc of 6th ICST, 2013.
[11] A. Maule, W. Emmerich, and D. S. Rosenblum, "Impact analysis of database schema changes," in Software Engineering, 2008. ICSE ’08. ACM/IEEE 30th International Conference on, 2008, pp. 451-460.
[12] S. Sumathi and S. Esakkirajan, Fundamentals of Relational Database Management Systems: Springer, 2007.

ISBN: 978-988-14047-3-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)