Aiding Maintenance of Database Applications Through Extracting Attribute Dependency Graph

Kaiping Liu, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

Hee Beng Kuan Tan, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

Xu Chen, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

ABSTRACT

Database is a major component of many software systems. This paper proposes a novel graph called the attribute dependency graph to reveal the dependencies between attributes in a database application and also the programs involved. The authors propose an approach to automatically extract the attribute dependency graph out of a database application from its source code through inter-procedural static program analysis. The extracted information is a straightforward indicator that benefits the maintenance process, particularly for impact analysis on the modifications in a database application. A tool has been developed to implement the proposed approach for PHP-based database applications. Case studies have also been conducted to demonstrate the use of our approach.

Keywords: Attribute Dependency Graph, Database Application, Extraction, Impact Analysis, Maintenance, Structured Query Language (SQL) Query

INTRODUCTION

Database is a major component of many software systems and database applications are widely applied in various areas. Due to the need for more functionality, integrity and other operating characteristics, database applications are becoming more and more complicated, and this rising complexity calls for more frequent updating in the application.

In practice, the database schema is continuously changing because system requirements keep changing in time. Moreover, since the data that is kept within the database is generally dynamic, the programs must always be modified to maintain the changeable data. An inadequate change on schema may possibly break the original system (Castro, Melnik, & Adya, 2007). However, due to necessities and benefits, changes on schema are widely applied.

DOI: 10.4018/jdm.2013010102
on database applications (Ambler, 2006). Over the modification process, impact analysis of changes must be conducted to identify their potential consequences. However, manual assessing of these effects is a tiring and complicated process (Law & Rothermel, 2003), and can be frequently incorrect. In addition, there are many dependencies between attributes maintained in a database which can introduce many obstacles for maintaining database applications. Considering these issues, it is essential to provide simple to use and straightforward information to aid the maintenance of database applications, especially by performing impact analysis after modification. However, little work (Gardikiotis, Lazarou, & Malevris, 2007) has been done in the field of software maintenance research targeting specifically on database applications. Our work aims to address this issue.

In general, graphs provide an intuitive and clear way to depict the relationships between entities and have been utilized in software engineering for a long time. A variety of graphs, such as control flow graph, function call graph, program dependence graph, have been introduced to aid software maintenance processes, and their usefulness has also been demonstrated. However, to the best of our knowledge, little research has been delved in the use of graphs to assist the maintenance of database applications.

This paper proposes a novel graph called attribute dependency graph to demonstrate the dependencies between attributes in a database application and also the programs involved. We propose an approach to extract dependencies between attributes and then construct attribute dependency graph by analyzing the source code of database applications. The extracted graph can be used to aid maintenance processes particularly in the impact analysis upon modification of a database application.

The paper is organized as follows. The next section introduces the attribute dependency graph. The follow section presents the proposed approach. The section after reports our evaluation through case studies followed by discussion of the related work in the next section. The final section concludes the paper.

ATTIBUTE DEPENDENCY GRAPH

Within a database application, a set of database attributes could be referenced by an SQL query in a program to decide whether other database attributes could be inserted, updated or deleted. Moreover, values of some attributes could also be used in the SQL query to assign other attribute values. When either of these two cases occurs, we note that dependencies exist among attributes. For every change that has been made to a database attribute, we must identify and investigate the change’s impact to other attributes that depend on the changed attribute and respond accordingly with necessary actions.

To support the above-mentioned process, we introduce attribute dependency graph to represent the dependencies between attributes. Our attribute dependency graph displays some similarities to the data dependency graph for program analysis in that both of them illustrate the dependencies between data. However, the data dependency graph only indicates the data dependency for database attributes but not the variables defined in a program.

In this paper, a program is defined as a basic independent executable unit in a database application. In the attribute dependency graph, a program is represented with a rectangular node, while an elliptical node represents a database attribute that is referenced, inserted, updated or deleted by a program. An edge from an attribute to a program indicates the program references to the attribute, while an edge in the opposite direction indicates that the program modifies the attribute. Here, the modification could be an insertion, update or deletion. Whereas incoming edge to a program is labeled with a reference number to identify a reference to an attribute made in the program, each outgoing edge from a program is labeled with both the type of modification and the reference numbers indicating the references identified by the incoming edges to the program. Figure 1 shows the basic notations of attribute dependency graph.

Since PHP is a widely used scripting language for developing web application, a large number of database applications are developed
Related Content

INDUSTRY AND PRACTICE: Solving the Partitioning Problem in Database Design
[www.irma-international.org/article/industry-practice-solving-partitioning-problem/51211/](www.irma-international.org/article/industry-practice-solving-partitioning-problem/51211/)

Checking Integrity Constraints in a Distributed Database
[www.irma-international.org/chapter/checking-integrity-constraints-distributed-database/11124/](www.irma-international.org/chapter/checking-integrity-constraints-distributed-database/11124/)

A Taxonomy for Object-Relational Queries
[www.irma-international.org/chapter/taxonomy-object-relational-queries/9212/](www.irma-international.org/chapter/taxonomy-object-relational-queries/9212/)

Enhancing the ER Model with Integrity Methods
[www.irma-international.org/article/enhancing-model-integrity-methods/51223/](www.irma-international.org/article/enhancing-model-integrity-methods/51223/)

A Measurement Ontology Generalizable for Emerging Domain Applications on the Semantic Web
[www.irma-international.org/chapter/measurement-ontology-generalizable-emerging-domain/4290/](www.irma-international.org/chapter/measurement-ontology-generalizable-emerging-domain/4290/)