Implementing Business Processes: A Database Trigger Approach

Wai Yin Mok, Department of Economics and Information Systems, University of Alabama in Huntsville, Huntsville, AL, USA

Charles F. Hickman, Department of Accounting and Finance, University of Alabama in Huntsville, Huntsville, AL, USA

Christopher D. Allport, Department of Accounting and Finance, University of Alabama in Huntsville, Huntsville, AL, USA

ABSTRACT

Database triggers are database procedures that are executed automatically when certain events occur and conditions are met. This paper presents a design methodology that helps users implement business processes using database triggers. The contributions of this paper are as follows. First, the proposed methodology uses the Unified Modeling Language (UML). UML is a standard modeling language for the software industry and many commercial CASE (Computer-Aided Software Engineering) tools support UML. Second, many expensive ERP (Enterprise Resource Planning) software systems are employed to implement business processes. The methodology proposed by this paper produces triggers that can be executed on MySQL, an open-source database system that is free for download. Third, as an example of the usefulness of the proposed methodology, the authors present a case study making use of database triggers in a tax audit process. This process involves many steps that require human intervention, and thus is typical of business processes.

Keywords: Business Processes, Computer-Aided Software Engineering (CASE), Database Triggers, Enterprise Resource Planning (ERP) Software, MySQL, Unified Modeling Language (UML)

INTRODUCTION

Due to the technological advances in the last twenty to thirty years, large corporations have come to realize that Information Technology (IT) is becoming a primary driving force in doing business (Chang, 2005). Automated business processes are guiding various stakeholders in making decisions and are able to instantly notify stakeholders of progress. There are many ERP software systems on the market today. Examples are Systems Analysis and Program Development (SAP), Oracle and PeopleSoft. Despite the benefits provided by ERP software, the business models supported by this software may not coincide with the culture of the adopting institutions. Many horror stories attest that either the culture of the institution has to change

DOI: 10.4018/ijkbo.2013040103
dramatically, or the software has to be modified significantly (Madara, 2007). Either way the result is undesirable.

As an alternative to expensive ERP software, this paper presents a design methodology that, if followed, will reduce the cost of developing a workflow system for a business process. The proposed methodology takes advantages of free open-source software. Since most ERP software cost tens of thousands of dollars, the proposed methodology provides substantial savings.

At the center of the proposed methodology are active database systems, which serve as platforms on which workflow systems are built. To understand their importance in our approach, note that a typical business process encounters a lot of human-generated events. These events are external to the resulting workflow system. Many of these external events occur simultaneously and any single one of them might lead to numerous modifications to the backend database. Therefore, if the platform on which the workflow system is built is able to respond to external events and act accordingly, most of the programming responsibilities can be delegated to the platform itself. As a result, a lot of programming effort can be saved. As they are designed to react to events and act accordingly, active database systems are therefore well-suited to be such supporting platforms.

To gain an understanding of active database systems, note that database systems in the 70s and 80s were passive, meaning that they did not act until instructed. Today’s database systems are different. Most of them support triggers. Consequently, they are able to react to external events; and as a result they have become active. Triggers, a relatively new addition to database systems, are named database procedures that are automatically executed when certain events occur and conditions are met. As such, triggers are able to monitor any changes to the business environment and modify the backend database accordingly to satisfy established business rules and logic. Example systems that support triggers include Oracle, IBM DB2, and MySQL. For this paper, MySQL is our choice because (1) it is free, (2) it supports triggers, and other procedures and functions, and (3) it is commonly used as a backend database system in many web applications.

Our methodology has several stages. Given a business process $P$, we first construct a model for $P$ by using an activity diagram in UML. UML is a standard modeling language in the software industry, and therefore has a well-understood syntax. Therefore, many software CASE tools on the market support UML. Hence, building our methodology upon UML will make it easier to integrate into existing CASE tools. After the model is complete, we then map it to a MySQL database, which contains a collection of tables and triggers. The triggers of the database will then implement the business logic of $P$. Lastly, we build a web application that lets the agents enter information into the database. The information will be further processed by the triggers and the MySQL database will provide feedback to the agents for their use in carrying out the activities in $P$.

This paper is organized as follows. The second section introduces activity diagrams and triggers. In the third section, we discuss how we map activity diagrams to MySQL tables and triggers. We study the property of termination for a collection of triggers in the fourth section. The case study for this paper is presented in the fifth section. We make concluding remarks in the sixth section.

**FUNDAMENTALS**

**Activity Diagrams**

The main tool in the UML for modeling business processes is activity diagrams (Booch, Jacobson, & Rumbaugh, 2005). Activity diagrams are similar to traditional flowcharts, but with additional modeling constructs. Like flowcharts, activity diagrams are able to show branches of control. However, activity diagrams are also able to show concurrency, which is not allowed in traditional flowcharts. Since concurrency is an important concept of business processes, activity diagrams are superior in this regard.

Activity diagrams have many modeling constructs. However, not all of them are rel-
Comparing Cultural and Political Perspectives of Data, Information, and Knowledge Sharing in Organisations
www.igi-global.com/article/comparing-cultural-political-perspectives-data/2679?camid=4v1a