Article can be accessed online at http://www.publishingindia.com

An Overview of Active Database: Model, Architecture and Challenges

Gambhir Halse*, Malakappa Shirdhonkar**

Abstract

Research in database management system results in new technology and opportunities to researchers. As a result of this researcher have developed many solutions to real time applications and advancement in hardware's. Database systems with production rules are referred to as active database systems and the field of active database systems has indeed been active. In this article we emphasize on evolution of active database system, its architecture and challenges faced by research community.

Keywords: Active Rules, Triggers, Rule Execution

Introduction

Early generation database system stored database passively, and perform only actions explicitly specified by a user transaction [1]. In contrast active database system not only stores data, but also carries out actions in response to events, such as changes in data. Incorporation of active rules in database is a need of database application development. Active rule specify when, and what action to carry out [2][3]. An active database system is a database system which detects situations of interest, evaluates the condition when they occur; and if the condition is true, then executes an action in timely manner [4]. In contrast, a conventional passive database system only executes queries and transaction explicitly submitted by the user or an application program.

The paper is organized as follows: section 2 states the basic model of active database systems, and describes the structure of active rules. Section 3 presents the architecture of active database systems. Section 4 focuses on issues and challenges of active database; finally, section 5 concludes the paper.

Basic Event-Condition-Action Model

Active database is conventional database in which active rules are incorporated[1]. The basic concept on which an ADBMS (Active Data Base Management System) relies is the concept of ECA, or active rules (ECA stands for Event-Condition-Action) needs to be considered. In active database when any event (EVENT) occurs and some conditions (CONDITION) satisfied then some action (ACTION) initiated automatically. These actions are performed without any need for the user's intervention [2] [5]. At the conceptual level people often talk about ECA rules; these rules are mostly implemented using triggers in some concrete ADBMS The event-condition-action model for active rules is widely used. The general form of rule in this model is as follows:

on event,

if condition,

then action.

Changes to the database such as insertion, deletion, and updates to tuples are modeled as events. In an object oriented database, an event could be an action such as creating or deleting of an object, or execution of a method on the object. When an event occurs, one or more rules may be triggered. The event is called the triggering event of the rule. Once a rule is triggered, the conditions of

* Associate Professor, Department of Computer Science and Engineering, KLE Dr. M. S. Sheshgiri College of Engineering and Technology, Belgaum, Karnataka, India. E-mail: g_halse@rediffmail.com

** Professor, Department of Computer Science and Engineering, BLDE's College of Engineering and Technology, Bijapur, Karnataka, India. E-mail: ms_shirdhonkar@rediffmail.com

the rules are checked. If the conditions are satisfied, the actions of the rules are executed.

Active rules can be used for diverse purpose [6]. An example application is alerting, where the rules monitors the system and notify the administrator or user, if an unusual event has occurred. Active rules can also be used for checking integrity constraints. Another example of the use of active rule is in maintenance of derived data, such as indices and materialized views. If a view has been materialized (i.e. captured and stored) it needs to be updated in response to changes to the database relations on which it is defined. The actions needed for keeping the view up-to-date can be encoded as active rules.

The syntax for active rule not yet standardized and may differ across system. The trigger facility is an example of active rule facility [7]][8]. Rules are typically stored in the databases, just like regular data, so that they are persistent and accessible to all database operation.
Database system in which triggers not supported, polling of database is carried out. Polling is a process periodically queries (polls) the database to see whether any event of interest has been occurred and then carries out necessary condition checks and actions.

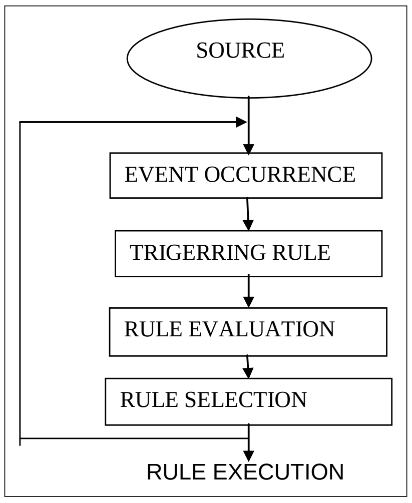
A trigger can be thought of as a ‘daemon’ that monitor a database, and is executed when the database is modified in a way that matches the event specification [8]. Why triggers can be hard to understand? Triggers offer a powerful mechanism for dealing with changes to a database, but they must be used with caution. The effect of a collection of triggers can be very complex and maintaining an active database can become very difficult. Often a judicious use of integrity constraint can replace the use of triggers.

In an active database system, when the DBMS is about to execute a statement that modifies the database, it checks whether some trigger is activated by the statement. If trigger is activated then it evaluates its condition part, if satisfied then corresponding action is performed[10].

Architecture of Active Database

The architecture of rule execution model shown in Fig.1 specifies how the set of rules are treated at run-time. While the execution model of rule system closely related to the underlying DBMS [10].

Fig. 1: Basic Architecture of Active Database Rule Execution



There are number of phases in rule execution:

1. The signaling phase refers to event occurrence caused by event source.
2. The triggering phase takes the event occurred and triggers the corresponding rule from rule base.
3. The evaluation phase evaluates the condition of triggered rule. Rule conflict set is formed from all rules associated with event occurred.
4. The scheduling phase indicates how the conflict set processed.
5. The execution carried with action specified in selected rule.

The phases are not necessarily executed contiguously but depends on the Event-Condition (EC) and Condition-Action (CA) coupling modes. EC coupling mode indicates when the condition is evaluated relative to the event that triggers the rule. CA coupling indicates when the action is to be executed relative to the evaluation of the condition.

The options for coupling modes supported are:

a) Immediate in which case condition is evaluated immediately after the event.
b) Deferred in which the condition is evaluated in the same transaction but not necessarily at the earliest.
c) Detached in which the condition is evaluated within a different transaction from the event.

Challenges in Active database

In this section we analyze the realities of active database system in solving the real life problems [11]. We decompose the problems found with active DBMSs into following categories.

- Challenges concerned with the design of active applications.
- Challenges concerned with security, reliability, and unpredictability.
- Challenges in addressing the performance problems.

Challenges in Active Applications

A first challenge is the lack of standards for trigger languages in the existing database. First, condition-action rules are usually not directly expressible. This problem is emphasized by restrictions of the trigger language, such as the event part of a rule must be associated with a single relation, or a disjunction of elementary events (even for the same relation) is not allowed. Coding a rule, such as "if an employee earns more than his manager then notify", may entail the definition of many triggers because one trigger is needed for every data modification event capable of violating the database constraint. The proliferation of rules renders more difficult the verification of their correctness. Most development guides recommend not to use triggers for coding integrity constraints that can be expressed by means of assertions in the data definition language.. Some degree of automatic generation is already available in several commercial database design tools. In many applications, we found business rules that could not be implemented in the trigger language because of the restrictions imposed to the event part

Challenges in Security Applications

Many researcher and database developers are often reluctant to use active DBMS facilities because they consider triggers as insecure, unreliable and unpredictable [9]. In this respect, their reaction is the same as with. Production rules in expert systems or knowledge base systems because they wonder how a set of individual, isolated rules will interact with each other and with other relevant application programs in concrete situations. With active rules, this issue is more challenging because these rules "act on their own" and may directly affect the real world [12]. For mission-critical financial applications where triggers may automatically execute stock deals, influence the structure of large portfolios or rate customers as non-credit- worthy, this attitude is well founded. The same is true for other applications like plant control, patient care or aviation systems. Without guarantee of correctness and predictable, unambiguous behavior, triggers will not be used in these fields.

Challenges in Database Performance

One of the main reasons that make researchers and developer reluctant to use triggers in the development of large applications is their anxiety about performance [13]. This feeling is consolidated by recent experiences conducted with the development of applications, that involve several hundreds of triggers on various DBMS platforms. When developers compare the performance of the same application coded with and without triggers, they observe that the trigger-based version runs slower than the without trigger system. As a consequence, many researchers suggested not to use triggers intensively although they are convinced by the functionality. This disquiet deserves some analysis. A natural question is to wonder if the immaturity of the implementations of triggers suffices to explain such a gap of performance. In fact, the overhead taken by the binding between events and rules, and the retrieval of rules remains quite smalls. Another issue is the lack of experience of developers in the programming of triggers [14].

Conclusion

Conventional database system stores the information passively. Now a day incorporation of active rules in active database is required to meet the challenges in the real world scenario. Active database system supports mechanism that enables them to respond automatically to the events that are taking place either within or outside the database itself. This paper discusses model, architecture and challenges in the field of active database.

REFERENCES

Ramakrishnan, J., & Gehrke, R. (2000). *Database management system* (2nded) McGraw Hill.

CODASYL Data Description Language Committee. (1973). CODASYL Data Description Language: Journal of Development.

Dayal, U. (1988). *HiPAC: A Research Project in Active, Time-Constrained Database Management, Interim Report*. Technical Report XAIT-88-02, Xerox Advanced Information Technology.

Bouazir, T., & Wolski, A. (1997). *Applying Fuzzy events to approximate reasoning in active databases*. In Proceedings of 6th IEEE International Conference on Fuzzy System.

Stonebraker, M. (1986). *A rule manager for relational database systems*. The POSTGRES Papers, University of California, Berkley.

Dayal, U., Buchmann, A., & McCarthy, D. (1988). Rules are objects too: A knowledge model for an active, object-oriented database system. *Advances in Object- Oriented Database Systems*, 334, (129-143).

Eswaran, K. P., & Chamgerlain, D. D. (1975). *Functional specifications of a subsystem for data base integrity*. Proceedings 1st International Conference on Very Large Data Bases.

Eswaran, K. P. (1976). Specifications, Implementations, and Interactions of a Trigger Subsystem in an Integrated Data Base System. IBM Research Report.

Bouaziz, T., & Wolski, A. (1996). Incorporating fuzzy inference into database triggers. Technical Report, VTT information Technology.

Hsu, M., Ladin, R., & McCarthy, D. (1988). *An execution model for active database management systems*. Proceedings 3rd International Conference on Data and Knowledge Bases.

Kumar, M. (2012). Issues and challenges in database research. *Global Journal of Computer Science and Technology, Software and Data Engineering*, 12(11), 16-20.

Viana, S. (2007). A rule repository for active database systems. *CLEI Electronic Journal*, December, 10(2), 1-10.

Manola, F., & Dayal, U. (1986). *PDM: An object-oriented data model*. Proceedings of International Workshop on Object-Oriented Database Systems.

Saygin, Y., & Ulusoy, O. (2001). *Automated construction fuzzy event sets and its applications to active databases*. IEEE Transaction on Fuzzy systems, 9(3), 450-460.