International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463
Vol. 3 Issue 8, August-2014, pp: (248-253), Impact Factor: 1.252, Available online at: www.erpublications.com

Performance of monitoring and control application in Active Database Management Systems

Sumeer Kumar, Sumit Kumar

Abstract: In this manuscript, the authors have tried to sort out the performance analysis of monitoring and control application in Active Database Management Systems. Active Database Management Systems (ADBMSs) have been developed to support applications with detecting changes in databases. This includes support for specifying active rules that monitor changes to data and rules that perform some control tasks for the applications. Active rules can also be used for specifying constraints that must be met to maintain the integrity of the data, for maintaining long-running transactions, and for authorization control. This begins with presenting case studies on using ADBMSs for monitoring and control. The areas of Computer Integrated Manufacturing (CIM) and Telecommunication Nets have been studied as possible applications that can use active database technology. These case studies have served as requirements on the functionality that has later been developed in an ADBMS. After an introduction to the area of active database systems it is exemplified how active rules can be used by the applications studied. Several requirements are identified such as the need for efficient execution of rules with complex conditions and support for accessing and monitoring external data in a transparent manner.

1. Introduction

A Database Management System (DBMS) [39] is a general information management system that can manage many different kinds of data, stored in the database. By DBMS we here mean more than just a system that manages variables or files of data. The data can be both application data of different types and meta-data used by the DBMS to define the database layout, the database schema. In relational DBMSs (RDBMSs) data is defined with relations between data which are stored in tables and can be accessed through logical queries, or relational views. The DBMS provides support for logical views of data that are separate from the physical views, i.e. how the data is actually stored in the database. This separation is accomplished by allowing applications to define, access, and update data through a Data Definition Language (DDL) and Data Manipulation Language (DML) combined into a declarative query language such as the relational query language SQL [6]. The DBMS provides persistency of data by ensuring that no data is lost in the case of system failures. The persistency can be achieved in many ways, e.g. by storing data and a log of uncommitted changes to data on disk.

Another important aspect of functionality supported by a DBMS is transaction management. Transactions provide a mechanism for organizing and synchronizing database operations. Different users and applications can use transactions for defining sequences of database operations without, more or less, having to consider possible interaction with other users and applications. If, for some reason, something goes wrong during a transaction, the user or the application can choose to abort the transaction and all the database operations are undone. If a transaction is finished successfully, the DBMS can commit the transaction and make all the changes in the database permanent. In recent years there has been development of DBMSs with more data modeling support. This is often needed in technical and scientific applications where the schemas can be highly complex. In Object Oriented Database Management Systems (OODBMSs) OO programming languages have been extended to support database management through persistent object classes. A standard query language, OQL, has been defined for declarative access to the data. The OODBMS model does not, however, provide a fully declarative query language for both definitions, accessing and updating data (object definitions with attributes and methods are still defined in the OO programming language). OO programming languages such as C++ allow low-level operations on objects which makes the separation between a logical and physical view of data in OODBMSs difficult to support.

1.1. Active DBMSs (ADBMSs) versus Passive DBMSs

Traditional DBMSs are passive in the sense that they are explicitly and synchronously invoked by user or application program initiated operations. Applications send requests for operations to be performed by the DBMS and wait for the DBMS to confirm and return any possible answers. The operations can be definitions and updates of the schema, as well as queries and updates of the data. Active Database Management Systems (ADBMSs) are event driven systems where operations such as schema changes and changes to data generate events that can be monitored by active rules. An ADBMS

can be invoked, not only by synchronous events that have been generated by users or application programs, but also by external asynchronous events such as changes of sensor values or time. When monitoring events in a passive database, a polling technique or operation filtering can be used to determine changes to data. With the polling method the application program periodically polls the database by placing a query about the monitored data.

The problem with this approach is that the polling has to be fine-tuned so as not to flood the DBMS with too frequent queries that mostly return the same answers, or in the case of too infrequent polling, the application might miss important changes of data. Operation filtering is based on the fact that all change operations sent to the DBMS are filtered by an application layer that performs the situation monitoring before sending the operations to the DBMS. The problem with this approach is that it greatly limits the way rule condition evaluation can be optimized. It is desirable to be able to specify the conditions to monitor in the query language of the DBMS. By checking the conditions outside the database the complete queries representing the conditions will have to be sent to the DBMS. Many DBMSs allow precompiled stored procedures that can update the database. The effects of calling such a procedure cannot be determined outside of the database. If the condition monitoring is used to determine inconsistencies in the database, it is questionable whether this should be performed by the applications, instead of by the DBMS itself. In an integrated ADBMS condition monitoring is integrated into the database. This makes it possible to efficiently monitor conditions and to notify applications when an event occurred that caused a rule condition to become true and that is of interest to the application. Monitoring of specific conditions, represented as database queries, can be performed more efficiently since the ADBMS has more control of how to evaluate the condition efficiently based on knowledge of what has changed in the database since the condition was last checked. It also lets the ADBMS perform consistency maintenance as an integrated part of the data management.

1.2 Rule-Based Systems and Active Database Systems

In rule-based systems the rules can be used for different purposes. In fig. 1, the distinction is made between using rules for monitoring, control, and reasoning. We here make a distinction between active database systems and other rule based systems such as reactive systems (sometimes called real-time expert systems) and knowledge-based systems (often just referred to as expert systems). Active database systems are primarily database management systems with the main task of storing large amounts of data and providing efficient access to this data through a query language. In active database systems the rules are primarily used for monitoring changes to the data stored in the database. In reactive systems the rules are used for reacting to changes of some external environment and performing actions on (controlling) the environment in response to the changes. In knowledge-based systems the rules are usually used for reasoning using stored facts and by deducing new facts by using the rules. As can be seen in fig. 1, there is no sharp distinction between the three different kinds of rule systems. An active database system can do limited reasoning by using rules with more complicated rule conditions and which store new data in the database as new facts that signify that the rules have triggered. Control of the environment represented by the data in the database itself can also be performed, e.g. with constraint rules that modify the database to remove any inconsistencies. By allowing the active database manager to access an external environment that can be both accessed and updated, the rules in the active database can be used for control of an external environment as well.

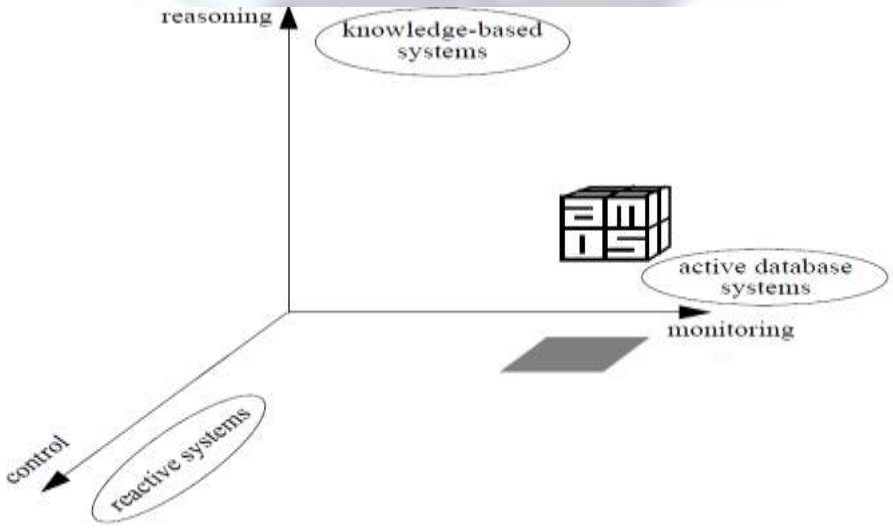

Figure 1: The relation between active database systems and other rule based systems

2. Background

2.1 DBMSs in Telecommunication Nets

In the area of telecommunications there are many different needs for DBMS support. Telecommunication nets already have DBMSs integrated with them and will have even more so in the future. Telecommunication nets are large heterogeneous systems with many, sometimes conflicting, needs that the DBMSs must fulfil. When discussing DBMSs in telecommunication nets it is important to separate between net traffic control, net management, and net applications.

The net traffic control involves the actual operation of the net in terms of setting up communication paths, maintaining them, and disconnecting them. In this work POTS (i.e. point-to-point communication) and standard subscriber services (e.g. wake-up call, call diversion, call waiting, malicious call tracing) are considered to belong to net traffic control. Net traffic control have requirements on DBMSs to provide high throughput and a large number of parallel transactions, main-memory storage, fast-path interfaces to programming languages, and real-time support. In net traffic control, availability and reliability is important, but losing a single connection is not a catastrophe.

Net applications that use a DBMS can include applications other than traditional telephone calls such as Internet access (e-mail, news, WWW, file transfer, and remote system access), text and voice mail, multi-media, and video-on-demand. Net applications will require support from a high-performance DBMS that can handle a large number of simultaneous transactions. To meet these requirements main-memory DBMSs can be considered. Disk based DBMSs that can store large amounts of data will also be needed for logging purposes and for applications needing to store large volumes of data. Support for new data structures will be needed for storing, for instance, voice data, graphical data, and video data in the databases. To support queries over these new data structures the DBMSs must support efficient indexing techniques and optimization of queries that access them.

2.2 DBMSs in CIM

CIM systems handle many different kinds of information for controlling the manufacturing process such as product data (e.g. what parts a product consists of), parts data (e.g. physical data such as size, weight, and number of parts available in stock), data related to the manufacturing equipment (e.g. configuration data), sensor and actuator data. Other data that can be handled by a CIM system, but which is not directly used in the process control can be product specification data (e.g. CAD-drawings), economic data (e.g. product and part costs), and sales data (e.g. how many products have been sold and thus have to be manufactured). The types of CIM applications that can be considered as candidates for the use of (active) database technology are applications where a fairly large amount of data access is needed during the automated process. This could be data such as information about the components involved in an assembly, data about the machines involved and sensor data stored or data directly accessible in the database. The data could also be information about the number of components in stock. This often involves applications where the level of autonomy has to be high and thus allowing the CIM system to operate without too much human intervention. This could be in a system that is more fault tolerant, e.g. by using sensors to detect abnormal situations and to deal with them without an operator having to restart the system, and that can also interact with other systems, e.g. to automatically order more components when the stock is running low.

2.3 Application Studies

This work presents application studies done as background research to find what requirements there are on an ADBMS for supporting various technical applications. The goal is to provide the ADBMS with general functionality that is suitable for the various needs of different applications. Some functionality might not be as important for one application as for another, but all functionality should be as general as possible instead of implementing very specialized functionality tailored for just one specific application. Two application areas were studied and are presented in this work:

• Computer Integrated Manufacturing (CIM)

• Telecommunication Nets

2.4 Computer Integrated Manufacturing (CIM)

Computer Integrated Manufacturing (CIM) is a broad term that covers all aspects of automated manufacturing from using welding robots in car manufacturing to using specialized equipment for making integrated circuits or controlling a steel- or

paper-mill. There are usually many computer systems used in manufacturing plants and the number of systems and level of automatization is constantly increasing. Most systems that are considered are directly involved in controlling the manufacturing process, usually called process control systems. These systems control a manufacturing process using actuators (e.g. conveyor belts, feeders, robots, lathes, or boilers) and monitor the progress using sensors (e.g. speedometers, position sensors, force sensors, image processing systems, thermometers, or pressure gauges). Some other systems that are also sometimes covered by CIM include various systems that are being used within manufacturing companies. These can be systems involved in the design process such as product specification and Computer Aided Design (CAD) systems, systems that handle parts and products in stock, and economic information systems such as product costs and sales information. A current trend in many manufacturing companies is to integrate all of these systems to provide better control of the whole manufacturing process, not just the process control.

2.5 Telecommunication Nets

Telecommunication nets consist of the infrastructure and the equipment needed to provide different telephony services. Traditional telecommunication nets provide transfer of low bandwidth analog data such as voice data in a point-to-point manner. The services provided by the telecommunication net can be divided into services provided to the end user and services provided to the net operator. Traditional user services include Plain Ordinary Telephony Service (POTS), i.e. basic point-to-point voice-based communication without operator assistance, different subscriber services such as call transfer, call waiting, and number presentation of who is calling. Traditional operator services include monitoring net usage and billing subscribers, adding/removing subscribers, load balancing the net (e.g. transferring traffic from heavily used sections to less used sections, sometimes by splitting one high bandwidth connection into several connections), reconfiguring the net without disrupting net traffic, and net supervision functions (e.g. monitoring net overload and equipment failure). In ISDN (Integrated Services Digital Nets) subscribers can be given a fixed medium bandwidth transfer. The basic idea in ISDN is that a digital bit pipe through an Integrated ISDN Transport Net is set up between users (fig. 2.). The bits can originate from any digital ISDN device such as a digital telephone, a digital fax, or a terminal (or any general computer). The connections connections to the end users are defined to use existing twisted pair connections using special ISDN interface hardware.

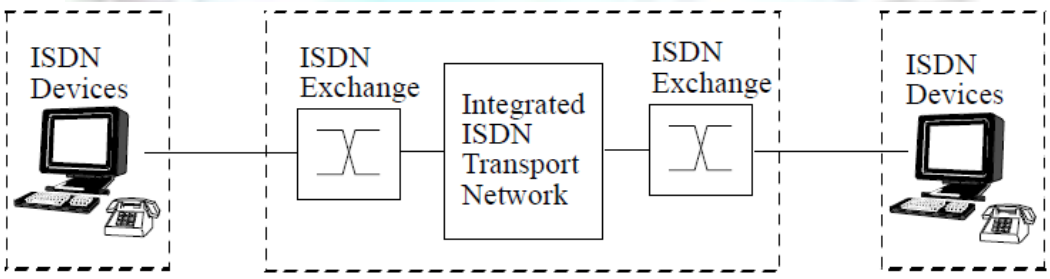

Figure 2: The basic ISDN net

3. Active Database Management Systems

In System R, a trigger mechanism was defined that could execute a prespecified sequence of SQL statements whenever some triggering event occurred. The triggering events that could be specified included retrieval, insertion, deletion, and update of a particular base table or view. Triggers had immediate semantics, i.e. they were executed immediately when the event was detected. In System R it was also possible to make assertions that specified permissible states or transitions in the database through integrity constraints that always had to be true after each transaction. Specific events had to be specified for when assertions were to be checked in the same way as with triggers. Assertions usually had deferred checking semantics, i.e. they were checked when transactions were to be committed. If an assertion failed, then the transaction was aborted. The term active databases was coined in as meaning “a paradigm that combines aspects of both database and artificial intelligence technologies”.

Active rules are defined as Event-Condition-Action (ECA) rules, where the Event specifies when a rule should be triggered, the Condition is a query that is evaluated when the Event occurs, and the Action is executed when the Event occurs and the Condition is satisfied. Events can be seen as signals that inform that a change to data in the database has occurred, e.g. an update of a table. In HiPAC coupling modes (fig. 3.) were defined which specify how the evaluation of rule conditions and the execution of rule actions are related to the detected events and the transaction in which the events occur.

3.1 The Iris Data Model and OSQL

The data model of AMOS and AMOSQL is based on the data model of Iris and OSQL. The Iris data model is based on objects, types, and functions (fig. 3.). Everything in the data model is an object, including types and functions. All objects are classified by belonging to one or several types, which equal object classes. Types themselves are of the type 'type' and functions are of the type 'function'.

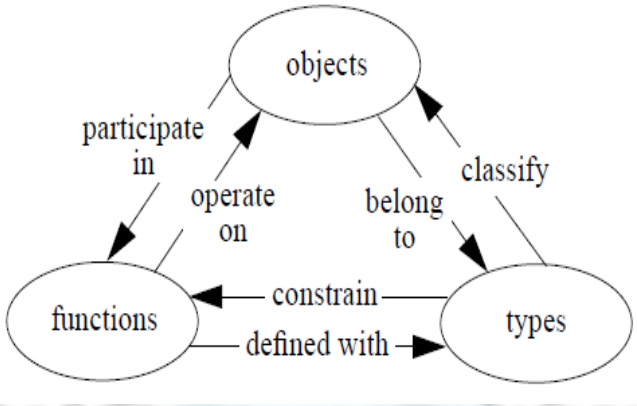

Figure 3: The Iris data model

The data model in Iris is accessed and manipulated through OSQL1. (All examples of actual schema definitions and database queries will here be written in a courier font.) For example, it is possible to define user types and subtypes: In the previous example the last query returns a single tuple. Queries, and subsequently functions, can return several tuples. Duplicate tuples are removed from stored functions if they are not explicitly defined to return a bag. We say that we have set-oriented semantics. Bag-oriented semantics is available as an option and can be specified along with the return type of a function as defined for the studies function.

4. Partial Differencing of Rule Conditions

This work presents a technique for efficient evaluation of rule conditions. The technique is based on incremental evaluation techniques and is named partial differencing. It is used for efficient monitoring of active rules in AMOS. The technique is especially designed for deferred rules, i.e. rules where the rule execution is deferred until a check phase that usually occurs when transactions are committed. The technique can also be used for immediate rule processing. Partial differencing is presented here based on a conference paper. (Paper VI) along with some continued. A difference calculus is defined for computations of the changes to the result of database queries and views. Queries and relational views are regarded as functions over sets of tuples and the calculus for monitoring changes is regarded as an extension of set algebra. Let P be a function dependent on the functions Q and R, denoted the influents of the affected function P. The problem of finite differencing [80][96] is how to calculate the changes to P, ΔP, in terms of the changes to its influents. With partial differencing, changes to P are defined as the combination of the changes to P originating in the changes to each of its influents. Thus, ΔP is defined in terms of the partial differential functions ΔP/ΔQ and ΔP/ΔR. We will define how to automatically derive the partial differentials ΔP/ΔQ and ΔP/ΔR, and how to calculate ΔP from them. The calculus is mapped to relational algebra by defining partial differentials for the basic relational operators.

Conclusions

In this work, overview from the application areas of Computer Integrated Manufacturing (CIM) and Telecommunication Net Management (TNM) have been presented. These applications served as motivation for the functions of an Active Database Management System (ADBMS) that were presented in the work. The major contributions within the field of active database systems are:

- Identifying the need of ADBMSs through the case studies of CIM and TNM. In the application studies the requirements for efficient execution of rules with complex conditions and the need for transparent access of external data were identified.

- Using active rules for monitoring and control in CIM and TNM.
- Identifying the need for mediators in CIM and TNM.
- Defining an ADBMS architecture.
- Identifying the need for generalizing the architecture towards active mediators.
- Adding active rules to an Object-Relational DBMS.
- Integrating (E)CA-rules into a query language.
- Rule modularization by grouping rules into rule contexts.
- Efficient rule evaluation techniques based on partial differencing.
- Defining external data in a transparent manner through the concept of foreign data sources.
- Defining external events through the concept of foreign events of foreign functions.
- On introducing time series for storing event histories.
- On new indexing techniques for inverse queries over time series.

Most of the ideas presented here, such as CA- ECA-, EA-rules, partial differencing for efficient rule condition monitoring, time series for storing time stamped events in event logs, and rule contexts have been implemented in the AMOS ADBMS.

REFERENCES

[1]. Cattell R. G. G: The Object Database Standard: ODMG-93b Release 1,2, Morgan Kaufmann Publishers Inc., 1994.
[2]. Ceri S., Gottlib G., and Tanca L.: What You Always Wanted to Know About Datalog (And Never Dared to Ask), IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 1, March 1989.
[3]. Ceri S. and Widom J.: Deriving Production Rules for Incremental View Maintenance, In Proceedings of the 17th VLDB Conference, Brisbane, Queensland, Australia, Aug. 1990, Pages 577-589.
[4]. Chakravarthy S., et. al.: HiPAC: A Research Project in Active Time-Constrained Database Management, Xerox Advanced Information Technology, Technical Report XAIT- 89-02, Cambridge, MA, Aug. 1989.
[5]. Chakravarthy S. and Mishra D.: An Event Specification Language (Snoop) for Active Databases and its Detection, UF-CIS Technical Report, TR-91-23, Sept. 1991.
[6]. Chakravarthy S., Krishnaprasad V., Anwar E., and Kim S-K.: Composite Events for Active Databases: Semantics, Contexts and Detection, in Proceedings of the 20th VLDB Conference, Santiago, Chile, 1994, Pages 606-617.
[7]. Chandra R. and Segev A.: Active Databases for Financial Applications, RIDE ´94, Houston, Febr., 1994, Pages 46-52.
[8]. Chawathe S. S., Garcia-Molina H., and Widom J.: A Toolkit for Constraint Management in Heterogeneous Information Systems, Proceedings of the Twelfth International Conference on Data Engineering, New Orleans, 1996, Pages 56-65.
[9]. Chimenti D., Gamboa R., and Krishnamurthy R.: Towards an Open Architecture for LDL, Proceedings of the 15th VLDB Conference, 1989, Pages 195-205.
[10]. Dayal U., Blaustein B., Buchmann A., Chakravarthy, Hsu M., Ledin R., McCarthy D., Rosenthal A., and Sarin S.: The HiPAC Project: Combining Active Databases and Timing Constraints, SIGMOD Record, Vol. 17, No. 1, March 1988.
[11]. Dayal U., Buchman A.P., and McCarthy D.R.: Rules are Objects too: A Knowledge Model for an Active, Object-Oriented Database System, Proceedings of the 2nd International shop on Object-Oriented Database Systems, Lecture Notes in Computer Science 334, Springer 1988.
[12]. Dayal U. and McCarthy D., The Architecture of an Active Database Management System, in Proceedings of the ACM SIGMOD Conference, 1989, Pages 215-224.
[13]. Dayal U., Hsu M., and Ladin R.: Organizing Long-Running Activities with Triggers and Transactions, Proceedings of the ACM SIGMOD International Conference on Management of Data, Atlantic City, May 1990.
[14]. Dewan H. M., Ohsie D., Stolfo S. J., Wolfson O., and Da Silva S.: Incremental Database Rule Processing in PARADISER, Journal of Intelligent Information Systems, 1:2, 1992.
[15]. Dewitt D.J., Katz R.H., Olken F., Shapiro L.D., Stonebraker M.R., and Wood D.: Implementation Techniques for Main Memory Database Systems, SIGMOD Record, Vol. 14, No. 2, 1984, Pages 1-8.