Some Key Techniques in Active Information System

Ping ZONG, Jun QIN, Qiong YU

Nanjing University of Posts and Telecommunications,
Nanjing 210003, CHINA

Summary
Active Information System (AIS) can describe the function specifications in the application environment more availably and make the reaction actively for the different events in the application environment. We propose the solution to realize AIS in the Event Definition and Processing, Definition and Processing of ECA Rule and Realization Strategy. With DSS-PCF the design methods are a very effective approach to develop the active information application system.
Key words:
Event, ECA Rule, ADB, Confluence, AIS.

1. Introduction

Traditional database systems have been passive and were designed to manage large collections of static data [1,2]. Data values in any database instance are related through certain dependencies and restricted through certain constraints. The focus of database design was to efficiently maintain these relationships among data elements whenever the database was updated. Updates to the stored data set happen in atomic operations, where each update is isolated from all other update operations. Query answering is a history-less, atomic operation. As long as there are no updates, a given query will result in the same response, regardless of how many other queries are running concurrently and how many times the same query has been asked before [3]. Active Databases (ADB) are a combination of traditional static databases and active rules, meant to be automated mechanisms to maintain data integrity and facilitate in providing database functionalities. Event-Condition-Action (ECA) rules comprise three components: event E, condition C, and action A. The event describes an external happening to which the rule may be able to respond. The condition examines the context in which the event has taken place. The action describes the task to be carried out by the rule if the relevant event has taken place and the condition has evaluated to true [4,5].

Information systems are to provide a service, which are semantic entities entailing consideration that span the life cycle of the application system. Services were initially defined and modeled as sequences of tasks resulting in various flow based representations. Services are interactive in nature involving user sessions with one or more users. An interactive service need not be a closed atomic operation and is also not isolated from one another. It may involve many intermediate states where the external environment may influence the flow of the computation. Two or more interactive services may be intertwined in such a way that their operations can not be serialized [6].

The distinction between a database and an information system is best appreciated when we consider their function. The task of a database is to store data and answer queries. The task of an information system is to provide services [7]. Continued active database system detects events and does related trigger actions as a result of this detection, according to the condition. Most of the active capabilities are provided by a set of ECA rules [5]. By contrast, Active information system has to maintain its dynamic integrity when it has only partial control over its own dynamics, and maintain an ongoing interaction with environment rather than transform a given problem to a solution. However there is a growing realization that services are best designed using models of interaction, developed largely in reactive system domain [8].

The community of information system researchers addressed issues of managing information within a large dynamic system. The general model of an information system is in form of a collection of semantic services [3]. As a result of the lack in the standardization, legible definition of execution semantics, advanced description ability and overfull restriction in business databases [9], we argue for a change in viewpoint, so active information system (AIS) is an interactive service-providing systems rather than mere data transformation engines. This change offers a promising approach for addressing ADB shortcomings, and reveals a roadmap of additional features for ADB.

In this paper, we propose the main practical key techniques in active information system that deals with the passive functions in traditional database systems and active functions in application environment together for active data processing and application service. So the active requirements can be implemented by unified

Manuscript received March 5, 2007
Manuscript revised March 25, 2007

mechanism. According to the internal status or the external event of database system, the active data processing and application service are afforded. Specially the requirements from users in the application domain can be described and notified. Active views provide active caching of materialized views to support subscription services and notification services, and to update user profiles specially. Furthermore, we can receive the high performance and flexibility in AIS. That means AIS can complete the most tasks from acquirement information phase to produce decision phase instead of decision-makers and provide the information services actively.

2. Key Techniques

2.1 Framework of AIS

The idea to implement AIS is defining the given events and related rules in conformity to the application condition in the information system. When any event which is needed by the rule occurs, AIS executes the database operation, system control and user exit routine in the agreement with the action in the related rule. Therefore the capability of active information processing is offered and satisfied for active requirements in application fields. According to this idea, we designed and realized the decision aided support system for preventing or controlling flood (DSS-PCF). The framework of DSS-PCF is as Figure 1 [7].

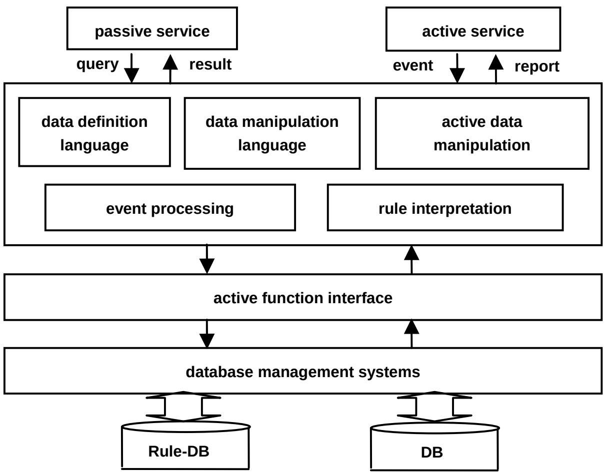

Figure 1 Framework of DSS-PCF

Here the new concepts are introduced other than ADBs, such as rule instances, state space of a rule and state change on rule [3]. AIS should support rule instances that run in their own contextual space and are logically isolated from other rule instances. Rule instances obviate the need for convenient approaches to differentiate rules from one another. Rule instances can also directly be mapped to different instances of services offered by the AIS. In addition to rule instances, AIS should support a state space of a rule. Rule instance is very useful and can help in tracking and managing each individual instance of a service more precisely. With the state space of a rule, it is possible to incorporate integrity constraints across rules depending on how many instances are there in which state of the state space. On the other hand, the state change such as enabling, disabling and waiting on rule is necessary in AIS. These three functionalities are possible for individual rule or rule instance to explicitly enable, disable and wait on the triggering of another rule. Enabling and disabling of rule are not controlled by the rules themselves, on the contrary by the rule processing interface. Rules need not be aware of this external control over their behavior. Enabling, disabling and waiting primitives can easily process the formulation of synchronization across disparate semantic services are incorporated in the model.

2.2 Event Definition

Events in DSS-PCF are separated into two kinds: atomic events and complex events. The atomic event is the elementary unit to define an event. It describes an event which can occur in any time from internal or external database system. The complex event is assembled with some atomic events by means of algebra.

The kinds of the atomic events are as follows:

1. A data processing event is the event which occurs after or before a database processing acts on the database item in database system. Its expression is:
   [BEFORE | AFTER] (INSERT | UPDATE | DELETE | QUERY) database_item
2. A transaction processing event is the event which occurs at beginning, end or cancel of a database transaction processing. Its expression is:
   event (BOT | EOT | ABORT | [transaction_name])
3. A time event is the event which occurs at appointed time or during a time period. Its expression is in two kinds:
   event (year. month. day, hour : minute)
   event (EVERY frequency [YEAR | MONTH | DAY | HOUR | MINUTE] time WITH interval
4. A sequence event is that m events occur in an event list. The sequence event is independent of the time sequence. Its expression is:
   SEQUENCE (m, e1, e2, …, en) $m \le n$

5. A count operator describes that event occurs fixed times. Its expression is:
   COUNT (event, times)
6. A user-defined event provides users to define some especial events which come from the application fields.
   DEFINE EVENT event_name
   BEGIN
   event_specification
   END

There are five arithmetic operators to construct the complex events. They are NOT, AND, OR, COUNT and SEQUENCE. The arithmetic operators of NOT, AND and OR are the same as relation algebra. The COUNT operator describes a complex event that event occurs fixed times. The SEQUENCE operator describes that several events occur in a determinate sequence. A user-defined event is defined and activated by oneself. Mostly, the main problem addressed here is the description of services. It is embedded in the application with user_exit. A service description specifies how task in the service ought to be processed. So a user-defined event is to resolve the extra active services, which are from the business rules and can not be afforded by DBMS itself.

2.3 Event Processing

An event appearance is identified by event processing. All occurring events are inserted in the event recorder. After the action related with the event has been processed, this event is deleted from the list of the event recorder. The data processing event and transaction processing event utilize the event trigger in database system to realize the event processing. The time event is defined in application system and inspected by the subsystem for the time inspection.

The complex event definition depends on the time. We design first matching algorithm as the distinguishing method for the complex event. That means the search in the event recorder is in the time order. The firstly researched event which accords with the condition is matched firstly. An example complex event of Flood_Alarm is defined as follows:

```
DEFINE EVENT Flood_Alarm
   BEGIN
      AFTER UPDATE Flood_ Flux AND Flood_
                         Flux ≥ 5000
   END
```

2.4 Definition of ECA Rule

An active information system must provide a knowledge model (ECA rule model) and an execution model for supporting the reactive behavior [10]. This is in contrast with the execution which determines how a set of rules behaves presented in AIS. The knowledge model essentially supports the description of the active functionality. ECA rule is a production rule. It is composed of the event, condition and action. The language for active data processing is realized with the mechanism of ECA rule. The mechanism of ECA rule includes the rule specification, rule execution and rule management. The rule specification describes the event characteristic, relevant condition, action execution and relationship among them. The syntax of ECA rule in DSS-PCF is as follows:

```
rule_definition ::= RULE   rule_id
                      ON   event
                      WHERE  condition
                      DO   action
                      [ PRIORITY number]
                    ENDRULE
```

The defined ECA is stored in the rule database after the processing of the rule interpreter. ECA rule is a data item and can be stored in DB. It is managed with DBMS, but we design a special program module in the component of the active function interface to search and identify the appointed rule. According to the concept of the transaction processing in DBMS, a transaction processing defined by user is actually the process structured with some data processing in the certain sequence. The action of ECA rule explains how the data is managed. In order that DSS-PCF keeps in active status at all times, there is a stipulation that the last command of the command sequence must be the commit command. After the action in ECA rule finished, the system control is transferred to the event processing, in order to keep the activity and timeliness of DSS-PCF. Because the action of ECA rule is not changed anymore in the execution, ECA rule is not allowed in the form of the self recursion [7]. A given example for the ECA rule expression is as follows:

```
RULE Flood_Schedule
   ON    Flood_Alarm
   WHERE   occur in 2 times in 1 day
   DO    start flood prevention process
   PRIORITY  20
ENDRULE
```

2.5 Processing of ECA Rule

Rule analysis deals with predicting how a set of rules behaves at run time [5]. Confluence analysis in AIS is an important problem, because the execution of any rule can cause the data change and then continue to trigger the other rules [11]. The rule set in AIS must be arranged to avoid that the mutual triggering among the rules can not be executed infinitely. However rules are defined by users in different time, so to assure the termination of the rule execution is difficult or impossible by manpower [12]. In DSS-PCF we designed a module to check the confluence.

For processing the Confluence of ECA rule, the interaction influence between rule trigger and rule priority should be considered. Even the execution order among rules directly or indirectly triggered by actions in ECA rule should be considered necessarily. Therefore we introduce the following definitions [13].

Definition 1: The rules r_i and r_j are conflicting, if one of the following conditions comes into existence:
(1) the action in r_i modifies the data related the condition in r_j, or contrarily the action in r_j modifies the data related the condition in r_i,
(2) the action in r_i and the action in r_j modify the same data.

Definition 2: If two rules are not conflicting, the reversal of their execution order can not cause the different results. Then these two rules are exchangeable.

Definition 3: T(e) is a rule set triggered by the event e. $T^*(r)$ is a rule set triggered by the action in rule r, when r executed.
P(r) is a database operation set fulfilled by the action in rule r.

Definition 4: Q is an interaction rule set including the following interactive operations:
(1) $r \subset r'$ indicates that r' is executed too, if r is executed.
(2) $\overline{r,\ r'}$ indicates that only one rule is executed, when both of r and r' are triggered.
(3) $r < r'$ indicates that both of r and r' are triggered and the execution of r' is prior to the execution of r, if only one of them can be executed.
(4) $r \prec r'$ indicates that the execution of r is prior to the execution of r', if both of r and r' are executed.

Definition 5: r (e, c, a) is an active rule. (e', c', a') $\in T^*$(e, c, a). For any state satisfied the condition c, the condition c' can be also satisfied, if and only if the action a executes, and the condition c' can be satisfied, then the condition c' is immutable to active rule r. If the condition c is always satisfied, then the condition c' is immutable to the action a.

According to the interaction relation in rules, we can utilize the exchange of rules to check the confluence problem of active rules. Here is the sufficient conditions that ensure AIS is confluent [12]: an active information system A is supposed as a static, determinable and terminable. E is an event set in AIS. For any $e \in E$ and two exchanged rules $r_1(e, c_1, a_1)$ and $r_2(e, c_2, a_2)$ in Q, If two triggered rules r(e, c, a) $\in T^*(r_1)$ and r'(e, c', a') satisfy the following conditions: (1) r and r' are exchangeable, (2) the condition c is immutable to rule r' and the condition c' is immutable to rule r. Then A is confluent.

Based on the above sufficient conditions, we design an algorithm to check the confluence in DSS-PCF:

```
PROCEDURE ( R: (r1, r2, … rn ), confluent: BOOLEAN )
  BEGIN
    FOR every ri DO count T*( ri ) ENDFOR
    FOR every e ∈ E DO
      count T(e),
      FOR every rule pair ( r1, r2 ), r1∈T(e), r2∈T(e) DO
        IF priority (r1) < > priority (r2)
          THEN confluent = true
          ELSE IF every r ∈ T*(r1) and r' ∈ T*(r2)
                  satisfy the following the conditions:
                  r1 and r2 are exchangeable, c is
                  immutable to r' and c' is immutable
                  to r
                THEN confluent = true
                ELSE confluent = false, EXIT
              ENDIF
        ENDIF
      ENDFOR
    ENDFOR
  END
```

The complexity of this algorithm depends on the amount of interaction rule set Q. In this algorithm the comparing time for Q is an expression of multinomial time, showed as $O((m + n)^t)$ [14]. We suppose the comparing time for Q as t', $|E| = k$, $|R| = m$ and $|Q| = n$. So the worst time complexity of the confluent algorithm is $O(k((m + n)^t\, t'))$.

3. Realization Strategy

At the present software environment, the effective approach is that the toolkit for designing AIS will be developed beforehand. Then the active application system is developed on this toolkit. As a result of the constant development of business database systems, for example ORACLE, the event definition and the stored procedure are realized in DBMS or in the developing tools. A developer controls the database via low layer interface, uses the SQL, database trigger or stored procedure to develop the interface of the database management. Certainly the developer can embed the programming language and the control mechanism from the operating system into the database applications with pro*C, user_exit, C++, etc [7].

Based on the component design method, we use directly the DBMS to fulfill the elementary data definition and processing function, which satisfies the passive function. The event or action of the non business logic can be defined as the component in DBMS [15]. In DSS-PCF there is an additional middleware, which is the active function interface, among communications. This middleware is in charge of the all communication between the active operation and DBMS. We encapsulate the active processing function in the middleware as a component, in order to adapt the change of the active processing function and provide the reuse of active data object. The message queue in DSS-PCF may easily be transferred between the database requirement and the component with the active function interface, so that it is easy to realize active application system. Because there is the commonness among the different active business logic, it is very effortless to maintain the active application system and realize the reuse technique.

Based on the framework of DSS-PCF in Figure 1, data definition language and data manipulation language adopt the extended SQL supplied by DBMS. The other modules were developed by the toolkit implemented with C++. This toolkit can be considered as a virtual machine above DBMS. This virtual machine extends the functions of the traditional database system and offers the processing ability of the active service requirements. The rule database, the rule processing subsystem and DSS-PCF toolkit make up a DSS-PCF core. DSS-PCF toolkit is the foundation to sustain the ECA rule processing and time management. It solves the problems that traditional database systems can not directly support the event management, rule realization and action execution.

4. Conclusions

AIS can describe the function specifications in the application environment more availably and make the reaction actively for the different events in the application environment. AIS advances the design quality and application functions of the information system. Some key techniques are introduced in this paper. Specially the solution for Confluence problem is analyzed and realized. With DSS-PCF the design methods is a very effective approach to develop the active information application system.

Acknowledgments

This research work is supported by Jiangsu University Natural Science Foundation under Grant No. 06KJB520079.

References

[1] U. Dayal, B. Blaustaints: The HiPAC Project:Combing Active Database and Timing Constraints. SIGMOD RECORD, Vol.17, No.1, March 1988

[2] Shaoyuan Li, Xiaopei Luo: New Database Technology. Tsinghua Publishing Company, 1997

[3] D. Goldin, S. Srinivasa, V. Srikanti: Active Databases as Information System. Proceedings of the International Database Engineering and Applications Symposium. 2004 IEEE

[4] K. R. Dittrich: Active Database System. Institute for Information, University Zurich, 1992

[5] N. W. Paton, O. Diaz: Active Database System. ACM Computing Surveys, Vol 31 No 1, March 1999

[6] H. J. Schneider, A. J. Wasserman: Automated Tools for Information systems Design, IFIP WG 8.1 Working Conference. Jan. 1991

[7] P. Zong, J. Qin: The Scheme Design for Active Information System. Knowledge-Based Intelligent Information and Engineering Systems (LNAI 4252). Springer 2006

[8] S. Srinivasa. Channel Sensitivity in Reactive Systems. Proceedings of International Conference on High Performance Computing, Embedded Systems Workshop, India, Dec. 2001

[9] P. Shao, S. Sun: Functions of Active Databases in Standard SQL and Business DBMS. Computer Engineering and Application. Sep. 1999

[10] C. Ryu, H.Han, Y. Kim, Yk. Kim, S.Jin: Kernel Structuring Using Time-Triggered Message-Triggered Objects for Real-Time Active DBMS in Layered Architecture. The 3th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. March 2000

[11] A. Vaduva, S. Gatziu, K. R. Dittrich: Investigating Termination in Active Database System with Expressive Rule Languages. Technique Report 97-03, University Zuerich, 1997
[12] W. Zuo, J. Liu, S. Liu: Relationship Graph and Termination Analysis for Active Rule Set. Journal of Software, 2001, 12(2)
[13] C. Jiang, Z. Hao: Confluence Property of Active Database. Journal Harbin University Science and Technology, Vol.9 No.2. Apr. 2004
[14] H. Jagadish, A. Mendelzon, I. Mumick: Managing Conflicts between Ruled. Proceedings of PODS. Canada, 1996
[15] Hui Xu, Min Chen, Xiaoyu Zhang: Study of Component-Based Active Database Models. Computer Engineering & Science, Vol.23, No.6. 2001

Ping ZONG is a professor of computer science and Technology at Nanjing University of Posts and Telecommunications. His research interests include intelligent information system, software engineering, network and so on.

Jun QIN is an associate professor of computer science and Technology at Nanjing University of Posts and Telecommunications. Her research interests include intelligent information system, multimedia technology and so on.

Qiong YU is a lecture of computer science and Technology at Nanjing University of Posts and Telecommunications. Her research interests include intelligent information system, software engineering and so on.