A New Transaction Scheduling Scheme for Active Real-time Database Systems: Space-Filling Curve

Sunil R. Gupta[#], Dr. M.S.Ali[*]

[#]*Department Computer Science & Engineering,*

P.R.M.I.T& R, Badnera.

[1]sunilguptacse@gmail.com

[*]*P.R.M.C.E.A.M, Badnera*

[2]softalis@hotmail.com

Abstract— **This paper presents the space-filling curves and their applicability in scheduling, especially in transaction scheduling in Active Real-time Database Systems (ARTDBS). Active real-time databases have emerged as a research area in which concepts of active databases and real-time databases are combined into a real-time database with reactive behaviour. A novel space-filling curve based approach for scheduling transaction in active real-time database systems is described along with generalized transaction model for active real-time database system is given which is used as base model for transactions in active real-time database system. This paper addresses proposed space-filling curve approach is detailed followed by its comparative evaluation with different algorithms used in ARTDBS.**

***Keywords*- active real-time database, real-time database, transaction scheduling, transaction model, trigger, space-filling curves, multi-dimension.**

I. INTRODUCTION

An active real-time database system (ARTDBS) is a database system where transactions have timing constraints such as deadlines and where transactions may trigger other transactions and where data may become invalid with the passage of time. Both active and real-time databases are considered as important technologies for supporting non-traditional applications such as process control, network database services, network management, air traffic control, cooperative navigation systems and computer integrated manufacturing (CIM) [1,2,3]. Generally active database systems support ECA (Event-Condition-Action) rules, which are triggered and executed within the context of database transactions. Applications that require execution of transactions with respect to time constraints require a real-time database system.

A brief introduction to active databases and real-time databases, subsequently, presenting description of the ARTDBS. The building block of an active database system is the event-condition-action (ECA) rule. The semantics of the ECA rule is that, if the specified event occurs and if the condition is true, then the specified action is to be executed. A condition is usually a predicate on the database state [4]. An action is the transaction that is executed in reaction to a specific situation, which is a combination of events and conditions. The transaction that fires the rules is called the triggering transaction, and the action that is executed because of the rule firing is called the triggered transaction. We refer to the transactions that trigger other transactions as active transactions or parent transactions. An active transaction has a set of triggered transactions that are executed either as part of the active transaction or separately, depending on the type of the coupling mode between the parent and the triggered transactions [5]. There are three types of coupling modes: immediate, deferred and detached (independent). The transactions triggered in those modes are referred to as immediate, deferred and detached transactions, respectively. Due to the rule firings, an active transaction dynamically generates additional work.

Active database systems support different mechanisms for triggering of transactions to react to the critical events occurred in the external environment [6]. The newly created transaction is called triggered transaction. However, active database systems are lacking mechanisms to guarantee that the triggered transactions can be completed before their deadlines. Their processing is "passive" even though their generations are "active". It is unpredictable and completely dependent on how the system schedules them. A real-time database system (RTDBS) can be viewed as a system, which inherits mechanisms of both traditional database systems and real-time systems. It is a transaction processing system that is designed to handle workloads where transactions have completion deadlines. The objective of such system is to meet these deadlines, that is, to complete processing transactions before their deadlines expire. On the other hand, RTDBS supplements the deficiencies of the unpredictability in active database systems by using different scheduling algorithms to minimize the number of deadline missing.

The integration of real-time database systems with active databases creates new scheduling problems. Triggering of transactions represent the generations of actions (in the form of triggered transactions) to respond to the occurrences of critical events in the external environment. It is important to commit these transactions. If the critical events are beneficial, missing the deadline of a triggered transaction means the loss of a good opportunity. If the occurred events are harmful,

missing the deadline may result in disasters. Triggering of transactions decreases the predictability of the system as the triggered transactions increase the system workload and the probability of data conflicts suddenly [7].

Many people have devoted their efforts to find a solution to the problem of efficiently scheduling task or transaction with multi-dimensional data. This problem has gained attention in the last years with the emergence of advanced database system and operating system such as real-time databases, real-time operating system which need to schedule and process the task or transaction in an efficient way. Hence, techniques that aim to reduce the dimensionality of the data usually have better performance. One such way of doing this is to use a space-filling curve. A space-filling curve can transform the higher dimensional data into a lower dimensional data using some mapping scheme [34].

In this section introduced ARTDBS, importance of scheduling transaction in ARTDBS and space-filling curve, section 2 describes ARTDBS model. In section 3 related issues of ATRDBS are given. Current research trends in ARTDBS are discussed in section 4. Finally conclude in section 5.

II. Related Work

Ideally, in an active and real-time context, transaction-processing mechanisms should integrate various components such as concurrency control (CC), scheduling, overload management (admission control), transaction dependencies (semantics) and conflict resolution as one seamless issue. Otherwise, incompatibilities among the components, may reduce the number of transactions completing on time, increase unnecessary restarts and wasted resource, and favour some classes of transactions. One of the very important components of ARTDBS transaction processing is transaction scheduling. Research in ARTDBS is in its initial stage not much work has been done [32,35].

Active databases and real-time databases have been important areas of research in the recent past. It has been acknowledged that many benefits can be gained by integrating both active and real-time database technologies. There has not been much work done in the area of transaction scheduling in an active real-time database system [12,25,31,35,36]. A rigorous effort that integrates various components of transaction processing in ARTDB context is reported in HiPAC [6]. In HiPAC, considers time constrained transaction processing, although time-based triggering is incorporated.

Two research projects, which take inspiration from the work on HiPAC, are REACH and DeeDS. REACH is a database system, which incorporates active features with timing constraints working in a heterogeneous environment. Their contributions to the development of the technology include the additional detached coupling modes and a suggestion on how to include timing information in the rule definition language. The DeeDS prototype integrates reactive mechanisms, distribution, dynamic scheduling of hard and soft deadlines and integrated monitoring. The approach is to take existing components and implement only those features necessary to obtain the required functionality.

RADEx is a real-time, active, temporal database simulator. It supports research on time cognizant concurrency control protocols, real-time transaction scheduling, and real-time logging and recovery [19]. ARTS-I (Active Real-Time DBMS prototype System) [37] is prototype system developed on AT&TULX System V4.0. It provides static and dynamic the transaction priority assignment policy, united scheduling mechanism for hard/soft, periodic/aperiodic transactions based on priority. Also forward step towards generalized transaction model. KRAFT is an Active Real-Time DBMS for Signal Streams. KRAFT is an extension of traditional DBMS [12]. It has distinguished features such as scheduler-level thread control mechanism for continual event monitoring, similar sequence retrieval for signal processing, UPS write ahead logging for predictable timely processing of transaction.

Another exciting and recent development was, the First International workshop on Active Real-Time Database Systems (ARTDBS95) held at Skovde, Sweden, 1995 and Second International workshop on Active Real-Time Database Systems (ARTDBS97) held at Como, Italy, 1997. The organization of these workshops is an indication of the timeliness and emerging importance of ARTDBS research area. Panel discussion on in these workshops also suggests that scheduling transaction is important aspect in ARTDBS.

The issue of deadline assignment in active real-time database system for active transactions i.e. triggered transaction is discussed in [20] otherwise not much is reported on this issue. Scheduling of triggered transaction in ARTDBS is discussed in [38]. In this work different approaches for assigning deadlines and priorities to the triggered transaction with the objectives to satisfy the timing requirements of the triggered transactions especially for more critical ones using Earliest Deadline First scheduling (EDF) policy for dynamic scheduling for aperodic transaction and Rate Monotonic Analysis (RMA) policy for static scheduling for periodic transaction. In [10] it suggest that more commonly suitable scheduling algorithm in ARTDBS is Earliest Deadline First (EDF) and Least Slack First (LSF). Also in [32] also used Earliest Deadline First (EDF) in their base model for performance study.

From above discussion and also we have investigated algorithms mainly used and suggested for ARTDBS are Earliest Deadline First (EDF) and Least Slack First (LSF).

III. Active Real-time Database System Model

An active real-time database system (ARTDBS) has to provide capabilities for timely trigger of time-constrained transactions and at the same time to process them, concurrently with others transactions, in a real-time manner. Mostly, triggered transactions are critical transactions. The newly created transaction is called triggered transaction.

An ARTDB system model given below (figure 1) uses a queuing model of a single-site shared-memory multiprocessor database system. This model is similar to those presented in other simulation studies e.g., [8,9].

In this section, an ARTDBS system model is presented as shown in figure 1 is referred from [10]. The figure 1 shows that model of ATRDBS consists of six components: include *six active components*, namely, Source (transaction generator), transaction manager, concurrency control manager, memory manager, resource manager, rule manager, and *one passive component*, namely, the database.

A Source also known as transaction generator that generate the non-triggered, or external workload of the system. A Transaction Manager (TM) that models the execution details of the transactions. It handles the various stages of transaction execution, such as begin, commit and abort. A Concurrency Controller that implements the (e.g. OPT-BC) protocol for managing concurrency. A Resource Manager that consist of the CPU Manger and the Disk Manager, that manage the system's CPUs, disks and the associated queues by using scheduling policy such as Earliest Deadline (ED) protocol, and a Rule Manager which implements the active functionality. Rules are triggered when a specified event occurs. When the event notification arrives at the rule manager, all rules triggered by this event must be retrieved from the rule base and be prepared for execution. The time to retrieve rules brings potentially unpredictable overhead to the system, and it is therefore critical to keep this time to a minimum. Methods for storing and retrieving rules must be carefully considered to allow predictable and efficient retrieval. There is always a compromise between flexibility and efficiency; the more static a system is, the more decisions can be made off-line, and thereby run-time overhead can be kept low. If a rule base can be made static, that is, no rules are added, deleted or altered at run-time, the rules can be analysed, optimised and compiled into the rule manager, thereby improving efficiency. Buffer Manager is associated with the use of main memory during reading and writing data from and to the disk. When the main memory has a plenty of space, the database is preferred to reside in the main memory ("memory resident database system") to enable fast and predictable access [12,13].

The transaction manager interacts closely with the concurrency controller, memory manager and the resource manager. The TM first interacts with the concurrency control manager to seek permission for data access or to see if marked for restart/abort. The rule manager manages the active workload of ARTDB. This module simulates a rule base in which a transaction triggers other transactions when a condition becomes true. The concurrency module simulates concurrency control mechanism to maintain database consistency [10].

IV. Generalized Transaction Model for Active Real-time Database Systems

An active real-time database system is a database system having dual capabilities, which seamlessly integrate the concepts, theories and technologies from active database and real-time database system. In some modern database application fields, especially in real-time computing, such as weapon guiding and navigation, automatic manufacture, radar tracking, traffic management, programming control telephone etc., there're many application requirements different from traditional business and transaction, which are timing of transaction, timing of data and activeness of response i.e. ability to automatically monitoring the system. The database system meeting these mentioned application requirement, should not be able to handle timing constraints of transaction or data, but also have active computing ability.

In active real-time database system literature study there is no generalized transaction model framework is mentioned explicitly, which accommodate the requirement of all type application including different types/class of transaction, deadline, temporal validity of data, criticality etc. and active database capability in terms of E-C-A model. The generalized transaction model for active real-time database system is given following.

An active real-time database system can be describe Database management (traditional database) + Active Action (trigger mechanism) + Real-time computing (timing constraint). The formula is not a simple "+" computation, but an integration of their concepts, theory and technologies.

Timing constraints in RTDBS can be defined as RTDB-TIME

RTDB-TIME = <AT, WCET, TT, TP, Pr, V, SL, DD, DL >

For a transaction, AT is a arrival-time or execution beginning time, WCET is the execution time evaluation in worst case, TT means transaction type triggering and non-triggering, TP is transaction type i.e. Periodic and Aperiodic, Pr defines fixed priority of transaction, V defines value of transaction higher value means higher importance also used to define criticality (reciprocal of value V), SL is slack time of transaction, D represents data deadline or temporal validity of data and DL is the deadline.

DL can be defined as

DL = <Du, Ds>

where Du is the urgency degree of deadline, which can be defined, further as

Du = <DuH, DuS, DuF>

where DuH means hard real-time deadline, DuS means soft real-time constraints and DuF means firm real-time deadline.

Ds = <DsA, DsR>

where DsA is a absolute timing property and DsR is a relative timing property of a transaction execution.

Active Database System supports the dual control stream of application and database system, i.e. application can invoke database and vice-versa. From its component, ADBS can be describe as

ADBS = <Eset, Aset, Etype, EAexp>

where Eset is a set of event, Aset is a set of transactions or actions, Etype is the event type, it can be insert | update | delete | select, EAexp is a condition expression on database state of an associated transaction or action in Eset and Aset.

Action can be transaction, sub transaction, components, a certain independent system behaviour program or user-defined

independent operation sets (action) activated directly by triggering mechanism (without transaction scheduling).

The dynamic execution model of ADBS based on the above definitions can be described as following. By simplifying the usual E-C-A (Event-Condition-Action) model to S-A (Situation-Action) model, thus the event and condition are unified into situation, which avoids the complex E-C (Event-Condition) match processing

ON SITUATION
DO ACTIONS
where,
SITUATION = <Eset,CONSTRAINT,*COUPLING_MODE* >
CONSTRAINT = < EAexp, *RTDB-TIME*>

SITUATION has three components (Eset and CONSTRAINT) as a whole and *COUPLING_MODE*, which defines the dependency between action (triggered) and triggering transaction, in form when to execute the action. Coupling modes are immediate, deferred and detached. Eset and EAexp, has the same meaning as the above, *RTDB_TIME* parameter is also associated with action, when one event is associated with many triggers.

The above execution model can be interpreted as: event detector detects the event occurrence signal pool at any time (before or after a system logical behaviour, at a period), once a certain event or some events happens, the associated constraints are evaluated at once. The associated action is activated (not always at once) with considering coupling mode, if the evaluation result is true.

The generalized transaction model for active real-time database system can be defined as real-time database timing properties and active database reactive mechanism with time constraint. ADBS model for representing reactive mechanism we use the SITUATION-ACTION model. RTDB-TIME represents real-time timing constraints.

Active real-time database dynamic transaction model is as follows.

ARTDBS-TRANS = <RTDB-TIME, ADBS>

Expanded ARTDBS model is as follows

ARTDBS-TRANS = (<AT, WCET, TT, TP, Pr, V, SL, DD, Du, Ds >,
<Eset, **EAexp, RTDB-TIME, COUPLING_MODE** >)

The generalized transaction model has as discussed above has capability to cater the need of each type of applications which require reactive mechanism with timely response to transaction according to database state.

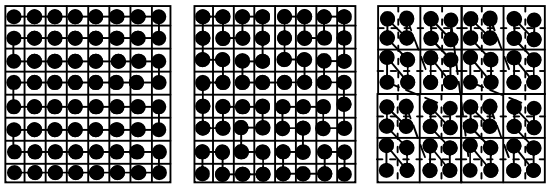
V. SPACE-FILLING CURVE

Many people have devoted their efforts to find a solution to the problem of efficiently scheduling task or transaction with multi-dimensional data. This problem has gained attention in the last years with the emergence of advanced database system and operating system such as real-time databases, real-time operating system which need to schedule and process the task or transaction in an efficient way. Hence, techniques that aim to reduce the dimensionality of the data usually have better performance. A space-filling curve can transform the higher dimensional data into a lower dimensional data using some mapping scheme.

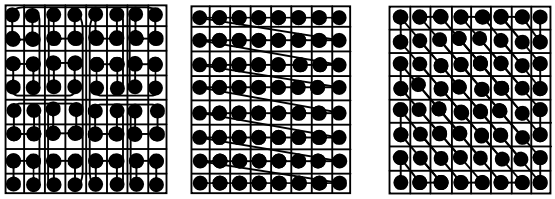
Space-filling Curves (SFCs) have been extensively used as a mapping from the multi-dimensional space into the one-dimensional space. A space-filling curve (SFC) [34] maps the multi-dimensional space into the one-dimensional space. Mapping the multi-dimensional space into one-dimensional domain plays an important role in every application that involves multidimensional data. Multimedia databases, geographical information systems, QoS routing, image processing, parallel computing, data mapping, circuit design, cryptology and graphics are examples of multi-dimensional applications. Modules that are commonly used in multi-dimensional applications include searching, scheduling, spatial access methods, indexing and clustering [39,40,41].

A space-filling curve is a way if mapping the multi-dimensional space into the one-dimensional space. An SFC acts like a thread that passes through every cell element (or pixel) in the multi-dimensional space so that every cell is visited exactly once. Thus, space-filling curves are adopted to define a linear order for sorting and scheduling objects that lie in the multi-dimensional space. Figure 2 gives examples of six two-dimensional space-filling curves. Using space-filling curves as the basis for scheduling has numerous advantages, like:

- Scalability in terms of the number of scheduling parameters,
- Ease of code development and maintenance,
- The ability to analyse the quality of the schedules generated, and
- The ability to automate the scheduler development process in a fashion similar to automatic generation of programming language compilers.



a. C-Scan b. Hilbert c. Peano



d. Gray e. Sweep f.

Figure 2. Space-Filling curves examples.

VI. SFC BASED PROPOSED TRANSACTION SCHEDULING

With growing complexity of applications an intelligent scheduler design and algorithm design should be needed, which not only use the value information, deadline and criticalness of the system for decision-making but also consider another parameter of transactions. As we know some of the parameter of transactions are associated with transaction, which is static and some are known at run time or prior to run time utilizing those information in terms for processing and scheduling transaction is beneficial. A design method for integrating the value, the deadline, the criticalness etc. is required [42,43].

Hence, the use of a new transaction-scheduling scheme is proposed for Active Real-time Database System based on five-dimensional design by integrating the characteristics of priority (provided by external environment), deadline, slack-time, value and criticalness. Here space-filling curves can be used as they are adopted to define linear order for sorting or scheduling. The space-filling curves unnaturally consider value, deadline, priority, slack time, temporal data deadline, and criticalness information and give a scheduling sequence.

Each transaction is having basic parameters, which are: transmission time, transaction arrival time, total transaction time, transaction arrival rate, total number of transactions, transaction arrival fashion (or distribution), average execution time, transaction inter arrival time, actual arrival time, transaction turn around time [44]. In active real-time database transaction from or change in external environment causes condition-action evaluation. That means in ARTDBS externally changed situation or sudden event occurs, external transaction known as triggering transaction and triggering transaction may or may not generate actions known as triggered transaction.

An incoming transaction i.e. triggering transaction and action i.e. triggered transaction are modelled by multiple parameters, (e.g., the real-time deadline, the criticalness, the priority, the value etc.) and represented as a point in the multi-dimensional space where each parameter corresponds to one dimension. Using a space-filling curve, the multi-dimensional in form of parameter, associated with transaction is converted to a one-dimensional value. Both triggering and triggered transaction parameters i.e. multi-dimension in form of parameter may get mapped into single dimension.

A space-filling curve with N priority levels in each dimension of a D-dimensional space is a contiguous path from the first point, say point 0, to the last point, say point N^D. The thread passes through all the N^D points exactly once. Thus, space-filling curves are adopted to define a linear order for sorting and scheduling objects that lie in the multi-dimensional space. A transaction request T takes a position in the thread path according to its space-filling curve value. They are then stored in the priority queue q according to their position in the thread path. The transaction scheduler walk through the thread path by serving all transaction requests in queue according to their path position, which is their one-dimensional value with a lower value indicating a higher priority. Figure 3 gives an illustration of an SFC-based transaction scheduler.

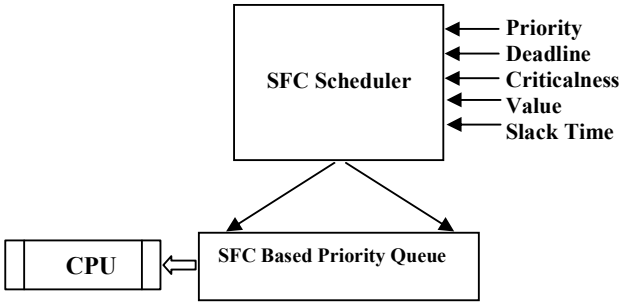


Figure 3. Space-filling curve based on Transaction Scheduler.

The space filling curves converts 3-dimensional space using the idea of bit interleaving, which is used and described in [40,41,42]. Every point in the space takes a binary number that results from interleaving bits of the two dimensions. The bits are interleaved according to the interleaving vector (0,0,0,0,1). This corresponds to taking the first and third bits from dimension 0 (x) and taking the second and fourth bits from dimension 1(y). The result of applying this bit interleaving is shown Table 1.

The sequence of few transactions obtained after mapping from 5-D to 1-D is shown in table 1 below.

TABLE I

AN EXAMPLE OF MAPPING TABLE FROM 5-D TO 1-D.

Point	Dimensions					Bit Interleaving	Decimal code
	0	1	2	3	4		
(0,1,3,1,2)	000	001	010	011	100	000010011001010	1226
(1,0,0,6,0,)	001	000	000	110	000	000100001010000	2128
(1,2,1,4,2)	001	010	001	100	010	000100100110100	2356
(1,0,0,0,1)	001	000	000	000	001	000000000010001	17
(3,0,1,1,0)	011	000	001	001	000	000001000010110	534

A. The Conditionally Preemptive Space Filling Curve Scheduler Algorithm

The *Non-Preemptive Scheduler:* Using this approach, once the server starts to serve the transaction requests from the priority queue q, no more requests are inserted in q. Arriving requests are grouped together in another priority queue q'. Once q is empty, q and q' are swapped, and the server starts serving transaction requests from q again. This scheduler is non-preemptive in the sense that the process of serving disk requests from q is not preempted by the arrival of new requests.

The *Fully Preemptive Scheduler:* This is a trivial approach. All requests are inserted in the same priority queue q regardless of their arrival times. This scheduler is fully preemptive in the sense that the process of serving requests from q is preempted by the arrival of any new request.

As a trade-off between the fully-preemptive and the non preemptive schedulers, in the conditionally-preemptive transaction scheduling algorithm: a newly arrived transaction R_{new} preempts the process of walking through a full thread path if and only if it has significantly higher priority than the currently served request R_{cur}. The value of a transaction is the priority of a transaction, given in above formula. We have used the value of a transaction as a function of its criticalness start time, deadline, and the current system time. When Rnew arrives while the scheduler is going to serve R_{cur}, then one of the following three cases takes place:

1. $R_{cur} < R_{new}$: This means that R_{new} has lower priority than R_{cur}. Hence, Rnew is inserted into q as inserting it into q will not preempt the thread path.
2. $R_{cur} - w < R_{new} < R_{cur}$: This means that R_{new} lies inside the blocking window w. Although R_{cur} has a priority higher than that of R_{new} but it is not high enough to preempt the space-filling curve thread path. So, R_{new} is inserted in the waiting queue *qL*.
3. $R_{new} < R_{cur} - w$. This means that R_{new} has a priority that is significantly higher than that of R_{cur}. So, it is worth to preempt the space-filling curve cycle by inserting R_{new} in q.

The space-filling curve thread path is preempted only by inserting the newly arriving request R_{new} of significant high priority into q. After preempting the cycle and serving R_{new} the process of serving the cycle is resumed. This algorithm is mainly suited for scheduling transaction in ARTDBS, where triggering and triggered transactions having dependency in terms of coupling mode. Depending upon situation, action will be taken in terms of scheduling a transaction.

B. Metric for Comparison

The performance metric we use in our study is missed deadline percentage (MDP), i.e., the percentage of transactions that miss their deadlines, which is a traditional metric used to evaluate performance in real-time systems. In ARTDBS to improve performance dynamic priority assignment methods are used. In ARTDBS processing and scheduling transaction is complex aspect. Changes in external environment represent in form of triggering transaction (i.e. user transaction), which may trigger another transaction in form of triggered transaction having commit dependency. Scheduling such transactions need to evaluate missed deadline percentage (MDP) of user transaction (i.e. triggering transaction) and triggered transaction generated due to ECA condition evaluation. Number of missed deadlines is an influential metric in scheduling algorithms for soft real-time systems. The transaction, which complete on time and follow the deadlines is taken into consideration. The success rate of transaction is also considered. Out of the total transactions submitted, how many actually complete their execution is considered

Another performance metric is to calculate the MDP having mixed load of input transaction i.e. Triggering (T) and Non-Triggering (NT) against coupling mode immediate, deferred and detached. Also finding out MDP for having mixed load of input transaction i.e. Triggering (T) and Non-Triggering (NT) against coupling mode combination such as Detached and Immediate mode, Detached and Deferred mode, and Immediate and Deferred mode.

These all metric are evaluated with different algorithm which are newly devised and classical algorithm used mostly in ARTDBS. The main aim is to evaluate the MDP of transactions under different situation mentioned above.

VII. EXPERIMENTAL EVALUATION

In order to find out performance impact of algorithms for transaction scheduling in active real-time database system, following present the experimental evaluation. There are a wide variety of algorithms for scheduling the transaction in traditional database systems. These scheduling algorithms are not adequate for real-time transactions. Scheduling transaction to be *'scalable', 'adaptive'* and *'flexible'*, we need to consider more transaction attributes other than deadline, which are most important factors for active real-time transactions.

This section describes the active real-time database model for scheduling transaction, experimental parameter and measures and evaluation of proposed SFC algorithm and its comparison with different algorithms like Earliest Deadline First (EDF) and Least Slack First (LSF). The comparison is further explored with different types of input transaction such as triggering and non-triggering with different coupling modes.

A. Experimental model parameters and performance measures

The ARTDBS model is built as mentioned above in Java with XML support utility. Database is main memory providing active real-time functionalities, mainly store data in XML format. In our experimental evaluation assigning a priority triggered transactions Transaction Data Deadline approach is used. Triggering transaction is unpredictable event and are resulted from the state of database, which is highly

dynamic. Thus, the system cannot anticipate the additional resource requirements of the triggered transactions. If the system schedules the triggered transactions in the same way as the other transactions, which do not trigger any transactions, the probability to commit the triggered transactions may be very low. If triggering and triggered transaction have commit dependency in form coupling mode such as immediate or deferred, then if anyone from triggering and triggered already miss the deadline then performance of system may degrade. Also there are temporal constraints of the data objects. The deadlines of the triggered transactions are also constrained by the temporal constraints of the data objects, which are responsible for their triggering. If the triggered transactions cannot be committed before the data objects become invalid, they have to be aborted, as the conditions for the triggering may not be true when the temporal data objects are invalid. These are rationale behind using Transaction Data Deadline approach.

In Transaction Data Deadline approach, the deadline of the triggered transaction is the minimum of the deadline of the triggering transaction and the temporal validity of the data object responsible for the triggering:

$$\textbf{Deadlinet}_t = \textbf{min (Deadline}_t\textbf{, Temporal}_D\textbf{)}$$

where $Deadline_t$ is the deadline of the triggering transaction and $Temporal_D$ is called the data deadline i.e. earliest data deadline of data object D, among the temporal data objects accessed by the transaction.

The model parameter and their baseline values are follows:

System

CPU Scheduling: Priority based Scheduling (SFC based table driven, EDF, & LSF)
Time to process page (Tprocess): 4.7 ms.
Block copy time (Tcopy): 0.5 ms.

Transactions

Arrival Rate of user transaction (ARup): 05-35 transactions per second.
Transaction size is (Noper): Calculated from database table size.
Slack Factor: 1.0 to 4.5 uniformly distributed.
Arrival rate of update transaction (ARup):400 transactions/sec
Avi (absolute validity Interval): 400-1000 ms.

In brief following given information about experimental setup:

- Addressing only soft real-time systems where the value of the transaction is lessened their values after the deadlines expire, though generalized transaction model supports hard, soft and firm.
- Considering active real-time databases that trigger only detached, immediate and deferred transactions.
- There are three types of class Transactions: IMM which trigger only immediate subtransactions, DEF that trigger only deferred transactions, and DETACHED that trigger both triggered and triggering transactions are independently execute.
- There are primarily two types of transactions considered for ARTDBS: nontriggering (NT) and triggering (T). Further triggering (T) is classified into triggering (user transaction) and triggered (transaction generated due to triggering or also called sub transaction). Triggering and triggered transactions are not mutually exclusive, i.e. a triggered transaction may trigger other transactions. Non-triggering (NT) transaction is a simple transaction, which does not cause any rules to fire.
- Simulating main memory ARTDBS model where there are no data conflicts.

B. *Performance Result*

Scheduling triggering and triggered transaction play important role in reducing missed deadline of transaction by considering priority and deadline assignment. Also scheduling the transaction by involving more attributes or parameters improves the performance. Following we explore the experimental evaluation and outcome result for scheduling transaction with different algorithm such as Earliest Deadline First, Least Slack First and proposed SFC table driven algorithm.

Experiments have been performed on different coupling mode such as detached, immediate and deferred, also using above-mentioned algorithm with input type transaction Only Triggering. We will evaluate these all with respect to base metric miss deadline percentage (MDP).

Following given experimental evaluation for Coupling Modes with Triggering (T) transaction as Input.

1. Detached Coupling Mode

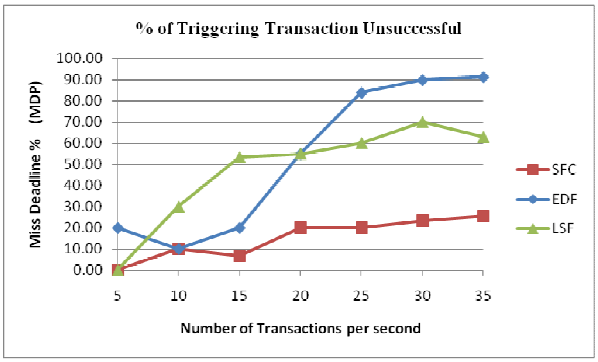


Figure 4. Impact of User/Triggering Transaction on Miss Deadline Percentage of user/triggering transaction with Detached Coupling Mode.

From figure 4, we draw a conclusion that proposed space-filling curve based table driven performs better as compared to other algorithm such as EDF and LSF, which mainly used in ARTDBS. In figure 4, it shows that success rate of triggering transaction is better, that is more user transaction commits in time.

Figure 5 is also helpful to draw conclusion about on large scale about committing of overall transactions and also about

triggered transaction i.e. "action" triggered due user transaction or external transaction. Figure 5 graph represent that triggered transaction miss deadline percentage is lower for SFC based table driven algorithm. As input load increases (user/triggering transaction) and same amount of load generated due to triggering of transaction other algorithms EDF and LSF algorithm performance degraded and leads towards higher miss deadline percentage. The coupling mode set for above performance evaluation is detached where transaction (i.e. triggering and triggered) executed independently.

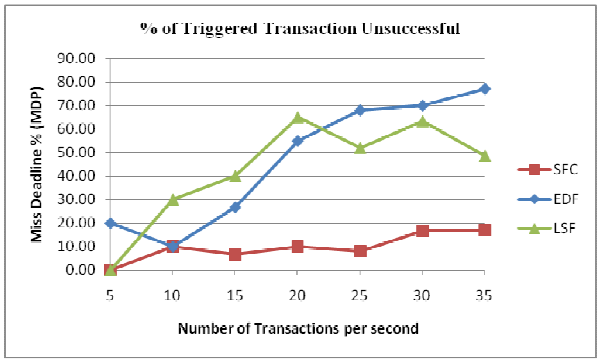


Figure 5. Impact of Triggered Transaction on Miss Deadline Percentage of triggered transaction with Detached Coupling Mode.

2. Immediate Coupling Mode

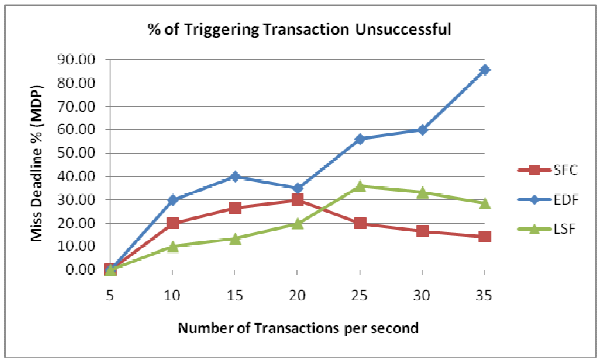


Figure 6. Impact of User/Triggering Transaction on Miss Deadline Percentage of user/triggering transaction with Immediate Coupling Mode.

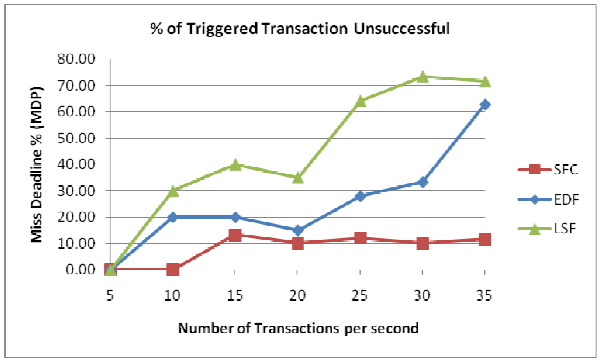


Figure 7. Impact of Triggered Transaction on Miss Deadline Percentage of triggered transaction with Immediate Coupling Mode.

In figure 6 input triggering transactions for immediate coupling mode where at low load EDF and LSF performance better as compared to SFC. The LSF performance is consistently moderate. The reason is in immediate coupling mode triggering transaction is executed at latter stage and execution of action or triggered transaction is done first and after that triggering transaction is release for execution, though slack-time or deadline of transaction is continuously decreasing but as an when triggered transaction finished the corresponding triggering transaction gets executed here on slack-time based scheduling gets an edge over deadline because though transaction is eligible for execution in EDF deadline time is valid but due to resource constraint it may delay other triggering transaction which one is waiting for execution. At high load of triggering transaction SFC based algorithm perform better.

In figure 7 shows triggered transaction impact with immediate coupling mode for miss deadline percentage is evaluated. But in figure 7 scenario is changed, SFC based algorithm perform moderately and LSF performance is degraded. EDF works consistently at high load its MDP increases. Our proposed algorithm works well and having good performance as compared to triggered transaction.

3. Deferred Coupling Mode

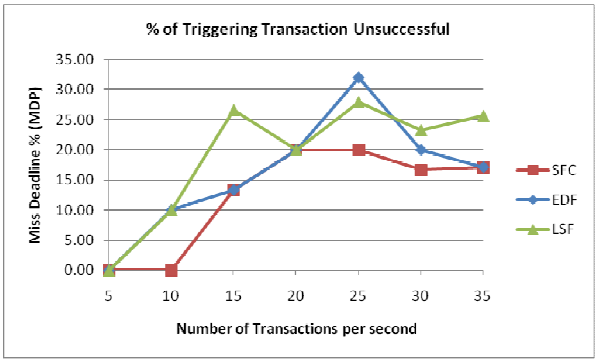


Figure 8. Impact of User/Triggering Transaction on Miss Deadline Percentage of user/triggering transaction with Deferred Coupling Mode.

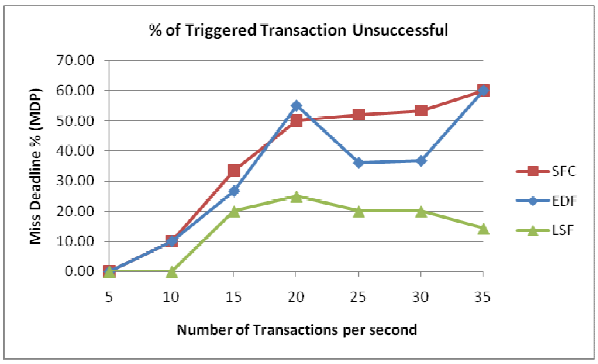


Figure 9. Impact of Triggered Transaction on Miss Deadline Percentage of triggered transaction with Deferred Coupling Mode.

In figure 8 SFC based proposed algorithm perform good, but EDF algorithm perform good at medium load and LSF performance is poor because in deferred transaction the triggering transaction executed first and action or triggered transaction will initiate after execution of triggering and LSF works or schedule according to remaining slack time.

Now in figure 9 dramatic changes are seen LSF works well in all type of triggered or action load and MDP is low as compared to other algorithms. The reason for LSF good performance is as triggering transaction finishes the execution action part or triggered transaction initiates the execution and in that case LSF schedule transaction according to remaining slack time of transaction compared to current time and also triggered transaction is having advantage of slack time get remain from triggering transaction though we are allocating remaining slack time of triggering transaction to triggered but had advantage when initiation of triggered transaction is done after completion of triggering transaction.

VIII. CONCLUSION

The area of ARTDBS has received much attention in recent times because of its vast applications. To achieve predictability and timeliness with respect to reactive behaviour in active real-time database, scheduling transaction is having great concern to improve performance. The evaluation results and comparison with different algorithms for scheduling shows that the utilization of our algorithm (SFC based table driven) achieved predictability and better success ratio with respect to different coupling modes. The SFC based table driven algorithms shows better performance as compared to other algorithms. This approach is 'scalable', 'adaptive' and, 'flexible' because it can be easily extended when new parameter is consider by increasing the dimension. Also transaction model is generalized which is flexible enough to accommodate different type, class and deadline urgency of transaction.

REFERENCES

[1] M. Berndtsson and J. Hansson, "Issues in Active Real-Time Databases", *in Proceedings First International Workshop on Active Real-Time Database Systems, (ARTDB-95)*, Skovde, Sweden, Workshops in Computing. Springer Verlag (London), pp. 142-157, June 1995.

[2] Joakim Eriksson, "Real-Time and Active Databases: A Survey", *in Proceedings Second International Workshop on Active, Real-Time, and Temporal Database Systems, (ARTDB-97)*, Como, Italy, Springer-Verlag (London), pp. 01-23, 08-09 September 1997.

[3] Jorgen Hansson and Mikael Berndtsson, "Active Real-Time Database Systems", in *Active Rules in Database Systems*, Editor Norman W. Paton: Springer, 1999, pp. 405-426.

[4] Thomas Heimrich and Gunther Specht, "Enhancing ECA Rules for Distributed Active Database Systems", *in Proceedings the Node 2002 Web and Database-Related Workshops on Web, Web-Services, and Database Systems*, Erfurt, Germany, pp. 199-205, 07-10 October 2002.

[5] Umeshwar Dayal, Meichun Hsu and Rivka Ladin, "Organizing Long-Running Activities with Triggers and Transactions", *ACM SIGMOD Record*, volume 19, issue 02, pp. 204-214, June 1990.

[6] U. Dayal, B. Blaustein, A. Buchmann, U. Chakravarthy, M. Hsu, R. Ledin, D. McCarthy, A. Rosenthal, S. Sarin, M. J. Carey, M. Livny and R. Jauhari, "The HiPAC Project: Combining Active Databases and Timing Constraints", *ACM SIGMOD Record,* volume 17, issue 01, pp. 51-70, March 1988.

[7] Kam-yiu Lam, Tony S.H. Lee and Sang H. Son, "READS: A Prototyping Environment for Real-Time Active Applications", *in Proceedings Eight International Workshop on Database and Expert Systems Applications*, Toulouse, pp. 265-270, 01-02 September 1997.

[8] J. Lee and S. H. Son, "Using Dynamic Adjustment of Serialization Order for Real-Time Database Systems", *in Proceedings of IEEE Real-Time Systems Symposium*, Raleigh Durham, NC, USA, pp. 66-75, 01-03 December 1993.

[9] Prabhudev Konana and Sudha Ram, "Transaction Management Mechanisms for Active and Real-Time Databases: A Comprehensive Protocol and a Performance Study", *Elsevier Journal of Systems and Software*, volume 42, issue 03, pp. 205-225, September 1998.

[10] Anindya Datta, Sarit Mukherjee, Igor R. Viguier, "Buffer Management in Active Real-Time Database Systems", Department of Management Information Systems, University of Arizon, Arizona, Tucson, 21 January 1996.

[11] Mohamad Ridha, Active Database Implementation For Real-Time Computing, *E-Learning Free Computer Science Community,* Kuliah Umum IlmuKomputer.Com, Indonesia, pp. 01-12, March 2005.

[12] H. Kawashima, "KRAFT: A Real-Time Active DBMS for Signal Streams", *in Proceedings of Fourth International Conference on Networked Sensing Systems (INSS 2007),* Braunschweig, pp. 163-166, 06-08 June 2007.

[13] S. Chakravarthy, B. Blaustein, A. Buchmann, M. Carey, U. Dayal, D. Goldhirsch, M. Hsu, R. Jauhari, R. Ladin, M. Livny, D. McCarthy, R. McKee and A. Rosenthal, "HiPAC: A Research Project in Active Time-Constrained Database Management'", Final Technical Report, XAIT-89-02, XAIT Reference Number 187, Xerox Advanced Information Technology, Cambridge, July 1989.

[14] S. F. Andler, J. Hansson, J. Eriksson, J. Mellin, M. Berndtsson and B. Eftring, "DeeDS Towards a Distributed and Active Real-Time Database System", *ACM SIGMOD Record*, volume 25, number 01, pp. 38-51, March 1996.

[15] Holger Branding, Alexander Buchmann, Thomas Kudrass and Jurgen Zimmermann, "Rule in an Open System: The REACH Rule Systems", in Norman W. Paton and M. Howard Williams, editors, *Rules in Database System,* Edinburgh, pp. 111-126, Springer-Verlag, 1993.

[16] J. Zimmermann, H. Branding, A. P. Buchmann, A. Deutsch and A. Geppert, "Design, Implementation and Management of Rules in an Active Database System", *in Proceedings of the 7th International Conference on Database and Expert Systems Applications,* Lecture Notes in Computer Science, Springer-Verlag, volume 1134, pp. 422-435, 1996.

[17] J. Stankovic, S. H. Son and J. Liebeherr, "BeeHive: Global Multimedia Database Support for Dependable, Real-time Applications", *in Proceedings of Second International Workshop, ARTDB97,* Lecture Notes in Computer Science, volume 1553, pp. 51–69, Springer, 1997.

[18] Suhee Kim, Sang H. Son and John A. Stankovic, "Performance Evaluation on a Real-Time Database", *in Proceedings of the Eighth*

IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'02), pp. 253-265, September 2002.

[19] R. Sivasankaran, "Design of RADEx: Real-Time Active Database Experimental System", Technical Report, University of Massachusetts, 1994.

[20] Rajendran M. Sivasankaran, John A. Stankovic, Don Towsley, Bhaskar Purimetla and Krithi Ramamritham, "Priority Assignment in Real-Time Active Databases", *The VLDB Journal - The International Journal on Very Large Data Bases*, volume 05, issue 01, pp. 19-34, January 1996.

[21] Xinjie Lu, Xin Li, Tian Yang, Zaifei Liao, Wei Liu and Hongan Wang, "QoS-Aware Publish-Subscribe Service for Real-Time Data Acquisition", *in Proceedings of Business Intelligence for the Real-Time Enterprise,* Lecture Notes in Business Information Processing, volume 27, pp. 29-44, Springer, 2009.

[22] A. Munnich, M. Birkhold, G. Farber and P. Woitschach, "Towards an Architecture for Reactive Systems Using an Active Real-Time Database and Standardized Components", *in Proceedings of the International Symposium on Database Engineering & Applications,* Montreal, Que., Canada, pp. 351-359, 02-04 August 1999.

[23] Marcus Brohede and Sten F. Andler, "Distributed Simulation Communication through an Active Real-Time Database", *in Proceedings of 27th Annual NASA Goddard Software Engineering Workshop (SEW-27'02),* Greenbelt, Maryland, pp. 147-155, 05-06 December 2002.

[24] Marcus Brohede and Sten F. Andler, "Using Distributed Active Real-Time Database Functionality in Information-Fusion Infrastructures", *in Proceedings of Real-Time in Sweden 2005 (RTiS2005),* Skovde, Sweden, 16-17 August 2005.

[25] Hauke Fuhrmann, Reinhard von Hanxleden, Christian-Albrechts, Jorn Rennhack, Jens Koch, Airbus Deutschland GmbH, "Model-Based System Design of Time-Triggered Architectures - Avionics Case Study", *in Proceedings of IEEE/AIAA 25th Digital Avionics Systems Conference,* Portland, OR, pp. 01-12, 15-19 October 2006.

[26] Aloysius K. Mok, Prabhudev Konana, Guangtian Liu, Chan-Gun Lee and Honguk Woo, "Specifying Timing Constraints and Composite Events: An Application in the Design of Electronic Brokerages", *IEEE Transactions on Software Engineering*, volume 30, issue 12, pp. 841-858, December 2004.

[27] M. Beck, Prabhudev Konana, Guangtian Liu, Yanbin Liu and Aloysius K. Mok, "Active and Real-time Functionalities for Electronic Brokerage Design", *in Proceedings of the International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems*, Santa Clara, California, pp. 30-35, 08-09 April 1999.

[28] AnnMarie Ericsson, "Verifying transformations between timed automata specifications and ECA rules", *Master's Thesis,* University of Skovde, Sweden, September 2003.

[29] Ying Qiao, Kang Zhong, HongAn Wang and Xiang Li, "Developing Event-Condition-Action rules in Real-time Active Database", *in Proceedings of the 2007 ACM symposium on Applied computing,* Seoul, Korea, pp. 511-516, 2007.

[30] Ying Qiao, Xiang Li, Hongan Wang and Kang Zhong, "Real-Time Reasoning Based on Event-Condition-Action Rules", *in Proceedings of the OTM Confederated International Workshops and Posters on, On the Move to Meaningful Internet Systems, OTM 2008 Workshops,* Lecture Notes in Computer Science, volume 5333, pp. 01-02, Springer, 2008.

[31] Ying Du, "Active Behavior in a Configurable Real-Time Database for Embedded Systems", *Master's Thesis,* Department of Computer Science, Linkoping University, Sweden, April 2006.

[32] Aindya Datta and Sang H. Son, "A Study of Concurrency Control in Real-Time, Active Database Systems", *IEEE Transactions on Knowledge and Data Engineering,* volume 14, issue 03, pp. 465-484, May-June 2002.

[33] Chenggang Zhen and Baoqiang Ren, "The Realization of Active Mechanism in Real-Time Database based on Micro-Kernel", *in Proceedings of the 2009 WRI World Congress on Software Engineering,* Xiamen, volume 01, pp. 172-176, 19-21 May 2009.

[34] Hans Sagan, "Space-Filling Curves", New York, Springer-Verlag, 1994. ISBN: 0-387-94265-3.

[35] Kam-Yiu Lam, Gary C. K. Law and Victor C. S. Lee, "Priority and Deadline Assignment to Triggered Transactions in Distributed Real-time Active Databases", *in Journal of Systems and Software, Elsevier publication,* volume 51, issue 01, pp. 49-60, April 2000.

[36] Krithi Ramamritham, Raju Sivasankaran, John A. Stankovic, Don T. Towsley and Ming Xiong, "Integrating Temporal, Real-Time, an Active Databases", *ACM SIGMOD Record*, volume 25, issue 01, pp. 08-12, March 1996.

[37] Yansheng Lu Yunsheng, Liu Qi Han Guoling and Hu Zhenyu Li, "The Architecture and Execution Model of An Active Real-time Database Management System", *in Proceedings of International Conference on Information, Communications and Signal Processing (ICICS-1997),* Singapore, 09-12 September 1997.

[38] [Lam & Lee, 1998] Kam-yiu Lam and Tony S. H. Lee, "Approaches for Scheduling of Triggered Transactions in Real-time Active Database Systems", *in Proceedings of the 24th Conference on EUROMICRO,* Vasteras, Sweden, volume 01, pp. 476-483, 25-27 August 1998.

[39] Mohamed F. Mokbel, Walid G. Aref and Ibrahim Kamel, "Performance of Multi-Dimensional Space-filling Curves", in Proceedings of the 10th ACM international symposium on Advances in Geographic Information Systems, McLean, Virginia, USA, pp. 149-154, 2002.

[40] Mohamed F. Mokbel, Walid G. Aref, Khaled Elbassioni and Ibrahim Kamel, "Scalable Multimedia Disk Scheduling", in Proceedings of the 20th International Conference on Data Engineering, pp. 498-509, 30 March-02 April 2004.

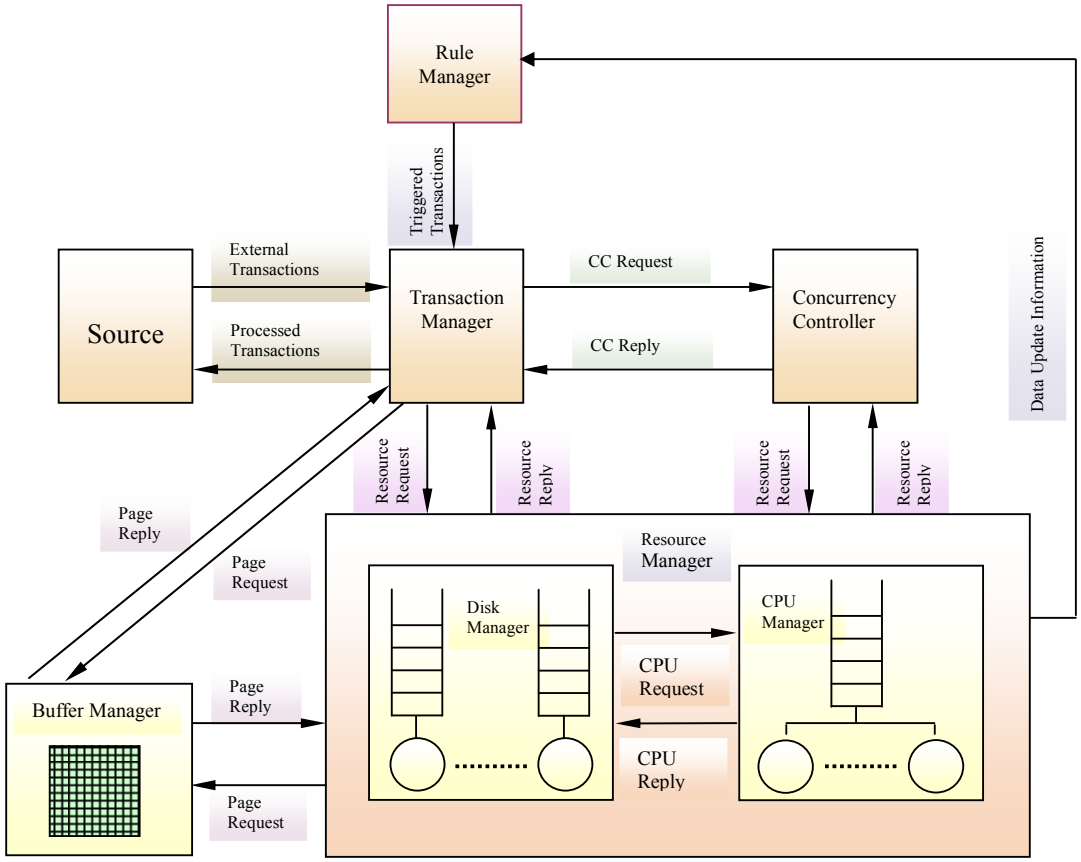
[41] M. Ahmed and S. Bokhari, "Mapping with Space Filling Surfaces", IEEE Transactions on Parallel and Distributed Systems, volume 18, issue 09, pp. 1258-1269, September 2007.

[42] M. F. Mokbel and W. G. Aref. "Irregularity in Multi-Dimensional Space-Filling Curves with Applications in Multimedia Databases", *in the Proceedings of the 10th International Conference on Information and Knowledge Management, CIKM*, Atlanta, Georgia, USA, pp. 512-519, November 2001.

[43] Guohua Jin and John Mellor-Crummey, "Space-filling Curve Generation: A Table-based Approach", *in Proceedings of the 2005 International Conference on Algorithmic Mathematics and Computer Science,* Las Vegas, Nevada, pp. 40-46, June 2005.

[44] Qiang Wang, Hong-An Wang, Hong Jin and Guo-zhong Dai, "Design and Evaluation of Priority Table Based Real-Time Scheduling Algorithms", *in Proceedings of the 2003 IEEE International Conference on Robotics Intelligent Systems and Signal Processing*, Changsha, China, volume 02, pp. 1294-1299, October 2003.

Fig. 1 An Active Real-time Database System Model.


Rule Manager
Triggered Transactions
Source
External Transactions
Processed Transactions
Transaction Manager
CC Request
CC Reply
Concurrency Controller
Data Update Information
Resource Request
Resource Reply
Resource Request
Resource Reply
Page Reply
Page Request
Resource Manager
Disk Manager
CPU Manager
CPU Request
CPU Reply
Buffer Manager
Page Reply
Page Request