Towards the Evaluation of Algorithms used in Real-Time Databases

VÁCLAV KRÓL [1], JINDŘICH ČERNOHORSKÝ [2], JAN POKORNÝ [2]

[1] Institute of Information Technologies
Silesian University in Opava, Faculty of Business Administration in Karvina
Univerzitní náměstí 1934/3, 733 40 Karviná
CZECH REPUBLIC

[2] Department of Measurement and Control
VŠB Technical University Ostrava, Faculty of Electrical Engineering and Informatics
17. listopadu 15, 708 33 Ostrava
CZECH REPUBLIC

Abstract: - There is a noticeable interest in merging standard database technology and real-time technology recently, resulted in combined systems known as Real-Time Database Systems. They are similar to the conventional databases, but they must ensure some degree of reliability regarding the real-time response requirements. It is very difficult to achieve guaranteed real-time database services in terms of both deadline miss ratio and data freshness because various components can compete for system resources. Transactions travel in a RTDB system through various components until their termination. Our objective is to design an experimental real-time database system which would be suitable enough to study the most important real-time database issues, including CPU scheduling, concurrency control and conflict resolution. Because of the strong interactions among the processed components, proposed system can help us to understand their effect on system performance and to identify the most important factors.

Key-Words: - Real-time database, transaction processing, CPU scheduling, concurrency control

1 State of the art

The data in a traditional real-time system are managed on an individual basis by every system task within structures which are fully application dependent and which were created in accordance to particular requirements. However, with the continuous advancement of technology, such systems require increasingly larger amount of information to be handled and managed in a timely manner. Thus, data can no longer be treated and managed on individual basis and an efficient data management mechanism is necessary. Traditional database management systems are designed around such a concept and that is why there is a noticeable interest in merging standard database technology and real-time technology recently. Such integration of the technologies mentioned resulted in combined systems known as Real-Time Database Systems (RTDBS).

RTDBS are similar to the conventional databases in that they encapsulate data as a resource, provide a central control of data and provide efficient algorithms for data storing, searching and manipulation. However, as being a part of real-time systems, they must ensure some degree of reliability regarding the real-time response requirements. Consequently, instead of managing data in an application-dependent manner, database systems offer the more structured data management with many necessary facilities like the elimination of redundancy, separation of data and application code, the maintenance and integrity checking.

Generally, there are two application categories where real-time database systems could be used to advantage:

1. The real-time control systems working with large amounts of data and requiring strict time response, such as air-traffic, defence systems, computer integrated manufacturing, telecommunication, etc.
2. The conventional information systems, where at least some operations have time critical deadlines, i.e. stock market and banking applications.

Each RTDB application has different system requirements depending on its domain. Many control applications involve strong time-constrained data access, whereas the data loses its validity after a certain time interval. However these operations have usually a periodical character and require only fundamental database functions. The conventional information systems involve real-time operations that usually require less time-constrained response time, have an aperiodical character and required response functions are more complex.

The field of RTDB systems is bringing together concepts from two research fields: traditional database management systems and real-time systems. During the latest years, the area of real-time databases has attracted the attention of researchers in both of them.

The most important issues in RTDB transaction processing are briefly explained in the next chapter.

2 Real time database systems issues

In contrast to a conventional database management systems where the goal usually is to minimize transaction response times, a real-time database system may be evaluated based on how often transactions miss their deadlines, the average lateness of transactions, the cost incurred in transactions missing their deadlines or data external and temporal consistency. It is very hard to meet these fundamental requirements due to potentially time-varying workloads and data access patterns. It is very difficult to achieve guaranteed real-time database services in terms of both deadline miss ratio and data freshness because various components can compete for system resources. RTDB model is presented next to outline the involved resources and to indicate the particular components of real-time transaction processing. The model is adopted from [3].

2.1 RTDB model

While tasks are considered as the schedulable unit in real-time systems, transactions are the schedulable unit in database management systems. Transactions in a RTDB system travel through various components until their termination.

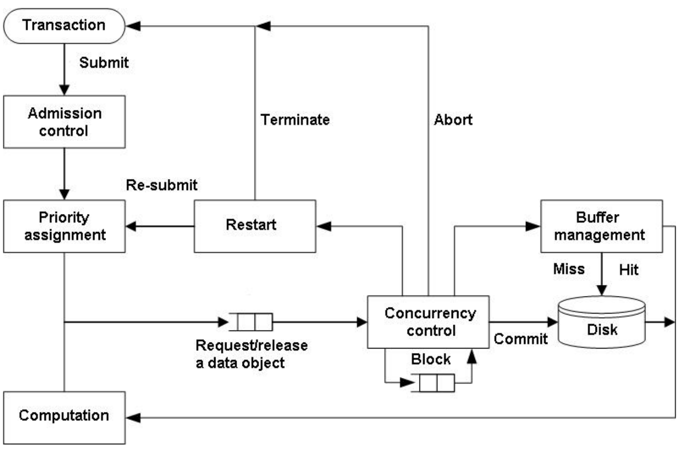

Fig. 1. RTDB model

2.1.1 Admission control

Any new transaction must pass through an admission control mechanism, which monitors and regulates the total number of concurrently active transactions within the system in order to avoid overloading. There are various approaches to control the load, standard techniques known from conventional database systems are not as effective in RTDB systems due to the inherent variation in the load. So setting the maximum number of concurrent transactions as a fixed system parameter is not an appropriate solution. More convenient seems to be some feedback control, monitoring the concurrency level and the behaviour of the system along with changing some runtime parameters. There have been also proposed some other special algorithms, for example a modified version of EDF called Adaptive-Earliest-Deadline (AED) [4], which uses a feedback mechanism in order to stabilize the overload performance of the traditional EDF policy.

2.1.2 Priority assignment

In the next step a priority level is assigned to every new or resubmitted transaction, meaning its preference relative to the other concurrent transactions within the system. Priorities can be assigned to real-time transactions by many of the same strategies used to assign priorities to real-time tasks, such as Rate Monotonic Scheduling (RMS) or Execute Deadline First (EDF) algorithm. However, transactions are less predictable than standard real-time tasks, so the priority assignment strategies using information about runtime behaviour, such as execution time or resource requirements, may not be feasible for RTDB systems. Therefore the new strategies were developed, as for example mentioned AED or AEVD[2].

Nevertheless there are some other characteristics that could be assigned to a transaction and which could influence its priority. The most important among them are deadline, criticality or amount of computing operations already spent or transaction age. Beyond these characteristics there are transaction dependent characteristics which can (or should) be taken into account when setting up priority combining thus both dominant types of influences. The frequently used combination is deadline + criticality. Also periodicity or aperiodicity of transactions are very important for determining the priority assignment mechanism. In hard RTDBS there are mostly periodic transactions which could be scheduled markedly easier than sporadic ones.

It has been shown that priority assignment policies can greatly influence the overall system's performance [5].

Remark: There are essentially three problems with which RTDB systems must deal: resolving resource contention, resolving data contention, and enforcing timing constraints [3]. So one of the basic concepts of RTDBS consists in integration of all these inevitable aspects. Due to the heterogenity of the DB resources it is necessary to modify the priority assignment algorithms in RTDB systems. In a database system, there are several components where priorities should be incorporated: CPU, memory and disk access, admission control, concurrency control and others. The studies conducted in [15] showed that regardless of which resource tends to be the system's bottleneck, priority scheduling on the critical resource must be complemented by a priority-based management policy of other system shared resources.

2.1.3 Concurrency control

In addition to resources such as the CPU and memory, DB transactions compete for access to the data stored in the database. To obtain reasonable performance, multiple transactions must be able to access data concurrently while requiring that the outcome appears as if it were the result of their serial execution. It produces a problem which is quite distinct from ordinary contention for operating system resources: contention for data. So before a transaction performs an operation on a data object, it must be processed by concurrency control component in order to achieve the required synchronization. If the transaction's request for an item is denied, the transaction must wait. The waiting transaction will be reactivated when the requested data block becomes available. Similarly, if a transaction requests an item that is currently not in main-memory, an I/O request is initiated and the transaction will wait. The waiting transaction will continue when the requested item becomes available in main-memory, and there is no active higher-priority transaction.

As opposed to conventional database systems transactions have a deadline. The another difficulty with blocking mechanisms is that a higher-priority task can be blocked by a lower priority task – a problem known as the priority-inversion problem. That is why the conventional concurrency algorithms must be slightly modified for real-time environment. The most often used strategies are:

1. Pessimistic protocols, based on locking (2PL-HP, 2PL-WP).
2. Optimistic protocols, based on validation during transaction commit (OCC-BC, OCC-WAIT).

Since the concurrency control strategies belong among the most important parts of the RTDB transaction processing, they are very often used as a subject for simulation studies. One of the main goals of the proposed experimental system will be evaluation of used algorithms.

2.1.4 Buffer management

Data buffering plays an important role in reducing transaction response time in disk resident database systems. In traditional database systems the distribution of available buffer frames among concurrent transactions and capturing transaction reference behaviors are the main concerns in buffer management. In a real-time environment, however, the buffer management may need to consider also the timing constraints imposed on referencing transactions. For example, it may be necessary to allocate available buffer frames favoring transactions with earlier deadlines.

When a transaction completes all of its operations, it commits its results and releases all of the data items in its possession. A transaction may abort/restart a number of times before it commits.

2.1.5 Operating system support

The major part of RTDB research was focused on evolution and evaluation of transaction processing algorithms, priority assignment strategies and little on concurrency control techniques. Evaluation was usually based on simulation studies. Much less attention was paid to architecture aspects of the operating systems, developed especially for real-time systems and for better support of time critical operations. Without an adequate low level support of this kind none of the scheduling algorithm can guarantee predictable transaction behaviour. No generally accepted paradigm exist for real-time operating systems and real-time database systems cooperation. It results in lower performance and worse predictability. Ideally, RTDB system blocks should be integrated with the real-time operating system kernel and runtime environment.

For unbiased evaluation of studied algorithms the experimental system will be developed for target platform VxWorks.

2.1.6 Specialized techniques

Due to highly specialized orientation of RTDB systems there are often used various alternative techniques to improve the overall system performance. If the database is not too large it can be placed into main memory while eliminating I/O operations. Memory access is much faster and more predictable than disk operations. But there is one significant drawback of possible data loss in case of power failure.

There have also been presented studies that cover some special indexing techniques for main memory databases. The most widely used and studied are so called T-trees, proposed by [14]. T-tree is an ordered index structure based on AVL tree, optimized for main-memory databases. Unlike AVL trees, T tree nodes store a range of values instead of a single value.

3. Concept reasoning

To date, research has focused primarily on finding the appropriate operational algorithms for one of the functional parts of the RTDB system. The research projects deal mostly with concurrency control mechanisms which are one of the most important blocks of a database management system. For the most part, presented research efforts are based only on simulation studies except a few exceptions ([5]). Simulations usually consist of a number of parameters. The parameters specify maximal count of data items, average count of one transaction data pages, processor time needed to manipulate one data item, average disk access time, probability of read vs. write transaction, etc. The project RADEx ([12]) is the best-known example. There is even a study where all the functional blocks are designed as object-oriented and described by means of classes with a number of attributes ([13]). However, there are two basic drawbacks of the presented research:

1. *For the most part there is only one functional part considered for investigation without any interaction* with other system parts. Because of the strong interactions among the various processing components in RTDBS, an integrated approach is necessary.
2. Research work at real-time transaction processing *is based on simulation studies only*. It is necessary to investigate the real-time transaction processing algorithms in their natural environment to achieve really relevant results. It means that the operating platform for RTDBS is a real-time operating system and the particular RTDBS functional blocks communicate with each other by means of this operating system.

3.1. Project goals

The main goal of the project is to design and implement an experimental real-time database system suitable for study of real-time transaction processing. The experimental system will be implemented as an integrated set of the most important functional parts of a veritable real-time database system. It will enable testing and performance evaluation of known and new algorithms of the particular functional parts to understand the effect of various processing components on system performance and to identify the most important influencing factors. This is the key point of the proposed experimental system.

3.1.1 Integrated approach to system construction

One of the most important aspects of our research is to consider the intended experimental real-time database system as one whole composed of integrated particular components. An integrated approach is necessary because even a single component in the system which ignores timing issues can undermine the best efforts of algorithms which do account for timing constraints. Although only the most important blocks are assumed to be created with respect to demanding implementation, this will not affect overall system behaviour because the transaction processing is mostly affected by the parts that are going to be implemented.

3.1.2 Implementation in RTOS environment

Without adequate support from the underlying subsystems, none of the scheduling algorithms can guarantee predictable transaction performance. RTDBS building blocks must be integrated with the real-time operating system kernel and other run-time environment building blocks in order to avoid wasteful duplication and provide predictable services. Intended experimental system will be built upon a real-time operating system platform VxWorks to provide reasonable results.

3.2 The fundamental idea

The system will be implemented as a centralized system with memory resident database and following parts :

<u>Transaction generator</u>

This part will be responsible for simulation of input transactions. Since it is necessary to monitor the system behaviour for various types of input transactions, the transaction generator would be able to produce periodical and aperiodical transactions or some combination of both. The generator would make use of the defined database schema.

<u>Admission control</u>

It is assumed only implementation of the algorithm „Fixed upper bound“.

<u>Priority assignment</u>

Priorities will be assigned according to deadline or according to some combination of other transaction properties (criticality + deadline). There will be implemented basic used algorithms, i.e. RM, EDF and EDF variations. The transaction priorities will be mapped to the operating system priorities.

<u>Concurrency control</u>

It is the most important database system part, so we must pay great attention to this control mechanism. At least it is assumed implementation of two variants of the pessimistic protocol 2PL and two variants of the optimistic protocol.

<u>Main memory database</u>

The database will be placed in main memory without the ability to backup or movement to another medium. We run short of traditional “data items” used in simulation studies if we want to cover database structure-access overhead. So it is assumed possibility of creating a simple database schema and transactions will approach the database with knowledge of the schema through an internal interface.

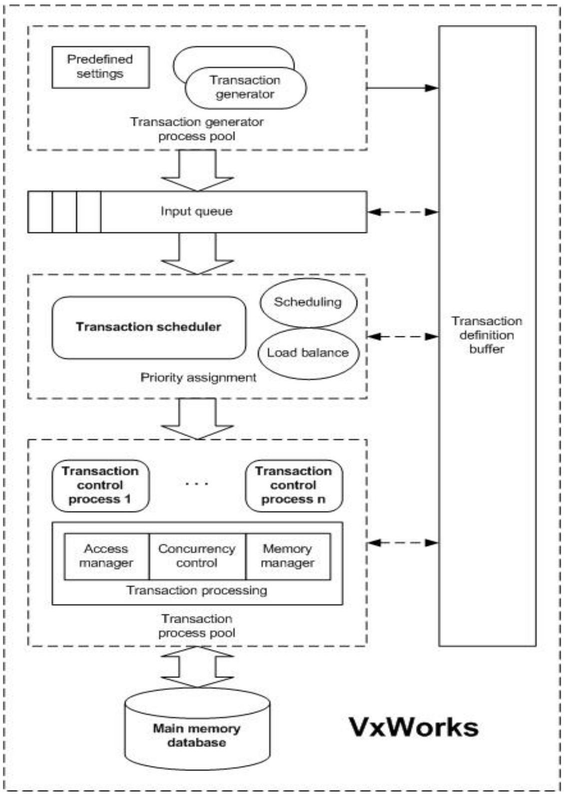

Fig. 2. Experimental RTDB system

Nevertheless the main goal of the project is not to fully implement a new professional database system. This would be unrealistic considering needed resources. Indeed this is not necessary, since the primary focus must be given only to the implementation and binding together all of those real-time database system parts essentially influencing the transactions behaviour. It is assumed that the created system will be used as the experimental rudiment for future research and for further development of advancement to study another, more complex properties of RT databases.

4. Conclusion

Real-time database systems continue to be a subject for research, there is no standard how to develop them. Relatively little research on real-time database systems is probably caused by their highly specialized orientation. Such systems are developed on an individual basis for particular applications because the standard database systems are not suitable for real-time applications. Due to continuous advancement of technology, many applications require large amounts of information to be handled and managed in a timely manner and great RTDB expansion could be expected in the future.

Acknowledgement

The work and the contribution were supported by the project AV1ET101940418 of the Academy of Sciences of the Czech Republic.

References:

[1] Kam-Yiu Lam, Tei-Wei Kuo : Real-time database systems: Architecture and Techniques. Kluwer Academic Publishers, 2001.
[2] Krishna C. M., Kang G. Shin : Real-Time Systems. McGraw-Hill, New York, 1997.
[3] Aldarmi S.A. : Real-Time Database Systems : Concepts and Design. The University of York. 1998.
[4] Haritsa J. : Earliest Deadline Scheduling for Real-Time Database Systems. Proceedings of the 12th IEEE Real-Time Systems Symposium, 1991.
[5] Huang J. : Real-Time Transaction Processing : Design, Implementation, and Performance Evaluation. PhD thesis, University of Massachusetts. 1991.
[6] Abbott R., Garcia-Molina H. : Scheduling Real-time Transactions: A Performance Evaluation, Proc. of 14th Intl. Conf. on Very Large Data Bases (VLDB). 1988.
[7] Eriksson J. : Real-Time and Active Databases : A Survey. University of Skovde. 1996.
[8] Young-Kuk Kim : Predictability and consistency in real-time transaction processing. PhD thesis. University of Virginia. 1995.
[9] Lee J. : Concurrency Control Algorithms for Real-Time Database Systems. PhD thesis. University of Virginia. 1994.
[10] Kao B., Garcia-Molina H. : An Overview of Real-Time Database Systems. Advances in Real-Time Systems. Prentice Hall, 1995, str. 463-486.
[11] Yu P., Wu K., Lin K., Son, S. H. : On Real-Time Databases: Concurrency Control and Scheduling. Proceedings of IEEE, Special Issue on Real-Time Systems. Leden 1994, str. 140-157.
[12] Sivasankaran R. : Design of RADEx : Real-Time Active Database Experimental System. University of Massachusetts. 1994.
[13] Taina J., Son S.H. : Towards a General Real-Time Database Simulator Software Library. University Of Virginia. 1999.
[14] Lehman T. J., Carey M. J. : A Study of Index Structures for Main Memory Database Management Systems. Proceedings of Twelfth International Conference on Very Large Data Bases, August 25-28, 1986, Kyoto, Japan.
[15] Carey, M.J., Jauhari R., Livny M. : Priority in DBMS Resource Scheduling, Proceedings of the Fifteenth International Conference on Very Large DataBases, pp. 397-410, 1989.
[16] Kim S., Son S.H., Stankovic J.A. : Performance Evaluation on a Real-Time Database. Proc. IEEE Real-Time Technology and Applications Symp., 2002.