Advances in Real-Time Database Systems Research

AZER BESTAVROS
(best@cs.bu.edu)

Computer Science Department
Boston University
Boston, MA 02215

1 Introduction

A Real-Time DataBase System (RTDBS) can be viewed as an amalgamation of a conventional DataBase Management System (DBMS) and a real-time system. Like a DBMS, it has to process transactions and guarantee ACID database properties. Furthermore, it has to operate in real-time, satisfying time constraints imposed on transaction commitments. A RTDBS may exist as a stand-alone system or as an embedded component in a larger multidatabase system. The publication in 1988 of a special issue of ACM SIGMOD Record on Real-Time DataBases [23] signaled the birth of the RTDBS research area—an area that brings together researchers from both the database and real-time systems communities. Today, almost eight years later, I am pleased to present in this special section of ACM SIGMOD Record a review of recent advances in RTDBS research.

There were 18 submissions to this special section, of which eight papers were selected for inclusion to provide the readers of ACM SIGMOD Record with an overview of current and future research directions within the RTDBS community. In the remainder of this paper, I will summarize these directions and provide the reader with pointers to other publications for further information.[1]

2 Concurrency Control

In [52], Ramamritham presents the real-time (or temporal) characteristics of data in a RTDBS. These characteristics may give rise to stringent timing constraints that must be satisfied when transactions are executed. These constraints are *in addition to* the logical constraints imposed by the concurrency control protocol to ensure the database consistency requirements. In [67], Lam examines the properties of RTDBS and specifies correctness criteria for different types of real-time transactions using the ACTA framework [21].

The satisfaction of both timing and logical constraints is inherently difficult due to the fact that concurrency control algorithms may introduce unpredictable delays due to transaction restarts and/or blocking. Early attempts to solve this problem have focussed on relaxing either the deadline semantics (thus suggesting best-effort mechanisms for concurrency control in the presence of *soft* [30] and *firm* [27] but not *hard* deadlines), or the transactions ACID properties (serializability in particular) [54, 45, 66]. Two instances of this latter approach are described in separate papers in this issue. In [42], Kuo and Mok overview their similarity-based concurrency control, which uses the semantic-based correctness criteria defined in [41]. In [46], Lin and Peng present another semantic-based concurrency control scheme for OO RTDBS that favors *external consistency* over serializability.

Various concurrency control algorithms differ in the time when conflicts are detected, and in the way they are resolved. Pessimistic Concurrency Control (PCC) protocols detect conflicts as soon as they occur and resolve them using *blocking*. Optimistic Concurrency Control (OCC) protocols detect conflicts at transaction commit time and resolve them using *rollbacks*.

Most real-time concurrency control schemes considered in the literature could be viewed as extensions of either PCC-based or OCC-based protocols. In particular, transactions are assigned priorities that reflect the urgency of their timing constraints. These priorities are used in conjunction with PCC-based techniques to make it possible for more urgent transactions to abort conflicting, less urgent ones (thus avoiding the hazards of blockages). Examples include the Priority Abort (PA) technique [3], Priority Inheritance (PI) technique [56], and variations of these techniques [55, 32, 7, 58, 64]. These priorities are also used in conjunction with OCC-based techniques to favor more urgent transactions when conflicting, less urgent ones attempt to validate and commit (thus avoiding the hazards of restarts). Examples include the Broadcast Commit (BC) technique [40, 27] and the Wait-50 technique [26]. Performance evaluation studies of these concurrency control techniques can be found in [3, 29, 57, 31, 1, 28, 30].

Other priority-driven real-time concurrency control protocols, which are not direct extensions of PCC or OCC, were also suggested in the literature. In [35], Kim and Srivastava studied and evaluated the potential performance improvement of using several protocols based on multiple-version two-phase locking concurrency control in RTDBS. In [47, 62], Son *et al.* propose a hybrid protocol that combines OCC and timestamp ordering. Using that protocol, the decision regarding the exact serialization order of transactions is delayed as much as possible to allow urgent transactions to commit. This is done through the dynamic allocation and adjustment of timestamp intervals [16]. In a recent study [12], Bestavros proposed the use of Speculative Concurrency Control (SCC), whereby a new dimension (namely redundancy) is exploited. By allowing a transaction to use

[1] Additional on-line information about RTDBS research can be found on the Web through the Real-Time Systems Page at "http://cs-www.bu.edu/pub/ieee-rts", and through the RTDBS Interest Group at "http://www.eng.uci.edu/ecc/rtdb/rtdb.html".

more resources, it can achieve better *speculation* and hence improve its chances for a timely commitment. Thus, the problem of incorporating transaction deadline and criticalness information into concurrency control is reduced to the problem of rationing system resources amongst competing transactions, each with a different payoff to the overall system.

Real-Time concurrency control is not a problem restricted to RTDBS data access activities. In a recent effort, summarized in a separate paper [25], Haritsa and Seshadri discuss and propose solutions to the important issues of real-time index concurrency control problem [24].

3 Resource Management and Operating System Support

The interaction between a RTDBS and its underlying operating system (OS) is another important topic of research because the correct functioning and timing behavior of RTDBS cannot be guaranteed without a thorough understanding of the impact of OS internals—including resource management in general, and scheduling in particular.

The interplay between OS and RTDBS can be best understood through implementation efforts. In a separate paper in this issue [6], Adelberg, Kao and Garcia-Molina describe their implementation of the Stanford STRIP platform. The main philosophy underlying STRIP is that soft RTDBS are likely to be part of larger open systems (*i.e.* not a monolithic stand-alone system) consisting of many heterogeneous databases. Towards that end, STRIP is designed on top of UNIX and provides support for value function scheduling and for temporal constraints on data. Son *et al.* developed a suite of database systems on several platforms, including UNIX, ARTS, and Real-Time Mach.[2] The main focus of their work has been to apply current real-time technology to architect an actual RTDBS [59]. The issues they considered included OS-RTDBS interface [36], flexible control of concurrent transactions [43], resource and data contention [44, 60], and predictable transaction execution [38]. Database security is another important issue that is often ignored in RTDBS work. In a separate paper in this issue [61], Son, David and Thuraisingham investigate the trade-offs that need to be made between security and timeliness.

The main challenge in applying real-time technology (*e.g.*, scheduling) to DBMS is that the resources needed to execute a transaction are not known *a priori*. Assuming *a priori* knowledge of transaction requirements is necessary for a *predictable system*, which in turn is necessary to meet hard deadlines. This *a priori knowledge* is the underlying assumption taken by Ulusoy and Buchmann in their efforts described in a separate paper [50] to improve timeliness by exploiting main memory DBMS features. Possessing complete knowledge of transaction requirements reduces resource management problems (*e.g.*, concurrency control, memory and buffer management) to scheduling problems. In many applications, however, the set of objects to be read (written) by a transaction may be dependent on user input (*e.g.*, in a stock market application) or dependent on sensory inputs (*e.g.*, in a process control application). In such systems, the *a priori* reservation of resources (*e.g.*, read/write locks on data objects) to guarantee a particular Worst Case Execution Time (WCET) becomes impossible—and the non-deterministic delays associated with the on-the-fly acquisition of such resources pose the real challenge of integrating scheduling into DBMS technology. This non-determinism led to a wealth of work (*e.g.*, [19]) on scheduling and resource management techniques for *best-effort systems*.

In a recent effort [14], Bestavros and Nagy proposed an admission control paradigm for RTDBS that attempts to strike a middle ground between predictable performance and best-effort performance. In their model, a transaction is *submitted* to the system as a pair of processes: a *primary task*, and *a recovery block*. The execution requirements of the primary task are *not* known *a priori*, whereas those of the recovery block are known *a priori*. Upon the submission of a transaction, an *Admission Control Mechanism* is employed to decide whether to *admit* or *reject* that transaction. Once admitted, a transaction is guaranteed to *finish* executing, either by completing its primary task (*successful commitment*) or by completing its recovery block (*safe termination*). Committed transactions bring a profit to the system, whereas terminated transactions bring *no* profit. The goal of the admission control, and scheduling protocols (*e.g.*, concurrency control, I/O scheduling, memory management) employed in the system is to *maximize* system profit. This notion of "cost consciousness" is similar to that investigated by Chakravarthy, Hong, and Johnson in [20], where a Cost Conscious Approach with Average Load Factor (CCA-ALF) is proposed and evaluated. CCA-ALF is a best-effort scheduling strategy (*i.e.* no guarantees are given) that takes into account the dynamic aspects of transaction execution (*e.g.*, system load) in addition to its static aspects (*e.g.* soft/firm deadlines) when making scheduling decisions.

Scheduling issues permeates several facets of a RTDBS. One such facet is I/O scheduling and memory management. Example work includes the development of time-cognizant variants of the traditional SCAN disk scheduling algorithm by Abbott and Garcia-Molina [5] and by Carey, Jauhari, and Livny [19], the development of time-cognizant broadcast disk organizations by Bestavros [11], the development of priority-based buffer managers by Abbott and Garcia-Molina [4] and by Kim and Srivastava [35], and the development of page replacement strategies for real-time memory managers by Carey, Jauhari, and Livny [19, 33] and by Abbott and Garcia-Molina [5]. In [51], Pang, Carey and Livny consider memory management at a higher level. They propose an admission control algorithm for real-time queries with large memory requirements, in which the multiprogramming level is related to the dynamic demand on the system's resources (memory).

4 Models and Paradigms

Two recent PhD theses have proposed novel transaction processing frameworks for RTDBS. In [39, 37], Kim establishes a RTDBS model which includes both hard and soft real-time transactions, maintains temporal and logical consistency of data [52], and supports multiple guarantee levels. Under this model, an integrated transaction processing scheme is devised, providing both predictability and consistency for RTDBS such that every application in the system is assured to achieve its own performance goal (the guarantee level) and maintain consistency requirement. A simulation study shows that higher guarantee levels require more system resources and therefore cost more than non-guaranteed trans-

[2] The ARTS and RT-Mach real-time operating systems are developed at Carnegie-Mellon [49].

actions. In [17, 13], Braoudakis takes a different approach, whereby transactions are associated with value functions that identify the nature of their timing constraints, as well as their overall importance to the system's mission. Under this framework a whole spectrum of transactions could be specified, including transactions with no timing constraints, as well as transactions with soft, firm, and hard deadlines. The novelty of this approach is that it allows transaction processing to be carried uniformly on all types of transactions. The efficacy of this approach has been demonstrated by applying it to the concurrency control problem in RTDBS. In particular, speculative concurrency control algorithms [12] were extended to work under this framework and were shown—in detailed simulation studies—to yield superior performance. The notion of transaction values and value functions [34, 48] has been utilized in both general real-time systems [15, 18] as well as in RTDBS [2, 29, 63]. In [15, 18], the value of a task is evaluated during the admission control process. The decision to reject a task or remove a previously guaranteed task is based upon tasks' values. A task that is accepted into the system is *conditionally* guaranteed to complete its execution provided that no higher valued (critical) task (with which it conflicts) arrives.

The increasing interest in Object Oriented (OO) systems has prompted a number of researchers to investigate the suitability of the OO paradigm for RTDBS. In [68], Zhou, Rundensteiner, and Shin propose ROMPP, a Real-time Object Model with Performance Polymorphism, to capture the characteristics of real-time control applications. In [65], issues of temporal and logical consistency, and precision are investigated within an OO framework.

5 Active Databases

Typically, a real-time constraint is imposed on a transaction to guarantee that the system's response to a *trigger* is committed in a timely manner. If the generation of this trigger depends on the state of the database, then the database is characterized as being both real-time and active. Application areas for Active RTDBS include automated manufacturing, air traffic control, and stock market trading, among others.

Early work on active RTDBS include Dayal *et al*'s High Performance ACtive (HiPAC) Database System project [22] and Korth *et al*'s active RTDBS paradigm [40]. Over the last few years, interest in active RTDBS has intensified as evidenced by the inaugural ARTDB'95 meeting [9], which is detailed in a separate report [10]. In this issue, two papers describing on-going projects on active RTDBS are included. The first paper [53], describes the work undertaken at the University of Massachusetts at Amherst to study the confluence of real-time constraints, temporal consistency constraints, and concurrency control and recovery constraints on Active RTDBS. In particular, they show that exploiting the characteristics of data for transaction processing, placing the data at the appropriate level of the memory hierarchy, and performing appropriate logging and recovery for each type of data is crucial to achieve high performance in RTDBS. The second paper [8], describes DeeDS—a research prototype under development at the University of Skövde in Sweden. Through the use of *lazy replication*, *main memory residency*, and contingency plans, DeeDS boosts the predictability of distributed active RTDBS.

Aknowledgments:

I would like to thank all the authors who submitted contributions to this special section of SIGMOD Record. Also, I would like to thank Jennifer Widom for all her help in fleshing out this idea of a special section dedicated to RTDBS. This work was partially supported by NSF (grant CCR-9308344).

References

[1] R. Abbott and H. Garcia-Molina. Scheduling real-time transaction: A performance evaluation. *ACM Transactions on Database Systems*, 17(3):513–560, September 1992.

[2] Robert Abbott and Hector Garcia-Molina. Scheduling real-time transactions. *ACM, SIGMOD Record*, 17(1):71–81, 1988.

[3] Robert Abbott and Hector Garcia-Molina. Scheduling real-time transactions: A performance evaluation. In *Proceedings of the 14th International Conference on Very Large Data Bases*, pages 1–12, Los Angeles, Ca, 1988.

[4] Robert Abbott and Hector Garcia-Molina. Scheduling real-time transactions with disk resident data. In *Proceedings of the 15th International Conference on Very Large Data Bases*, Amsterdam, Holland, 1989.

[5] Robert Abbott and Hector Garcia-Molina. Scheduling I/O requests with deadlines: A performance evaluation. In *Proceedings of the 11th Real-Time Systems Symposium*, December 1990.

[6] Brad Adelberg, Ben Kao, and Hector Garcia-Molina. Overview of the STanford Real-time Information Processor (STRIP). *ACM SIGMOD Record*, 25(1), March 1996.

[7] D. Agrawal, A. El Abbadi, and R. Jeffers. Using delayed commitment in locking protocols for real-time databases. In *Proceedings of the 1992 ACM SIGMOD International Conference on Management of Data*, San Diego, Ca, 1992.

[8] S. Andler, M. Berndtsson, B. Eftring, J. Eriksson, J. Hansson, and J. Mellin. DeeDS towards a distributed and active real-time database system. *ACM SIGMOD Record*, 25(1), March 1996.

[9] Mikael Berndtsson and Jorgen Hansson, editors. *Proceedings of ARTDB-95: The 1st International Workshop on Active Real-Time Database Systems.* Workshops in Computing. Springer Verlag, London, June 1995.

[10] Mikael Berndtsson and Jorgen Hansson. ARTDB-95 Workshop Report. *ACM SIGMOD Record*, 25(1), March 1996.

[11] Azer Bestavros. AIDA-based real-time fault-tolerant broadcast disks. Technical Report TR-96-001, Boston University, CS Dept, Boston, MA 02215, January 1996.

[12] Azer Bestavros and Spyridon Braoudakis. Timeliness via speculation for real-time databases. In *Proceedings of RTSS'94: The 14^{th} IEEE Real-Time System Symposium*, San Juan, Puerto Rico, December 1994.

[13] Azer Bestavros and Spyridon Braoudakis. Value-cognizant speculative concurrency control. In *Proceedings of VLDB'95: The International Conference on Very Large Databases*, Zuiich, Switzerland, Spetember 1995.

[14] Azer Bestavros and Sue Nagy. An admission control paradigm for real-time databases. Technical Report BUCS-TR-96-002, Boston University, Computer Science Department, January 1996.

[15] Sara Biyabani, John Stankovic, and Krithi Ramamritham. The integration of deadline and criticalness in hard real-time scheduling. In *Proceedings of the 9th Real-Time Systems Symposium*, December 1988.

[16] C. Boksenbaum, M. Cart, J. Ferrié, and J. Francois. Concurrent certifications by intervals of timestamps in distributed database systems. *IEEE Transactions on Software Engineering*, pages 409–419, April 1987.

[17] Spyridon Braoudakis. *Concurrency Control Protocols for Real-Time Databases.* PhD thesis, Computer Science Department, Boston University, Boston, MA 02215, expected June 1994.

[18] G. Buttazzo, M. Spuri, and F. Sensini. Value vs. deadline scheduling in overload conditions. In *Proceedings of the 16th Real-Time Systems Symposium*, December 1995.

[19] M. J. Carey, R. Jauhari, and M. Livny. Priority in DBMS resource scheduling. In *Proceedings of the 15th International Conference on Very Large Data Bases*, pages 397–410, 1989.

[20] S. Chakravarthy, D. Hong, and T. Johnson. Incorporating load factor into the scheduling of soft real-time transactions. Technical Report TR94-024, University of Florida, Department of Computer and Information Science, 1994.

[21] Panos Chrysanthis and Krithi Ramamritham. Acta: A framework for specifying and reasoning about transaction structure and behavior. In *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, pages 194–203, May 1990.

[22] U. Dayal, B. Blaustein, A. Buchmann, U. Chakravarthy, M. Hsu, R. Ledin, D. McCarthy, A. Rosenthal, S. Sarin, M. Carey, M. Livny, and R. Jauhari. The HiPAC project: Combining active database and timing constraints. *ACM SIGMOD Record*, 17(1):51–70, 1988.

[23] Sang Son (ed.). Acm sigmod record: Special issue on real-time databases, 1988.

[24] B. Goyal, J. Haritsa, S. Seshadri, and V. Srinivasan. Index concurrency control in firm real-time dbms. In *Proceedings of the 21st VLDB Conference*, pages 146–157, September 1995.

[25] J. Haritsa and S. Seshadri. Real-time index concurrency control. *ACM SIGMOD Record*, 25(1), March 1996.

[26] Jayant R. Haritsa, Michael J. Carey, and Miron Livny. Dynamic real-time optimistic concurrency control. In *Proceedings of the 11th Real-Time Systems Symposium*, December 1990.

[27] Jayant R. Haritsa, Michael J. Carey, and Miron Livny. On being optimistic about real-time constraints. In *Proceedings of the 1990 ACM PODS Symposium*, April 1990.

[28] Jayant R. Haritsa, Michael J. Carey, and Miron Livny. Data access scehduling in firm real-time database systems. *The Journal of Real-Time Systems*, 4:203–241, 1992.

[29] J. Huang, J. A. Stankovic, D. Towsley, and K. Ramamritham. Experimental evaluation of real-time transaction processing. In *Proceedings of the 10th Real-Time Systems Symposium*, December 1989.

[30] Jiandong Huang, John A. Stankovic, Krithi Ramamritham, and Don Towsley. Priority inheritance in soft real-time databases. *The Journal of Real-Time Systems*, 4:243–268, 1992.

[31] Jiandong Huang, John A. Stankovic, Krithi Ramamritham, and Don Towslwy. Experimental evaluation of real-time optimistic concurrency control schemes. In *Proceedings of the 17th International Conference on Very Large Data Bases*, Barcelona, Spain, September 1991.

[32] Jiandong Huang, John A. Stankovic, Krithi Ramamritham, and Don Towslwy. On using priority inheritance in real-time databases. In *Proceedings of the 12th Real-Time Systems Symposium*, December 1991.

[33] R. Jauhari, M. J. Carey, and M. Livny. Priority-Hints: An algorithm for priority-based buffer management. In *Proceedings of the 16th International Conference on Very Large Data Bases*, pages 708–721, 1990.

[34] E.D. Jensen, C.D. Locke, and J. Tokuda. A time-driven scheduling model for real-time operating systems. In *Proceedings of the 6th Real-Time Systems Symposium*, pages 112–122, December 1985.

[35] Woosaeng Kim and Jaideep Srivastava. Enhancing real-time DBMS performance with multiversion data and priority based disk scheduling. In *Proceedings of the 12th Real-Time Systems Symposium*, December 1991.

[36] Y. Kim, M. Lehr, D. George, and S. H. Son. A database server for distributed real-time systems: Issues and experiences. In *Proceedings of the IEEE Workshop on Parallel and Distributed Real-Time Systems*, pages 66–75, Cancun, Mexico, April 1994.

[37] Y. Kim and S. H. Son. An approach towards predictable real-time transaction processing. In *Proceedings of the 5th Euromicro Workshop on Real-Time Systems*, pages 70–75, Oulu, Finland, June 1993.

[38] Y. Kim and S. H. Son. Predictability and consistency in real-time database systems. In S. H. Son, editor, *Advances in Real-Time Systems*, pages 509–531. Prentice Hall, New York, New York, 1995.

[39] Young-Kuk Kim. *Predictability and Consistency in Real-Time Transaction Processing.* PhD thesis, Department of Computer Science, University of Virginia, May 1995.

[40] Henry Korth. Triggered real-time databases with consistency constraints. In *Proceedings of the 16th International Conference on Very Large Data Bases*, Brisbane, Australia, 1990.

[41] Tei-Wei Kuo and Aloysius Mok. Application semantics and concurrency control for real-time data-intensive applications. In *Proceedings of the IEEE 13th Real-Time Systems Symposium*, 1992.

[42] Tei-Wei Kuo and Aloysius Mok. Real-time database—similarity semantics and resource scheduling. *ACM SIGMOD Record*, 25(1), March 1996.

[43] J. Lee and S. H. Son. Using dynamic adjustment of serialization order for real-time database systems. In *Proceedings of the 14th IEEE Real-Time Systems Symposium*, pages 66-75, Raleigh-Durham, North Carolina, December 1993.

[44] M. Lehr, Y. Kim, and S. H. Son. Managing contention and timing constraints in a real-time database system. In *Proceedings of the 16th IEEE Real-Time Systems Symposium*, Pisa, Italy, December 1995.

[45] Kwei-Jay Lin. Consistency issues in real-time database systems. In *Proceedings of the 22nd Annual Hawaii International Conference on System Science*, pages 654–661, 1989.

[46] Kwei-Jay Lin and Ching-Shan Peng. Enhancing external consistency in real-time transactions. *ACM SIGMOD Record*, 25(1), March 1996.

[47] Yi Lin and Sang Son. Concurrency control in real-time databases by dynamic adjustment of serialization order. In *Proceedings of the 11th Real-Time Systems Symposium*, December 1990.

[48] C. Locke. *Best Effort Decision Making for Real-Time Scheduling*. PhD thesis, Carnegie-Mellon University, Department of Computer Science, May 1986.

[49] Clifford W. Mercer and Ragunathan Rajkumar. Interactive interface and RT-Mach support for monitoring and controlling resource management. In *In Proceedings of the Real-Time Technology and Applications Symposium*, May 1995.

[50] Özgür Ulusoy and Alejandro Buchmann. Exploiting main memory dbms features to improve real-time concurrency protocols. *ACM SIGMOD Record*, 25(1), March 1996.

[51] H. Pang, M. J. Carey, and M. Livny. Managing memory for real-time queries. In *Proceedings of the 1994 ACM SIGMOD Conference on Management of Data*, pages 221–232, 1994.

[52] Krithi Ramamritham. Real-time databases. *International journal of Distributed and Parallel Databases*, 1(2), 1993.

[53] Krithi Ramamritham. Integrating temporal, real-time, and active databases. *ACM SIGMOD Record*, 25(1), March 1996.

[54] L. Sha, J.P. Lehocsky, and E.D. Jensen. Modular concurrency control and failure recovery. *IEEE Transactions on Computers*, 37(2):146–159, February 1988.

[55] L. Sha, R. Rajkumar, and J. P. Lehoczky. Priority inheritance protocols: An approach to real-time synchronization. *IEEE Transactions on Computers*, 39(9):1175–1185, 9 1990.

[56] Lui Sha, R. Rajkumar, and J. Lehoczky. Concurrency control for distributed real-time databases. *ACM, SIGMOD Record*, 17(1):82–98, 1988.

[57] Lui Sha, R. Rajkumar, Sang Son, and Chun-Hyon Chang. A real-time locking protocol. *IEEE Transactions on Computers*, 40(7):793–800, 1991.

[58] S. Son, S. Park, and Y. Lin. An integrated real-time locking protocol. In *Proceedings of the IEEE International Conference on Data Engineering*, Tempe, AZ, February 1992.

[59] S. H. Son, Y. Kim, and R. Beckinger. Mrdb: A multi-user real-time database testbed. In *27st Hawaii International Conference on System Sciences*, pages 543–552, Maui, Hawaii, January 1994.

[60] S. H. Son and S. Koloumbis. A token-based synchronization scheme for distributed real-time databases. *Information Systems*, 18(6):375–389, December 1993.

[61] Sang Son, Rasikan David, and Bhavani Thuraisingham. Improving timeliness in real-time secure database systems. *ACM SIGMOD Record*, 25(1), March 1996.

[62] Sang H. Son, Juhnyoung Lee, and Yi Lin. Hybrid protocols using dynamic adjustment of serialization order for real-time concurrency control. *The Journal of Real-Time Systems*, 4:269–276, 1992.

[63] John Stankovic and Wei Zhao. On real-time transactions. *ACM, SIGMOD Record*, 17(1):4–18, 1988.

[64] O. Ulusoy and G. G. Belford. Real-time transaction scheduling in database systems. *Information Systems*, 18(8), December 1993.

[65] V. Fay Wolfe, L. Cingiser DiPippo, and J. K. Black. Supporting concurrency, timing constraints and imprecision in objects. Technical Report TR94-230, University of Rhode Island, Computer Science Department, December 1994.

[66] Kun-Lung Wu, P. Yu, and Calton Pu. Divergence control for Epsilon-Serializability. In *IEEE Eighth International Conference on Data Engineering*, Tempe, Arizona, February 3-7 1992.

[67] Kam yiu Lam. Specifying the correctness criteria for processing of real-time transactions. Technical Report TR-9601, Department of Computer Science, City University of Hong Kong, Hong Kong, 1996.

[68] L. Zhou, E.A. Rundensteiner, and K.G. Shin. OODB support for real-time open-architecture controllers. In *Proceedings of the Fourth International Conference on Database Systems for Advanced Applications (DASFAA'95)*, pages 206–213, Singapore, April 1995.