Report on DART '96:
Databases: Active and Real-Time (Concepts meet Practice)

Krithi Ramamritham
Department of Computer Science
Univ. of Massachusetts, Amherst 01003, U.S.A.
`krithi@cs.umass.edu`

Nandit Soparkar
The University of Michigan
Ann Arbor MI 48109
`soparkar@eecs.umich.edu`

DART '96 was held in conjunction with the Conference on Information and Knowledge Management (CIKM) on Nov 15th in Baltimore. Its goal was to provide a forum for researchers and practitioners involved in integrating concepts and technologies from active and real-time databases to discuss the state of the art and chart a course of action. To this end, nine speakers from academia, industry, and research laboratories were invited to provide a perspective on the theory and practice underlying active real-time databases. In addition, some selected papers were presented briefly to complement the invited speakers' talks. The second half of the workshop was devoted to discussions aimed at identifying the problems that still need to be addressed in the contexts of the diverse target applications.

The advantages of incorporating databases into real-time applications stem from the support provided within databases for data management, persistence, and distribution of data, querying and transaction management. These are important for real-time applications involving significant amounts of data, with needs to manage temporal data, or achieve predictability. Often, such data (e.g., data from sensors) may lose their validity after a length of time. The need to ensure predictability calls for efficient support for overload management and for processing queries and transactions under time constraints.

To stimulate discussions, four types of real-time database applications were considered: telecommunications, avionics / air traffic control, manufacturing automation, and internet-based applications. The major characteristics of these applications, as discussed from the perspective of the Workshop, are presented below.

In telecommunications applications, real-time accesses to databases currently occur mainly during 800-number translations. The associated queries, at present, are simple and involve fetching a record from the database given a key (the 800 number). The end-to-end delay for connection establishment is in the order of 50 msec, and with required simple disk access, this responsiveness is guaranteed by analyzing the code to ensure that it takes no more that 50 msec to complete. By placing the needed data in main memory this delay can be brought down to 10 msec. Also, the telecommunications applications have strict availability requirements (e.g., less than one call in a million may be lost), and this is usually achieved through the replication of resources. In some aspects of the application, due to the heavy update volume, with logging overheads consuming a large proportion of the update costs, novel methods (e.g., writing to the memory of a hot standby) are used. While the overall approach works well currently, with the advent of "intelligent" networks, this application as a whole is likely to get quite complex.

In typical avionics applications, the system keeps track of 3000 objects, and this data must be temporally coherent with a deviation (from the actual state of the environment) of no more than 200 msec. A maximum read/write response time of 1 sec is expected from the database. Some onboard systems also have inherent issues of robustness and manageable physical size. In contrast, an air traffic control application deals with 20000 objects, and the timing requirements are not as stringent. Temporal coherency of 3 to 6 sec with 5 millisec for read/write response times are acceptable. Also, in these applications, the associated data has an inherent period of validity.

In manufacturing applications, to effectively support real-time interactions between different physical components, a distributed real-time database (preferably with multimedia capabilities) is indicated. There is need for the management and retrieval of historical information as well as state information under timing constraints. The manufacturing control systems by necessity use real-time data, and some of the archived and derived information is used to make long-term decisions. Furthermore, a combination of the real-time data and the stored data may be used to alert operators to maintenance needs or abnormal fault-like situations. The information is maintained either as a single stored entity or distributed across multiple logical information entities.

The commercial uses of the internet/intranet environments typically involve interactions with different parts of an enterprise. Generally, it is advantageous to have an accurate and temporally coherent view of the distributed state. Furthermore, the users' view should be up-to-date since the users are cognizant of their current dealings. A similar situation arises when OLAP analysts interact with a real-time database. In internet applications, predictability refers to the capability of assessing how long a task is likely to take given current system conditions. This may arise due to pricing policies – e.g., if a system indicates how long it will take to respond to a request given different acceptable qualities of service (QoS), users may choose appropriate ones depending on their needs.

These applications suggest the need for the following capabilities:

- Modified consistency models are needed to address the temporal consistency requirements, and also, temporal data require new concepts for persistence. Admission control policies for real-time databases are necessary. This would prevent the admission of tasks that cannot be completed due to system or time resource constraints.

- New and more relevant metrics are required for assessing performance for active and real-time database applications. Metrics such as missed deadlines, as used currently, have limited practical utility. There is a need to systematically derive timing requirements arising from the applications. Currently, deadlines or other performance criteria, are assigned in a largely *ad hoc* manner.

- While implemented applications attempt to avoid concurrency control issues using special-purpose solutions, a large number of general solutions have been proposed in the research literature. It would be useful to re-examine traditional transaction processing in the context of the target applications, and to develop concurrency control techniques that can be used selectively.

- It is important to categorize data, such as main memory data vs. disk resident data, real-time data vs. non real-time data, etc. – and be cognizant of how these arise from the application requirements. With respect to the various system components, there is a need for predictable access and execution costs coupled with low overheads.

- Practical economic factors, deployment environments, and projected needs indicate that the developed services be amenable to commercial off-the-shelf systems, and be scalable and interoperable. This implies that the technology must be able to co-exist and meld with available and future non-active or non-real-time data management systems as well.

Given the monolithic structure of off-the-shelf databases, real-time applications have often resorted to developing special-purpose database support. That is, by requiring transactions to be short, with known execution times and resource needs, and placing temporal data in main memory, many applications are made to meet their predictability and performance requirements. While this approach could be justified for tasks and transactions with very tight deadlines, it is not cost-effective in general. This implies that real-time applications have not yet benefited from the advantages provided by database technology. An example of the few exceptions is SmallBase, the general-purpose main-memory database system from HP Labs, which has been used in a telecommunications application. The performance obtained was far superior to typical commercial databases.

A recurrent theme in the workshop was the need for *unbundling* the services provided by traditional databases so that tailored database support can be provided – by mixing and matching needed mechanisms – depending on the requirements of the applications. For a database to be useful in many different real-time application scenarios, it is necessary that the following services be unbundled: query processing, transaction processing, schedulers and overload managers, meta-data services, event manager, admission control, rule engine, buffer manager, and disk scheduler. Unbundling these services, while allowing for rebundling to produce efficient database support, form important research problems. Ideally, the result would provide a menu of choices allowing application developers to mix and match services identified early in the development process of building the database support for specific applications. Such a building-block based approach should help avoid paying the price for functionality not needed or not used.

The services provided by an active real-time database must be cognizant of important characteristics of the target applications. These include the application-specific QoS and timeliness needs; the ability to tradeoff quality (i.e., the consistency, coherency, completeness, and currency) for timeliness; the cost considerations and atypical performance characteristics (e.g., in many real-time applications the time taken for the first response is important); the possibility of relaxing the traditionally provided ACID properties of transactions; and the associated temporal consistency, coherency, and currency properties of data. The services offered must also account for market forces and user requirements that demand that the services be built atop commercial off-the-shelf operating systems.

There are several other issues related to the practical domains where these systems may be deployed. Security is an issue that will assume growing importance with internet-based applications. The performance penalty due to the provision of security becomes a consideration. The flexibility provided by triggers must be weighed against the unpredictability introduced by triggered transactions. Finally, to ease the interaction with these systems, real-time cognizant GUIs for monitoring and control are needed as well.

Details regarding the workshop as well as transparencies of the invited presentations can be found at the following web site:

`http://www.eecs.umich.edu/~soparkar/Dart96`