Naval Command,
Control and Ocean
Surveillance Center RDT&E Division San Diego, CA
92152-5001

Technical Document 2747
February 1995

Final Report of the DBSSG Predictable Real-Time Information Systems Task Group

D. K. Fisher
NRaD
San Diego, CA 92152–5001
Dfisher@cod.NOSC.mil

P. J. Fortier
University of Massachusetts Dartmouth
North Dartmouth, MA 02747–2300
Pfortier@umassd.edu

D. K. Hughes
DBx Inc.
Cherry Hill, NJ 08002–0446
DBx@world.std.com

M. Roark
Martin Marietta
Syracuse, NY 13221–4840
Roark@gw.syr.ge.com

Technical Document 2747
February 1995

Final Report of the DBSSG Predictable Real-Time Information Systems Task Group

D. K. Fisher
NRaD
San Diego, CA 92152–5001
Dfisher@cod.NOSC.mil

P. J. Fortier
University of Massachusetts Dartmouth
North Dartmouth, MA 02747–2300
Pfortier@umassd.edu

D. K. Hughes
DBx Inc.
Cherry Hill, NJ 08002–0446
DBx@world.std.com

M. Roark
Martin Marietta
Syracuse, NY 13221–4840
Roark@gw.syr.ge.com

NAVAL COMMAND, CONTROL AND OCEAN SURVEILLANCE CENTER
RDT&E DIVISION
San Diego, California 92152–5001

K. E. EVANS, CAPT, USN
Commanding Officer

R. T. SHEARER
Executive Director

ADMINISTRATIVE INFORMATION

This work was performed by the Advanced Concepts and Systems Technology Division, Code 41, of the Naval Command, Control and Ocean Surveillance Center, RDT&E Division, San Diego, California 92152–5001, during the period of FY 92 – FY 95. The work was funded by the Space and Naval Warfare Systems Command, program element 0604574N, project number X0706.

Released by
M. B. Vineberg, Head
Deputy for Business

Under authority of
R. B. Volker, Head
Advanced Concepts and Systems
Technology Division

ACKNOWLEDGMENTS

This technical document was compiled through the contributions of Dr. Paul Fortier, University of Massachusetts Dartmouth; David K. Hughes, DBx INC; Mayford Roark, Martin Marietta; Christy Selvester, NRaD Code 4122, and Lisa J. Payne, NRaD Code 4101.

Contents

1 Introduction and Goals of the PRIS-TG

The Predictable Real-time Information Systems Task Group (PRIS-TG) is a task group of the X3 (Computers and Information Processing Committee) Database Systems Study Group (DBSSG). PRISTG's relationship to the X3 standards organization is described below. The purpose of the PRIS-TG is to determine if the technology for real-time systems in general and for real-time information management is sufficiently mature for standardization.

The PRIS-TG was tasked to develop a Predictable Real-time Information Systems Reference Model, evaluate existing Predictable Real-time Information Systems technology, and determine the need for standardization in this area.

The objective of the PRIS-TG study was to establish a framework for future predictable real-time information management standards activities, both extensions to ongoing SQL (Structured Query Language), IRDS (Information Resources Dictionary System), RDA (Remote Data Access), ODP (Open Distributed Processing), OIM (Object Information Management), POSIX (Portable Operating Systems Interface for Computer Environments), Ada, and computer security development, as well as related future standards.

The task group reviewed and evaluated existing and developmental products claiming to be real-time information management systems, as well as, real-time products with information management capabilities. The task group also reviewed and evaluated published literature and research activities concerning real-time information management technology world wide. In this report the PRIS-TG will discuss the recommendations for standardization in the area of real-time information management technology.

1.1 Overview of Document

Beyond this introduction section, this document is broken up into three additional sections. The section 2 indicates the issues with conventional database technology when applied to real-time systems. Section 3 outlines PRIS-TG findings on the state of real-time database technology and which technologies need to be addressed to realize real-time database management systems and standards in particular. Section 4 indicates PRIS-TG recommendations for X3 work towards real-time database management systems standards. Appendix A shows the schedule followed by PRIS-TG. Appendix B lists some annotated references used in this document. Appendix C lists the members of the PRIS-TG. Appendix D highlights the findings from an industry survey on real-time databases. Appendix E contains the PRIS-TG real-time database management reference model.

To highlight the needs and show where work must progress, an annotated bibliography is included on papers that have been generated by PRIS-TG members and other organizations on real-time database topics of interest. These papers include works on: flexible transaction structure and specifications, real-time structured query language concepts, semantic database management systems models, other ongoing standards activities in real-time database systems, the operating system and real-time database man-

agement interface issues and real-time scheduling and evaluations. These represent just a sampling of ongoing efforts in real-time database management.

1.2 Terminology Used

The PRIS-TG uses the following terms and meanings within this document.

- Data item - the smallest separable unit recognized by the database representing a real-world entity.
- Database - a collection of data items which have constraints, relationships and a schema.
- Schema - a description of how data, relationships and constraints are organized for user application program access.
- Constraint - a predicate that defines all correct states of the database.
- Transaction - a partially ordered sequence of database operations that represent a logical unit of work and which access a shared database.
- Transaction Properties - ACID properties are defined as:
  - Atomicity - either all the actions of a transaction occur successfully or the transaction is nullified by rolling back all updates.
  - Consistency - a transaction moves an object ... from one valid state to another valid state, and if the transaction is aborted, the object is returned to its previous valid state.
  - Isolation - the actions carried out by a transaction against a shared database cannot become visible to other transactions until the transaction commits.
  - Durability - once a transaction completes successfully, its effects cannot be altered without running a compensating transaction. The changes made by a successful transaction survive subsequent failures of the system.
- Relationship - a correspondence among two or more data items.
- Features of data items in a database
  - Shared - data items in a database are shared among several users and applications programs.
  - Persistent - data items in a database exists beyond the scope of the process that created it, the data exists permanently.
  - Security - data items in a database are protected from unauthorized disclosure, alteration or destruction.

  - Validity - also referred to as data integrity or correctness. Data items in a database should be correct with respect to the real-world entity that it represents.
  - Consistency - whenever more than one data item in a database represents related real-world values, the values should be consistent with respect to a defined relationship.
  - Non-redundancy - no two data items in a database represent the same real-world entity.
  - Independence - data items in the database should exhibit physical data independence and logical data independence.

- Concurrency control - the activity of coordinating actions of concurrently executing transactions so that the correctness of the database is maintained and transaction properties are not violated.
- Recovery - the activity of the database management system that provides restoration of the database when transactions fail. The affects of aborted or failed transactions must be isolated to only failed transactions no others should be affected.
- Real-Time - that area of computer systems design and implementation in which the correctness of a result is dependent on the validity and timeliness of data and of the operations on that data.
- Real-Time Database Management System - a database management system which is capable of managing time-constrained queries. Such a DBMS may form the foundation in support of time-constrained transactions.
- Real-Time Transaction - a complex query made up of more than one action that must be performed within a given period of time. The transaction must be completed or aborted as a "single unit".

2 Real-Time DBMS Problem Statement

Real-time information processing requirements are found in a wide spectrum of applications, from small embedded systems to large transaction oriented systems. The common factors amongst these systems is the need for determinism and high speed in information accesses and manipulations. The need for *real-time database management systems* is becoming more evident as large data intensive projects such as the National Information Infrastructure (NII, or information highway), are being developed. NII applications include electronic commerce, digital libraries, advanced manufacturing, environmental monitoring and health care. Beyond NII applications other real-time database application areas include telephone, energy management, automated factories, airplane information management, defense oriented systems, prison management, cable television, medical and financial management. These applications require application

derived data structures, high data availability and time constrained access to data that may also have time constraints on data consistency and correctness. Requirements for real-time database management systems encompass features for database and transaction structure, transaction execution policies and architectural interaction requirements with the system's infrastructure [16].

Applications such as those above require a different model of database structure and database processing than that found in conventional database management systems. Adherence to the conventional database model correctness criteria based on transactions executing to preserve *ACIDity*[1] properties must be altered if real-time service is to be provided. Real-time does not simply imply *FAST*, but timeliness and predictability in all aspects of transaction and database operations. The need for new solutions for real-time data storage and processing has spurred real-time database systems research. In recent years, significant research has been conducted to develop real-time database models [27, 28], real-time transaction scheduling [2, 11], real-time concurrency control [6, 38], real-time transaction structuring [10, 4] and real-time database recovery [13].

Presently there is a trend in industry to develop products that conform to open systems standards. Database management systems are no exception, though standards in this technology area are relatively new and still evolving. Presently there are real-time database products available (DBx inc, Martin Marrietta, Westinghouse). These products do not yet address all the needs of real-time database applications, though they are a step in the right direction. These products however, do not conform to real-time database standards as none exist. These products have concepts for time driven transactions, time constraints on data and some architectural dependency considerations. Additionally they interact with the hardware systems through real-time open system operating systems standards such as POSIX 1003.4 [3].

3 Findings of the PRIS-TG

The Predictable Real-time Systems task group (PRIS-TG) performed a study to determine the maturity and viability of real-time database management systems to include such technology in existing and emerging ANSI standards. The findings highlight the lack of support for real-time in conventional database systems, the need for such support, the maturity of real-time database technology, the availability of products and supporting standards. Presented are synopsis for recommended real-time concepts for database management systems standardization.

3.1 Real-time Database Systems Technology

Nearly all existing database standards and existing products do not have support for real-time service to applications. Conventional SQL products are based on the model of a monolithic database (persistent storage) with serialization of transactions acting on

[1]Transaction ACID properties guarantee the Atomic, Consistent, Issolated and Durable execution of transactions on the database which a database manager maintains in a consistent and correct state.

the database as the correctness criteria. Conventional database recovery is based on the check pointing of the database with the application of log actions to redo or undo the effects of transactions on the database at the time of a failure.

While conventional database management implementations support a wide range of application areas, it has been shown by numerous researchers, product developers and vendors that this model limits the ability of databases to be applied to a wider array of computer based information processing and management applications such as those for real-time computing systems. Numerous examples of how real-time database management could improve performance within the computer aided design, computer aided manufacturing, medical monitoring and diagnostics, department of defense mission critical systems, on-line transaction processing systems and numerous other applications areas have been defined in the literature [18].

Work has progressed beyond the research phase into testbed systems and currently into available off the shelf products that exhibit features necessary for the support of real-time in the database environment. These products and research address the need to extend conventional database concepts to include time as a component for database consistency, correctness and manipulation and to address the requirement for predictability of access. In the database specification area the move is to limit the flexibility of on-line alteration of database structures to deliver predictable access to the database while giving more control to the database designer to limit how the database can be used, where it will be stored, how it will be structured in storage, how the database is partitioned and how constraints affect correctness and consistency.

Research, prototyping efforts and product developments address the need of the real-time programmer to exhibit more control over the specification, structure, access, manipulation and recovery of the database and transactions. The focus of these efforts is on the loosening of the conventional *ACID* properties defined for a database's transactions and the development of more flexible transactions to increase concurrency, increase data availability, limit data blocking, not cause cascading aborts and exhibit controlled recovery all under transaction control [5, 8, 10].

Standards in other areas of an information infrastructure which support real-time are also evolving. The IEEE has recently released a real-time extension (IEEE 1003.1b) to the POSIX operating system interface standard (IEEE 1003.1). This standard provides support for real-time scheduling of tasks, the concurrent execution of tasks and for extended control over numerous elements of the computer system. Evolving SQL standards are looking into the support of concepts which will aid in the introduction of real-time into the standard [10, 12]. For example there are additions in the SQL-3 standard for objects, triggers[2] , time reference, more refined constraint definitions and enhanced database and transaction structuring [14, 23].

When taken together these indicators demonstrate the need for real-time databases and the viability of the technology. Real-time database management systems will become commonplace in the computing arena in next five to ten years. To ensure that the development of such systems result in interoperable, open products requires standard-

[2]The concept for triggers presently in SQL are sketchy at best.

ization of application interfaces. It is a recommendation of the PRIS task group that the technology is sufficiently mature (concepts well understood) that the standardization efforts should begin now. The next section summarizes some of the findings of this task group for work areas for extensions to existing and evolving ANSI and ISO data management standards.

3.2 Work Areas for Real-time DBMS Standardization

If real-time is to find its way into the present and evolving SQL standards for databases, a working group must be formed to refine concepts defined by the DBSSG PRIS-TG as required for real-time database management systems support. In particular the PRIS-TG has defined five (5) major functional areas within a database management system that need enhancement for real-time to be incorporated into the SQL3 standard. These areas are:

1. Database Structure
2. Transaction Structure and Operational Properties
3. Transaction and Database Recovery
4. Architectural Dependencies
5. Remote Data Access

Each of these will be briefly covered in this executive summary section leaving details for the referenced documents.

3.2.1 Database Structuring

As in all database applications, the writer of a real-time application should be able to access data from the database in a natural form for the application. While the two dimensional table object type of the relational model is very well suited for most conventional applications, it is often insufficient for real-time applications [1, 27]. Relational tables are a poor fit for such real-time applications as contour maps, satellite images, and sensor data. For real-time applications, a database management system must also support the definition of abstract data types and binary large objects (blobs), user specified database consistency constraints, (data type constraints, temporal constraints, spatial constraints, storage constraints, access constraints, data and transaction criticality), as well as database decomposition and distribution.

3.2.2 Transaction Structure and Operational Properties

Transaction structuring must be altered from the conventional straight line sequence of database operations to provide real-time services tailored to the needs of real-time applications [5, 11]. For example if a real-time process is monitoring some number

of sensors, a transaction supporting that task must be structured so it can predictably, concurrently and consistently access and update data in support of the task regardless of all other actions in the system. Application-dependent transactions may require the ability to specify and control a variety of transaction processing schemes such as; serial transaction partitions [30], nested transaction partitions [24], interleaved transaction partitions [7] and parallel transaction partitions [26]. A general model of a transaction may consist of boundary markers, a specification, a body, recovery handlers and pre and post conditions on execution. The specification provides data structure definition, timing requirements specification, resource limits specifiification (e.g. max CPU, Disk, I/O execution time limits), data dependencies, criticality of transaction, atomicity of transaction, premptability of transaction and execution dependencies definition. Transaction pre-conditions and post-conditions are predicates which define the conditions upon which this transaction should or must begin execution and conditions upon which this transaction's execution is considered correct. The recovery body is used to hold user or system defined recovery procedures for user or system specified failures [5, 10, 13].

A new paradigm for transaction execution and concurrency control is needed to support the needs of real-time applications [4, 8, 15, 33, 36]. Transactions must execute to maximize the timeliness and predictability of applications, yet must also maintain database consistency and correctness. The difference from conventional databases is an altered definition of transaction *ACIDity* properties. Real-time transactions must be able to specify their own consistency, correctness criteria, database and transaction partitions, and commit criteria. Transaction initiation and transaction operations concurrency management must be driven by applications temporal, data dependencies and transaction relationships [37]. Transaction writers must be able to control how a transaction is selected for invocation [25], if a transaction can be preempted, if a transaction must be atomic, if a transaction must be recoverable, if data manipulation will trigger other database actions, how a transaction is partitioned and how interleaved conflicting operations will be ordered [10, 17].

3.2.3 Transaction and Database Recovery

Conventional means for recovery of committed and active transactions use check pointing of data items, with redo for committed and undo for active transactions. This model of recovery is not adequate for real-time database management systems where availability and timeliness of data may be more important than strict serializability [5, 35]. Correctness criteria for transaction execution and recovery must be altered to support the unique needs of real-time databases [13, 32]. It may be more desirable in the real-time database environment to do nothing (e.g. delay and wait for the next periodic update), do an application dependent recovery, or recover to a future correct state [5, 13] using forward recovery techniques.

Transaction recovery policies and mechanisms for real-time need to be added to the specification of database's data definition language (data temporal consistency constraints and enforcement rules) and to the specification of a database's data manipulation

language for transaction specification (transaction temporal consistency constraint and enforcement rules). The ability of transaction writers to specify exception conditions for software transaction aborts, conflict resolution, and hardware faults must be added to database languages [10].

Additional recovery mechanisms for recovery from failures for memory-resident database management systems are required, for bounded recovery in the event of catastrophic failure, the manage the effect of distributed architectures on recovery in real-time, and to the semantics of time-bound recovery (partial recovery, discard and reinitialize, etc.) must be provided within a standard for real-time databases.

3.2.4 Architectural Dependencies

Real-time systems must interact with an environment that they do not control. A real-time system must respond to detected conditions within time frames defined by the physical system so as to affect applications operations in a predictable manner. To provide this the real-time computer systems operations must be totally defined and bounded. To predict the actions of the real-time computer system requires the bounding of all aspects of execution, and to control performance so that actions *always* behave in the same way and take the same bounded time.

To deliver such capability the database system must request the operating system to perform in specific fashions. For example, to bound execution time may require limiting the size of a database table and to force the table to be stored in a specific place in memory and stored in a particular fashion. In addition, the database may require certain external sensors to trigger database actions on particular boundaries of time or events outside of the domain of the database system. To limit how the database executes may require the specification of CPU time, I/O time, stack usage and numerous other formerly operating system controlled resources to be managed for the databases purposes.

3.2.5 Remote Data Access

The I/O in a real-time system is not limited to memory and disks. It also includes the network and external elements. These external elements must be properly integrated with any database management scheduler scheme. Gigabit networking technology is making distributed real-time systems possible. Real-time scheduling of data packet/message transmissions will allow for dynamic selection and correlation of distributed data across the system.

Monolithic database management will be more of the exception than the norm in the future. Client/Server and distributed systems are necessary to meet the requirements of next generation products (as seen in the down sizing of corporate databases). Across these distributed components transactions must maintain their predictable behavior for real-time systems. The database management scheme should be able to maintain temporal constraint requirements as well as access timing requirements accross remote access conduits and protocols.

Realtime systems are subject to an overall system design. They are dedicated to a single application, which may consist of one or more concurrent processes. Resources are still managed by the operating system, but at the direction of the applicaiton, and in accordance with the overall system design. This is especially true of large real-time systems, such as the US Navy's BSY-2 system and the Air Force's AWACS programs.

Providing a framework of integration will make for a more unified approach to developing a real-time system. It also helps foster use of standards and less hand coding by providing mechanisms to control other performance components. The remote data access protocol standards [20, 21] are examples of standards for remote access of data which provide such a framework.

4 Recommendations of the PRIS-TG

The X3/OMC DBSSG PRIS-TG has found that real-time data management technology has a solid foundation for standardization. At the present time there are existing products [19, 29] which exhibit fundamental features for real-time data management. In addition numerous governmental, industrial and academic research programs are developing further prototypes and refinements of the numerous areas mentioned in this report. For example, the University of Massachusetts Amherst has developed real-time concepts, and prototypes for scheduling protocols, real-time transaction processing and real-time recovery; the University of Rhode Island, University of Virginia and Carnegie-Mellon University have been researching real-time database management systems and constructing testbed systems, the US Air Force's FIRM (Functionally Integrated Resource Management) program and numerous others have established a solid foundation for real-time data management standards development.

The technology is at a point where many basic concepts have reached stability as indicated by the increase in product developments. The further refinement of these basic features will enhance the stability of any developed standard. Some areas within this technology will require addition work to refine and stabilize the concepts to a point where a specific and lasting standard could be developed. In the time frame it will take to develop a standard, current and potential advances in the real-time database management area should have matured and stabilized.

4.1 Need for Standardization Effort for Real-time Database Management

Industry and government have a need for real-time information management systems and in the absence of standards are developing proprietary products to meet real-time application needs. The proliferation of one of a kind solutions for real-time information management will only make interoperability harder later on. A real-time information management systems, is one which provides the information or response at a known time, but generally not before. Requirements for real-time information management are found in numerous military, credit card validation, medical, airlines and nuclear power

systems. Presently there is a large movement in the DOD and industry to capture legacy systems and to incorpoate their information into on-line information infrastructures. Without real-time database management systems standards in place as soon as possible, capturing legacy real-time systems databases may not be possible.

4.2 Existing Practice

This is a relatively new technology area. In the past (and in many existing systems) the data necessary for real-time applications (especially military) was hard coded into the system. Now, organizations are developing their own, proprietary solutions. Standards in this area would facilitate applications portability. New products being developed today and existing real-time database products are built upon existing real-time operating systems standard products such as POSIX 1003.4. As the market for proprietary, one of a kind solutions dwindles, real-time database vendors will realize the need and see the market for standard, commercial off-the-shelf real-time database systems. Organizations are unwilling to place large projects at risk of failure to support unique infrastructure services. The mandated trend is to use commercial off-the-shelf products to the maximum extent possible. This trend does not appear to be a short lived policy, rather a glimpse of what will become a common systems development policy.

4.3 Expected Stability With Respect To Current and Potential Technological Advance

Presently there is a trend in industry, academia and government to develop products and technologies that are based on open systems philosophies. A real-time operating systems standard, POSIX 1003.4, has been developed and is quickly being adopted into off the shelf products. The use of standards will enhance the ability of developers to build real-time applications that will be portable across multiple platforms. Present practice of hand crafting real-time systems will be replaced with design philosophies that result in readily available commercial off-the-shelf open systems products. Presently real-time applications developers lack a consistent methodology for developing the information management portion of their application. Real-time database management systems standardization will result in products being developed that can operate on a variety of machines and provide a consistent platform for new applications developments. If the same open system philosophy of the operating systems domain is applied to the database management systems domain then current and future technological advances can be more readily supported within real-time database management systems products.

5 References

[1] M. Aksit, J. Bosch, W. Van der Sterren and L. Bergmans. Real-time Specification Inheritance Anomilies and Real-time Filters. In *Proceedings of European Conference on Object Oriented Programming*, 1994.

[2] R. Abbott and H. Garcia-Molina. Scheduling real-time transactions. In *Proceedings of ACM SIGMOD*, March 1988.

[3] ANSI. Portable operating systems interface standard. Technical Report ANSI, ANSI, Washington, DC., September 1993.

[4] A. Biliris, S. Dar, N. Gehani, H. Jagadish, and K. Ramamrithham. Assel: A system for supporting extended transactions. In *Proceedings of the ACM SIGMOD*, March 1994.

[5] G. Bundell and G. Trivett. "Real, real time transactions." *The Bulletin of the IEEE Technical Committee on Data Engineering*, 17(1), March 1994.

[6] L. Cingiser DiPippo and V. Fay Wolfe. Object-based semantic real-time concurrency control. In *Proceedings of the 14th Real-time Systems Symposium*, Dec 1993.

[7] P. Fortier. *D.Sc. Thesis: Early Commit.* University of Massachusetts Lowell, 1993.

[8] P. Fortier, J. Prichard, and V. Fay Wolfe. Flexible Real-Time SQL Transactions. *Proceedings of the Real-Time Systems Symposium*, December 1994.

[9] P. Fortier, D. Pitts, and T. Wilkes. Experiences with data management in real-time C^3 systems. *Proceedings of the SEDEMS II Conference*, April 1993.

[10] P. Fortier and J. Prichard. Concepts for a real-time structured database query language (RT-SQL). *In the Proceedings of the IFIP/IFAC Workshop on Real-time Programming*, June 1994.

[11] P. Fortier and J. Rumbut. Issues and concepts for a real-time database management. *In the Proceedings of the First International Conference on Electronics and Information Management*, August 1994.

[12] P. Fortier and CDR. G. Sawyer. DISWG a new player in NGCR open systems standards. *to appear in Computer Standards and Interfaces*, 1995.

[13] P. Fortier and J. Sieg. Recovery protocols for real-time database management systems. *In the Proceedings of the International Conference on Information Management (ICIM94)*, May 1994.

[14] L. Gallagher. Object SQL: Language extentions for object data management. *International Society for Mini and Microcomputers CIKM-92*, 1992.

[15] H. Garcia-Molina, D. Gawlick, J. Klein, K. Kleissner, and K. Salem. Modeling long-running activities as nested sagas. *Bulletin of the IEEE Technical Committee on Data Engineering*, 14(1), March 1991.

[16] K. Gordon. *DISWG Database Management Systems Requirements.* NGCR SPAWAR 331 2B2, Alexandria, Va., 1993.

[17] M. Graham. Real-time data management. *IEEE Technical Committee Real-Time Systems Newsletter*, 9(1/2), Spring/Summer 1993.

[18] IITA Task Group. *Information Infrastructure Technology and Applications.* National Coordination Office for HPCC, Executive Office of the President, February 1994.

[19] D. Hughes. ZIP-RTDBMS a real-time database management system. Technical Report PRISTG-93-011, ANSI DBSSG PRIS-TG, San Diego, CA, 1994.

[20] ISO/IEC. RDA - part 1: Generic model, service and protocol. Technical Report 9579-1, ISO/IEC, Washington, DC, 1993.

[21] ISO/IEC. RDA - part 2: SQL specification. Technical Report 9579-2, ISO/IEC, Washington, DC, 1993.

[22] H. Korth, E. Levy, and A. Silberschatz. A formal approach to recovery by compensating transactions. In *Proceedings of the 16th VLDB Conference*, 1990.

[23] J. Melton and A. Simon. *Understanding the New SQL: A Complete Guide.* Morgan Kauffman Publishers, San Mateo, CA., 1992.

[24] J. Moss. *Nested Transactions: An Approach to Reliable Distributed Computing.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1981.

[25] H. Nakazato. *Issues on Sychronizing and Scheduling Tasks in Real-Time Database Systems.* PhD thesis, University of Illinois at Urbana- Champaingn, Urbana, Il., 1993.

[26] T. Ozsu and P. Valduriez. *Principles of Distributed Database Systems.* Prentice Hall Inc., Englewood Cliffs, NJ, 1991.

[27] J. Prichard, L. Dipippo, J. Peckham, and V. Wolfe. RTSORAC: A real-time object oriented database model. *To Appear in: Proceedings of the Database and Expert Systems Applications*, Sept 1994.

[28] K. Ramamritham. Real-time databases. *International Journal of Distributed and Parallel Databases*, 1(2), 1993.

[29] M. Roark. RTDM: A real-time database management systems. Technical Report PRISTG-93-012, X3 DBSSG PRIS-TG, San Diego, CA, 1993.

[30] L. Sha. *Ph.D Thesis: Modular Concurrency Control and Failure Recovery – Consistency, Correctness and Optimality*. Carnegie-Mellon University, Pittsburg, PA, 1985.

[31] L. Sha, J. Lehoczky, and E.D. Jensen. Modular concurrency control and failure recovery. *IEEE Transactions on Computers*, 37(2), 1988.

[32] S. Son, S. Yannopoulos, Y. Kim, and C. Iannacone. Integration of a database system with real-time kernal for time-critical applications. *International Conference on Systems Integration*, June 1992.

[33] J. Stankovic and W. Zhao. On real-time transactions. *SIGMOD Record*, 17(1), March 1988.

[34] H. Tokuda. Compensatable atomic actions in object-oriented operating systems. In *Proceedings of the Pacific Computer Communications Symposium*, October 1985.

[35] P. Watson. The challenge of response time management in real-time distributed systems. In *IEEE Proceedings of the Fourth Israel Conference on Computer Systems and Software Engineering*, June 1989.

[36] V. Wolfe, L. Cingiser, J. Peckham, and J. Prichard. A model for real-time object-oriented databases. *IEEE Technical Committee on Real-Time Systems Newsletter*, 9(1/2), Spring/Summer 1993.

[37] V. Wolfe, S. Davidson, and I. Lee. RTC: Language support for real-time concurrency. *Journal of Real-time Systems*, 5(1), March 1993.

[38] P. Yu, K. Wu, K. Lin, and S. Son. On real-time databases: Concurrency control and scheduling. In *Proceedings of the IEEE*, volume 82, January 1994.

Appendix A: Schedule Followed by the PRIS-TG

- Jul 1993
  - Formation of the Task Group.
- Oct 1993
  - Statement of issues to be addressed.
  - Listing and classification of concepts going by the name Real-time Information Systems.
  - Initial draft of survey form.
  - Outline of Reference Model and Report.
- Jan 1994
  - Preliminary survey data collection.
  - Initial draft Reference Model.
- Apr 1994
  - Complete survey data collection and analysis.
  - Interim Draft Reference Model.
  - Preliminary recommendations formulated.
- July 1994
  - Draft Reference Model completed.
  - Draft Final Report.
- Oct 1994
  - Final Report submitted to DBSSG.
- Jan 1995
  - Revise final report and submit to OMC.
  - Initiate standards task groups in sub-areas as needed.
  - Disband after responding to questions and comments from OMC and X3 Technical Committees.

Appendix B: Annotated References

- A.Billiris, S. Dar, N. Gehani, H. Jagadish, K. Ramamritham, "ASSET: A System for Supporting Extended Transactions", ACM SIGMOD Conference, May, 1994.

  Abstract *Extended transaction models in databases were motivated by the needs of complex applications such as CAD and software engineering. Transactions in such applications have diverse needs, for example, they may be long lived and they may need to cooperate.*

  We describe ASSET, a system for supporting extended transactions. ASSET consists of a set of transaction primitives that allow users to define custom transaction semantics to match the needs of specific applications. We show how the transaction primitives can be used to specify a variety of transaction models, including nested transactions, split transactions, and sagas. Application-specific transaction models with relaxed correctness criteria, and computations involving workflows, can also be specified using the primitives.

  We also describe the implementation of the ASSET primitives in the context of the Ode database.

- P. Fortier, J. Pritchard "Concepts for a Real-time Structured Database Query Language (RT-SQL).", *Proceedings of the IFAC/IFIP Workshop on Real-Time Programming*, June 1994.

  Abstract. - *Real-time database management systems have become a hot topic in the research and development community of late. In addition there has been a movement in the standards community to examine and develop extensions to existing and proposed query languages to support real-time.*

  This paper examines the state of research into real-time database management systems in the areas of database structuring, transaction structuring, transaction processing, concurrency control, recovery and real-time transaction scheduling. We then extend the findings and trends of this work into the high level specification of data definition language, data manipulation language and data control language extensions for the standard SQL2 and emerging SQL3 database query languages.

- P. Fortier, M. Murphy "Simulation Analysis of Real-time Task Scheduling" *Proceedings of the 27th HICSS Conference*, January 1994.

  Abstract - *In a distributed real-time command, control and communication C^3 system, tasks execute to fulfill both local and system wide computational goals. Satisfying system wide goals imposes requirements on local tasks to operate in a predictable manner, within restricted timing ranges. In addition, local tasks themselves may vary within a wide operational envelope in terms of their criticalities of performance.*

Traditional solutions to scheduling, use mechanisms such as FIFO, round robin, or simple priority to provide sequencing. These techniques are not adequate in a time constrained environment, where failure could lead to catastrophic results.

This paper surveys a collection of scheduling algorithms and examines their performance via simulation for a class of real time C^3 tasks. Our value-phase-deadline (VPD) algorithm is described and analyzed against five well known schedulers.

- Fortier, P. "Application of RMA for Next-generation Database Management Systems", *Proceedings of the 2nd RMA Users Forum*, Software Engineering Institute, CMU, November 1993.

  Abstract - *Rate Monotonic Analysis (RMA) is being increasingly used as a tool for architecting real-time systems operations. This technology has been used solely for the optimization of operating systems task scheduling for real-time systems. The question this paper posses is "can RMA be used effectively for real-time database management analysis". The answer is not that simple or clear. Databases pose a new problem to RMA that of dynamic behavior which cannot be apriori engineered. We look at some of the issues and pose some uses for RMA within real-time database management.*

- P. Fortier, J. Prichard, V. Wolfe, " A Methodology for Extending SQL for Real-time", *Submitted to the ACM SIGPLAN Workshop on Real-time Programming Languages*, 1994.

  Abstract - *SQL is a standard language which supports the definition, manipulation, and control of data in a relational database systems. This paper proposes a methodology for extending SQL to provide support for real- time database systems. Current wok on real-time SQL (RTSQL) is based upon the ANSI/ISO standard SQL2. We outline the requirements for real-time databases, then outline extentions for SQL's data definition, data manipulation and data control languages*

- P. Fortier, J. Prichard, V. Wolfe, "Flexible Real-time SQLTransactions", *Proceedings of the Real-time Systems Symposium*,December, 1994.

  Abstract - *This paper presents flexible transaction structuring capabilities that allow relaxed ACID properties for better support of real-time transactions. The specification of these flexible transaction structures is demonstrated through proposed extensions to the standard SQL database language.*

- P. Fortier, J. Sieg, "Recovery Protocols for Real-time Database Management Systems", *Proceedings of the 5th International Conference on Information Management (ICIM94)*, May 1994.

  Abstract - *In a distributed real-time command, control and communication C^3 system, database transactions execute to fulfill both local and system wide information management goals. Satisfying real-time system wide goals imposes requirements on transactions to operate in a predictable manner, within restricted*

timing, correctness and consistency ranges. In addition, local transactions themselves may vary within a wide operational envelope in terms of their criticalities of performance.

Traditional solutions to transaction operation and recovery, use mechanisms revolving around serialization of transaction executions and database chechpointing with redo and undo of transactions for recovery. These techniques are not adequate in a time constrained, highly concurrent environment, where failure could lead to catastrophic results. This paper outlines protocols that increase concurrency, limit cascading aborts and minimize the impact on recovery for a decomposed database and transaction system.

- P. Fortier, J. Sieg, "Simulation Analysis of Early Commit Concurrency Control Protocol' *To appear in the Proceedings of the 28th Annual Simulation Symposium*, April, 1995.

  Abstract - *This paper describes results of a simulation model for decomposition of concurrency control enforcement in databases. The database is partitioned into atomic data sets using constraints defined during database design. For each atomic data set A, the transaction writer declares a point in his transaction after which there will be no more accesses to A. This location is a candidate for early commitment. We present three new concurrency control protocols: early-commit versions of conventional locking, timestamp ordering, and optimistic protocols, and one new recovery protocol: merged-commit. A simulation model used to model these protocols is described. The new protocols performance is compared to that of their conventional counterparts using the described simulator.*

- P. Fortier, G. Sawyer, "DISWG a New Player in NGCR Open Systems Standards", *To appear in the journal of Computer Standards and Interfaces*, 1995.

  Abstract - *Real-time command, control, and communications (C^3) systems are being used in many industrial, medical, business and military applications. Real-time systems differ from conventional, general purpose computer systems in that the consistency and correctness of the controlling process is dependent on the timliness and predictability of the controlling processes effects on the controlled system. Availability of data may be as important as correct and consistent execution in real-time C^3 systems.*

  Real-time C^3 systems traditionally have not adequately addressed these requirements in a methodical fashion; systems have been hand-crafted and fined-tuned until they met testing requirements.

  In this paper, we review ongoing efforts by the U.S. Navy in cooperation with private industry and academia to develop standards and concepts for next-generation real-time database management systems. We examine what is the NGCR effort, what is DISWG, and how we will foster the evolution of conventional standard database management into real-time database management systems standards.

- P. Fortier, J. Rumbut, "Issues and Concepts for Real-time Database Management", *Proceedings of the The 1st International Conference on Electronics and Information Technology*, August 1994.

  Abstract - *Real-time database management systems are becomming an important issue within the computer science research community. The community though has not focussed efforts on the specific needs of real-time predictable systems, nor have they adequately delineated the database management systems requirements for these systems. Due to the unique needs of real-time predictable computer systems and their increased reliance on information, database management systems are being examined for applicability.*

  We are interested, in the definition of requirements for information management in real-time predictable systems and in the review of policies and mechanisms being researched to meet these unique requirements.

- P. Fortier, J. Peckham, "Operating System Support for Next Generation Database Management Systems" Working report of the NGCR DISWG, 1995.

  Abstract - *To realize the benifits of standards for next generation database management systems requires the interoperability of the database management systems and the host operating system. This paper examines the basic features of database management systems, operating systems and what a database system need from an operating system to perform its functions better. The paper also examines issues in how to evaluate the performance of an operating system in relation to a database management systems being supported.*

- P. Fortier, Victor Fay Wolfe and Janet Prichard, "Real-time SQL: Extensions for Support of Real-time Database Systems", *To appear in the journal of Computer Standards and Interfaces*, 1995.

  Abstract *Real-time database management systems have recently received a great deal of attention in the research and development community. In addition there has been a movement in the standards community to examine and develop extensions to existing and proposed query languages to support real-time. This paper examines the state of research into real-time database management systems in the areas of database structuring, transaction structuring, transaction processing, concurrency control, recovery and real-time transaction scheduling. We then extend the findings and trends of this work into the high level specification of data definition language, data manipulation language and data control language extensions for the standard SQL2 and emerging SQL3 database query languages.*

- Janet Prichard, Lisa Dipippo, Joan Peckham and Victor Wolfe, " RTSORAC: A Real-time Object Oriented Database Model", *Proceedings of the 5th International Conference on Database and Expert Systems Applications*, Sept. 1994.

  Abstract *A real-time database is a database in which both the data and the operations upon the data may have timing constraints. We have integrated real-time,*

object-oriented, semantic and active database approaches to develop a formal model called RTSORAC for real-time databases. This paper describes the components of the TRSORAC model including objects, relationships, constraints, updates and transactions.

- A. Pruitt, "A Real-time Database Manager for Mission Critical Combat and Sensor Systems, Martin Marietta document number F94-0293

  Abstract *Real-time database systems differ from conventional database systems in many ways. These differences restrict the use of conventional databases in real-time systems. The major problems are lack of adequate performance and unpredictable behavior. Martin Marrietta has developed a real-time database manager which is a marriage of the best attributes of conventional database management systems and real-time systems. This database manager provides high-performance, guaranteed response-times, deterministic resource consumption and a fault tolerant architecture. while it was developed for application in Navy mission critical, real-time combat/control and sensor systems, it has application in commercial products. Its ability to provide superior performance using very limited resources makes it ideal for embedded applications.*

- A. Pruitt and M. Moore, " The Emergence of Real-time Database Management Technology, *Proceedings of the Conference on High Performance Computing*, Singapore, 1994.

  Abstract *Real-time database management systems require added support from the underlying systems support infrastructure in order to deliver guaranteed response times and deterministic behavior. To provide this, systems must be designed such that resource consumption can be predictaed ahead of time and can therfore be used as a metric in systems design. Predictable resource use requires the preallocation of CPU cycles as well as required peripheral elements. This paper outlines issues in the managed design of real-time systems from a database management systems perspective.*

Appendix C:

PRIS-TG Member List

KEY: P-Principle, A-Alternate, L-Liaison, C-Correspondence, E-Ex Officio

Baudoin, Claude
A
Software Engineering Director
Schlumberger ATE
1601 Technology Drive
San Jose CA 95110-1397
(408) 437-5228
Fax: (408) 437-5246
Internet: baudoin@san-jose.ate.slb.com

Brown, Todd, 1Lt.
C
USAF SSC/XOCR
Bldg. 856, Room 154
Gunter AFB, AL 36114

Burns, Thomas
A
MITRE Corporation
MS W-649
7525 Colshire Drive
McLean, VA 22102
(703) 883-6214
email: tburns@mitre.org

Dumais, Joe
C
Martin Marietta Western Systems
4041 N. First 95134
email: dumais@ralph.mdso,vf.ge.com

Emerson, John
C
Empress Software Inc.
6401 Golden Triangle Drive
Greenbelt, MD 20770
(301) 220-1919
Fax: (301) 220-1997
email: jermerson@empress.com

Fisher, Donna K.
Chair
P
NCCOSC RDTE DIV 411
49180 Transmitter Road Rm 2
San Diego, CA 92152–7341
(619) 553-4095
Fax: (619) 553-6288
e-mail: dfisher@cod.nosc.mil

Fortier, Dr. Paul
P
University of Massachusetts Dartmouth
Department of Electrical and Computer Engineering
North Dartmouth, MA 02747-2300
(508) 999-8490
Fax: (401) 841-2431
e-mail: Fortier@ada.npt.nuwc.navy.mil

Glickstein, Ira
C
IBM Federal Systems Co.
9002 IBM FSC
Owego, NY 13827
(607) 751-2008
Fax: (607) 751-2008
e-mail: ira@vnet.ibm.com

Graham, Marc
C
CMU
SEI
Pittsburg, PA 15213
(412) 268-7784
Fax: (412) 268-5758
email: marc@sei.cmu.edu

Hollowell, Glenn

P
SEMATECH
2706 Montopolis Drive
Austin, TX 78741-6499
(512) 356-7166
FAX: (512) 356-3575
e-mail: glenn_hollowell@sematech.org

Hughes, David
P
DBx, Inc.
P.O. Box 8446
Cherry Hill, NJ 08002-0446
(609) 667-9322 (+fax)
e-mail: DBx@world.std.com

Kilov, Haim
P
Bellcore, MRE-2F049
445 South Street
Morristown, NJ 07960
(201) 829-4509
Fax: (201) 829-5883
email: haim@bcr.cc.bellcore.com

Kinsley, Kate
P
Datawise, Inc.
1915 E. Colonial Drive
Suite 22
Orlando, FL 32803
(407) 894-7203
FAX: (407) 275-5381 (business hours)

McCabe, Patrick
C
Rome Labs
32 Hanger Rd.
Griffis AFB, NY 13441-4110
(315) 330-3222
FAX: (315) 330-3913
e-mail: mccabep@el.af.mil

Meisenbacher, John
C
Strata Group
3301 Rider Trail South
Suite 170
St. Louis, MO 63045
(314) 770-9659
FAX: (410) 765-7900

Moore, Mike
C
SofTech Inc.
P.O. Box 143
Clinton, NY 13323-0143
(315) 456-2484
FAX: (315) 853-4546

Pritchard, Janet
C
Dept. of CS
258 Tyler Hall
University of Rhode Island
Kingston, RI 02881

Richards, Paul
C
Martin Marietta Western Systems
4041 N. First St.
San Jose, CA 95134
e-mail: richards@ralph.mdso.vf.ge.com

Roark Mayford
P
Martin Marietta
P.O. Box 4840, EP7 MD63
Syracuse, NY 13221-4840
(315) 456-7565
FAX: (315) 456-0107
e-mail: roark@gw.syr.ge.com

Smith, Maurice
L, X3T2
Allied-Signal
P.O. Box 419159
Kansas City, MO 64141-6159

(816) 997-3590
e-mail: msmith@vax2.cstp.umkc.edu

Son, Dr. Sang
C
University of VA
Dept. of Computer Science
Thorton Hall
Charlottesville, VA 22903
(804) 982-2205
FAX: (804) 982-2214
e-mail: son@virgina.edu

Taylor, Clavin
C
US West Advance Tech
4001 Discovery Drive
Boulder, CO 80303
(303) 541-6369
FAX: (303) 541-6384
e-mail: ctaylor@advtech.uswest.com

Thuraisingham, Dr. Bhavani
C
The MITRE Corp.
A156
Burlington, Rd
Bedford, MA 01730
(617) 271-8873
FAX: (617) 271-7722
e-mail: Thura@security.mitre.org.

Trout, Ray
C
Superior Prgramming Services
18100 Upper Bay Road, Ste 105
Houston, TX 77050
(713) 335-0366
FAX: (713) 335-9444

Trujillo, Ed
C
Hughes Aircraft Co.
P.O. Box 92426
M/S RE/R10.S532
Los Angelos, CA 90009-2426
(310) 607-1892
FAX: (310) 607-5699

Wolf, Dr. Victor
C
Dept. of CS
258 Tyler Hall
University of Rhode Island
Kingston, RI 02881
(401) 792-2499
FAX: (401) 792-4617
e-mail: wolfe@cs.uri.edu

Appendix D: Industry Survey

PRIS-TG Survey Report PRISTG 94-009

Introduction and Background

The Predictable Real-Time Information Systems Task Group (PRIS-TG) is writing a final report to the Operational Management Committee (OMC) on the need and readiness for a standardization of real-time database concepts. The PRIS-TG final report will:

1. Investigate the a need for standardization real-time database concepts.
2. Evaluate the current level of standardization within the real-time database field.

The PRIS-TG project surveyed real-time database professionals to reveal attitudes toward a real-time database standardization. To obtain baseline data for PRIS-TG, a survey was created and distributed to the real-time database community. The results of this survey will be presented below.

Survey Description

The survey consisted of 12 questions requiring short written responses. The questions forcused on two goals:

1. Revealing the kinds of real-time database projects currently being worked on within the database community.
2. Measuring respondent attitudes toward standardizing real-time database concepts and principles.

The following section presents the result of the survey.

Survey Statistics

Surveys were distributed to real-time systems professionals in industry and DOD. In all, twelve (12) completed surveys were returned.

PRIS-TG Survey Statistics

1. Are you familar with Real-Time Database Management Systems (RTDBMS)?
   RESULTS: Yes: 8 No: 4

2. Are you desigining or planning applications (of any size) that would require this type of technology?
   RESULTS: Yes: 8 No: 3

3. If yes, will you implement your own?
   RESULTS: Yes: 3 No: 5

4. If yes, is this a main memory RTDBMS?
   RESULTS: Yes: 4 No: 0

5. How important would standardization of general RTDBMS concepts and principles be to your procurement or implementation?
   RESULTS: The following were responses given to this question:

   - "Very!"
   - "Quite useful"
   - "Very important"
   - "Very important. It will help produce other production quality real-time DBMSs."
   - "Very helpful"
   - "Such standardization would probably lead to reduced maintenance costs for the family of systems in JMCIS and this would be a tremendously important aspect for SPAWAR PD60 and the Navy."
   - "Not important at this time."
   - "Relatively important"
   - "Standarization can mean S/W reuse and interoperability."
   - "RTDBMS concepts and principles should be standardized."
   - "Good idea."

6. Does your application have long duration transactions (CAD or target tracking, for example)?
   RESULTS: Yes: 7 No: 4
   Survey respondents gave the following long duration transaction examples:

   - Target tracking
   - Targeting and OPNL force readiness status.
   - ODBMS supports long transactions with check in/out and persistent locks.
   - Ground target tracking.
   - Threat location tracking.
   - Mission replan.

- Determine position and motion from state-vector tracking data and time-space position information.
- Collect sensor, weapon/combat direction system, and C3I data.
- Provide data storage/archiving capability with the means for subsequent retrieval.

7. Does your database application required recovery? If so, must it be full or may it be partial?
   RESULTS: Yes: 6 No: 1
   Must be full recovery: 6
   May be partial recovery: 0

8. Does your application required heterogeneous database interaction?
   RESULTS: Yes: 6 No: 2

9. Does your application requires special scheduling?
   RESULTS: Yes: 5 No: 2

   - Tell us what your think is important about RTDBMS.
     RESULTS: The following were responses given to this question:
   - Provides most current data for OPNL command decision support
   - Lowers software costs
   - Provides improved data handling and protection
   - Allows for the design of a system that must synchronize itself with the external world under conditions the system cannot control.
   - Has the ability to access data in a timely fashion.
   - We need an evaluated multi-level secure (MLS) COTS product with performance (i.e., response time) equivalent to today's non-MLS products.
   - Standardization of RTDBMS can be effective means of controlling cost for duplicative software development efforts.
   - Standardization should not impinge on initiative and flexibility to implement operational requirements in software. Rigid guidelines can become restrictive and counter productive.
   - An RTDBMS is vital to the operations within the TCTS program.

10. Does (or will) your real-time database have to be compatible with either the relational or the object-oriented model of databases, or does it matter?
    RESULTS: Both: 2 Does not matter: 5
    Relational only: 2
    Object-oriented only: 0

11. Would you attend a real-time information systems workshop?
    RESULTS: Yes: 9 No: 3

12. Would you contribute a paper to a real-time information systems workshop?
RESULTS: Yes: 6 No: 6

Appendix E:

Real-time Database Management Reference Model

Database Systems Study Group (DBSSG)

Predictable Real-Time Information Systems Task Group (PRISTG)

Real-Time Database Management Systems Reference Model

Paul J. Fortier[1]
University of Massachusetts Dartmouth
North Dartmouth, Massachusetts, 02747

February 9, 1995

[1]I wish to thank The US Navy's Next Generation Computer Resources Database Management Systems Interface Standards Working Group and The Naval Undersea Warfare Center for their support in this documents development.

Abstract

This document describes a database management systems reference model for the American National Standards Institute (ANSI), Accredidted Standard Committee (ASC), X3, Operational Management Committee (OMC), DataBase Systems Study Group (DBSSG) Predictable Real-Time Information Systems Task Group (PRISTG). This document is meant as a general view of a database management system to allow the PRISTG to discuss the focus of real-time standardization efforts.

Contents

List of Figures

Chapter 1

Introduction

The American National Standards Institute (ANSI), formerly the American Standards Association (ASA), was formed in 1918 by five engineering societies. The ANSI charter is to serve as an independent, voluntary institution for the development, management, and coordination of national standards. Further, ANSI is to provide a focal point for industry and government on standards and to represent the US. in non-government international standards organizations.

One of the major ANSI Accredited Standards Committees (ASCs) is the Computers and Information Processing Committee (X3). The Computer and Business Equipment Manufacturing Association (CBEMA) serves as the X3 Secretariat and has performed that function since X3's formation in 1961.

The scope of X3 is: standardization in the areas of computers and information processing and peripheral equipment, devices, and media related thereto; and standardization of functional characteristics of office machines, plus accessories for such machines, particularly in those areas that influence the operators of such machines.

While X3 serves as the consensus body, its actual standards writing is delegated to its technical committees and their task groups. There are some eighty five or more entities and over 3200 volunteers working on approximately 700 projects. The majority of these projects fall within a systems concept of information processing technology, including Open Systems Interconnection (OSI), and deal with such items as programming languages, ASCII, Networks, and Office Text technologies.

In addition to its domestic standards responsibility, X3 serves as ANSI's Technical Advisory Group (TAG) to X3's international counterpart of the International Standardization Organization (ISO), Joint Technical Committee 1 (JTC1). X3's technical committees serve as the TAGs to the subcommittees of the JTC1.

One of the X3 management committees is the Operational Management Committee (ASC/X3/OMC). OMC plans, studies, reviews, and makes recommendations to X3 regarding standards projects and drafts proposed standards. The study groups of OMC provide in-depth studies and ongoing assessment of particular areas of technology.

The ASC/X3/OMC Database Systems Study Group (DBSSG) is one of the OMC

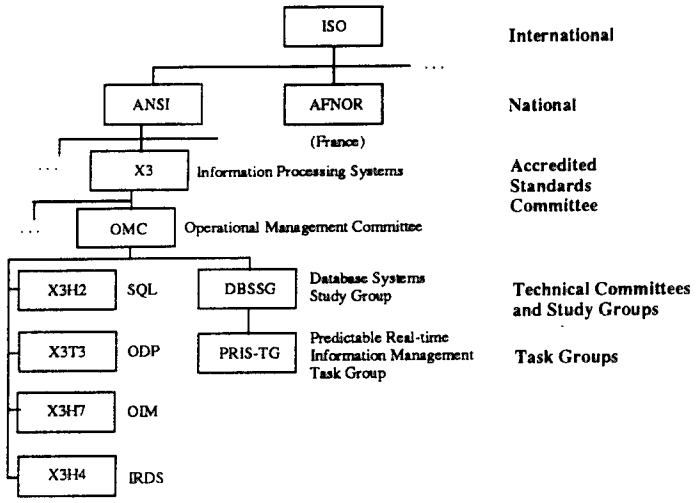

Figure 1.1: X3 Organization Tree

study groups. The Predictable Real-time Information Systems Task Group (PRISTG) is a DBSSG task group (See Figure 1.1).

1.1 PRISTG Scope

In May 1993, OMC approved the following scope for the Predictable Real-time Information Systems Task Group.
The Task Group will:

1. Develop a predictable real-time information systems reference model.
2. Evaluate existing predictable real-time information systems technology.
3. Determine the need for standardization in this area.
4. Determine the need for security in this area.

1.2 Objectives of PRISTG

The objectives of the Predictable Real-time Information Systems Study will be:

1. To establish a framework for future predictable real-time information management standards activities, both extensions to ongoing SQL, IRDS, RDA, ODP, OODB, POSIX, ADA, and computer security development, as well as related future standards.

2. The task group will review and evaluate existing and developmental products claiming to be real-time information management systems, as well as, real-time products with information management capabilities. The task group will also review and evaluate published literature and research activities concerning real-time information management technology world wide, and seek to generate discussion among practitioners in the field.

3. This study project is for the development of recommendations for standardization in the area of real-time information management technology.

1.3 Purpose and Scope of This Document

The purpose is to document the PRISTG adopted reference model to help guide recommendations for standards development and requirements refinement for real-time database management, as well as define a common view upon which to discuss PRISTG recommended standards activities.

This document is intended to provide an unbiased view of database management systems architecture, in terms of a reference model for use in recommendations for real-time information management standards development. The intent is to not define a systems model tied to one of the present state-of-practice database systems models. To this end the real-time database management systems reference model must provide a general view of a database system, its functions, services and interfaces. The specifics of implementations and operations are left to implementors and to specific standards potentially wrapped around one of the existing database structural models.

The systems model presented in this document was developed by a consensus of the PRISTG industry, academic and governmental members. The model will be used in the definition and discussion of recommended standards and products being constructed, or accepted by the PRISTG and the OMC standards activities.

1.4 Organization of Report

The PRISTG reference model report is organized into four sections. Section one is the introduction to this document and defines the goals, purpose, scope and approach to the production of the document. Section two describes the reasons behind the need for real-time DBMS standards, the process for identifying potential standards, the areas for potential standards, current efforts in DBMS standards (derived from the current technology subgroup), and open issues. Section three describes the real-time reference model. Section four provides a description of the PRISTG standards selection process and criteria. This is followed by references and a bibliography.

1.5 Structure of the Reference Model

The baseline PRISTG reference model consists of two sub models. A meta database structural model and an active services execution model. The meta database model is an adaptation of the ANSI/SPARC three schema architecture [10]. The meta database model of ANSI/SPARC was modified to cover both distribution of data and heterogeneity of data which were basic requirements of a real-time predictable database management system [6].

The active services database model is a refinement of the DISWG [5] database model[1]. The services database model consists of seven layers. These seven layers cover the functions of the database management system from applications down to physical interface with computing resources. The layers have been developed so as to be general in functionality in order to capture as many database designs and implementations as possible. Not all of the layers have application programming interfaces (API) access. Those that do have APIs provide some set of services to applications. Those that do not have API access provide services in support of other PRISTG reference model layers.

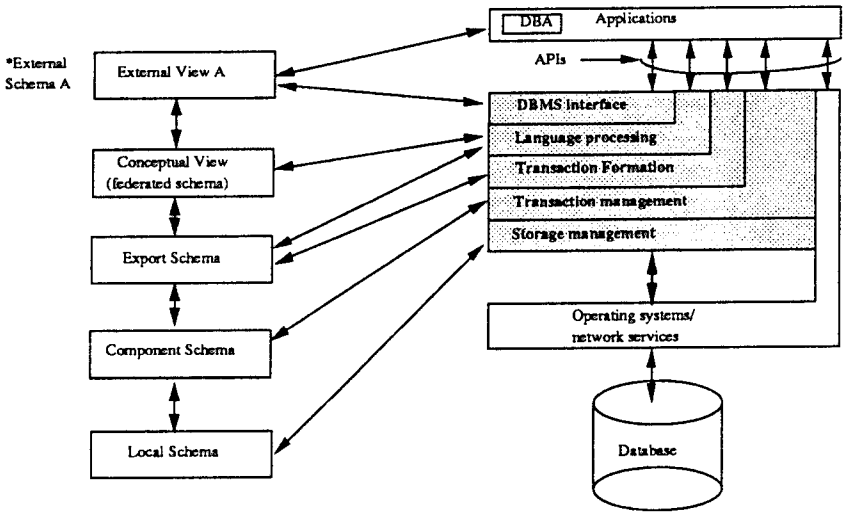

Figure 1.2: PRISTG Database Systems Reference Model

The real-time database reference model is shown in Figure 1.2 Starting from the highest layer, the layers include: the applications layer, database management system interface layer, language processing layer, transaction formation layer, transaction management layer, storage management layer, and operating system and network layer. The

[1]The DISWG is a working group of the US Navy's Next Generation Computer Resources (NGCR) commercial computer standardization effort

lowest two layers, the operating system/network layer and the storage management layer provide management, access and control over the physical database assets. The top layer, applications is where applications are developed and the layer through which real-time data management services are accessed. These layers interface with the meta database model to extract pertinent structural and logical information for use in manipulating stored information. The local schema provides information on the logical to physical mapping odf data items. The component schema provides information on composite logical structures. The export schema provides information on data views external from a host site. The conceptual view provides information on mapping of sites to logical data, and the external view is the view provided to applications for interpretation.

Chapter 2

Need for Real-time DBMS Standards

2.1 Directions in Real-time DBMS Standards

To date most work in standards for database systems has been at the programming language level. Standards have been specified for the SQL relational database system query language, for NDL network database systems languages wrapped around the CODASYL model, and RDA remote database access for distributed access to databases using the client server model of distributed access. PRIS-TG must not only look at current standardization activities, but must examine research and product development efforts in real-time systems and un-conventional database systems now evolving. Real-time systems will require protocols and services much different than those seen in conventional database implementations.

2.2 The need for standards

Standards provide a means to specify a set of services, protocols, and interfaces. In an ideal world, the use of standards allows multiple products to be developed which may connect and interact uniformly without requiring alteration to any of the products.

2.3 Process for Identifying Potential Standards

The PRIS-TG has been established to examine current efforts in standardization for database structure and data management for real-time systems. To reach these goals the PRIS-TG has; 1) examined current standards and available technology, 2) identified requirements for database and database management needs for real-time systems

[6], and 3) developed a database management system model and approach to develop standards.

2.4 Potential areas for Standardization

The PRISTG must determine what level(s) in the DBMS reference model are appropriate for real-time database management systems standardization. The options can be broken down as follows:

- Applications to DBMS manipulation and definition language interface.
- Applications to DBMS distinct layers' protocols and services. This level requires examining the purpose of database components and to develop a set of services, protocols and their interfaces for standardization. These layers represent the API and external environment interfaces in conventional database systems.
- Database management system to operating system and network.

PRIS-TG should keep an open mind and examine all of these issues to determine the benefits and pitfalls of each possibility. To this end a study is being performed by Naval Undersea Warfare Center, The University of Massachusetts Dartmouth and the University of Rhode Island to define requirements specifically from each option [4, 3, 8].

2.5 Current Efforts Towards Real-time Data Management Standards

This information is provided in the NGCR DISWG current standards and available technology documentation [2].

2.6 Open Issues

This section maintains a list of open PRIS-TG issues and direction for closure.

Operating system to database management system:

- memory control
- memory faults and recovery
- process over ride
- real-time access
- trigger invocation and control

The PRIS-TG is studying these issues and will document findings. Features of an operating system not provided will be directed towards the IEEE POSIX real-time standards working group for comment and or action.

Chapter 3

The Real-time DBMS Reference Model

3.1 Introduction

A reference model provides a general means to view and discuss interfaces and functions of various discrete components of a system. The real-time database reference model will aid in the development of interface standards for future real-time database management systems components. The real-time reference model represents a database management system as a set of distinct layers, each of which has a specific function different from all other layers, with clear interfaces between the layers.

The number of layers, and the function of each will vary from one database management systems implementation to another. However, in all database management systems the purpose of each layer is to offer certain services to the higher layers, shielding those layers from details of how the offered services are actually implemented.

3.1.1 Motivation

A layered reference model reduces the complexity of the whole system by breaking the system into manageable pieces. The goal is not to capture every database management systems implementation's structure totally, but to develop guidelines that generally meet the needs of specific interfaces to functional software elements found in the majority of database management systems implementations. The layers in a reference model are developed to generally meet the following goals:

1. A layer should perform a well defined function.
2. Interfaces with a layer are cleanly definable.
3. Layers minimize information flow across interfaces.

4. Layers are created when abstraction is needed.

5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer, nor so large that the model becomes unwieldy.

This reference model specifies the number of layers and how these layers are connected. Interfaces that facilitate the passing of information between the layers of software and hardware need to be defined. Between corresponding layers at different database management sites, protocols are needed to be defined to specify how the two database management systems separate layers exchange information and perform their defined functions. If these interfaces and protocols are adhered to, legacy database management systems can be linked together to form larger integrated database systems.

3.1.2 Purpose of Reference Model

The purpose of the reference model is twofold, first, during the early stages of the PRIS-TG efforts, the reference model focuses the PRIS-TG's standards viewpoints, which areas of a database system are standardized and what these standards will provide. The reference model will highlight the interfaces, protocols and API layers services provided by these protocols.

Second, a reference model will aid in the ability to convey the requirements and extensions needed for real-time database applications, as well as focus on which extensions are needed and how they impact a database system.

3.1.3 Intended Audience

This document is aimed at PRIS-TG participants as well as general DBSSG members. The developed model will aid in the development of recommendations for existing standards enhancements for real-time database management systems.

3.1.4 Relationship of Model to Standards

The model is designed to map layers to standards. As in the IEEE 802 standards [7], it is the intent for the model's layers to represent services, functions and protocols relevant for database management features available at each layer. It is the intent to specify these layers in relation to interfaces and services available to these APIs. Once accomplished these services and APIs can be mapped to standards, (to be adapted, adopted or developed), which will support the specified functionality.

3.1.5 Section Organization

The remainder of this section is broken up into two subsections. The first subsection addresses generic database management functionality within a database system, and does not focus on specifics of the interfaces between these functions. The second subsection

describes the PRIS-TG reference model in relation to these generic functions, focusing on the protocols and interfaces to and between these layers.

3.2 General Characteristics of a DBMS

3.2.1 Applications

Any application program which interfaces with and uses a database manager to store and retrieve data.

3.2.2 Applications Program Interface

The interface between the application software and any of the set of resources (including hardware and software) across which services are provided. [POSIX.0] An API is used to access basic services or to access facilites of other applications. The interface specifications enables the developer to write a program with little or no knowledge of the underlying implementation. [Quarterman–'UNIX, POSIX, and Open Systems'].

3.2.3 Translation services

Typical software services include tools to translate one data manipulation language into another, or translate one database definition language (relational-model) to another database definition language (object-model or network-model).

3.2.4 Transaction Formation

Data manipulation code must be organized into units of work for the database system on which to coordinate activities. Transaction formation provides this service. Database manipulation requests are bundled into transactions which are the units used for database concurrency control and recovery.

3.2.5 Transaction Management

Transaction management provides services for concurrency control, transaction recovery, and update synchronization.

3.2.6 Storage Management

In a typical database system the storage manager performs the accesses to physical storage and manages the physical storage allocated to the database system. This includes memory buffer management and input and output from secondary storage to the database buffers.

3.2.7 Remote access

Database systems are stored in a variety of ways. The database may be on a central server with a variety of client sites which make requests to the server, or the database may be distributed over the sites. In either case software must be written to access the database. Remote access services provide for client server interaction, distributed and heterogeneous database access and coordination.

3.2.8 Architecture Dependencies

Database systems have been developed which take advantage of special architectural features. For example specialized processors assist various transaction processing functions such as query processing. Specialized back-end storage devices enhance access to database's stored on secondary storage. These features must be addressed at the lower levels of the reference model.

3.2.9 Database

A database management system manages data stored in a database. A database consists of data, relationships amongst data, constraints on data and a schema defining how the data, relationships and constraints are organized [1].

3.3 The Real-time DBMS Reference Model

3.3.1 Introduction

The real-time database management systems reference model represents a baseline effort to define the static and active structure of a generic database management system. The model was not meant to be specific to any present or projected database management system or data model nor database management system. The model is intended to direct the reader in the understanding of the approach this task group will recommend for standardization. The model should be complete and of sufficient detail to allow standards development and discussion at various levels of a DBMS. PRIS-TG does not limit itself to the interface to the entire DBMS alone as have other standards activities. This would limit the ability to develop a meaningful real-time database management system standard which will serve ANSI into the year 2000 and beyond. PRIS-TG focuses on the interface to services and protocols available at each layer in the model.

3.3.2 Layered Oriented Approach to Model

In the layered approach services and protocols are partitioned into layers. The interface between these layers become the areas to examine for standardization. This method was chosen over the service-oriented approach because the layered model provides a clearer picture of how elements of a system communicate and interact.

3.3.3 PRIS-TG database management system Model Structure

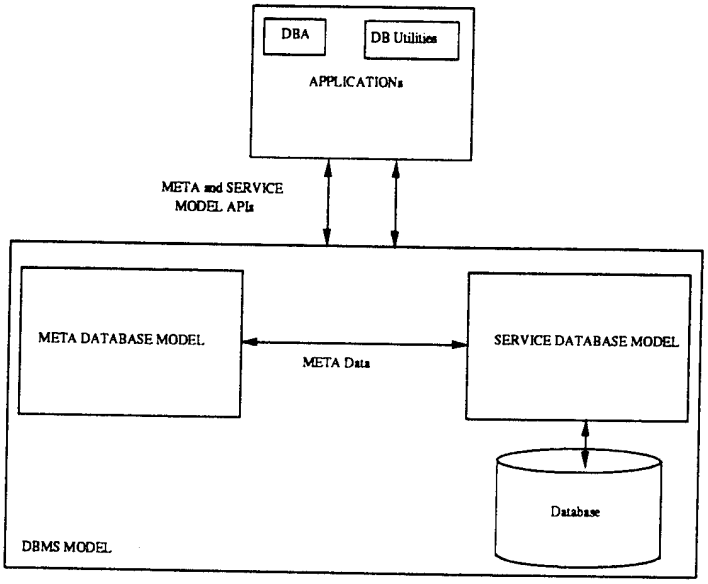

Figure 3.1: The PRIS-TG High Level DBMS Reference Model

The PRIS-TG database management systems reference model is composed of two elements; a meta database model element and a services database model element. The meta database model describes the structure of the data in the database. The services database model describes the operational components of a database system. (See Figure 3.1).

The meta database model captures the features of database structure as described in the ANSI/SPARC 3-schema architecture [10], but is extended for heterogeneous and distributed databases.

The services database model describes the operational features of the database system such as language processing, transaction formation, transaction execution, and storage management.

3.4 Real-time Database Management Systems Model

3.4.1 Structure of the real-time DBMS Reference Model

The baseline real-time database reference model consists of a meta database structural model and an active services execution model. The meta database model is an adaptation of the ANSI/SPARC three layer schema model [10]. The services database model is a refinement of the POSIX 1003.0 database transaction processing model.

The services database model consists of seven layers. These seven layers cover the functions of the database management system from applications down to physical interface with computing resources. The layers have been developed so as to be general in functionality in order to capture as many database designs and implementations as possible. Not all of the layers have API access. Those that do have API interfaces provide some set of services to applications. Those that do not have API access provide services in support of other PRIS-TG reference model layers.

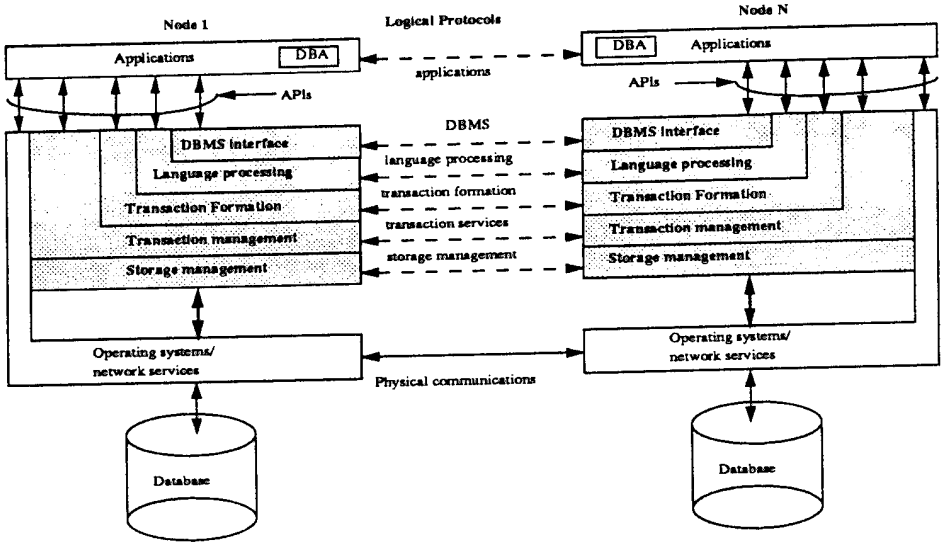

Figure 3.2: The PRISTG DBMS Services Reference Model

The real-time active database reference model is shown in Figure 3.2 There are seven layers, these are the applications layer, database management system interface layer, language processing layer, transaction formation layer, transaction management layer, storage management layer, and operating system and network layer. In the following subsections we discuss each of these layers in further detail.

3.4.2 Five Layer META Database Schema Model

The PRIS-TG reference model must be capable of allowing for data distribution and heterogeneity of database systems. The PRIS-TG has adapted the five schema model described in Sheth [9], which is an adaptation of the ANSI/SPARC three schema model [10].

The five schema model includes a local schema, component schema, export schema, federated schema and external schema. (See Figure 3.3).

The local schema is the conceptual schema for a local database in its native data

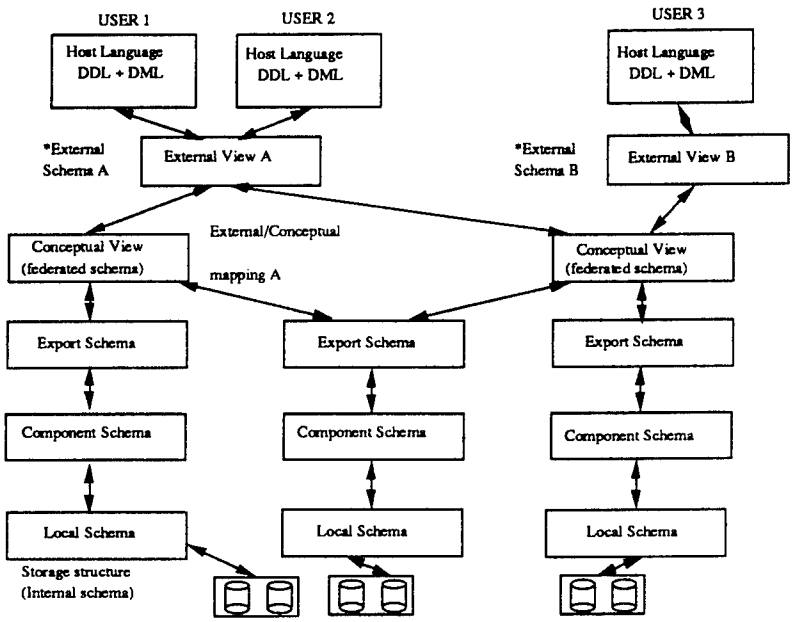

Figure 3.3: The PRISTG META DBMS Reference Model

model. For example a relational database described in the relational model, or an object oriented database described using object oriented techniques.

The component schema is a representation of the local schema in a common data model for the federated database system. A common data model provides the means by which different data from non-homogeneous databases can be translated and shared. The component schema provides the mapping between the local schema objects and universal schema objects which can be translated by other local database managers using the component schema.

The export schema provides the database administrator of a local schema the capability of limiting access to a subset of the local schema. The export schema describes which portion of the local schema is viewable to the outside world. This provides control over what is accessible.

The federated schema integrates multiple local export schemas into a single schema. This schema captures information about distribution and data organization. Data can be integrated into a federated schema due to usage patterns or data similarities. This schema provides a means to organize data from numerous databases dealing with similar data into a global unified schema.

The external schema defines user or applications views of a federated schema. This schema allows tailored views for specific applications, to provide additional access control or to provide additional integrity constraints on data being managed.

This decomposition of the passive structure of a database system provides needed meta data information about the database, which is required by the services database management elements to manage data in the database. The management functions, protocols and APIs are described in the following subsections.

3.4.3 Selectable API Service Policies

For all layers of the PRIS-TG reference model APIs have available to them a set of service policy qualifiers. These service policy qualifiers are set by a database administator in most cases, but may be available for some applications to set. The service policy qualifiers may include semantic specifications for control over; concurrency control protocol applied, the real-time scheduling of transactions and operations of transactions, data consistency, transaction recovery, access authorization, database partitioning and allocation, transaction execution granularity, active database triggers, replicated data, update synchronization, and transaction optimization methodology.

These service extensions provide the ability to allow the loosening of conventional database systems ACID properties. The ACID properties of transactions include Atomicity, Consistency, Isolation, and Durability. Atomicity ensures that a transaction is treated as a unit of operation. Consistency of a transactions operations ensures correctness of transformation of the database from an initial consistent state to a new consistent state. Isolation is the transaction property that requires transactions see a view of the database as if they were being performed alone. Finally Durability is the property of transactions which ensures that once a transaction is committed its results are permanent and cannot be removed from the database.

To allow loosened transaction ACID properties requires a means to allow either the database administrator, applications transaction writer or an ad hoc user the ability to select what service policy they wish. Some requirements for these selectable service policy parameters are addressed in the annotated references included in the PRIS-TG final report.

Real-time database management systems have a need for services that may span all layers or even bypass layers due to real-time systems unique needs. Selectable service policy extensions would provide services and protocols for applications to embed information allowing access to any of the layers described without the aid of the other layers.

Such service may be necessary where time is of the essence and strict adherence to conventional database management requirements for the ACID properties of transactions are not needed, or cannot be tolerated due to unique requirements of an application. Control over these ACID properties will be essential for next-generation database applications such as CAD, CAM, real-time control, medical monitoring, communications systems, and long duration transactions systems.

3.4.4 The PRIS-TG DBMS Reference Model Layers

The Applications layer

The applications layer contains user code and user provided services. When two user applications synchronize operations or communicate information, they alone determine how the applications operation's synchronization is to be performed, as well as how the information transferred will be used. Protocols are specified and adhered to by applications writers, not by the database system. The database system is accessible from this layer by the user applications through the database management system interface layer.

The Database Management System Interface layer

The DBMS interface layer makes functionality available to extract database manipulation requests from applications code. This layer provides services to format requests and responses into appropriate forms, and to present these reformatted responses or requests to the applications layer or the language processing layer respectively. This layer provides isolation and transparency of the database management systems details from the applications using these services. The database administration application accesses this layer to taylor or tune a DBMSs execution and structures.

For example if a graphics program is requesting an object from the database, the function of this layer is to interpret the request, form it into a database only request, and once the response is returned, organize it into an appropriate form for the graphics application.

The interface to the language processing layer is the local resident database manipulation language and database definition language, SQL, NDL, or an object based data manipulation language and data definition language

The Language Processing Layer

The language processing layer provides services and protocols to translate one data manipulation language or data definition language into another target data manipulation or data definition language. For example if the host machine is a network database management system and the requested data is on a relational system, the language processing layer needs services to translate the NDL request into an appropriate SQL request. In addition the response from the relational system would need to have services to reformat the response into the target network structure.

The application interface to the language processing layer is an API with data interchange services for the translation process in the native DDL or DML along with quality of service parameters and target DDL or DML.

The interface to the transaction formation layer is transactions in DML form, along with selectable service policy parameters.

The Transaction Formation Layer

The transaction formation layer provides protocols and services for local and remote service of database manipulation requests. The functions performed at this layer include execution plan computation and generation, access plan code optimization, decomposition into local and remote data manipulation components and transaction delineation.

The API to the language processing layer is in data manipulation language form with selectable service policy parameters. The interface to the transaction processing layer is in the form of transaction data access code which contains control parameters for execution control derived during optimization, translation and access code generation. As an example access codes could include the type of service required from the underlying concurrency control protocols, recovery protocols, real-time control protocols, security protocols, and triggering protocols.

The Transaction Management Layer

The transaction management layer provides services and protocols to enable control over execution of transaction execution plans, and bundling these into transactions based on provided information. The transaction management layer provides protocols for transaction concurrency control, recovery control, security enforcement, real-time access, active triggers, remote synchronization, update control, and consistency enforcement.

The interfaces with this layer are from applications layer, the transaction formation layer and the storage management layer. The applications layer and transaction formation layer interact with the transaction execution layer through database manipulation code and formatted responses along with selectable service policy parameters. The interface with the storage management layer includes requests to open data items to read data items, to restrict access to data items, requests to update data items, to set triggers to release triggers, to close data items, commands to release data items for others use, and control parameters to indicate which selectable service policy is needed from underlying storage management services.

The Storage Management Layer

The storage management layer provides protocols and services to manage data storage for the database management system. These services include protocols to interface with remote storage managers to read, write, and control access to used data, services to interface and control the level of service from the local and remote operating systems layer, services to interact with and control the selected service policy from the network communications layer.

The storage management layer interfaces with the transaction management layer and the operating system/network layer with selectable service policies. The interface with the transaction management layer consists of responses to read and write requests as well as to other command requests. For example if the transaction manager wishes

to configure a set of triggers, the storage manager must set these up and control their actions, only providing responses to the transaction services when appropriate. The interface with the operating system and network layer consists of requests for services to open or close files, to allocate database buffer working space, to alter the selected service policy given to a database task, such as a real-time access request, to provide a level of service from the communications protocols, or to provide selectable security or access control policies for database controlled assets.

The Operating Systems/Network Layer

This layer is the lowest level of the PRIS-TG reference model. The operating system component controls the use of computer resources, and software executions.

The network component controls the execution of network control and configuration services, and manages the transfer of information from one node in a network to another.

Interfaces to these layers would consist of calls and parameters which request services for the PRIS-TG reference models' other layers. Further details of these components will be extracted from the ANSI operating systems and network working groups. The PRIS-TG feels that these efforts may need to reflect evolving requirements from any real-time data management standards efforts into their operating systems and network standards.

Chapter 4

Summary

This presentation of the PRIS-TG database management systems reference model is meant as a starting point from which can evolve into an element of future standards for real-time databases and database management systems. Please review this document and provide comments to Dr. Paul Fortier at The University of Massachusetts at Dartmouth, Electrical and Computer Engineering Department, North Dartmouth, Mass. 02747. Email address Pfortier@umassd.edu.

Submission of Comments

When you submit comments on the reference model, please send them by electronic mail to the following address:

Pfortier@umassd.edu

If you do not have access to an electronic mail network, please send the comments by

postal mail or FAX to:

Dr. Paul J. Fortier
at The University of Massachusetts at Dartmouth
Electrical and Computer Engineering Department
RM 213-c
North Dartmouth, Mass. 02747
(508)-999-8544
FAX:(401)-841-2431

To assist us in the processing and tracking of your comments, please use the format below for each comment.

Commentors name
Commentors phone number
Commentors FAX number
Commentors Email address
Commentors Postal mail address
Date of comment generation
Section number commenting on
Version number of document
Topic of comment (e.g. Concurrency control)
Comment details
Rational for comment
End of comment marker

Acronymns

ACID	Atomicity, Consistency, Isolation, Durability
ANSI	American National Standards Institute
API	Application Program Interface
ASC X3	Accredited Standards Committee X3
ASC	(ANSI) Accredited Standards Committee
BLOB	Binary Large Object
DBA	Database Administrator
DBMS	Database Management System
DBSSG	X3/OMC Database Systems Study Group
DDL	Data Definition Language
DISWG	Database Management System Interface Standards Working Group
DML	Data Manipulation Language
DoD	Department of Defense
FIPS	Federal Information Processing Standard
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
NBS	National Bureau of Standards
NCCOSC	Naval Command Control Ocean Surveillance Center
NDL	Network Data Language
NGCR	Next Generation Computer Resources
NIST	National Institute of Standards and Technology
NRaD	Naval Command, Control and Ocean Surveillance Center, RDT and E Division
NUWC	Naval Undersea Warfare Center
OMC	Operational Management Committee
OMG	Object Management Group
OODB	Object-Oriented Database
OODBTG	X3/SPARC/DBSSG Object-Oriented Database Task Group
POSIX	Portable Operating System Interface
RT	Real-Time
SPARC	Standards Planning and Requirements Committee
SQL	Structured Query Language

Bibliography

[1] L. Cingiser. What is a database management system? Technical Report Working TM 93-1, SPAWAR, Arlington, Va., September 1993.

[2] D. Fisher. *Current Standards and Available Technology*. NGCR SPAWAR 331 2B2, Alexandria, Va., 1993.

[3] P. Fortier and J. Prichard. Concepts for a real-time structured database query language (RT-SQL). *Proceedings of the IFIP/IFAC Workshop on Real-time Programming*, June 1994.

[4] P. Fortier and J. Rumbut. Issues and concepts for a real-time database management. *Proceedings of the First International Conference on Electronics and Information Management*, August 1994.

[5] P. Fortier and CDR. G. Sawyer. DISWG a new player in NGCR open systems standards. *to appear in Computer Standards and Interfaces*, 1995.

[6] K. Gordon. *DISWG Database Management Systems Requirements*. NGCR SPAWAR 331 2B2, Alexandria, Va., 1993.

[7] IEEE. IEEE standards for local and metropolitan area networks: Overview and architecture. Technical report, IEEE Standards, Piscataway, NJ, January 1990.

[8] J. Peckham and P. Fortier. Operating systems support for next generation database management systems. Technical Report Working TM 93-2, SPAWAR, Arlington, Va., September 1993.

[9] A. Sheth and J. Larson. Federated database systems for manageing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3), September 1990.

[10] D. Tsichritzis and A.Klug. The ANSI/X3/SPARC DBMS framework. *Information Systems*, 3(4), 1978.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 1995	Final: Feb 1995

4. TITLE AND SUBTITLE

FINAL REPORT OF THE DBSSG PREDICTABLE REAL-TIME INFORMATION SYSTEMS TASK GROUP

5. FUNDING NUMBERS

PE: 0605866N
PN: X0706

6. AUTHOR(S)

Donna Fisher, Paul Fortier, David Hughes, Mayford Roark

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Command, Control and Ocean Surveillance Center (NCCOSC)
RDT&E Division
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TD 2747

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Space and Naval Warfare Systems Command
SPAWAR 331
2451 Crystal Drive
Arlington, VA 22245–5200

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report is the result of the study conducted by the American National Standards Institute (ANSI), Accredited Standards Committee (ASC), X3, Operational Management Committee (OMC), Database Systems Study Group (DBSSG), Predictable Real-Time Information Systems Task Group (PRISTG). Predictable real-time information systems are systems in which the correctness of a result is dependent on the validity and timeliness of data and of the operations on that data. The report describes a reference model for real-time information systems; but more importantly, the report discusses the recommendations for standardization in the area of real-time information management technology.

14. SUBJECT TERMS

Real-time Information Systems
Reference Model

15. NUMBER OF PAGES

63

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
Donna K. Fisher	(619) 553–4095	Code 411

INITIAL DISTRIBUTION

Code 0012	Patent Counsel	(1)
Code 0271	Archive/Stock	(6)
Code 0274	Library	(2)
Code 411	D. K. Fisher	(250)

Defense Technical Information Center
Alexandria, VA 22304–6145 (4)

NCCOSC Washington Liaison Office
Washington, DC 20363–5100

Center for Naval Analyses
Alexandria, VA 22302–0268

Navy Acquisition, Research and Development
Information Center (NARDIC)
Arlington, VA 22244–5114

GIDEP Operations Center
Corona, CA 91718–8000