TINA Workshop *Heidelberg, September 2, 1996*

RODAIN: A Real-Time Object-Oriented Database System for Telecommunications

Kimmo Raatikainen
University of Helsinki, Department of Computer Science
P.O. Box 26 (Teollisuuskatu 23), FIN-00014 University of Helsinki, Finland
Phone: +358 0 708 {44247,44243}
Fax: +358 0 708 44441
Email: {kimmo.raatikainen}@cs.Helsinki.FI

Abstract

Future telecommunication services will extensively exploit database technology. The persistent and temporal information needed in operations and management of the telecommunication networks and services will be in databases. The current Intelligent Network (IN) Recommendations of ITU-T imply that real-time transaction processing capabilities should be provided. Telecommunications Management Network (TMN) and Telecommunications Information Networking Architecture (TINA) are object oriented. Current and future mobile telephone systems have their own requirements for database services. An ideal database system supporting various telecommunications applications should be a fault-tolerant distributed real-time object-oriented database system.

In the research project Rodain the ultimate objective is to design and to specify a real-time object-oriented database architecture for telecommunications applications and to implement a prototype based on that architecture. The design and specification take place in 1996 while the prototype will be implemented in 1997.

Key Words: Telecommunications, Real-time databases, Object-orientation.

Premises

Databases will already in the near future have a central role in telecommunications networks. The information needed in operations and management of the nets will be collected into a logically uniform database. The world-wide nature of telecommunications prescribes that the only possibility to obtain the logical uniformity is the co-operation of autonomous databases.

During the next ten years great changes are expected in the ways of doing business on the telecommunications markets. The relative importance of traditional transmission and switching services will decrease. This is due both to liberalisation of telecommunications and to developments in the technology. We anticipate that in the future transmission and switching are done using generic technology based on international standards.

The ability of a telco operator to survive in the international competition will base either on huge volume of the few bulk services or on enhanced services. The ability of introducing enhanced services will primarily be based on the competence of responding in time to the needs on the markets. Future telecommunication architectures like IN and TINA are developed to meet the goal of rapid and efficient creation, deployment, and management of new services. An efficient and reliable database system is one of the key factors in meeting those goals.

Most of the recommendations issued in the 90's by ITU-T and other standardisation bodies are object oriented. Therefore, object orientation is a natural starting point for the telecommunications database system, too. However, current object-oriented databases are primarily designed for situations where objects have complex structures and transactions are long-living. In telecommunications the situations are different. Most objects have quite simple structures and transactions are short. In addition, we anticipate that real-time properties, guaranteed response time in particular, are soon needed.

Key Requirements for Databases in Telecommunications

In the research projects Darfin[1] and RODAIN[1] we have examined database needs in telecommunications. Based on our results and experience the most important features will be the *real-time* and *object-orientation*. Real-time transactions are really needed in telecommunications. In telecommunications we will need both *soft transactions* that may continue their execution after deadline but with a reduced lower priority and *firm transactions* that are terminated when their deadline expires. We do not believe that *hard transactions* will be used in near future because systems supporting hard transactions are too expensive for open telecommunication markets.

Below we summarize the key requirements while detailed discussion can be found in [4].

Data distribution: A set of database nodes may cooperate to accomplish database tasks. A node may request information from other nodes in case information needed is not locally available. It is possible to implement the underlying database system without data distribution. In such an implementation all applications use a single database server. This differentiates logical data distribution from physical data distribution among database nodes. The former is necessary but the latter is a decision internal to the design of the database architecture. Our current belief is that only a few requests need access more than one database node. We also believe that the request distribution between different database nodes can be arranged to be almost uniform. Thus using a distributed architecture allows several transactions to be run in parallel on several database nodes and give better throughput.

Data replication: It is stated on [1] that a database node contains customer and network data for real time access. This implies that the underlying database architecture must be able to answer requests in real time. We believe that currently most distributed operations are reads. Therefore, replication is an effective way to speed up distributed operations because it gives better throughput.

Object orientation: It is a common belief that the best long term telecommunications architectures are object oriented. The implication is that a database system must support object access. In other words, the database system must either be object oriented or have an object interface.

[1]Darfin and RODAIN are acronyms of the project names *Database ARchitecture For Intelligent Networks* and *Real-time Object-Oriented Database Architecture for Intelligent Networks*, respectively.

Object directories: The conceptual model of IN Service Control Function [2] defines a Service Data Object Directory that is used to access data stored in the SDFs. This implies that object directories must be supported. The invocation mechanism of CORBA is a good alternative to implement the functionality.

Multiple application interfaces: In telecommunications different architectures define different access interfaces. The IN Capability Set 1 defines IN Application Protocol (INAP) [3]. TMN has its own access methods based on the OSI (X.700) management. Agents and managers exchange information using the services of *Common Management Information Service Element* (CMISE). CMISE uses the *Common Management Information Protocol* (CMIP) and utilizes the services of *Association Control Service Element* (ACSE) and *Remote Operations Service Element* (ROSE). The current definition in TINA is based on TINA ODL which is quite similar to the OMG IDL interface.

Fault tolerance: Real time access implies that data must be continuously available. The result of the implications is that the database system must be fault tolerant. In the current definitions the maximum allowed down time is a few seconds in a year. However, the actual allowed down time may be minutes instead of seconds. Nevertheless, a fault tolerant database system is a necessity.

Real-time transactions: A real time transaction has an explicit deadline that the transaction must meet. Although the real time data access as stated in [1] does not directly imply that the underlying database system must support real time transactions, we believe that the most convenient way to support real time access to data is to use a real-time database system.

RODAIN Database Architecture

The RODAIN database architecture is designed to fulfil database needs of future telecommunications apllications. The architecture It is a *real-time object-oriented database management system architecture*. It has three primary levels: global architecture, node architecture, and object-oriented database management system. Below we briefly summarize each of the levels. Detailed description can be found in [5].

RODAIN DBS consists of a set of autonomous database nodes that interact with each other. Each node may communicate with one or more applications, and an application may communicate with one or more database nodes. Each node is replicated into a mirror node (Figure 1).

A RODAIN DBMS node consists of a set of subsystems that communicate with each other. The subsystems are: 1)*User Request Interpreter Subsystem*, 2) *Distributed Database Subsystem*, 3) *Fault-Tolerance and Recovery Subsystem*, 4) *Watchdog Subsystem*, and 5) *Object-Oriented Database Management System*; see Figure 2.

The *Object-Oriented Database Management System* (OO-DBMS) consists of a set of database processes that use database services to resolve requests from other subsystems, and a set of manager services that implement database functionality. The functional structure of RODAIN OO-DBMS is shown in Figure 3.

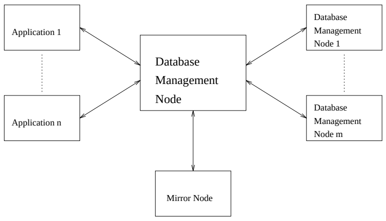

Figure 1: Overview of the RODAIN Database Architecture

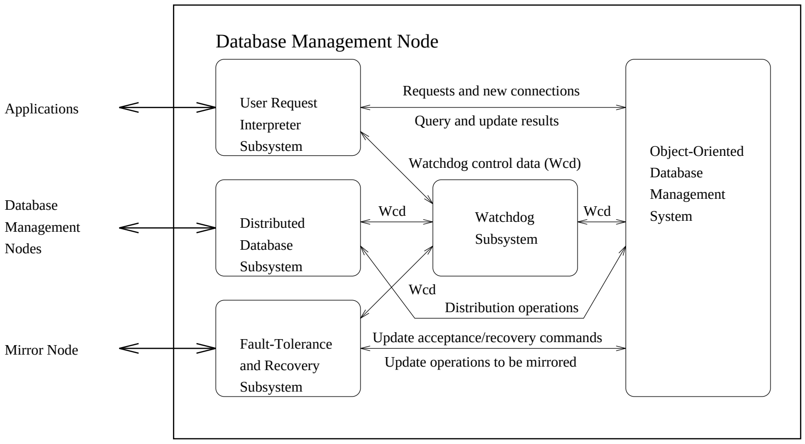

Figure 2: RODAIN Database Management Node

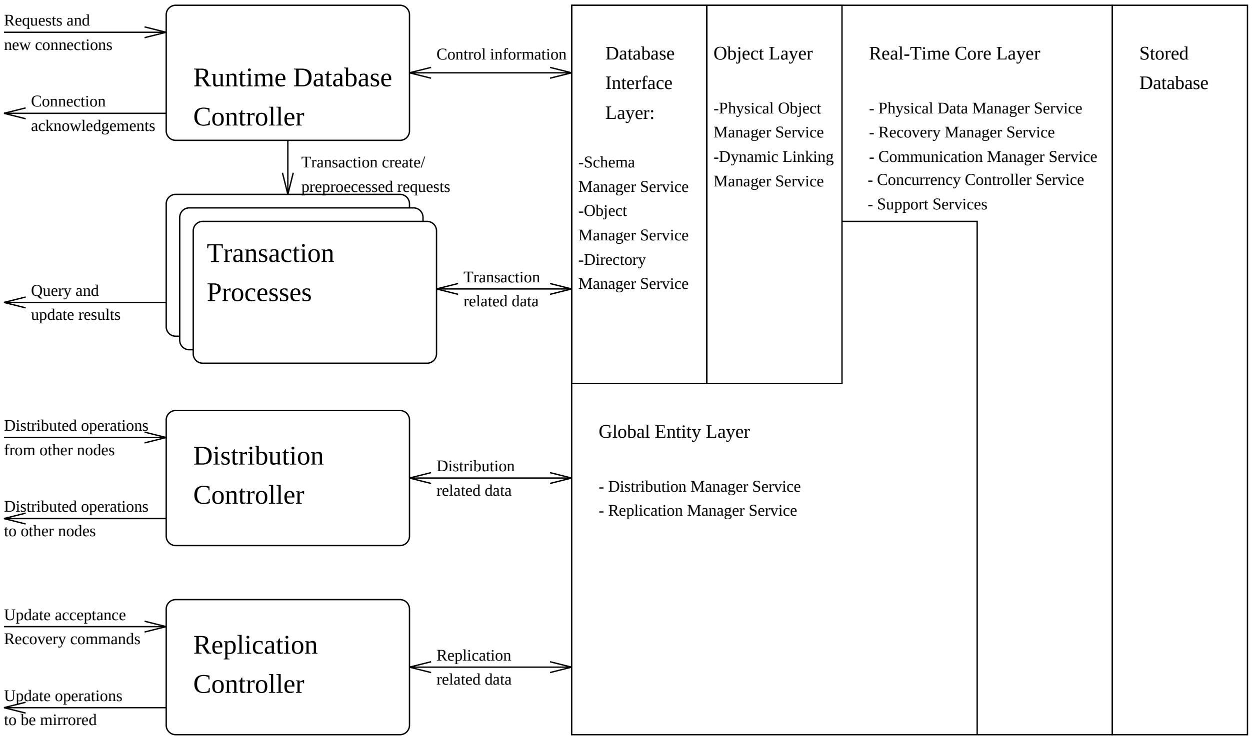

Figure 3: Processes and Database Services in RODAIN OO-DBMS Subsystem

Key Issues in Real-Time Databases

A *Real-Time Database System* (RT-DBS) is a transaction processing system that tries to satisfy the explicit timing constraints associated with each incoming transaction [6]. The timing constraint is usually given as a *deadline*. The ultimate goal of an RT-DBS is to maximize the fraction of transactions which meet their deadlines. This is in contrast to the goal of traditional database systems (DBMSs) which is to minimize the average response time.

At any time, all transactions in an RT-DBS can be divided into two classes: *feasible* transactions and *late* transactions. A feasible transaction still has a possibility to meet its deadline. A late transaction has either already missed its deadline or cannot any more meet its deadline. The RT-DBS executes feasible transactions until they complete in time or until the system detects that they can no more meet their deadlines. The policy how late transactions are handled depends on the application needs. The two primary policies are: 1) to continue the execution with a low priority and 2) to abort the transaction without restart.

Real-time systems are usually classified into three categories according to the effects that missing the transaction's deadline has. *Hard* deadline transactions are those which may result in a catastrophe if the deadline is missed. A hard real-time system must guarantee that any transaction never misses its deadline. *Firm* deadline transactions are those the results of which are not any more of any value if the deadline is missed but have no serious consequences. *Soft* deadline transactions are those the results of which gradually decrease their value when the deadlines are missed.

The deadlines raise several difficult questions. How are deadlines assigned? Are deadlines firm or soft? It must be remembered that a transaction accessing the database is only a subtask in a telecommunication service. Unfortunately *subtask deadline assignment* is still almost an unexamined problem; see e.g. [7]. The primary implication of transactions being subtasks is that the task can still meet (miss) its deadline even if the transaction misses (meets) its deadline.

Since the main goal of a firm or soft RT-DBS is to maximize the fraction of transactions that meet their deadlines, a key issue in transaction processing is *predictability* of transactions' execution and turnaround times. In a database system, data and hardware resource conflicts are an important source of unpredictability. Another significant source is the unpredictability of execution time caused by execution sequence depending on data values and by disk operations. Distributed databases have additional problems due to communication delays and site failures.

One of the primary difficulties in designing an RT-DBS is the fact that the concurrency control of data access must be combined with timing constraints. The two primary approaches to arrange the concurrency control in database systems are based on *locking* and on the *optimistic approach*. Most commercial DBMSs are based on *two-phase locking* (2PL) [8]. This is due to the fact that under most operating circumstances locking algorithms give shorter response times than optimistic algorithms; see e.g. [9].

Recent performance studies [10, 6, 11, 12, among many others] indicate that the results obtained from traditional DBMSs do not hold in RT-DBSs. The relative performance of concurrency control algorithms in real-time database systems is heavily affected by several factors including policy with late transactions, a priori knowledge of transaction resource requirements, and the availability of resources. This leaves the floor open to several design issues that must be evaluated.

The major design issues in concurrency control are conflict detection and conflict resolution. In addition, the run policy is an important issue. Below we discuss these issues separately although they are not independent. We also name several mechanisms proposed in the literature.

Conflict Detection. Conflicts in data granule access that violates the correctness criteria of a transaction can be detected either before the granule access (*pessimistic approach*) or after the granule access (*optimistic approach*). In the optimistic approach the correctness is later verified at the certification time. The mechanisms proposed to be used in the detection include *locks*, *time stamps*, and *serialization graphs*.

When a lock-based scheme is used, a lock must be obtained before an access is allowed. In pessimistic approaches, such as various 2PL schemes, the lock is either shared (*read*) or exclusive (*write*). When an optimistic approach is based on locks, there are *weak* locks as well as normal (*strong*) locks. When a transaction has a weak lock, either read or write lock, on a granule, it only indicates that the granule is accessed. A weak lock conflicts with a strong write lock but is compatible with other weak locks (both read and write) and with strong read locks. Typically, the weak locks are converted to strong locks at the beginning of the certification time.

In some situations two different kinds of write locks may be profitable: an *update lock* (ordinary write lock) and a *replace lock* (blind write lock). Two *blind write* operations do not conflict with each other, since the new value becomes outside the database.

Conflict Resolution. A conflict resolution mechanism is invoked once a conflict is detected. In the resolution at least one transaction is penalized through appropriate actions selected by the conflict resolution mechanism. The actions most commonly used are *blocking* (wait), *abort* (restart), *multiversioning*, and *dynamically readjusting the serialization order* (delayed commit or abort). It should be noted that the resolution mechanism is not independent from the detection mechanism. For example, if the conflict is detected after the granule access, the action must be abort.

Several conflict resolution mechanisms have been proposed in the literature. Variants of the 2PL include *no wait* [13], *wound-wait* [14], *wait-die* [14], *running priority* [15], *wait depth limited* [16], and *delayed abort* [17]. In addition to the pure *optimistic concurrency control* (OCC) *broadcast OCC* [18] and a combination of pure and broadcast OCC [19] have been proposed. *Locking with deferred blocking* [20] is a scheme that combines 2PL and OCC. Multiversioning opens a wide variety of conflict resolution mechanisms. A detailed comparison can be found in [21].

Run Policy. When abort is used as an action in the conflict resolution, a transaction may need to be restarted several times. The restarts create an effect known as *buffer retention* [22]. In the first run it may be profitable to run a transaction that will be aborted to end so that all granules accessed will be in the buffer and the CC manager learns all the locks the transaction needs.

Integrating Concurrency Control and Real-Time Scheduling

There are four major problem areas when a real-time transaction scheduler is to be developed: *priority inversion*, *scheduling priority*, *overload management*, and *IO scheduling*.

Priority Inversion. A phenomenon called *priority inversion* arises when a low-priority transaction blocks a high-priority transaction due to the concurrency control. At least three different approaches have been proposed to solve the problem. In *priority inheritance* [23] the low-priority transaction inherits the priority of the high-priority transaction. In the *priority ceiling protocol* [24] blocking time of the high-priority transaction is bounded. The *dynamic priority ceiling protocol* [25] is a modification of the previous based on earlist deadline first instead of fixed external priorities.

Scheduling Priority. *Earlist Deadline First* (EDF) is most widely used in the experimental real-time database systems as the scheduling priority. It can minimize the number of late transactions expect when the system is highly loaded [26]. Another drawback in the EDF is that large transactions are discriminated. Alternatives to the EDF include *Least Slack First* (LSF), *fixed priorities*, *criticality of transaction*, *weighted priority scheduling*: see e.g. [10, 27, 24, 28, 29]. The fundamental design criteria are the heterogenity of transactions and transaction characteristics available a priori.

Overload Management. If the EDF scheduling is used, then occasional periods of high load can significantly decrease the fraction of transactions that complete in time. In order to cope with the overload situations the scheduling needs an overload management policy. In [30] an *adaptive earlist deadline* (AED) scheduling scheme was proposed. In AED, arriving transactions are placed in a HIT group or a MISS group depending on the system condition. In the HIT group the transactions are treated in the normal way but in the MISS group the transactions get low priority. In [31] an *adaptive earlist virtual deadline* (AEVD) scheduling scheme was proposed. AEVD is an extension of AED that tries to take into account the fairness issue in an overloaded system.

IO Scheduling. When the primary copy of the database is on the disk, IO operations are extremely important for the RT-DBS. Traditional disk scheduling algorithms should be replaced by novel ones as in [32] and in [33]. When a large fraction of database can be held in the main memory buffers, the situation is completely different.

Correctness. Yet another interesting area is correctness. In real-time applications concurrent transactions do not necessarily need serializability to produce a correct result. In many situations the result of a transaction can be regarded as correct as long as the result corresponds to the real-world situation presented in the database even if the transaction does not serialize with other transactions.

Summary

The database system should be flexible due to different kinds of needs present in telecommunications. We believe that the required functionality is the easiest to implement with an object based approach. An open question is if a fully object-oriented architecture—something similar to ODMG-93—is needed. Currently we believe that it is better to use a reduced set of object features than to use a fully object-oriented database when we are developing a real-time database to be used in telecommunications.

We can gain several advantages when we combine real-time and object orientation. Data objects can have attributes that specify the correctness criteria. Operations can have attributes that tell the resource consumptions of the operations, what kind of locks they needed (read, update, or replace). Transactions are also objects; either transient or persistent. They can have attributes that specify the priority, deadline, criticality, correctness criterion, among other things.

Acknowledgements

This work has been carried out in the research projects Darfin (1994–5) and RODAIN (1996–). The former was funded by Telecom Finland and the latter by the Finnish Technology Development Center (TEKES) together with Nokia Telecommunications, Solid Information Technology, System Software Partners, and Telecom Finland.

References

[1] ITU-T, "Intelligent Network Distributed Functional Plane Architecture," Recommendation Q.1204, 1993.

[2] ITU-T, "Distributed Functional Plane for Intelligent Network CS-1," Recommendation Q.1214, 1993.

[3] ITU-T, "Interface Recommendation for Intelligent Network CS-1," Recommendation Q.1218, 1994.

[4] Raatikainen, K. and Taina, J., "Design Issues in Database Systems for Telecommunication Services," Proc. of IFIP IN'95 Working Conference, 1995, pp. 71–81.

[5] Taina, J. and Raatikainen, K., "An Experimental Real-Time Object-Oriented Database Architecture for Intelligent Networks," Int. J. of Engineering Intelligent Systems for Electrical Engineering and Communications, Special Issue on Databases and Telecom, to appear.

[6] Haritsa, J. R., Carey, M. J., and Livny, M., "On Being Optimistic about Real-Time Constraints," Proce. of the 9th ACM Symposium on Principles of Database Systems, ACM, 1990, pp. 331–343.

[7] Kao, B. and Garcia-Molina, H., "Deadline Assigment in a Distributed Soft Real-Time System," Proce. of the 13th Int. Conf. on Distributed Computing Systems, IEEE , 1993, pp. 428–437.

[8] Eswaran, K. P., Gray, J. N., Lorie, R. A., and Traiger, I. L., "The Notions of Consistency and Predicate Locks in a Database System," Commun. ACM 19, 11, 1976, pp. 624–633.

[9] Agrawal, R., Carey, M. J., and Livny, M., "Concurrency Control Performance Modeling: Alternatives and Implications," ACM Trans. on Database Systems 12, 4, 1987, pp. 609–654.

[10] Abbott, R. and Garcia-Molina, H., "Scheduling Real-Time Transactions: A Performance Evaluation," ACM Trans. on Database Systems 17, 3, 1992, pp. 513–560.

[11] Huang, J., Stankovic, J. A., Ramamritham, K., Towsley, D., and Purimetla, B., "Priority Inheritance in Soft Real-Time Databases," Journal of Real-Time Systems 4, 2, 1992, pp. 243–268.

[12] Lin, K.-J. and Son, S. H., "Concurrency Control in Real-Time Databases by Dynamic Adjustment of Serialization Order," Proc. of the 11th Real-Time Systems Symposium, IEEE, 1990, pp. 104–112.

[13] Tay, Y. C., Suri, R., and Goodman, N., "A Mean Value Performance Model for Locking in Databases: The No-Waiting Case," Journal of the ACM 32, 3, 1985, pp. 618–651.

[14] Rosenkrantz, D. J., Stearns, R. E., and Lewis II, P. M., "System Level Concurrency Control for Distributed Database Systems," ACM Trans. on Database Systems 3, 2, 1978, pp. 178–198.

[15] Franaszek, P. A. and Robinson, J. T., "Limitations of Concurrency in Transaction Processing," ACM Trans. on Database Systems 10, 1, 1985, pp. 1–28.

[16] Franaszek, P. A., Robinson, J. T., and Thomasian, A., "Concurrency Control for High Contention Environments," ACM Trans. on Database Systems 17, 2, 1992, pp. 304–345.

[17] Yu, P. S., "Effective Concurrency Control With No False Killing," IBM Tech. Disclosure Bull. 33, 2, 1990, pp. 193–194.

[18] Kung, H. T. and Robinson, J. T., "On Optimistic Methods for Concurrency Control," ACM Trans. on Database Systems 6, 2, 1981, pp. 213–226.

[19] Yu, P. S., Dias, D. M., and Lavenberg, S. S., "On the Analytical Modeling of Database Concurrency Control," Journal of the ACM 40, 4, 1993, pp. 831–872.

[20] Yu, P. S. and Dias, D. M., "Performance Analysis of Concurrency Control Using Locking With Deferred Blocking," IEEE Trans. on Software Engineering SE-19, 10, 1993, pp. 982–996.

[21] Wu, K.-L., Yu, P. S., and Chen, M.-S., "Dynamic Finite Versioning: An Effective Versioning Approach to Concurrent Transaction and Query Processing," Proc. of Int. Conf. on Data Engineering, IEEE, 1993, pp. 577–586.

[22] Yu, P. S. and Dias, D. M., "Analysis of Hybrid Concurrency Control for a High Data Contention Environment," IEEE Trans. on Software Engineering SE-18, 2, 1992, pp. 118–129.

[23] Sha, L., Rajkumar, R., and Lehoczky, J. P., "Priority Inheritance Protocols: An Approach to Real-Time Synchronization," IEEE Trans. on Computers C-39, 9, 1990, pp. 1175–1185.

[24] Sha, L., Rajkumar, R., Son, S. H., and Chang, C.-H., "A Real-Time Locking Protocol," IEEE Trans. on Computers 40, 7, 1991, pp. 793–800.

[25] Chen, M.-I. and Lin, K.-J., "Dynamic Priority Ceilings: A Concurrency Control Protocol for Real-Time Systems," Journal of Real-Time Systems 2, 4, 1990, pp. 325–346.

[26] Liu, C. L. and Layland, J. W., "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment," Journal of the ACM 20, 1, 1973, pp. 46–61.

[27] Buchmann, A. P., McCarthy, D. R., Hsu, M., and Dayal, U., "Time-Critical Database Scheduling: A Framework For Integrating Real-Time Scheduling and Concurrency Control," Proc. of the 5th International Conference on Data Engineering, IEEE, 1989, pp. 470–480.

[28] Huang, J., Stankovic, J. A., Towsley, D., and Ramamritham, K., "Experimental Evaluation of Real-Time Transaction Processing," Proc. of the 10th Real-Time Systems Symposium, IEEE, 1989, pp. 144–153.

[29] Huang, J., Stankovic, J. A., Ramamritham, K., and Towsley, D., "Experimental Evaluation of Real-Time Optimistic Concurrency Control Schemes," Proc. of the 17th VLDB Conference, Morgan Kaufmann, 1991, pp. 35–46.

[30] Haritsa, J. R., Livny, M., and Carey, M. J., "Earliest Deadline Scheduling for Real-Time Database Systems," Proc. of the 12th Real-Time Symposium, IEEE, 1991, pp. 232–242.

[31] Pang, H., Livny, M., and Carey, M. J., "Transaction Scheduling in Mulriclass Real-Time Database Systems," Proc. of the 13th Real-Time Systems Symposium, IEEE, 1992, pp. 23–34.

[32] Sprunt, B., Kirj, D., and Sha, L., "Priority-Driven, Preemptive I/O Controllers for Real-Time Systems," Proce. of Int. Symp. on Computer Architecture, ACM, 1988, pp. 152–159.

[33] Abbott, R. and Garcia-Molina, H., "Scheduling I/O Requests with Deadlines: A Performance Evaluation," Proc. of the 11th Real-Time Systems Symposium, IEEE, 1990, pp. 113–124.

Appendix: RODAIN Position Statement

Premises

Databases will already in the near future have a central role in telecommunications networks. The information needed in operations and management of the nets will be collected into a logically uniform database. The world-wide nature of telecommunications prescribes that the only possibility to obtain the logical uniformity is the co-operation of autonomous databases.

During the next ten years great changes are expected in the ways of doing business on the telecommunications markets. The relative importance of traditional transmission and switching services will decrease. This is due both to liberalisation of telecommunications and to developments in the technology. We anticipate that in the future transmission and switching are done using generic technology based on international standards.

The ability of a telco operator to survive in the international competition will base either on huge volume of the few bulk services or on enhanced services. The ability of introducing enhanced services will primarily be based on the competence of responding in time to the needs on the markets. Future telecommunication architectures like IN and TINA are developed to meet the goal of rapid and efficient creation, deployment, and management of new services. An efficient and reliable database system is one of the key factors in meeting those goals.

Most of the recommendations issued in the 90's by ITU-T and other standardisation bodies are object oriented. Therefore, object orientation is a natural starting point for the telecommunications database system, too. However, object-oriented databases are still emerging. They are primarily designed for situations where objects have complex structures and transactions are long-living. In telecommunications the situations are different. Most objects have quite simple structures and transactions are short. In addition, we anticipate that real-time properties, guaranteed response time in particular, are soon needed.

Due to the world-wide nature of telecommunications international standards—both official and de facto—are mandatory. In this project we have identified the following standards as our prerequisites:

- IN recommendations by ITU-T and ETSI
- TINA architecture by Tina Consortium
- TMN by ITU-T and ETSI
- RM-ODP by ISO and ITU-T
- CORBA by Object Management Group (OMG)
- ODMG-93 by Object Database Management Group (ODMG)

Objectives

In the research project Rodain we continue the work done in the Darfin project (Database Architecture for Intelligent Networks) funded by Telecom Finland. The ultimate objective of the Rodain project is to design and to specify a real-time object-oriented database architecture for Intelligent Networks and to implement a prototype based on that architecture. The design and specification take place in 1996 while the prototype will be implemented in 1997.

In particular, we will carry out in 1996 the following research activities:

1) Evaluation of Concurrency Control Algorithms for Real-Time Databases

2) Specification of Data Models for Real-Time Databases in Telecommunications
3) Evaluation of Platforms for Real-Time-Databases
4) Specification of RODAIN Database Architecture and Interfaces

Partners

The Rodain project is a collaborative enterprise of Nokia Telecommunications, Solid Information Technology, Systems Software Partners, Telecom Finland, and Department of Computer Science in University of Helsinki. The Rodain project is partially funded by the Finnish Technology Development Centre (TEKES).

Darfin Inheritance and Rodain Directions

The results of the Darfin project are published in several public reports and international publications. Below we briefly summarise the key results and their implications to the research to be done in the Rodain project.

High-Level Information Model

On an abstract level we can interpret that a telephone call is controlled by call processing logic and data. The data can be termed as Call Processing Data Object. It is a dynamic object that is created during the call establishment. During the call the object may change as a result of the so-called mid-call service control actions. Logically the data object can be divided into two parts: Calling Party Object and Called Party Object. Moreover, the data object may have attributes that trigger actions to modify and/or to create persistent data objects when the call is terminated.

Figure 4 summarises our abstract information model of a telephone call. It should be noted that the call is not necessarily between two telephones. The called party may also be an information processing system. User management of services is a typical case in which the called party is an information processing system.

The Call Processing Data Object is typically created in two phases. The first phase consists of creating the Calling Party Object. When the switch detects that the hook is lifted up, a query is triggered. The query builds up the object by retrieving the service profile for the originating line from the persistent subscriber data object.

The Called Party Object is created in the second phase. This object depends on the access number dialled. Typically, the dialled digits are given as the key in the query that creates the Called Party Object. The query builds up the object by retrieving the service profile for the destination line from the persistent subscriber data object.

Real-Time Transactions

Based on the IN Recommendations of ITU-T (Q.1204 and Q.1211) we have concluded that real-time transaction processing capabilities will be needed. Real-Time Database System (RT-DBS) is a transaction processing system that tries to satisfy the explicit timing constraints (deadlines) associated with each incoming transaction. The ultimate goal of any RT-DBS is to maximise the

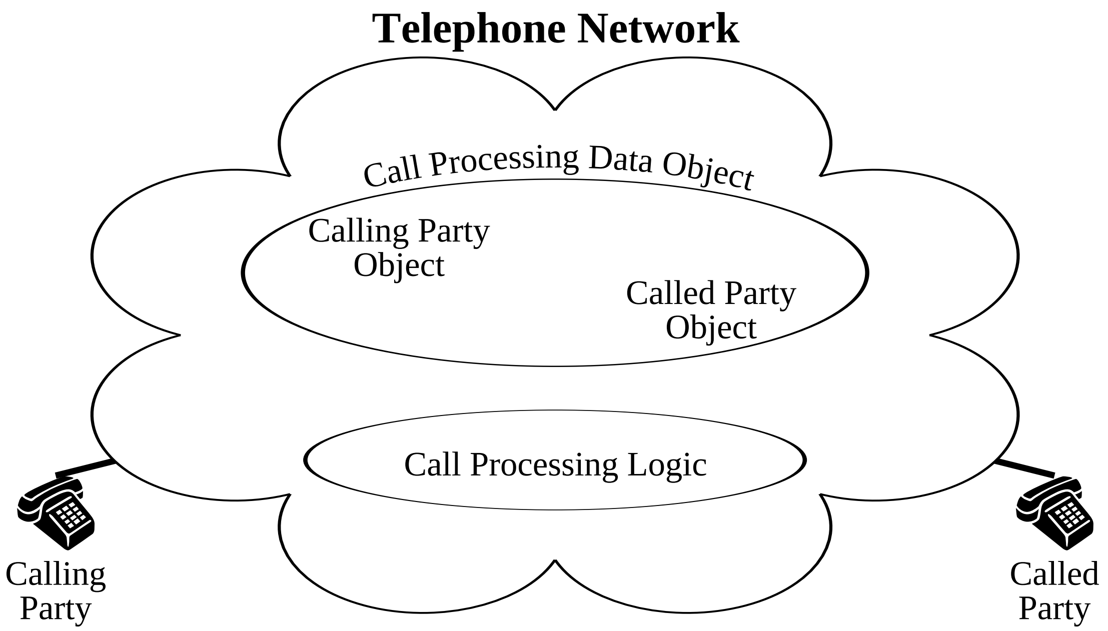

Figure 4: A High-Level Information Model of Phone Call

fraction of transactions that meet their deadline. This is very different from the goal of traditional database management systems (DBMSs) that try to minimise the average response time.

When a database system for telecommunication services is to be designed, the list of most important issues includes real-time. We anticipate that both firm and soft deadlines will be needed. When the deadline is firm (outcome of a late transaction is useless or harmful), then the late transaction (or transaction that can no more meet its deadline) can be aborted without restart. When the deadline is soft (outcome of a late transaction gradually loses its value), then the late transaction is continued with a lower priority. We do not believe that hard (meeting the deadline must be guaranteed) deadlines are needed in telecommunications applications.

One of the primary difficulties in designing an RT-DBS is the fact that the concurrency control of data access must be combined with real-time scheduling. The major design issues [see i.e. Raatikainen and Taina 1995; Yu et al. 1994] in concurrency control are conflict detection and conflict resolution. Of the algorithms proposed in the literature our current favourite is the 2PL-HP algorithm [Haritsa et al. 1992]. In that algorithm a transaction having a higher priority aborts the transactions with lower priorities when a lock conflict occurs. The transactions with lower priorities are later restarted if the system still estimates that their deadlines will not expire. However, the Rodain project will carry out an evaluation study of the concurrency control algorithms for real-time databases.

In the Darfin project we have learned the basic understanding of real time, object orientation, and database needs in telecommunications. In particular, the simple prototype implemented gave us valuable experience on the problem space. The database architecture should be flexible due to different kinds of needs present in telecommunications; for details see Taina [1995]. Based on the experience we believe that the easiest way to implement the required functionality is to use an object-based approach. An open question is if a fully object-oriented architecture—something similar to ODMG-93 [Cattell 1994]—is really needed. Currently we believe that it is better to use a reduced set of object features than to use a fully object-oriented database when we are developing a real-time database to be used in telecommunications. In Rodain we will start from a real-time database system and include the necessary object features.

Object Models

In the Darfin project we evaluated four different object models: X.500, X.700, OMG, and ODMG; for details see Raatikainen et al. [1995].

Of those models the X.500 model is the most mature. However, there are some major shortcomings in X.500. The two most serious ones are the formal organisation of data into a directed graph and the lack of atomic transactions. Minor shortcomings include the lack of multiple inheritance, operations, relationships, extents, keys, and transient objects.

The formal organisation as a directed graph makes it complicated to use X.500 as the object model of the database system for telecommunication services. As pointed out in the usage scenario, the queries have different criteria in the select operation: the line number and the dialed digits. Moreover, the management needs at least two different views into the data: one in which all services of a single subscriber can be found and another in which all subscribers of a single service can be found.

The Management Information Model defined in the X.700-series of ITU-T Recommendations is a full object model. Although a distributed management system as defined in X.700 is actually quite close to a database management system, most drawbacks identified in X.500 are also present in the X.700 Information Model. In particular, it is stated in the X.700 recommendations that the local databases containing the information are independent from the recommendations. When taking the management's point of view, this is flexible. However, the approach leaves everything open when the point of view is that of an object-oriented database system.

The ultimate goal in OMG is portability and interoperability. These goals are also very important in telecommunications. However, they are so general that the OMG Core Object Model does not take into account the specific needs of databases. Therefore, our choice as the baseline object model is the ODMG Object Model. However, the OMG - particularly the OMG Object Request Broker - will obviously have an important role in future telecommunication systems.

The ODMG-93 is designed to be a general-purpose object database system. Therefore, all features presented in the ODMG-93 Release 1.1 [Cattell 1994] are neither appropriate nor needed in a real-time database system; see Taina and Raatikainen [1995]. An "RT-ODMG", that is a real-time ODMG, will be a subset of ODMG-93 with predefined types of real-time object and real-time transaction. Type real-time object will include attributes that specify the correctness criteria and resource demands of operations. Type real-time transaction will include attributes that may overwrite the correctness criteria of operations and that specify the deadline and criticality for the instances of this type.

In Rodain we will develop a data model tailored for the needs of telecommunications. We will examine how the object model of ODMG-93 should be modified to meet those needs. The interfaces of data access will be flexible. In particular, applications using the database services can use INAP as specified in the IN Capability Set 1 of ITU-T (Q.1218), CMIP as specified in

X.700, ODP channels as specified in RM-ODP (X.900 series), and CORBA as specified by OMG. In addition, we will provide an Object Query Language (OQL) interface which will be a subset of the ODMG OQL.

Correctness Criteria for Real-Time Transactions

Correctness of transaction results in real-time databases is an interesting area of research. In real-time applications concurrent transactions do not necessarily need serializability [Eswaran et al. 1976] to produce a correct result. In many situations the result of a transaction can be regarded as correct as long as the result corresponds to the real-world situation presented in the database even if the transaction does not serialise with other transactions.

An interesting correctness criteria called τ-serializability was introduce in Raatikainen et al. [1995]. It can be used to reduce the number of serialisation conflicts. In τ-serializability a transaction is allowed to read old data as long as the update is not older than a predefined time. In the Rodain project we will further refine and examine the basic concept of τ-serializability.

References

Cattell, R. G. G. (Ed.). 1994. The Object Database Standard: ODMG-93. Version 1.1 edition. Morgan Kauffmann, San Francisco, Calif.

Eswaran, K. P., Gray, J., Lorie, R. A. and Traiger, I. L. 1976. The Notions of Consistency and Predicate Locks ib a Database System. Communications of the ACM 19, 11 (November): 624-633.

Haritsa, J. R., Carey, M. J., and Livney, M. 1992. Data Access Scheduling in Firm Real-Time Database Systems. Journal of Real-Time Systems 4, 3 (September): 203-241.

Raatikainen, K., Karttunen, T., Martikainen, O., and Taina J. 1995. Evaluation of Database Architectures for Intelligent Networks. Proc. of Telecom95 Technical Summit, Vol 2, 549-553. ITU, Geneva, Switzerland.

Raatikainen, K. and Taina, J. 1995. Design Issues in Database Systems for Telecommunication Services. Proc. of IFIP TC6 Working Conference on Intelligent Networks. August, Copenhagen, Denmark.

Taina, J. 1995. Requirements Analysis for Database Services in Telecommunications. Report C-1995-17. Department of Computer Science, University of Helsinki.

Taina, J. and Raatikainen, K. 1995. Design Issues and Experimental Database Architecture for Telecommunications. Intelligent Networks and New Services. Chapman and Hall, London, UK.

Yu, P. S., Wu, K.-L., and Son, S. 1994. On Real-Time Databases: Concurrency Control and Scheduling. Proceedings of the IEEE 82,1 (January): 140-157.

Darfin Publications

Kimmo Raatikainen: Database Access in Intelligent Networks, in Intelligent Networks (Chapman & Hall, 1995), pp.173-193

Juha Taina: Problem Classes in Intelligent Network Database Design, in Intelligent Networks (Chapman & Hall, 1995), pp.194-207

Kimmo Raatikainen: Information Aspects of Services and Service Features in IN CS-1, Technical Report C-1994-45, Univ. of Helsinki, Dept. of Computer Science

Juha Taina: Evaluation of OMG, ODMG, X.500, and X.700 Data Models, Technical Report C-1994-54, Univ. of Helsinki, Dept. of Computer Science

Kimmo Raatikainen and Juha Taina: Design Issues in Database Systems for Telecommunication Services, Technical Report C-1995-16, Univ. of Helsinki, Dept. of Computer Science. Also in Proc. IFIP IN'95 Conference

Juha Taina: Requirements Analysis for Database Services in Telecommunications, Technical Report C-1995-17, Univ. of Helsinki, Dept. of Computer Science

Juha Taina, Tapani Karttunen, Olli Martikainen, and Kimmo Raatikainen: Evaluation of Database Architectures for Intelligent Networks, Technical Report C-1995-29, Univ. of Helsinki, Dept. of Computer Science

Juha Taina, Mika Rautila, and Kimmo Raatikainen: An Experimental Database Architecture for Intelligent Networks, Technical Report C-1995-30, Univ. of Helsinki, Dept. of Computer Science. Also in Proc. IFIP IN'95 Conference

Juha Taina and Harri Töhönen: On Implementing a Prototype IN Database Architecture, Technical Report C-1995-42, Univ. of Helsinki, Dept. of Computer Science.