46

DATABASE MANAGEMENT FOR TELECONTROL SYSTEMS

Working Group 01
of Study Committee 35

April 1997

DATABASE MANAGEMENT FOR TELECONTROL SYSTEMS

Working Group 01
of Study Committee 35

Final Version - February 96

Authors :
David COOPER
Malcolm TUCKER
Peter LIGTVOET
António CARVALHO

Special thanks to
D. MARSH from WSL
R. SMITS from SEP
for their contribution to this work

INDEX

Page

Page

INDEX

DATABASE MANAGEMENT FOR TELECONTROL SYSTEMS

1. INTRODUCTION

1.1. Scope of the Technical Brochure

The scope of this Technical Brochure is:

- To point out the actual needs of utilities concerning real time databases, definition, implementation and handling, including the new needs created by the changes in customer needs.
- To give a realistic overview of the actual state of the art in the real time database field, aiding the users who need to buy a new system and those who have to upgrade the capacity of existing systems for new applications, new RTU's. etc..
- To try to foresee what can appear in the medium term in this field and how to prepare the evolution of the systems in service.

It aims to give a positive contribution to:

- Those who now have systems with several real time databases loosely coordinated and are facing increasing difficulty in maintaining the complete system.
- People who are going to upgrade their old systems and that have to find a solution to improve their database management capability.
- Users that have to buy new systems now when open systems, standardization, and new software techniques are available but the right way to follow is not yet totally defined.
- Utilities that want to foresee the real time database architecture of the future.

The Technical Brochure has the following organisation:
Chapter 1 - "Introduction" - gives the scope and motivation for the Technical Brochure.
Chapter 2 - "Database background" - gives a brief review of some database concepts and a set of useful definitions used in the Technical Brochure.
Chapter 3 - "Goals and requirements" - discusses database management from the system users point of view and deals with the specifications.
Chapter 4 - "Current solutions" - surveys the current solutions to the problems of defining a network under control.
Chapter 5 - "New OSA Databases" - looks from the general field of Information Technology to foresee new developments in the medium-term perspective that can be suitable for the real time environment.

1.2. Motivation for the Technical Brochure

This Technical Brochure is intended to help those who have to define and implement a database management structure for their telecontrol systems.

To perform real time control it is necessary to handle information. This has been true even in the past when the old fashion hardwired technologies were used. The way information is organised has changed with the evolution of technologies and the complexity of the applications.

When computers began to appear in control systems the information acquired near the process was organised in files with a format that was consistent with the system which used it. Each manufacturer, and even each equipment type of the same manufacturer, had its own data organisation.

Control centres process information relating to large installations. In the first generation of telecontrol systems, RTU's and central equipment were both part of proprietary solutions exclusively tailored for each particular application.

Progress in technology allowed a tremendous increase in the use of microprocessors in industrial equipment, hence information began to appear organised in databases structures, and at the Control Centres the software became more elaborate increasing the number of applications with specific requirements in terms of information performance.

With such heterogeneous environments it is difficult to have only one type of database structure. The information consistency has to be assured and so database management is of utmost importance.

Nowadays the age of many control systems in service creates extra difficulties concerning database management.

It is strongly desirable that systems can evolve instead of having major replacements at a particular stage of their lives.
Major replacements always entail considerable effort to minimise the time during which systems are non operational. Further, the investment in telecontrol systems is large and has to be preserved as far as possible.

For these reasons a smooth upgrade of existing systems is strongly recommended, replacing only those parts that became obsolete instead of a complete renewal.

This means that in the systems to be upgraded it is important to be able to share the databases of the old equipments with those of the new ones, frequently belonging to different manufacturers.

On the other hand, the changing framework of electricity utilities imposes a tremendous requirement for flexibility in handling the information available, to allow the implementation of several organisational scenarios for the new enterprises, many of them implying considerable changes in the role of existing control centres.

2. DATABASE BACKGROUND

2.1. Database role in telecontrol systems

Telecontrol systems are the main tool used by utilities to handle the electricity process and business.

These computer based systems do have two main areas of functionality that have to be specifically engineered:

- the Supervision Control And Data Acquisition - SCADA - which allows Central Supervision and control of the remote installations of the electrical network.
- the Energy Management System - EMS - which uses SCADA information in applications software to improve the grid management performance.

Telecontrol systems need to handle data to perform their work.

They all have the capability to process a large amount of information that is used in a range of different ways within the various applications software, and for that they use information stored in databases.

A database is a collection of data stored in a manner which enables information to be drawn from it in a range of different ways and formats in order to be used in different applications.

SCADA systems, like most other information systems, can be generically divided into three key elements:

- Data Acquisition: Collecting data pertinent to the processes being managed;

- Data processing: manipulating the data through calculations, algorithms for display/reporting, decision making or control functions;

- Data management: data storage, accessing/retrieving and archiving.

In all current SCADA system implementation or refurbishment programmes, there is a strong requirement that the SCADA system forms part of a larger integrated process management system. This implies that a significant portion of data interchange is required between the SCADA system and the external information systems, and this has to be done while assuring the data security and integrity aspects of the SCADA system.

The structure of the database and its manipulation, storage and retrieval functions are most critical to the functions of the integrated process management system. Together they completely define the way the physical processes are to be managed: In this context the information system is a pseudo-digitised image of the physical system from which management strategies can be received (or stored) and executed.

The database system structure, its functions and its management, directly impact on the information system operations and support, and directly translate into the ability to control and manage the physical processes.

The sensitivity of the database and its flexibility to accomodate the changing operational and business needs of the utility is critical to the long term viability/suitability of the SCADA system. Limitations of the database can rapidly render an expensive information system obsolete. Moreover, upgrading of databases or of the information systems generally incurs significant cost in procurement and in description of the operation/business processes.

2.2. Operational data for Power Utilities

This section provides a broad outline of the power system data types and their usage across a typical power utility. Some of the data is used in current applications, other data represents potential future usage. This analysis will provide some background on the structure and performance requirements of the operational database.

2.2.1. Power System Monitoring and Control

The data required for the monitoring and control of the power system can be classified into three types:

i) Real Time Data

. Real time data from the physical process: this includes alarms, events, status, analogue measures of generation, load, flow, voltage, current, etc,;
. Real time data generated by data processing and by operator input: This includes derived values operator controls/commands/comments, output of close-loop algorithms, etc.;
. Real time data imported from other information systems, e.g. from other SCADA nodes, from external SCADA systems, etc..

ii) Historical/Archived Data

Long term storage of selective real time data for trending, analysis and planning purposes.

iii) Semi-static Data

. Data describing the physical processes such as power system topology, asset parameters, etc.;
. Data supporting the processing functions such as conversion parameters, displays, etc..

2.2.2. Asset Management Data

Data is also required for the management of the assets, including the SCADA-communications assets collecting the operational data. This may include:

i) Asset Performance Data: Hours run, efficiency, power consumption, error rates, number of retransmissions, etc.;
ii) Asset Condition Data: Protection flags, sequence-of-events, ambient condition, mean-time-between-failure, vibration, equipment temperature, time-to-repair, last fault, etc.;
iii) Asset Support/Maintenance Data: Electronic Documentation, maintenance logs, etc..

2.2.3. Data Interchange with External Systems

The operation of power systems also requires the data import from and export to other external information systems. The degree and extent of data interchange is continually on the increase. Some current and potential data interchange scenarios include:

i) Off-line Modelling: Off-line modelling packages for transient and steady state stability, for neighbouring system modelling, for voltage profile and var optimisation, etc.;
ii) Communications Monitoring: Integrating with communications Network Management information systems for effective placement of polling calls, for channel usage and cost monitoring, etc.;
iii) Management Reporting: Aggregated power system performance data is directly exported to the Management Information System (MIS) for instantaneous or periodic management reporting;
iv) Facility Management - Customer Services: Interchange of data between the SCADA system and Geographical Information System (GIS) and Customer Services information system (customer complaints, fault reporting, service/maintenance despatch and jobbing, etc.) for effective management/maintenance of assets and for enhanced customer services;
v) Metering Management: Interchange of data with electronic metering systems for supervision and for backup data collection;
vi) Maintenance/Works Management: Interchange of information with Maintenance/Works Management systems on the Corporate platforms to improve preventive maintenance functions (condition monitoring) and to assist in coordination and logistics;
vii) Contract Management: Interchange of generation, load and instantaneous transmission cost calculations for the management of electricity delivery on long term or spot supply contracts.

2.3. DataBase Management System (DBMS)

A database management system is software which acts as an intermediary between the applications and the data files and which organizes the storage and retrieval of data in a database.

There are several types of databases. Separated by the way data is held, where it is held and how and to what degree data consistency is achieved.

Central database: All the information is located at one site and is centrally managed.

Duplicated database: Sometimes several parts of the main database are also stored at other sites and, at predefined intervals or whenever required, information snapshots update the local data.

Distributed database: In this type of database organization data is stored locally and is not duplicated to other sites but anyone that has authorization, anywhere within the system can gain access to any piece of data regardless of where it is stored.

To achieve the required performance within these organizations a considerable number of links have to be defined and the users need to have data dictionaries to define the rules for handling the information.

A different and more flexible approach is to use relational databases: this technic for information handling uses the concept of data entity and attribute. A data entity is a “thing” that is to be stored for future reference; a data attribute is a single piece of information that acts as one qualifier to describe a data entity.

The properties and capabilities offered by the the different types of DBMS will be thoroughly described in this Technical Brochure.

2.3.1. Users of Power Systems Information

Users of the power system information can be broadly classified into two types: Primary users who require data in real time for their work, and Secondary users who require small subsets of real time data or who rely only on periodic transfer of stored data.

Primary users are power system operators and control engineers. Secondary users comprise all other SCADA data users, which includes management, maintenance personnel, planning engineers, electricity market traders, service despatchers, etc..

The database access speed, data volume, data accuracy, data security and integrity requirements vary greatly between the Primary and the Secondary users, and vary significantly amongst the Secondary users themselves. Power system operators require instant access to real time data which would in general be no more than 2 seconds old (except for some less critical data which could be up to 15 minutes old). On the other hand, management and planning engineers generally work from a snap shot of real time data at a particular instant in time, or from aggregated data over time. Some Secondary users require fixed & periodic data transfer, others require conditional (event triggered) data transfer, or require ad hoc transfer on request.

The users impose diverse requirements on the performance and structure of the database, and to date, are yet to be satisfied with a single database product that is suitable for all purposes.

Databases used in SCADA systems started from code-embedded data residing in specific memory locations evoked through arrays and tables. As the software became more sophisticated and as the size of the software developing team grew, proprietary database structures emerged with database management systems to regulate database access. When SCADA systems’ customers became more educated and wished to develop their own applications software, the system suppliers were forced into providing more sophisticated Relational Database Management Systems (RDBMS), which were proprietary, with their own structure and manipulation commands. These proprietary databases were generally structured to optimised the speed of access by the real time programs, using non-RDBMS “memory reference” accesses. This imposed severe constraints on the usage of the database, especially for Secondary users. The resistance deterrence to using the SCADA database was due partly to the threat to database integrity/security, and partly to the steep learning curve required to become proficient in their usage.

With the advent of Open Systems Architecture and Standards, and with the fast paced development in hardware capacity and speed, SCADA system suppliers have commenced to develop products conforming to open systems standards which interface to commercial open systems databases on the market.

2.3.2. Maintainers of SCADA Databases

The drive towards a more open database for SCADA systems has also been greatly influenced by the people charged with the maintenance of these databases.

With the early generation of SCADA products, the database (and displays, reports) modifications/extensions were extremely difficult, time consuming and often hazardous to the continued operations of the system. The introduction of more sophisticated database systems made the database maintenance task somewhat easier, however, it was still time consuming and required data to be manually transferred between a number of media. One of the serious restrictions was the fact that database changes had to be performed from a central location, then duplicated and loaded at other sites. Thus the “owners” of the data at remote sites were totally dependent on a central service group to support their change requirements. Other restrictions were due to the poor version control and audit trail facilities provided by the DBMS.

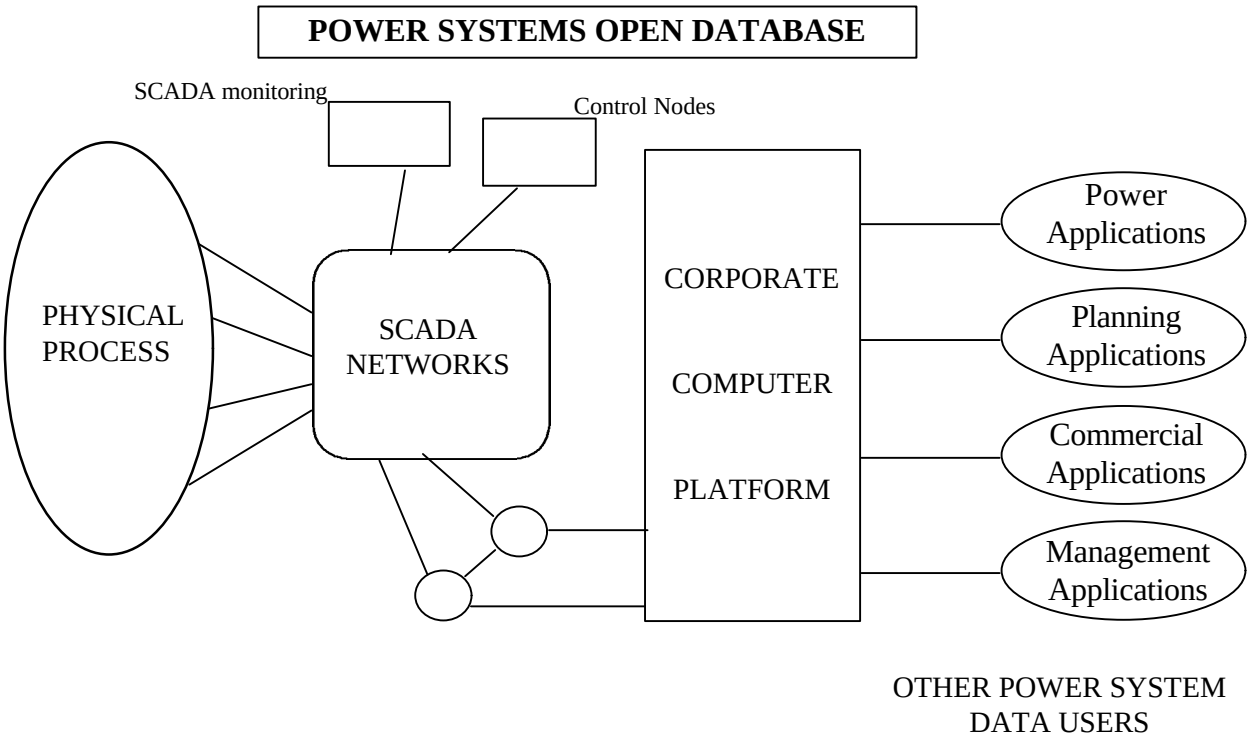
The decentralisation of most large power utilities and the devolution of functions now require that more flexibility is provided to allow ease of maintenance by the data “owners” and to avoid multiple handling of the maintenance process. The advent of a number of OSA databases, coupled with advancement in data communications hold the promise to fulfil these requirements.

3. GOALS AND REQUIREMENTS

A system specifier, normally working for an electricity supply utility company, is primarily concerned with specifying what is required of a DBMS to allow the efficient operation of the power system, the definition of the power system configuration and the setting up and defining of output reports to provide information for the efficient operation of the utility business. The user will not be too concerned with how the management of the data by the run time system is organized to meet his requirements, this being of more concern to the system suppliers. These run time aspects are covered in more detail in later chapters of this Technical Brochure.

3.1. Goals

The ultimate goal for the structure of a database for the operations and management of a power system is that of a single database serving the needs of diverse users. This database must be simple to maintain and must be equipped with measures to provide data integrity and security. A generic illustration of an architecture to incorporate such a database is given below:



We can imagine that there are different goals based upon the different users of data. Primary users are power system operators and control engineers. Secondary users are all other data users which include management, maintenance personnel, planning engineers, electricity market traders, service dispatchers, etc. The goals for these two classification of users are shown in the following table. Note that all goals are allocated to either Ease of Use, Openness, Integrity or Performance.

GOALS OF A TELECONTROL DATABASE MANAGEMENT SYSTEM		
GOALS	**PRIMARY USERS**	**SECONDARY USERS**
Ease of Use	No loss of data - data backup and recovery.	Intuitive DBMS operation. Database easily modified, data only entered once. Fully adaptable to meet every change of user requirements.
Openness	Applications to be independent of data storage structure.	Single DBMS supporting a common information model from substations to off-line power system applications. General purpose query / report facilities provided.
Integrity	Confidence that changes to the database have only the desired effect on the online system. Minimal online system downtime for database testing.	Tools to provide confidence that modifications to the database are controlled. Integrity checking at data definition time. Data migration facilities.
Performance	DBMS runs online and provides adequate data access performance for online applications.	Capacity for adequate long term storage for historical data.

3.2. Requirements

It is important not to put restrictions on suppliers by, for example, specifying that the database in a telecontrol system should be relational, object orientated or simply flat file. It is better to specify requirements, that is what is wanted in terms of functionality and performance rather than how the functionality should be provided. By keeping the specification at the functionality and performance level the supplier is afforded maximum freedom in the design and implementation.

However, the specifier requires to know what can be achieved with the current technology so that a realistic specification can be prepared. The practical way forward within individual utilities must depend on what systems they currently have installed and their individual end user requirements.

From the table in Section 3.1 **the user requirements are in summary**:

- **Support for Standard Applications (Open System)**
- **Single DBMS**
- **Common Information Model**
- **Adequate Performance**
- **Data Migration**
- **Database Integrity**
- **Data Backup/Recovery**

The following sections expand on these requirements.

3.2.1. Support for Standard Applications (Open System)

There is a requirement for standardisation of all power system applications. Currently some offline power network stability analysis applications are becoming standardised, but much work is required to standardise on the real time control applications. This standardisation will lead to the need for the database to support a standard set of data for each application. This in turn could lead to the more efficient definition of data.

3.2.2. Single DBMS

Telecontrol systems provide integrated functionality using coordinated substation control systems, substation Remote Terminal Units (RTUs), SCADA functions and power applications but it is unusual for all of this functionality to be supported by a single DBMS. It is desirable in the future to have a single DBMS that maintains data at all levels of the utility organisation. Improvements in computer technology means that advances are being made at all of these levels, for example at the substation and corporate levels within utility organisations. Substation functions such as event logging, alarm display and management, synchronising, etc. are now being integrated into substation control systems and these require to use the same data as the control centre SCADA system. At the corporate level one example of a user requiring particular data to manage a utility business efficiently is the requirement to obtain plant performance data from the system for asset management purposes. With utilities becoming more business focused and their operating environments changing dramatically there will be many other examples that require the DBMS to be adaptable at producing different reports for different users with different business needs which will change as the business develops.

Currently, because different DBMS's are used for different parts of the system much data entry is duplicated and although each database can be verified on its own it is difficult to verify data across different databases with errors often only being discovered at run time. Due to the lack of confidence in the validity checking of the different DBMS's used at control centres and substations it is necessary to do extensive testing after even a small change before the system can be handed over as operational.

- **A single DBMS will provide significant cost savings** as modification of the database should be possible with minimum retest time.

- **A single DBMS must provide a good security system to be in place** which should include the use of passwords, user roles and user profiles, security checks and an audit facility.

- **A single DBMS will need to ensure that any complete action allowed on the database will result in a set of self-consistent data.**

- **The single DBMS will require to be a multiple-user system**. Some kind of control mechanism will be needed to ensure that concurrent transactions do not interfere. If a deadlock occurs, it is desirable that the system should detect it and break it.

3.2.3. Common Information Model

The amount of effort to maintain a telecontrol database is large. Any parameter should have to be entered and validated only once. **It should be possible to build up the description of, say, a substation from a number of pre-defined feeders which in turn are built from a number of pre-defined breakers and isolators etc.**

With such a model utilities will be able to purchase applications from many suppliers and have these integrated with a common information model.

In the past substation and power network layouts were defined by labour intensive methods that required data to be entered relating plant symbols to screen position. It should now be possible to define substation and power network layouts using a GUI to define library held plant symbols. By combining such standard symbols it will be possible to build up the graphical representation of the network.

3.2.4. Adequate Performance

The choice of DBMS will be dependent on the specified performance requirements. These requirements vary from those needed by primary users for the real time operation and control of the power system to those required by secondary users for offline analysis. The specification of real time performance parameters needs to be undertaken in collaboration with the primary users to establish what performance and tolerance they will accept for each parameter, the tolerances being necessary as performance will depend on system loading. These primary users are mainly concerned with performance and currently their needs are met with a real time database derived from the master definition system. The requirements of secondary users have more emphasis on ease of use than performance and their requirements can be met with adequate performance by the use of standard access procedures on the definition database management system.

3.2.5. Data Migration

Since there is a large investment in the existing databases on RTU's, SCADA and EMS systems, there will be a requirement for data migration from existing systems to any new system.

The fulfilment of this requirement would need to be assessed to see if the production of conversion software is more economic than manually re-entering the data. It would seem that the best method would be to use conversion software since this will avoid human induced errors and the need to completely re-test the new database.

3.2.6. Database Integrity

Currently, many man-hours are spent verifying the correct operation of the system after a database change. This could be much reduced if the old database could be compared with the new database and the changes that have been made thereby confirmed. This kind of confidence boosting function, along with the fact that modern DBMS's are likely to be more user friendly than previous systems should reduce the time required for verification and testing.

3.2.7. Data Backup/Recovery

A database system has to prevent loss of data by either a system or media failure. A system failure occurs when anything prevents the system from continuing work. Examples are CPU failure, power failure or abnormal termination of vital background processes for database or system. Media failure is any hardware or software failure that prevents a required file to be read from or written to.

A database system should take care of automatic recovery from system failures and provide special backup and recovery features in order to be able to recover from media failures.

4. CURRENT SOLUTIONS

4.1. OPEN SYSTEMS

4.1.1. Platforms

With the arrival of powerful computers from rival manufacturers which are able to run common operating systems, current software solutions are available which may be transferred between hardware platforms with minimal effort. Similarly software upgrades should run on current hardware platforms. Thus changes in capacity or performance should be achievable by asynchronous changes to the hardware or software when enhancements are required.

4.1.2. Networks

In order to work successfully database applications (and many others) have adopted a client/server partitioning of the work load. This has allowed the system to become distributed with Front End clients connecting to more than one back end server.

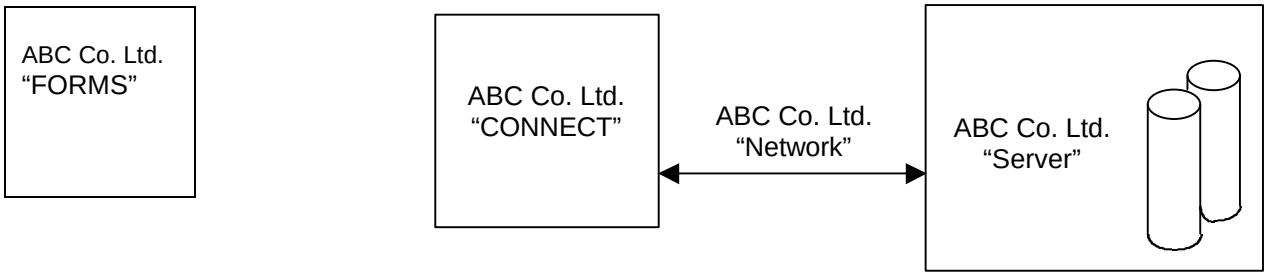
This splits the functionality cleanly between an application and its supporting environment on the client workstation and an environment designed to support the appropriate database engine. Additionally it requires a supporting network environment and this incurs additional costs.

As with all costs, they must be compared to alternative solutions. If a single front end has to bring together data from different databases, then this approach avoids the additional cost of gateways or programs that copy data from one database to another. This simplifies the system design, avoiding the requirement to manage either a gateway or to manage multiple copies of the same data in different databases. However, networking must always add additional I/O to write the data to and read it from the network. This costs more than a dumb terminal on a slow modem link, but people no longer want dumb terminals. The real choice is between moving the windows (X) traffic over the network or the data between the client and the server. Moving the data is easier and generates far less traffic, making it a reasonable option.

By staging a network through the intelligent implementation of a distributed database, it is possible to minimise the data movement over a Wide Area Network. By placing local copies of databases adjacent to the major work centres and handling updates in a secure manner, a client can read and write a local database copy and still securely update the system databases through a central master database.

4.1.3. Homogenous DBMS System

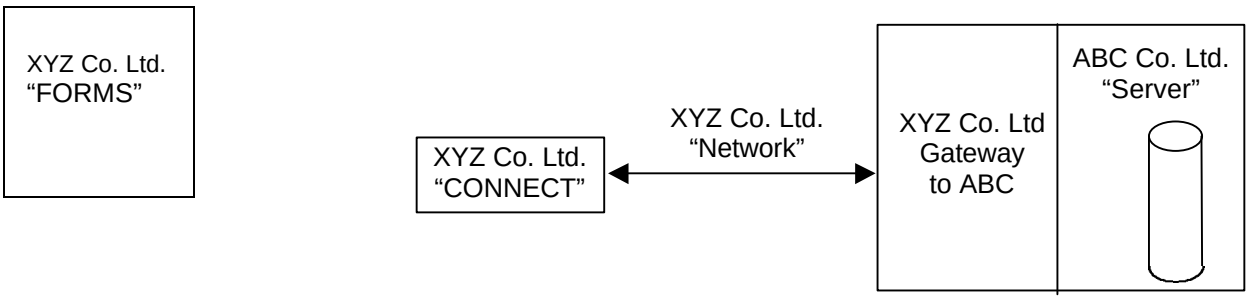
Here the only connectivity capability required is to allow clients on one processor to attach to database servers on another processor. The current generation of software from most RDBMS suppliers is capable of transparently using any one of a number of network transports (e.g. TCP/IP, DECnet).



All supplied by ABC Co. Ltd.

4.1.4. Gateways to Other DBMSs

Where data is held in database systems from multiple vendors, gateways are supplied, normally by the vendor of the tools (e.g. the forms system). A gateway is a specialised server that handles queries for clients in the manner of the native RDBMS of the client and places these queries to an RDBMS from a second vendor holding the data to be viewed or manipulated. Thus a client/server chain is built up e.g.:



4.1.5. Native Connections

All third party database tools and some RDBMS's vendor tools (e.g. CA OpenROAD) can connect to databases from other vendors. This supplies the same functionality as a gateway without the I/O overhead implicit in the additional client/server chain.

A native, relational, connection to operating system files is provided by some third parties (e.g. UNIX CISAM, VMS RMS are supported by UNIFACE). This should prove useful for small systems and development or prototype environments.

4.1.6. The Microsoft ODBC

This is an agreed API set for use on Windows and Windows NT platforms. The database tool (e.g. Microsoft Access) uses the API and links to the ODBC for the database required at run-time. This gives the closeness of a native connection without the tool vendor having to provide multiple versions of his connection software. It is to be hoped that this standard or one like it will emerge across all platforms. It is already available on some Unix platforms.

4.1.7. Classical Distributed Database

The ability to handle transactions as reliably in two or more separate databases as in a single centralised database is provided by two phase commit.

(Two Phase Commit allows more than one simple data transaction to be joined into a single complex distributed transaction. Normally the simple transactions are on different database servers/processors. The simple transactions may actually be in different databases on the same server; the key point is the combination of otherwise independent simple transactions into a single transaction.)

It should never be forgotten that this allows a user to access multiple local databases as a single centralised database. Thus the availability of the well distributed database is less than the availability of any of the local databases that form the distributed database.

Nevertheless, the technology currently exists and reliably provides the ability to access multiple databases through these specialised star database servers. These act as servers to true front end MMI applications and as clients to true back end database servers. The client server technology employed allows network links, gateways and star servers to be stacked at the cost of increased I/O.

4.1.8. Using Transaction Processing Monitors

In order for multiple databases/sites to coordinate their transactions, Transaction Processing Monitors have come into existence. These allow a wider variety of actions to be grouped into a transaction than proprietary two-phase commit software. The RDBMS server has to have the ability to allow transactional control to be provided by a third party TP Monitor in order for a TP monitor to be used. Preferably the emerging open standard interfaces will be used. Major RDBMS vendors are already moving in this direction.

It should be noted that this behaves similarly to the Classical Distributed Database described above in terms of availability.

4.1.9. The File System Cluster

For good availability an RDBMS that can use cluster technology and run multiple servers against the same database(s) is hard to beat. While more than one processor is available connections can be made to any available server and load sharing across processors is therefore possible. When a machine fails, the client applications will lose their open transaction and may have to reconnect. The RDBMS server has to be able to take specific advantage of a cluster file system in order to obtain maximum advantage from it.

4.1.10 Data Distribution - by Table Copy or Snapshot

In order to keep remote copy databases up to date, a simple and long-standing method has been to copy one or more tables (in a single transaction) to the remote site on a timed basis. Such data distributors allow all or part of a table to be shipped.

4.1.11. Data Distribution - by Replication

For constant update of remote database copies replication is now available and gaining ground. This does not involve two-phase commit. A transaction applied to a database at one site is replicated in a log into two or more other sites. There the transaction is applied in the same way as it was on the original database. Obviously these mechanisms require that the replicated transactions are correctly sequenced and applied to databases that originally match.

This approach allows one database to support several whole or part slave copies and to daisy-chain this support by having a copy database replicate on to one or more other copies.

If required the movement of transactions to remote copies can be bundled up so that all complete transactions could be moved to a remote copy on a timed basis.

4.2. SINGLE DATA BASE MANAGEMENT SYSTEMS

4.2.1. Background

Whilst there is movement towards the ideal requirement for a single database management system its achievement will be determined by the type of data to be manipulated.

Although some utilities are moving towards a linked or unified database for all their business activities, current systems divide the data into quite small areas of functionality and the linkages between these functional areas are either specifically tailored to the functions concerned or non-existent. In each functional area a different data store type may be used.

4.2.2. Creation and Modification

Current solutions include the provision of "Mapping Capability" to provide some form of linkage between two functional areas and "Data Collection" packages which allow the one way export of data from a separate collection system into the active databases.

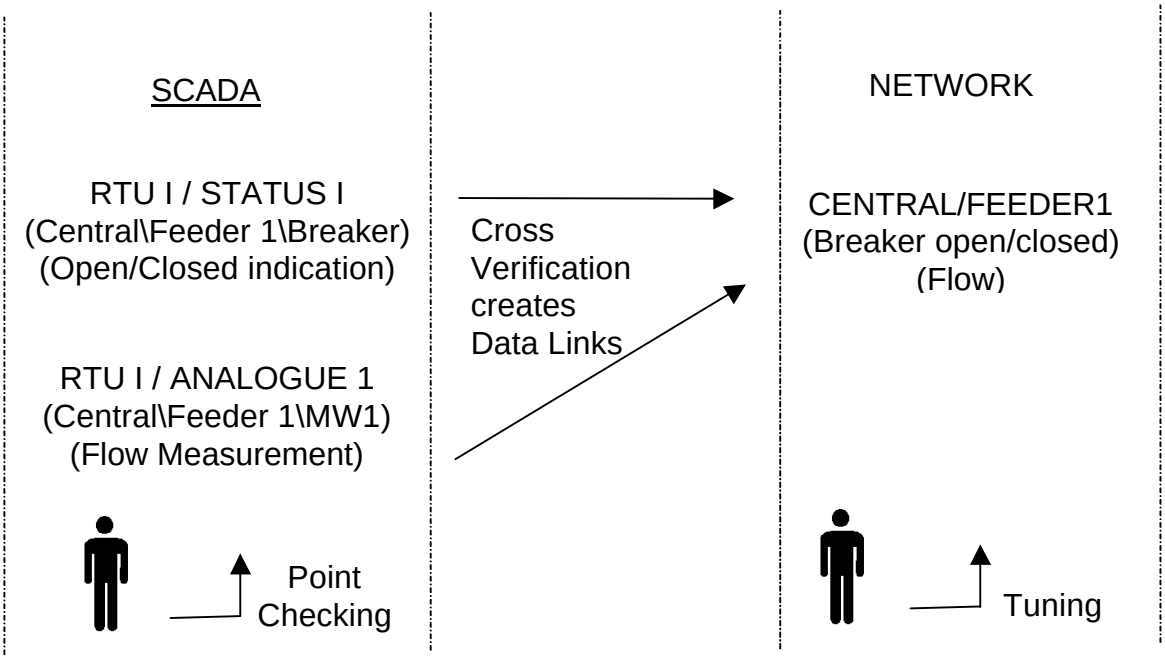
4.2.3. Mapping Approach

The mapping approach requires the provision of extra information in one or more databases.

This information is then used to provide a mapping from one namespace to the other (e.g. identifying the telecontrol voltage measurement for the Network Bus component). The naming conventions in use are normally driven by the functional aspect (e.g. Generation, telecontrol) of the system covered. A database describing an electrical network is going to see a bus voltage as an attribute of a bus, or for topological functionality simply a live/dead indication. Within the telecontrol database the voltage is an entity in its own right, with a name, address, raw to engineering conversion curve etc. This can lead to quite different names and complex mapping between the two databases.

The reason for the above problems can be seen as violations of the simply stated rule "Any single data attribute for an item must have a single master source", and the existence or otherwise of an object is its prime attribute. Thus in the data mapping approach constant operator intervention is required to maintain two databases in synchronisation. The operator and the paper documentation become the single master source and the operator is the control mechanism by which the databases are maintained in synchronisation.

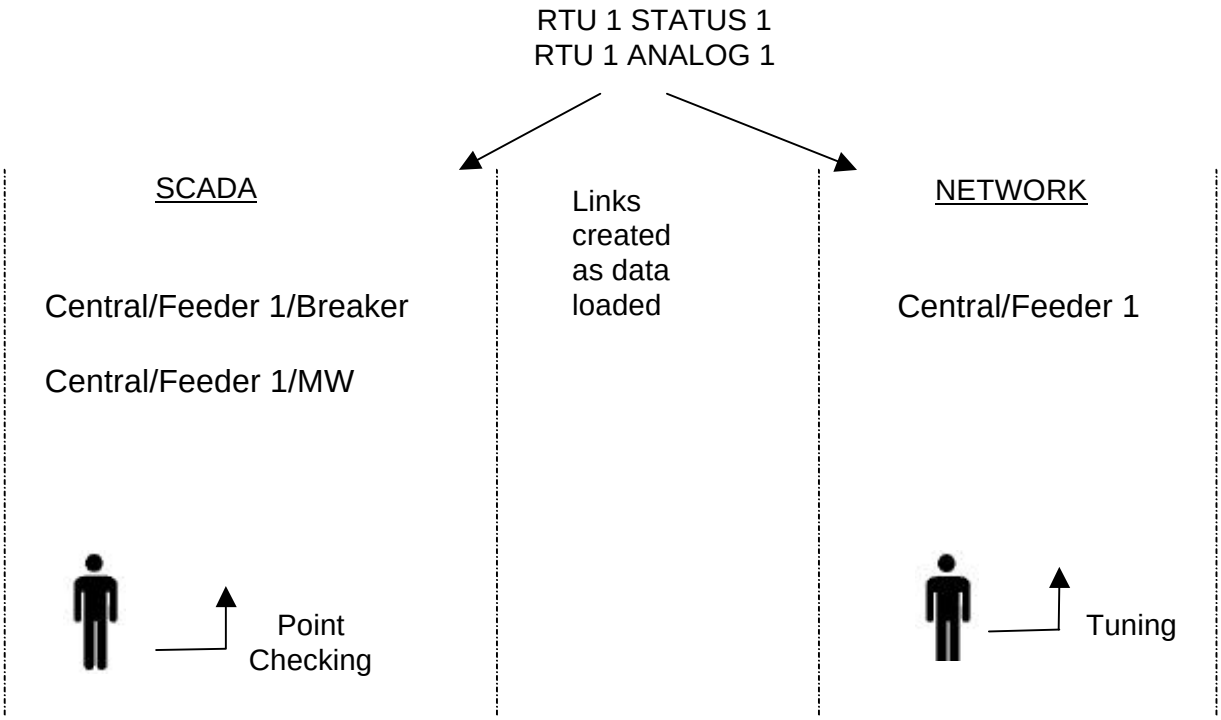
MAPPING EXAMPLE



4.2.4. Data Collection Approach

The separate data collection approach is certainly useful at the start of a project. The applications using the data will each have their own active interfaces, working practices, model/system snapshot storage mechanisms and documentation. Unless the data collection system can be fully integrated into these practices it will soon be left behind. This integration must include replacing the existing application interfaces and their associated documentation.

DATA COLLECTION EXAMPLE CENTRAL\FEEDER 1



In the data collection approach, before the system goes live the data collection system is a useful single master source. Once values are added to the active application databases as telemetry checking proceeds, studies are made and stored and different configurations created and studied, the data collection system will either go out of date or be manually maintained as a copy of the active application databases.

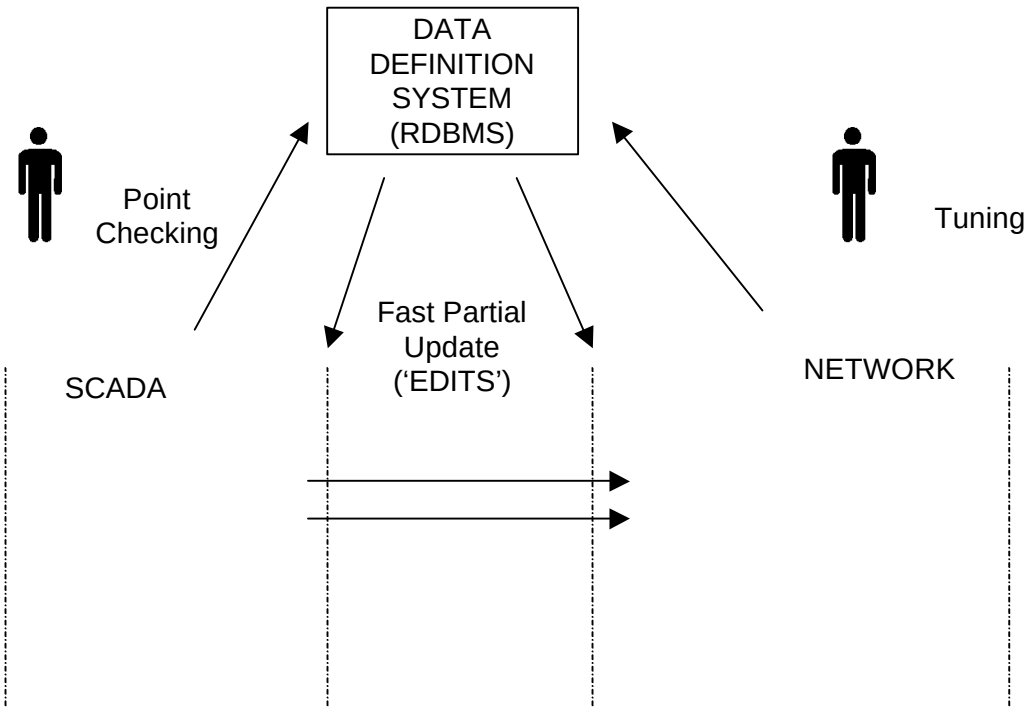
4.2.5. New Moves - Relational Databases

To obviate the above problem of maintaining links between functional areas the following actions are currently being taken:

- Projects or whole utilities are moving to a single data store style. This is normally a relational database system, often a single manufacturer solution.
- The users are requiring a degree of openness in the systems purchased.
- Areas of functionality are being merged for definition purposes.

The first action lays the foundations for the later actions. This centralised master database function leads to the controlled consistency of redundant data, a high level of data integrity, standard data types and the sharing of common data.

CENTRALISED DEFINITION SYSTEM



Apart from providing a controlled master source, a centralised modern RDBMS provides the following advantages:

- Isolates applications from the precise data storage structure.
- Provides common access to transaction processing functions including reliable data sharing through controlled data locking.
- A common source for data processing procedures.
- Centralised, flexible, user and user group access control .
- High Quality data backup.
- Eases the job of the Database Administrator.

Some of the above come about because a Relational Database system is not an empirical solution for an application but a pragmatic application of Relational Data calculus.

The separation of the Data Definition Systems from the active application databases has useful side effects, in particular the definition schema can be biased towards a good MMI and a high level of automatic integrity and the run-time schema can be optimised for speed with the assurance that the definition system will only supply good data.

4.2.6. Current Relational Database Products

In the above section, mention is made of how utilities are moving towards a single database type and often a single supplier. The reasons for the choice will vary and in the current fast changing marketplace will be outdated very quickly. The current state of the market is monitored by independent consultants e.g. the Butler Group, and reports on several DBMSs could be obtained to compare them.

In the area of the RDBMS server itself there is less to distinguish the available products than the suppliers would wish for. All the main RDBMS suppliers provides servers with a good degree of Reliability (low down-time), Data Security (log and journal support), user definable Data Integrity, efficient and secure Data Locking, user controlled Access Security, a good variety of table/index structures and availability across multiple hardware platforms and operating systems.

Attempts by the suppliers to differentiate their product by claiming performance or other advantages are not always evident in third party tests. As SQL standardisation progresses and the requirements of Object Orientation become better known the differences are likely to shrink rather than to increase. Product differentiation is most apparent in the areas of tools and networking where third party suppliers (e.g. Uniface and Cognos) play their part. These are discussed later.

4.3. COMMON INFORMATION MODEL

4.3.1. Background

In current systems multiple applications handle updates to different functional areas of the system independently. The combined databases are then finally verified by inter-application verification and placed on-line. By the application of current RDBMS technology it is possible to maintain one or more sets of changes (or proposed changes) to the system definition. This technique is independent of the functional diversity of the data stored. The links to the target application are as explored in section 3.5.8.

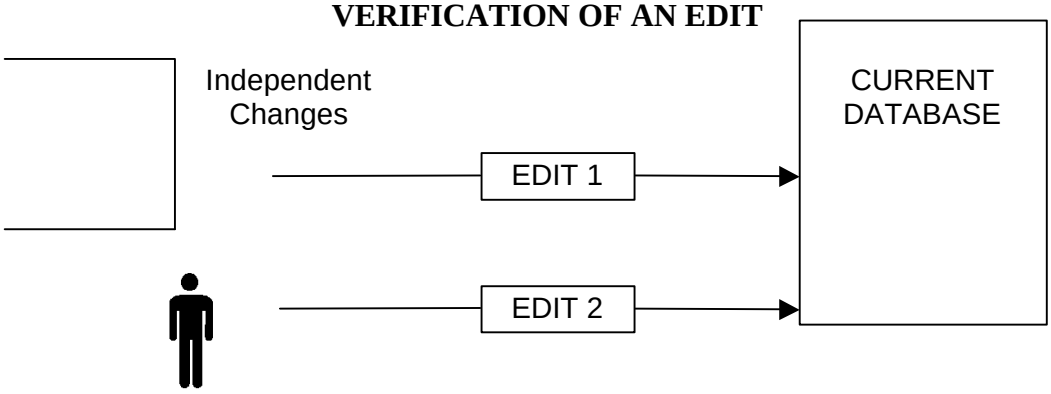
4.3.2. What is an Edit?

An edit is a set of changes to the current on-line system definition. It consists of a set of insert, update and delete actions only. That is, it complies with relational principles.

4.3.3. Verification of an Edit

Any verification procedures should be able to view the system through an edit and see the result of placing the edit into the current system definition.

VERIFICATION OF AN EDIT

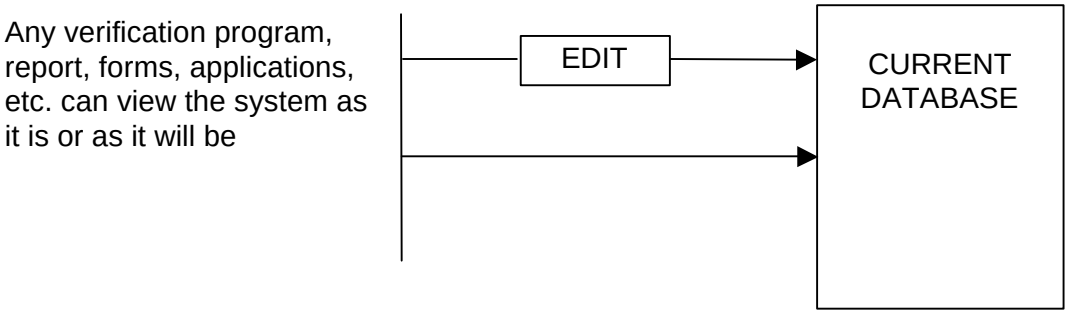


Any integrity definitions, procedural or declarative, need not be enforced until the edit is closed and placed on-line. This avoids the engineer having to perform a set of linked changes in an artificial order to avoid temporary integrity problems, problems that do not exist either before or after the complete change described by the edit.

4.3.4. Multiple Edits

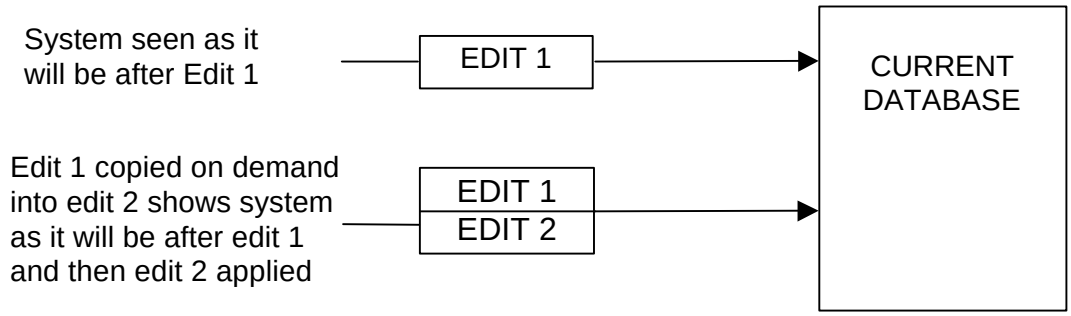
As each edit is independent, it is easy to push small 'express' changes past major changes that may take several days to prepare, study and implement. This has clear advantages over approaches where all the changes since the last major update are transferred together.

MULTIPLE EDITS 1



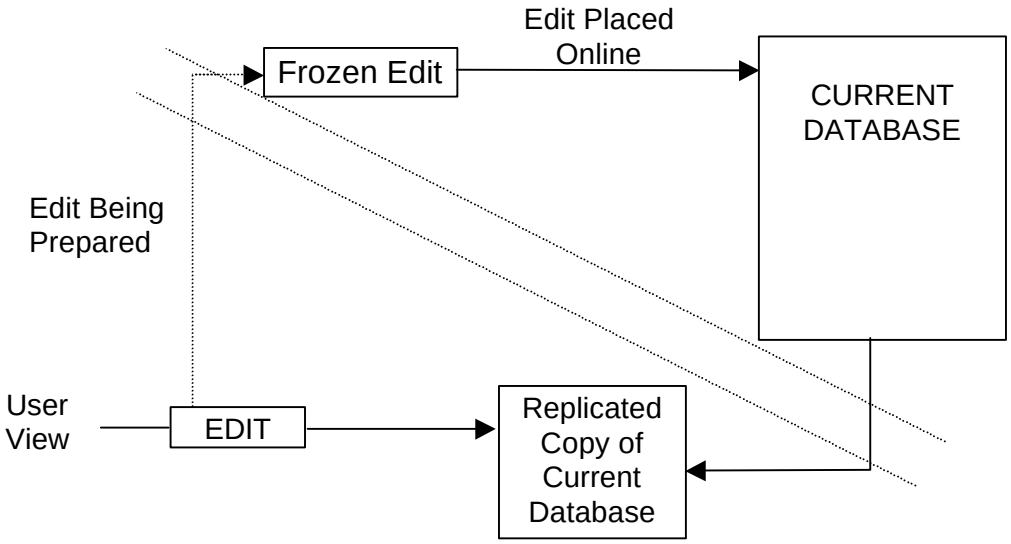
Edits of this type are independent, that is, the changes described by an edit to the current system can be applied more than once without resulting in damaged data. Thus if we have three edits to apply, a, b, c and for verification purposes we merge the edits to have the three edits, a, a+b, a+b+c, then these three combined edits can be applied safely.

MULTIPLE EDITS 2



4.3.5. Edits in a Distributed Environment

If multiple copies of the databases are maintained, then only one of these copies should be the master (see section 4.2 above). Any attempt to horizontally partition the mastership is likely to lead to management problems, particular if responsibilities are to be flexibly assigned, a common requirement.



With edits in a distributed environment, the proposed changes can be assembled and verified on a local database copy. The complete edit/change can then be frozen and transmitted to the master server. From there the current operations team can place it on-line, this action would distribute the edit to all copies before requesting its movement into the local current copy and any real-time database affected.

If a remote processor with a copy of the Master Database becomes isolated from the master then the likelihood is that the utility will want to carry on processing at that site. If modifications are made to this isolated database then it becomes out-of-synchronisation with the master copy. When the isolated remote processor rejoins the network the changes made on it should be sent to the master processor to be applied there and should be undone on the processor that was isolated.
Thus the master of the database, which is responsible for maintaining the integrity of distributed copies, is not compromised.

4.3.6. Archive of Edit 'Before Images'

If the edits and the before images associated with the actions within an edit are archived they can be used as a changes log. Additional the before images could be used to allow a quick reversion to a previous system if a system definition started to give problems after it was placed on-line. The before images for an edit are effectively the edit required to undo the object edit.

4.3.7. Preparation of Switching Schedules in Advance

This is a further example of how the ability to view the proposed system through an edit can be used to plan for the future. Without running a full simulation or creating a complete study database it should be possible to view the future configuration through an edit and run the display system and other programs in this way. Thus switching schedules can be prepared using the same tools as would be used on the current system model.

4.3.8. Placing Changes into Subsystems

This is partially covered in section 4.5.8. A subsystem would interrupt any on-line functions when it receives an instruction to place an edit on-line. This would ensure that each application and any backup systems process, any event, or operator action synchronously with respect to the placement of changes on-line. If the subsystem has no local data store the action is trivial, if it has some internal data store it must update it either directly using SQL to read the edit or it must use an intermediate transfer file.

4.3.9. Liaising with Non-Relational Data Stores

The concept outlined above gives a useful hook for liaising with external systems, including GIS systems from which live diagrams may be imported. If a diagram import is contained within an edit then final verification and fix-up can be achieved before the diagram is released in its current state to users. This implies that the external GIS (or other) system can itself efficiently produce a list of changes since the last update.

As such an import process does not provide a unified transaction system (e.g. locking, integrity) it remains second best to integrating the diagram and data definition completely through a single RDBMS using an edit mechanism.

When liaising with external systems that read from an RDBMS definition system it is possible for the application to read the data directly from the RDBMS or for an intermediate file to be produced from the RDBMS and the external system loads from this file. For any given application the method used to take in such an update whilst staying on-line could be different to the method used to start the application from cold. The following table compares the two approaches:

Application Load uses Intermediate File	Application uses Direct Load from RDBMS
Some people are uncomfortable with SQL and find file formats easier but file formats are not self-documenting	SQL is a standard language and the table formats are available with on-line help.
If a file format changes the programs reading and writing it must change	If columns are added or removed from a table only programs using the columns need to change
Can cope with optional fields at the cost of naming parameters (e.g. slope = 1.234) and increasing file size greatly	Using NULLs can cope with optional fields but does not cope well with multiple choices (e.g. angle, radius OR x,y OR [abc]+[x,y]...)
If files are held locally this allows the application to load independently of the main system (black box)	Applications can only load from the RDBMS, requiring the RDBMS server to be available.
Easy to keep good performance	Can be slow (correct techniques avoiding joins, complicated predicates and sorting can improve performance).
A mixed memory and RDBMS system would require strong liaison with the file reader and the RDBMS edit application	A system where only some items are copied into memory and manual items are managed on the RDBMS would be easier to support
Avoids concurrency problems on items exported (database locked for file write only)	Requires thought to avoid concurrency problems
The application internal database may be out of step with the RDBMS	Easy to keep the program in step with the RDBMS (this is what leads to concurrency problems)
Has to check version loaded once communication established to main system	Always picks up the correct version from the master RDBMS
Application developer has to build in some integrity and security mechanisms	The application has the 'natural' security and integrity of the RDBMS available to him
Any old snapshot values have to be read separately (possibly from the RDBMS)	Any old real-time values snapshotted into the RDBMS are available for a cold start (if the values come from a current on-line system and this system is going to be a hot standby this is very useful).

4.4. DATA MIGRATION

There are two stages to this process, the first is to collect a picture of the utility system as it exists and its definition in any existing telecontrol system or other existing database. The second state is to capture (reformat) this definition in the manner required for the (new) target telecontrol system.

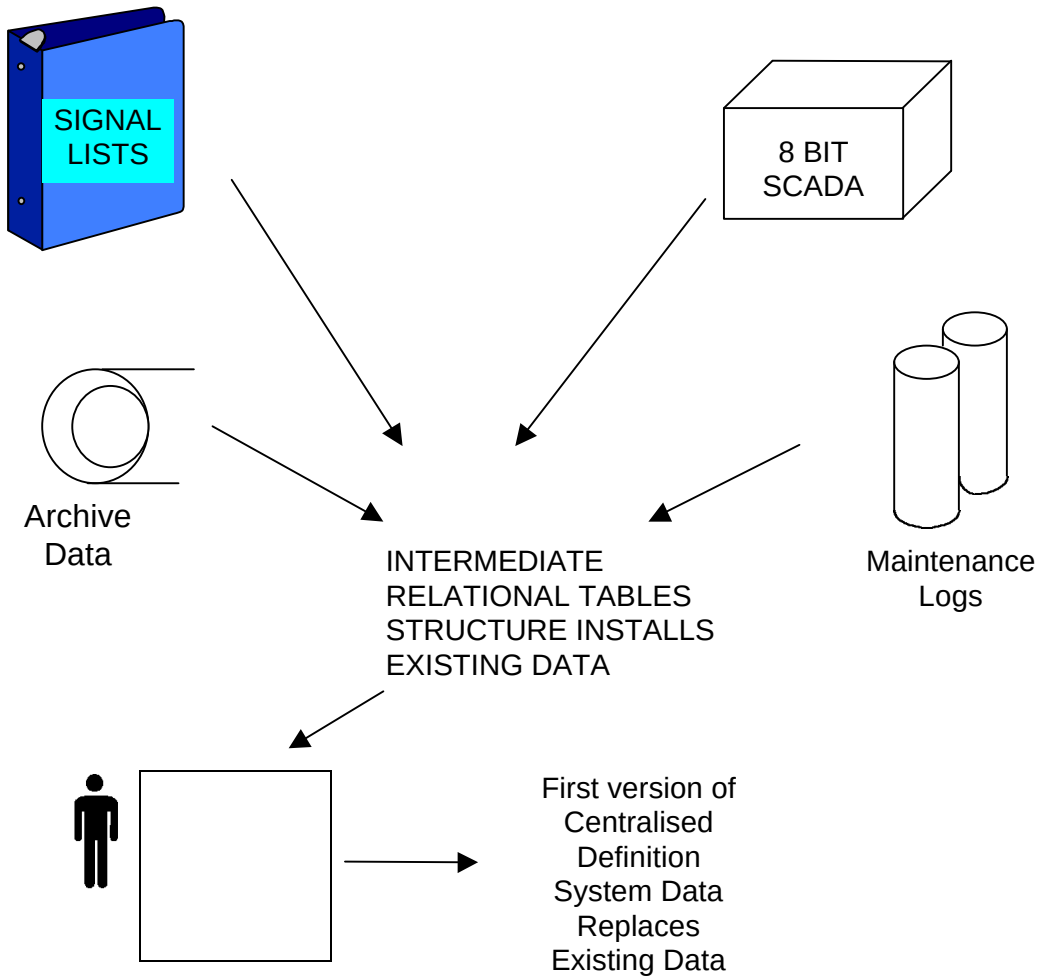
4.4.1. Current Systems

Most users of current systems keep separate records which are then entered into the computer systems. Where a modern RDBMS is used it is possible to give greater integrity and more flexible presentation of the data. Provided the system is well managed this should already be encouraging users to take the next step.

4.4.2. Conversion of Existing Data

Where existing data resides in old computer systems it will often need to be brought forward into a replacement system.

CONVERSION OF EXISTING DATA



This is currently achieved by one of three main methods:

- Manual - rekey the data into the system from lists from the old system.

- 3GL software - use 'C' or Fortran programs to read the data from the old system and load it into the new. This is normally non-trivial, due to the edit, recompile, relink, rerun cycle. It is time consuming to follow alternatives and even finding out the number of records that fulfil given criteria often requires a separate program.

- 4GL software - use report programs (or simple 'C' or Fortran programs) to produce Comma, Separated Values file and import these into the target RDBMS. These RDBMS tables are kept nearly identical to the source data formats. Standard SQL can then be used to query and 'understand' the specific source data and move it into the target table structures.

The 4GL approach has proved to be both much easier and quicker than the 3GL method.

The reasons for computerising the data transfer are not always obvious. Some of the things to consider are listed below:

- Improved Data Integrity - spot checks on 10% of the items can test the method.

- Fewer mistakes - the chance for one off errors due to typing inaccuracy is greatly reduced.

- The final cost - for anything other than a small database is much reduced.

- Inherent conversion assumptions - where the data formats require a transformation process then the 4GL approach should allow different transformation functions to be tried and the increased ability for ad hoc queries to obtain a mental picture of the data should eliminate any guesswork.

- Added system facilities to the transformation functions - where transformation functions are required, it is worthwhile spending time on thinking them through to reduce time spent on tuning the data after the initial transfer (e.g. sensible or context driven default settings for new attributes). Do not forget that the old system did run successfully without some of this information.

It is worth remembering the way in which some utilities are filling in details about their local feeder circuits from their fault call systems. An initial transfer of data can be used to get the system through its acceptance tests and commissioned. The data can be enhanced through the commissioning phase and on into the future.

4.5. DEVICE TYPE DEFINITIONS

Within any system the definition data splits into two types:

- Type data e.g. alarm types and behaviour, message definitions, object types.

- Operational Data, e.g. substation, device and analog instance definitions.

The majority of this document will concentrate on the second part of the data. Short consideration of the type data is in order. System suppliers already use a common telecontrol system for applications in the electricity, gas and water utilities. Between customers using such a system there is a wide range of device or object types used. Current systems already allow the user to define names for the device types in the system and a limited amount of

behaviour. A standard 4GL forms based application provides an ideal environment in which a user can both define device/object type behaviour and through the use of pop-up fields and frames use these definitions when creating the objects in the telecontrol system.

The type data is currently being extended to describe the device connectivity, the commands that affect this connectivity and the commands that inhibit or regulate the other commands. The relationship between these commands and the system state (live, dead, earth, pressure differentials) can also be defined. The key to successfully providing these definitions is the ability to store and manipulate a type domain defining the current dynamic state. The key to taking these capabilities forward lies in providing capabilities for the suppliers and the end users engineers to collaborate in defining the required device (or object) classes for the target system.

4.6. DATABASE VERIFICATION

Any database must be as accurate as possible. Databases involved in the telecontrol of utilities must represent:

- An external, changing, real world.
- A set of users and their (allowed) capabilities.
- Data for which the database is the master source (e.g. normally reported states, inhibits and overrides).

The RDBMS can only help in these ares, primarily in the area of 'Self-Consistency' defined below. The area described as 'Point Checks' remains a manual activity. Overall system design, including the use of the RDBMS as the master source for wiring lists and other information areas should reduce the amount of manual checking required.

4.6.1. Self-Consistency

A modern RDBMS can readily ensure that switches are not connected to non-existent cables within its own database. This is internal to the database and cannot be used to ensure that either the switch or the cable really exist.

4.6.2. Point Checks (End to End Testing)

This term is commonly used to describe the actions of modifying field equipment states (directly or by manual intervention in the monitoring equipment as close as possible to the field equipment) and seeing the correct changes take place in the following areas:

- The current state database.
- Any schematic displays.
- The alarm lists (including ensuring any text description is correct).
- The system logs (including ensuring any text description is correct).
- The wallboard mimic.
- Any digital instruments and other secondary displays.
- Mapped secondary applications (e.g. EMS databases).

If the power of the RDBMS has been used to ensure internal self-consistency and the centralised database controls all of the above, then the amount of checking can be reduced to point checks to the database or schematic displays and sample checks between the areas to ensure the database enforced rules are operating correctly.

4.7. PERFORMANCE

4.7.1. Using a Relational Database for a Master Database

In sections 4.4 and 4.5 the notion of using a Relational Database as the Master Definition for a system was implicitly assumed. Here, some of the ramifications of that assumption are explored.

Due to the performance requirements of telecontrol systems it is likely that a real-time database will exist. In order to maintain this in-line with the master definition database it will have to be kept up to date and these updates synchronised across the master and backup systems. It would be allowable for the real-time subsystem to have a restart under a small number of scenarios but the majority of updates should cause only a processing pause. Such real-time databases can be expected in both the main control centre processors and the smaller processors forming the telecontrol network.

As the relational system becomes the master database it becomes the permanent repository for telecontrol (and DMS) actions or notes.

Thus it should also become the place where Operational Restrictions, Inhibits, Tags and other such notes are placed. The type and usage of these notes should also be defined in the relational database in order that only appropriate notes may be placed on objects of a given type.

The suggestion is often made that as a relational database exists it should have the current values from the real-time system copied into it. If this is possible at other than a slow background rate and without severe performance implications then the requirement for a real-time database has probably expired.

4.7.2. Study and Historical Data Handling

The edit mechanism described in section 4.5 gives a user a view of the system as it might be at some time in the future. Using this view mechanism it should be possible to allow studies to be run against an edit to analyse its usefulness, for instance a contingency analysis could be run and reviewed before the proposed change was accepted or dismissed.

With the master definition in the relational database and possibly a log of the (recent) changes made to that system also held, the storage of system snapshots, event sequences and archive data can be managed with the definition and its history.

As the archive storage requirements are often very much larger than the definition requirements for telecontrol systems, the storage of complete copies of the definition database is practical. However, it is possible that the distribution systems, with their low telecontrol element, may have a definition database of a similar size order to the archive requirements.

4.7.3. Performance Limitations of Relational DBMSs

There is no doubt that a modern relational database server, although more efficient than its predecessors, is necessarily going to give a lower performance than a purely memory based real-time system that does not support reliable transaction processing.

However, given sufficient resources and programmed well, a modern RDBMS will certainly provide sufficient power to drive a primarily manual distribution system. Whenever an evaluation is performed to ascertain whether an RDBMS could support a function, the evaluation must be performed on a machine with adequate resources in terms of processor power and I/O throughput (including multiple discs). It is easy for the more visually attractive areas of the system to be given more resources than the database engine that provides a shared service to the whole system. The RDBMS server should obviously be allowed more resources than either a GUI workstation or a telemetry front end.

Further, the level of integrity and flexibility provided by the RDBMS needs to be considered. If the same requirements are placed on a real-time database (e.g. reliably saving all changes to disc to survive both a system reboot and a disk failure) will it perform better. A 3GL approach is also unlikely to provide the full flexibility of an SQL served RDBMS at a similar cost.

To summarise, a modern RDBMS would provide sufficient performance for:

- Manual distribution systems
- The manual portions of a mixed telemetry/manual system
- The MMI portion of all systems (e.g. issuing controls)
- The capture of archive data (possibly with buffering required).

For high performance telecontrol systems a modern RDBMS is unlikely to be cost effective apart from the functional areas listed above.

Further, the flexibility and performance of a modern RDBMS is going to make it useful for study and future system assessment work.

Indeed, an RDBMS may well be expected to serve not just the telecontrol system but the whole enterprise once power engineers and management information preparation is considered.

4.8. DATA BACKUP/RECOVERY

All modern RDBMS's support an adequate to excellent level of transaction level support for maintaining a database in working order through a server processor failure. Most also support complete database backups, sometimes without interrupting service coupled with continuous journalling of data changes to independent storage to protect from media failure.

Useful as most of this is, it is not absolutely necessary in a telecontrol or distribution management environment to have these facilities or rely solely on them. The transaction level mechanism is, however, essential. Without, actions consisting of multiple data changes could end up half applied, leaving a database without even self-consistency.

The options available to provide against media failure or major control centre disaster are outlined below.

4.8.1. The RDBMS supplied database backup features

The first requirement is to take a complete copy of the database(s) concerned to other media with minimum disruption to the system. If the RDBMS supports a true online backup capability then this is the obvious choice, far preferable to a server shutdown. However, the user should check that this capability does not cause concurency problems by locking all or part of the database while the copy takes place. For consistency reasons, the offline copy should look as if it was taken at a single point in time. The easy way to do this is to wait for an opportuniy to lock the whole database, lock it and block out all online activity, take the offline copy and then unlock the database. Whilst the server will be available throughout this process, the lack of allowed database activity means this can not be viewed as an uninterrupted service. A true online backup capability will allow normal user activity to continue whilst the backup is taken, using some form of before imaging technique to secure a consistent backup. To support the backup taken at a single point in time, perhaps performed daily or even weekly, transaction journalling can be provided. This copies database activity not only to the online database but also, at a leisurely pace, to a separate journal on separate media. Within this journal the position of the complete backups are known. Thus, when a complete backup has to be restored, it can be walked forward in time to the last secured transaction from the journal. This is unlikely to be the last committed transaction but should be fairly close.

The most sophisticated versions of these mechanisms that have existed over the years, allow databases to be turned backwards and forwards in time from journals to enable a good starting point to be selected for the resurrected system.

4.8.2. Multiple Servers

In most systems for telecontrol and active distribution management, at least two processors are provided to allow the system to proceed in case of processor failure. Increasingly, a second emergency control centre is provided, requiring another one or two processors. The data is then continuosly secured on two of these processors and copied on a more leisurely, but still transaction orientated basis, to the others. This gives an excellent degree of data security from media and control centre failure without recourse to the RDBMS backup and journal facilities.

If all the location configurations are identical, or have an identical core, one of the standby or emergency systems can be regularly shut down and complete system backups performed. These backups could be used to restore any system which would have to be given the old processors network identity before being returned to service. Any backup data from the master to the processor being restored should be automatically buffered while the processor is absent from the network.

The RDBMS facilities or other backups may still be required depending on the system design. Inadvertent but self-consistent data changes could be introduced into the system and successfully percolate to all the databases. In extreme cases the system design may offer inadequate means of recovering from this, although if all processors / servers are affected, presumably a complete operating system restore could be performed.

4.8.3. Intelligent Storage Systems

A more straightforward approach is to use RAID or cluster technology. Here an intelligent bulk storage looks like a single filestore to the main system but has within it backup and redundancy features. The storage system is able to continuously store data on redundant, independent media whilst providing a continuous service to the main system. Not all RDBMS servers are able to provide multiple RDBMS servers on different processors sharing a single such storage system.

4.8.4. Backup / Recovery Support where multiple servers or storage are unavailable

Where only a single server on a single processor is provided, the system design can normally provide simple backup provisions.

If the single processor is purely for data preparation, then the changes can be copied to the central system during preparation. If an archive or warehouse system has no redundant processor, then all data can be secured to tape on this system before it is removed from the source system. Since such system will probably support the ability to offline and reload sections of historical data, this same mechanism can be used to backup and restore data.

5. NEW OSA DATABASES

The open system architecture (OSA) states that it should be possible to interface power application programs to a variety of database packages.

Relational databases are available today that provide excellent tools and a standard language SQL (Structured Query Language) to access the data.
Object Oriented databases have become available providing a more flexible database structure with promised increased performance.
Even, the best of both worlds (the hybrid approach) may be combined to maintain the database with the export of data to the proprietary real-time application databases.

The main question this section deals with is how we should integrate different database systems in a telecontrol environment viewed from the medium-term perspective.
No restriction on whether the database is relational, object oriented or a flat file should be imposed in advance.

Aspects of the different database systems will be highlighted according to the main goals as explained before:

5.1. Relational databases

The first question which comes into mind is why and when we should choose relational technology ?

5.1.1. Ease of use

The support of set-manipulation has been one of the most important reasons for the evolution of relational products.
Data engineering is simple because objects are always managed in the shape of two-dimensional tables. This allows a user to analyze and manage data independently of the original data design and analysis. The question to be asked may not be known precisely in advance. Because there is no need to reformat or relink the data to try different queries, it is relatively easy to try several queries in order to improve the understanding of the data held. This feature is powerful. A good example is the way it enables database engineers, sometimes with the help of special programs, to gain a better understanding of a company's data than that company itself.

The preceding hierarchical and network (CODASYL) database systems used pointers to maintain the associations. Queries on the data could therefore only be supported where the pointers had been provided.
Further, data corruption of the pointers led to a useless database. It should be noted that this is where relational databases came in.
With relational technology, the index replaces the pointers and is an enhancement of the relationships defined by the data, keys and indexes held.

5.1.2. Openness

Relational products are suitable in an OSA environment.

All relational products have a standard language to access the data (SQL) and have excellent tools for developing data management and query applications. All SQL operations are set-oriented, allowing powerful commands to be built to manipulate data.
Nowadays, relational databases are available on a wide range of hardware platforms and operating systems. This is complemented by the support for most network protocols, which provides a good opportunity for tools from one manufacturer to access data from one or more databases managed by another manufacturers RDBMS. A measure of this is that the RDBMS community has contained third party tool developers for some time. These have thrived without providing an RDBMS engine of their own, surely a good indication of genuine openness.
The development of the Open Data Base Connectivity (ODBC) standard is furthering the inter-operability between different manufacturers products.

Thus, the simplification and standardisation of data engineering is one of the main arguments to choose for relational products in a power system management environment.

5.1.3. Integrity (and distributed integrity ?)

Recently, great stress has been laid on providing quality implementations of user controlled data checks, most obviously referential integrity, where the existence of cross referenced objects is guaranteed. Whilst this may be done declaratively or by user implemented triggers and stored procedures, the benefit to the integrity of the database is the same.

This benefit is the centralisation of the checks concerned into the centralised database. Thus any user accessing the database with any tool is controlled by the set of checks.
This should be compared to implemented checks in application programs or verification programs, where there is no built in guarantee of uniform applications of the checks concerned.
This is sometimes referred as providing an enterprise with a single unified set of business rules.

The current Ansi/Iso standard for SQL, called SQL-2, supports the definition of database integrity in a declarative form, requiring minimal user programming. This support does not extend however to a distributed database.

5.1.4. Performance

Why has it taken so long for relational products to be accepted as a general tool ? The poor performance of many systems built with this technology is the most common answer. For real-time applications the performance of pure relational systems is currently inadequate.
Vendors of relational products are therefore reshaping the RDBMS, using specialised data-base access techniques (like optimizers, row level locking, shared SQL etc.) and specialised database servers (multi-threaded server architecture) in order to improve the performance profile.

Despite this, we believe it will remain difficult to satisfy the central performance requirements of a telecontrol system with relational databases.

5.2. Object Oriented databases

Currently, Object Orientation is described as an improvement on all existing data processing techniques, and Object Oriented Databases an improvement on all other Databases.
An OO-DBMS can support complex objects and is not restricted to the (normalised) two-dimensional tables of a relational system.
Complex objects certainly exist in the real world of telecontrol systems. For example stations, bays, power network equipment, telemetry outstations all are simple objects with descriptive and identifying data. These objects are also composed of other simple objects which interact in non-trivial ways. The interaction makes the behaviour of the equipment or device level objects complex.
Thus it is apparent that a Power Network control system, incorporating telecontrol, should benefit from the use of an Object Oriented Database.

So, why and when should we choose object-oriented technology ?

5.2.1. Ease of use

The basis of Object Orientation is to classify the problem domain into sets of objects with the same data (or attributes) and the same behaviour (or operations).
These 'CLASS'es can be related to each other in similar ways as objects in network or relational systems.
Especially the possibility to store behaviour (using methods) into an Object Oriented database adds a new dimension providing a more realistic data structure, which is closer to the users imagination.

Furthermore, with the aid of its methods an object classes data structure will be separated (encapsulated) from the applications using the class.
Thus a change to the database structure does mean the methods needs reworking, but the class users should be insulated from the change in data structure.
An object class is allowed to manage its contained, private data in private and this data is protected from access by other classes.

5.2.2. Openness

Object Oriented products are emerging in an OSA environment.
Standards for the use of Object Orientated databases are evolving, like the Common Object Request Broker Architecture (CORBA)

5.2.3. Reliability/Integrity

Within an Object Oriented system it is less common to store foreign keys or data in an object. The main reason is the sole reliance on pointers between related objects to maintain the association, which is the same as in preceding hierarchical and network (CODASYL) database systems.

Theoretically, the existence of an object is its identity. In practice, this means the value in the pointer to an object is that objects identity. and will be the same in any pointer to that object. With object identity, changes can be made to objects without impact on related tables.

Inheritance allows similar classes to be grouped in a hierarchy. For example, a switch device_class can provide the basic switching capabilities. Other specific classes can inherit this basic behaviour, adding extra capabilities or replacing the basic capabilities for different types of switch. The latter is achieved by implementing the operations with different methods at different points in the hierarchy.

5.2.4. Performance

With their origins in the memory based 3GL world and their management of logically and physically complex objects, OO-DBMS systems offer potentially large performance gains over a purely relational system.

5.3. Hybrids ?

The number of OO-DBMS vendors is currently quite small. But it is a growing area and, as mentioned above, promises to remove some of the problems of relational databases, particularly performance.
The relational world has a strong background for providing data safety and flexible data access, and became popular because of drawbacks in existing data handling systems. Some of these drawbacks are built in to the OO model. The obvious solution is an integration of the strong parts of relational systems with the strong parts of object oriented systems. Such systems are commonly referred to as hybrids.
Other hybrids do exist, network-relational, hierarchical-relational. In the early days of relational databases, some network and hierarchical systems added relational features. The 'pure' relational systems and their supporters decried these systems as 'impure', however such databases are still appearing in the market place amongst post-relational and other offerings.

Already the first products exist, which completely support Ansi/Iso SQL plus object oriented concepts like inheritance, methods and object identity.
Somehow, the database system customers must ensure that the purists, relational or Object-Oriented, do not prevent the emergence of useful systems drawing on both models.

The activities of various standardisation committees and vendors of known relational products shows a convergence of ideas. In the relational world, even if OO capabilities are not placed in the end-users hands, the tools themselves use OO techniques. The next Ansi/Iso standard for SQL, informally called SQL-3, is focused on the provision of object oriented concepts within SQL.

5.4. The choice of database technologies

At present, no easy choice is available.
A proprietary real-time database, an object-oriented database and a relational database all have different advantages to offer in a telecontrol environment.

The following table summarises some of the differences between the database technologies.

Feature	Proprietary	RDBMS	OO DBMS
Openness	Weak	Good	Improving
Tool Availability	Weak	Good	Variable
4gl Tool support (Catalogs)	Weak	Good	Variable
3gl support			
-Procedural SQL	n/a	Reasonable	Improving
-Embedded SQL	n/a	Ugly	Good
Skill Availablity	Weak	Good	Improving
Training	Variable	Good	Improving
Ease of Use	Variable	Good	Improving
Performance	Good	Improving	Good
Concurrency Support	Weak	Good to Excellent	Unclear
Complexity	Variable	Medium	Good
Large Volumes of data	Variable	Medium	Unclear

Before a single DBMS can supply all the system needs the best features of the different breeds of DBMSs need to meet in the evolving standards.
To some extent this is coming true in the Ansi/Iso standard for SQL (SQL-3). The performance of such a database, even with current computing speeds, is still likely to leave a proprietary real-time database in any telecontrol system.

6. CONCLUSIONS

6.1. Technical Brochure overview

Real time data in telecontrol systems is used by many application programs each one with particular constraints concerning database performance.

In spite of the efforts of some utilities to create linked or unified databases for all their activities, the market in general is biased towards using the data divided into small areas of functionality tailored to the specific applications. Further, the changes in business environment that utilities are experiencing everywhere creates the need for range of software applications that handle SCADA and non SCADA applications and gives extra interest to the flexible approach of having information functionality tailored to the specific applications. This is a problem to the users engineers that have to assure the consistency of information all over the telecontrol system. Therefore the DataBase Management System is the cornerstone of the SCADA/EMS.

A clear specification of users requirements for DBMS is of extreme importance. The utmost requirement is the use of a single database management system that shall cement the different proprietary databases in service in the telecontrol field so that data can be entered and modified only once and that standards, security and data integrity can be maintained at all time.

Chapter 3 of this Technical Brochure describes the more important requirements of the users of telecontrol databases, and gives a short explanation of each one. Some engineering requirements related with implementation and maintenance are also mentioned. All of them try to achieve flexibility and reliability for data manipulation.

Although there are presently in service many proprietary solutions for databases, the techniques used are fairly well known and the review of the current solutions used in the existing systems will help the utilities to define the kind of improvements they can do in their SCADA/EMS concerning databases.

Chapter 4 describes in some detail many of the current solutions and refreshes some useful concepts about database management and the problems users usually face in their systems. The aim is to give a contribution to the utilities to decide what kind of modifications they will have to perform in their database management to move to more flexible and open systems.

6.2. General Guides to practical solutions

In general a considerable effort and investment is needed to modify a totally proprietary system so that it can support software in a fairly easy way and so the solutions to be adopted will depend on the particular situation of the utility system.

We can consider four typical situations:

6.2.1. Managing the improvements of existing Databases

That is the case of those who now have systems with several real time databases loosely coordinated and are facing increasing difficulty to maintain the global system.

The main problem is that such systems have very expensive maintenance costs because they need intensive manpower to assure the information consistency. On the other hand, it is very expensive to make a major enhancement to the databases software.

Depending on the cost benefit analysis three main alternative solution can be adopted:

a) Do not change the databases software and keep coordinating manually (i.e., cross referencing information by paper);

b) Use some kind of "gateways" to full or partially coordinate the databases. This usually implies building application software especially adapted to the situation, which can be expensive and difficult to test;

c) Build a completely new centralised DBMS taking account of the actual and future needs. This is obviously a very costly solution and time consuming to implement. Usually it can be only justified if the system is going to undergo a major renewal.

6.2.2. System Upgrade

That is the situation of those utilities that are going to upgrade their old system and have to find a solution to improve their database management capability.

A utility that has a SCADA/EMS system in service would want to protect their investment as far as possible, especially the investment in application software. Nevertheless, the initial investment so that a proprietary system can conform to standards is very important.

Two main solutions can be adopted:

a) Stay proprietary if the cost benefit analysis shows that this is the more effective solution;

b) Put a shell around the proprietary database and grow from there into an open environment, allowing third party software to interface with the system database, using standard Application Software Interfaces (API). Cost and life time evaluations need to be done very carefully to be sure that this solution is more effective than going directly to an open environment with a major renewal.

6.2.3. Buy a new SCADA/EMS now

Users need to decide whether to buy new systems now when openness, standardisation and new software techniques are available but the way found is not yet totally defined.

In a pragmatic approach, openness means that:

- a system is able to perform the required functionality, based on software that uses the existing standards so that third party standard software can interface with it;
- if the system uses any proprietary solution, namely at database level due to speed restriction, there is a strategy to fully maintain public interfaces;
- the system can be engineered in the future to follow the evolution of the open environment. This is not easy because the open environment is not yet stable. In the future it will be more stable.

General recommendations to face this situation are:

- Go to a system as "Open" as you can. Try the system that exist in the market to test their standard solutions and to identify any problems with interfacing third party software;

- Try to get an environment for source data in which you can develop the non real time applications using PC tools and relational data handling.

Relational databases and PC applications software are "de facto" standards nowadays. On the other hand most of the EMS applications do not need to handle real time data. Snapshot information with a relative slow refresh rate are enough for many applications, provided the information read was sampled at the same instant. This way systems can have Relational Database Management with real time proprietary databases that are read back to the relational environment at sampled intervals so that information can be used in EMS applications.

6.2.4. Telecontrol Databases in the future

What about the utilities that want to foresee the real time database architecture of the future?

The trends of future standard architectures can be obtained from the report. Nevertheless utilities must be alert for the evolution of the market and follow very carefully all users associations and study committees that are trying to standardise databases definitions of EMS.

ABBREVIATION LIST

3GL	-	Third Generation Language
4GL	-	Fourth Generation Language
API	-	Application Programming Interface
C/S	-	Client/Server
CA	-	Computer Associates
CCS	-	Coordinated Control System
CISAM	-	Compressed, Indexed, Sequential Access Method (A File Structure)
CPU	-	Central Processor Unit
DBA	-	DataBase Administrator
DBMS	-	Database Management System
DECnet	-	Registered name of Digital Network
DMS	-	Distributed Management System
EMS	-	Energy Management System
GIS	-	Graphical Interface Systems
GUI	-	Graphical User Interface
I/O	-	Input/Output
IMS	-	Information Management System
INGRES	-	Trademark of a Database
LAN	-	Local Area Network
MIS	-	Management Information System
MMI	-	Man Machine Interface
NGC	-	National Grid Company
ODBC	-	Open DataBase Connectivity
OO-DBMS	-	Object Oriented Data Base Management System
OSA	-	Open System Architecture
OSF	-	Open System Foundation
PC	-	Personal Computer
RDBMS	-	Relational DataBase Management Systems
RMS	-	Remote Management System
RTU	-	Remote Terminal Unit
SCADA	-	System Control And Data Acquisition
SNA	-	Systems Network Architecture
SQL	-	Structured Query Language
SQL-2	-	Current ANSI/ISO standard for SQL
SQL-3	-	Informal designation for future release of SQL-2
TCP/IP	-	Transmission Control Protocol and Internet Protocol
TP	-	Transaction Processing
UNIX	-	Standard Operating System for Open System
VMS	-	Virtual Management System

Le CIGRÉ a apporté le plus grand soin à la réalisation de cette brochure thématique numérique afin de vous fournir une information complète et fiable.

Cependant, le CIGRÉ ne pourra en aucun cas être tenu responsable des préjudices ou dommages de quelque nature que ce soit pouvant résulter d'une mauvaise utilisation des informations contenues dans cette brochure.

Publié par le CIGRÉ
21, rue d'Artois
FR-75 008 PARIS
Tél. : +33 1 53 89 12 90
Fax : +33 1 53 89 12 99

The greatest care has been taken by CIGRE to produce this digital technical brochure so as to provide you with full and reliable information.

However, CIGRE could in any case be held responsible for any damage resulting from any misuse of the information contained therein.

Published by CIGRE
21, rue d'Artois
FR-75 008 PARIS
Tel : +33 1 53 89 12 90
Fax : +33 1 53 89 12 99