December 31, 1992

The Architecture of a Distributed Substation Management System

Andrew McRae

A description of the Megadata Substation Management System.

Introduction.

This paper describes the software and hardware architectiure of the Megadata distributed Substation control and management system. The Substation Management System is the result of some 9 years of continuous product development, with Prospect Electricity providing the impetus for development (and providing the primary proving ground). The system as described is a leading edge total solution for the complex demands of substation automation, control and management, and provides significant potential for constructing sophisticated applications. Several examples of such applications are given in an accompanying paper (authored by Michael Rybinski of Prospect Electricity).

Initially the 9 year evolution of the system will be described, showing how various hardware and software architectures have effected the overall cost, response time and field reliability. An indepth discussion of the current system will review the design choices in the light of field experience.

The hardware, software and communication architecture is described, and compared to other systems such as Programmable Logic Controller (PLC) based systems or traditional Supervisory Control And Data Acquisition (SCADA) systems. Areas of comparison include the database architecture, the use of Megadata's Sequential Control Language (SCL), and the level of distribution.

Finally some future directions are outlined, which discuss the continuing refinement and design improvement incorporating emerging techniques and technology such as direct measurement of primary quantities, substation to substation data interchange through the use of Wide Area Networks (WAN), and greater use of energy management and load control algorithms in the substation.

Evolution of SMS

As a result of high growth and an ever expanding customer base, Prospect Electricity in the early 1980's desired to maximise their investment in substation installations by providing intelligence within the substation, as opposed to traditional SCADA systems, which implement centralisation of control and data processing through the use of a single master station obtaining telemetered data from relatively simple Remote Telemetry Units (RTUs). Megadata was awarded the initial contract for development of the Substation Terminal Unit (STU), and the first systems were installed in 1985.

Since this was a new venture and development for both Megadata and PE, it was seen as a longer term evolving project that could grow and undergo refinement as experience was gained in developing and testing the target system.

The First System

The first systems that were installed were designed as a single centralised telemetry and control computer that had self-contained data acquistion I/O (Figure 1).

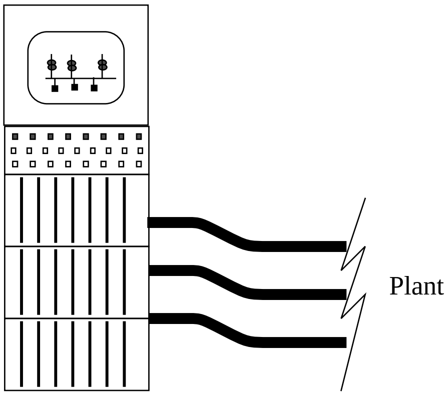

Figure 1: Initial Configuration

The single CPU handled all aspects of the substation management system, such as Man Machine Interface (MMI) functions, telemetry acquisition and data processing, and Local Control Routines (LCRs). The LCRs were a key element in the PE control strategy, where specialised routines would allow customised control of the substation to take place.

The hardware of this system was based on the Megadata MD3000 range, 19" rack mounted proprietary hardware; the CPU was a Motorola 68000 processor running at 10 Megahertz, with limited RAM and non-volatile memory. A typical configuration included a CPU card, power supply, non-volatile memory expansion card (based on EEPROM), a semi-graphic character display generator, and a mixture of digital input modules, analogue input modules and pulsed output modules. A typical STU would scan 270 digital inputs, 90 analogue inputs, and could operate 60 outputs. This configuration catered for a typical PE zone substation, containing three transformers, 28 circuit breakers and miscellaneous telemetry. Some digital inputs were specified as higher speed scanning (time tagged to 10 millisecond resolution).

Due to the centralised nature of the I/O, all acquisition and control cabling needed to be marshalled back to the main STU cubicle. In the initial configuration, this was not a problem, since most of the cabling already existed.

In retrospect, the hardware of this configuration operated suprisingly well; there were a number of obvious deficiencies which later systems attempted to overcome:

- In newer substations the cost of cabling back to a central cubicle was high, mainly due to the number of cabling connections that needed to be made and the length of cable.
- Robustness was seen as a problem, where all control and telemetry was invested in a single CPU, memory and I/O subsystem.
- Scaling of the system to meet new requirements and applications was very limited. Performance limitations would be quickly reached if sophisticated LCRs were designed.

More importantly, the software and database architecture was originated from a traditional SCADA system (reproduced in minature), where the telemetry I/O was mapped to a monolithic flat analogue and digital point database. Whilst the point database model was sufficient at the time for larger scale SCADA systems where simple telemetry and control was performed, it was woefully inadequate for representing substation data in a meaningful and useful manner.

An example is a circuit breaker, which in a PE substation is telemetered by four digital inputs; open, close, available and protection trip. These four inputs would each be mapped to a separate point in the digital database, and each point would be processed individually. LCRs or applications dealing with the circuit breaker as a piece of equipment would have to deal with each of the four separate digital points, and understand how each related to the other points. It became even more complicated when equipment such as transformers had to deal with the state of circuit breakers connected to the transformer, as well as analogue measurements such as voltage, current or oil temperature.

Local Control Routines were written in the C programming language, and interfaced to the rest of the system by directly accessing the analogue and digital point database. The key difference between a LCR and other pieces of application software was that the LCR program itself was part of the database, and new LCRs could be loaded by installing a new database. LCRs had a well defined and limited interface to the rest of the application software and database, which was a mixed blessing. On one hand LCRs fitted into a distinct (almost formularised) framework, but conversely accessing arbitrary data in the system or interfacing to application software become convoluted and difficult.

PE and Megadata spent a considerable length of time developing and testing the initial LCRs in the early phase, as there was a steep learning curve due to the use of C as the implementation language. As with all such sharp tools, C allowed flexibility and power as a programming language, but did not provide robustness in the face of programming or design errors. Also essentially being a procedural language, it did not lend itself to easy implementation of control algorithms.

Since the LCRs needed to be part of the database, it mandated the inclusion of obvious system dependencies, such as compilation to the object code of the target processor, as well as knowledge of the database structures as processed by the resident application software. It was clear that C was a convenient but non-optimal and undesirable solution to the LCR problem.

One of the major problems with the point database system was the intimate intertwining of dependencies between the database structures, LCRs, application software and MMI functions. Maintainability became a problem, especially as new applications required new database fields, or differing processing of telemetry processing; keeping database and LCR revisions in synchronisation with application software was a problem, as subtle errors could occur if a data structure differed between the two, as there was no way of enforcing or checking the correctness of interfaces.

It took some time for these problems to be recognised for what they were, but by then the system had undergone a fairly typical transformation that occurs in many large software projects; too much had been invested in the software design and structure to make radical changes without endangering existing sites that needed to be maintained. The performance of the system was good enough to be considered a success, and enough invaluable experience had been gained that would ensure evolution of the system into a new phase.

Hardware Distribution

One of the main drawbacks to the centralised system was the lack of robustness in the event of failure, and also the plant cabling problem. Part of the next stage of the project was to attempt to solve both these problems by distributing the telemetry and control throughout the substation itself. And so about 3 years after the initial installations, various schemes of distribution were examined to find one that would meet the criteria laid down. In surveying the available offerings, the following goals were held in view:

- Some form of networking instead of radial point-to-point connection was desirable. It was intended eventually that peer to peer access was required, so a master-slave arrangement was unacceptable.
- Readily available and inexpensive components should be used in the construction of the network, so that the per-unit cost of each node is kept low.
- It was highly desirable that a non-proprietary solution should be used; Megadata held the view (radical at the time) that Open Systems would become increasingly important in the SCADA world, and enterprises would become concerned with the interoperability question. Rather than base a product on dead-end proprietary technology, the strategic view was held that standards based technology should be employed.
- Electrically robustness and physical strength were of high priority. Since the environment was rather inimical electrically, and high availability was a goal, the networking technology should be chosen accordingly.
- Reasonably high throughput, so that the network does not become a bottleneck when responding to many system events.

At the time these considerations were taking place, the Manufacturing Automation Protocol (MAP) was being touted as being the OSI protocol of choice for factory floor and real time applications. Much energy and effort was being placed into this standard, and many vendors were beginning to align themselves with the standard. And so in anticipation of MAP being widely adopted and implemented throughout the real time world, Megadata decided to offer MAP as the networking backbone to a distributed STU solution for PE.

In the initial instance, the physical layer of the MAP standard was implemented on a new range of Megadata products, the MD1000. The MD1000 was smaller and more configurable than the MD3000, and consequently cheaper. Each MD1000 consisted of a CPU card (containing a 8 MHz 68000, 128 Kbytes of EPROM, 128 Kbytes of RAM), a networking adaptor, and up to 6 telemetry I/O cards. The physical networking media adopted was 802.4 Token Bus over 5 Megabit carrier band coaxial cable, which provided the physical and electrical robustness required.

The goal of this phase of the project was to distribute the telemetry I/O of the STU into MD1000 units termed Distributed Interface Units (DIUs) over the 802.4 carrier band network, forming a hardware architecture shown in figure 2.

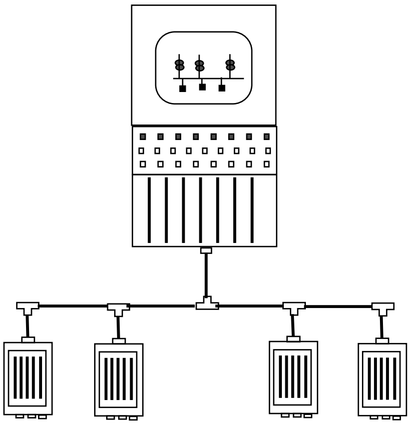

Figure 2: Hardware distribution

The DIUs were doing nothing more at this stage except telemetry I/O. The data was being transceived back to the central STU cubicle using a Megadata telemetry protocol. The performance of the initial system was disappointing; the integration of software designed for telemetry scanning over traditional communication media meant that the network was poorly used. The network was in effect being used as a slow (and expensive) multi-dropped communications channel. The rest of the software archicture remained the same; if anything it moved towards more traditional SCADA, where the telemetry is done in RTUs and the master station processes all the raw data.

One underlying motivation was to save costs on the cable marshalling, and to provide a hardware platform for a future generation of distributed software to operate on, yet not cast out the old software architecture at this stage. However the unit cost of each DIU mandated that each DIU interfaced to more than one item of plant; this led to a ‘mini-marshalling’ problem, where the cabling from each unit of plant had to be marshalled to a DIU marshalling cubicle close by. The use of DIUs has not entirely eliminated the cabling problem; utimately the use of low cost single board DIUs will allow the installation of one DIU per item of plant, and thus solve the marshalling problem.

Part of the future plan was to implement a complete MAP protocol stack, so that other vendor’s equipment could interoperate. It is a truism that hindsight is 20/20. MAP as an industry standard has not penetrated to anywhere near the depth that many predicted, which supports the notion that just because someone calls something a standard, it doesn’t mean that it will be used as a standard (even if that someone is General Motors). As a result of the glacial acceptance of MAP (and the associated Token Bus physical network), semiconductor vendors were slow in delivering working silicon, and as a result there were early teething problems with the distributed networking hardware. Again, a steep and long learning curve had to be climbed before the networking was fully functional (another truism is that if you work at the leading edge, sometimes you fall off). However with recent technology such as running 802.4 Token Bus over RS-485 twisted pair, it seems that MAP is finally maturing and stablising into the favoured real time networking technology. The advantages of operating over a standard non-proprietary networking technology are manifold.

Software Distribution

With the hardware distibution of the telemetry I/O solved, the next phase in the evolution was to migrate the software to allow distributed processing to take place. The Distributed Interface Unit would become the Distributed Intelligence Unit. This would allow many benefits to be realised. Distributing the hardware was easy; distributing the software meant that some form of distributed database was required, peer to peer access to remote data, LCRs operating within each node etc.

Some goals of the distributed system were formulated:

- Transparent database access across the network.
- Robustness in the event of failure; no failure of any node should affect any other node (except in the case when data from the failed node is required).
- Automatic remote configuration of each node in the event of a new database being loaded.
- Revision control so that all nodes would be synchronised with the correct copy of the database.
- Separation of the dependency between the LCRs/database, and the resident application code. This would allow new databases and LCRs to be downloaded without concern for incompatibility.
- Provide a much improved LCR subsystem that was more robust, better integrated with the database, and much easier to program.
- Provide support for extensible database and applications so that a good foundation could be laid for developing much more sophisticated applications.
- Scaling of the architecture so that much larger sites could be accomodated.

A long hard review of the installed software architecture highlighted a number of problems, some of which have already been discussed. The essential question was: How much to keep of the old, and how much to replace? One overriding concern was the investment in existing LCRs, databases and software; an early goal was to preserve as much as possible of the database and LCR structure, but it was quickly discovered that it is extremely difficult to distribute a piece of software that has not been designed for it. Another nagging problem was the continued use of a point database. With the exposure to more modern relational data models, it was more and more obvious that the existing database structure was clumsy and difficult to use. This would be magnified once more sophisticated applications were designed and developed.

Given the stated goals for the distributed system, it was hard to see how any of the older architecture could be used. The dilemma essentially was: Do you build on a fundamentally flawed foundation, or rebuild from the ground up? Many of the distributed systems that exist in the real time world are simply non-distributed systems that have been modified to operate on a distributed platform; it was felt that a much sounder product would result if the first one was simply thrown away, and a totally new software architecture developed. The attraction was that the new product would be much easier to use and provide a solid foundation for developing new applications. It would give the opportunity for the known limitations to be removed, and employ newer techniques and technology.

The Current SMS

The Substation Management System is a completely new implementation of a fully distributed substation control and management system, built on industry standards that allow easy interconnection with other technologies for a flexible and powerful solution.

The software architecture includes some unique and interesting features, such as the use of the Internet Protocol (TCP/IP) suite as the main communication backbone, which allows complete and high performance networking to be achieved in a minimal package such as a DIU. An object oriented distributed database allows peer-to-peer access of data across the network, and a sophisticated Sequential Control Language (SCL) allows distributed control algorithms to be easily developed and operated across the entire substation network.

The SMS has in operation for over a year, and in that time has proven itself to be unequalled in providing a robust and flexible foundation for development of sophisticated new applications, as well as providing fast response and exceptional robustness within the substation.

Some closer examination of various subsystems of the SMS will highlight the architectural features of the system.

Communications

Whilst the physical media that is employed as the network in the SMS is 802.4 Token Bus carrier band, the upper layer communication protocols are not MAP protocols. The small size of the DIU and low processing power effectively barred the use of ISO protocols. Other vendor boards that implement MAP typically employ a 25 MHz 68020 (minimum) along with at least 3 Megabytes of RAM. The requirement still existed for a networking standard to be used within the SMS, and so TCP/IP was considered.

Arguments rage back and forth between *de jure* and *de facto* standards, but when it comes down to availability, performance, packaging and interoperability, it is very hard to better the Internet Protocol suite. A complete TCP/IP implementation along with drivers, embedded operating system etc. totals less than 60 Kbytes of code, so it was a perfect choice for the distributed Substation system. The performance is also very pleasing, allowing throughput in the order of 50 to 100 Kbytes/second for database download. TCP/IP operates over 802.4 using ISO Standard Network Access Protocol (SNAP) headers. Many vendors provide networking products, both hardware and software, that interface to the TCP/IP protocol standard, such as routers, terminal servers, workstations etc.

There are a multitude of benefits for running a commonly implemented protocol suite within the SMS. The Megadata developed embedded networking operating system implements an Application Programming Interface (API) the same as the Berkeley '*socket*' interface, so much software could be ported to the DIU system with little effort. It also provided a useful development path, as much of the prototyping, coding and testing could take place on Unix workstations, and then simply recompiled for the DIU system. Perhaps the most obvious benefit is shown in figure 3, where a test Substation network can be easily connected via an Ethernet gateway to a development network, allowing instant access to the distributed database, downloading of new databases, remote login into any of the DIUs on the network etc

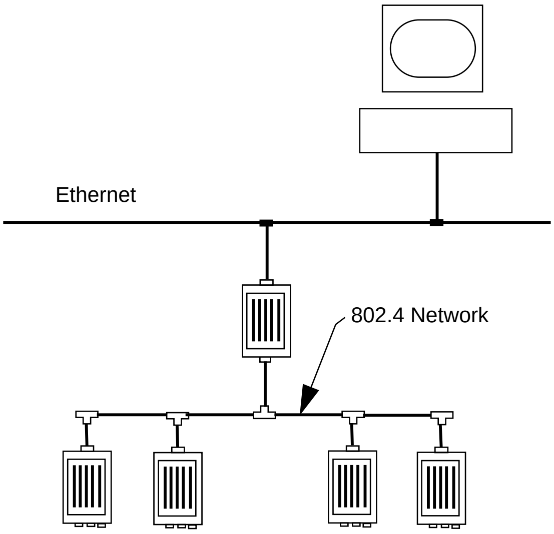

Figure 3: Gateway to Ethernet

Standard protocols such as Telnet, Trivial File Transfer Protocol (TFTP) etc. may be used to interact with the DIU network. Instead of using the semi-graphic character display generator, it allows display of mimics on any X Window System display such as workstations or X terminals. The implementation a Simple Network Management protocol (SNMP) agent allows the DIUs to be managed just like any other network-cognizant host.

TCP/IP can be layered on a wide range of physical media, allowing the SMS to transparently operate across interconnected networks. Thus the substation network can form part of a wider area network (termed a *catenet*) comprising a number of networks, each of which can be different media, such as 802.4 carrier band, 802.4 twisted pair, Ethernet, serial point-to-point connections, ISDN etc.

One example of this is currently undergoing commisioning within the Central Business District of Parramatta (west of Sydney). Figure 4 shows how the concept of a network goes beyond a single physical network into a Internet of networks.

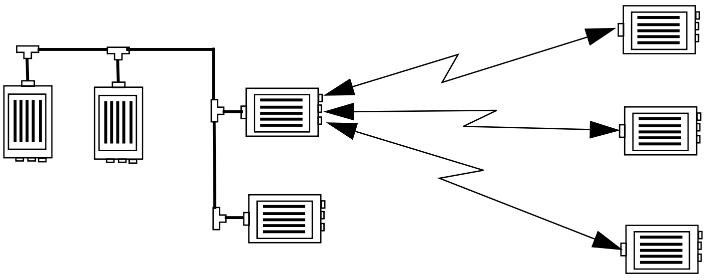

Figure 4: Internetting via serial lines

A number of remote DIUs are connected to central 802.4 networks via serial gateways; the remote DIUs contain the same application code and database as would be found in any other DIU (with the addition of a directory of remote records and addresses), and the services that operate across the network such as database access, control algorithm initiation, remote login etc. run transparently across the serial link.

The use of a common standard such as TCP/IP allows the easy integration of SMS in an enterprise wide network, which allows peer to peer database access throughout the entire network, and allows substation to substation database access for interchange of zone information. The emerging networking technology such as ISDN, ATM etc. can be easily integrated since standards exist for operating TCP/IP over these media.

Distributed Database

Probably the most powerful subsystem comprising the SMS is the database architecture underlying the applications and LCRs. The whole thrust of the database architecture is to provide a tool for application development rather than a *fait accompli* flat database structure.

Three key concepts underly the database architecture; object oriented data modelling, data distribution and LCR integration.

Object Oriented Data Modelling

Whereas the previous point database system simply had analogue or digital points, the new database architecture allows any number of different *classes*, each of which can be connected together to give a hierarchial representation of the actual plant items being telemetered and controlled.

Consider a circuit breaker. Each circuit breaker has 4 digital inputs which describe what state that circuit breaker is currently in, **open**, **close**, **available** and **protection trip**. Rather than trying to deduce the state of the breaker by examining four separate digital database points, a circuit breaker *database class* can be created. Each class has a number of standard class *methods* attached, and a structure for the data that is associated with each instance of the class. So a circuit breaker has a number of standard routines attached for dealing with the inputs to the class, and a data record that contains the particular data required for each circuit breaker (such as the name, current attributes and state etc.). The standard routines only operate on that class's data records. Each such data record and associated circuit breaker class methods is called a circuit breaker *object*.

An object is comprised of a data record that contains the unique data associated with that object, and the methods of the class that object belongs to. Objects can be as simple as a single digital input, or as complex as a complete transformer bay. All objects that belong to the same class share the same methods. The group of methods attached to a class is called a class *handler*, and is responsible for interfacing that class to the rest of the database. Every class handler is part of the database i.e downloading a new database will allow a new set of class handlers to be installed.

The nature of these objects is that they can be interconnected to provide a hierarchial model of the actual plant being telemetered, so that the designer or configurator does not have to deal with a narrow and restricting concept of digital or analogue points, but can deal with data *models* representing the real world objects within the substation.

In figure 5 is an example of such a hierarchy, where each object is connected to other objects in a standard way. Each raw digital input does not know what kind of object that it is connected to, but simply passes the object's status values whenever the telemetered values change.

Each class handler is defined as part of the database, and is not wired into the resident code of the DIU. This allows complete freedom in modifying objects, and adding new classes and objects along with their methods.

There are many benefits to the object orientated nature of this database architecture, as listed below.

- Robustness. Because each object is accessed through a tightly defined and narrow interface, there is little interference and consequently much reduced risk of error through accidental modification.
- Modularity. Each class handler deals only with the type of class being processed, and so it can be designed and tested in isolation. It also allows modification of the handling of the class with impunity, as it will not affect other classes e.g if in the above

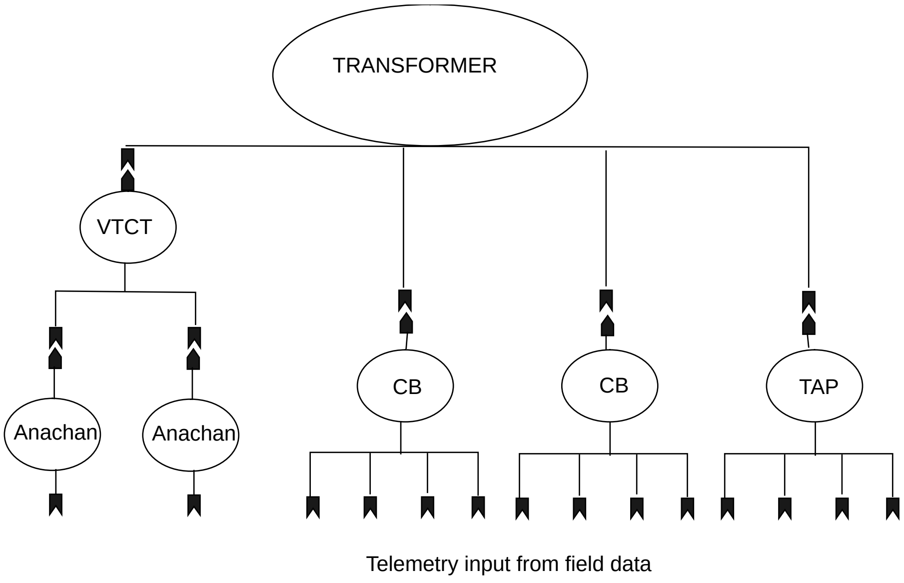

Figure 5: Database Object Hierarchy

example a new form of tap was being used that required 6 tap indicator bits, then a new class and handler could be developed and installed without any modification to any of the other objects.

- Ease and accuracy of configuration. Since the designer is dealing with the actual real-world objects, they are much more comfortable with configuration, and the designer can include whatever data is required in each object's record.
- Reduced data handling. Since each object performs the necessary processing to represent the real state of this object, higher level objects (such as transformers etc) do not need to deal with large amounts of data. By way of example, for the tap number calculation, the tap handler performs all the necessary processing to calculate the tap number from the multiple digital input values, which it simple passes to the transformer object as an integer.
- Extensibility. A major benefit of object oriented technology is that new data classes and objects can be readily created as new applications are developed, and the new database downloaded without modification of the resident DIU code.

Data Distribution.

The distribution of the database allows different object hierarchies to exist on separate DIUs, and each DIU to service access requests from client DIUs, and also to remotely access data records on another DIU. Central to the distribution of the data is the notion of *ownership*. The node that contains a particular data record, and is responsible for updating that record is said to *own* that record. Other DIUs may access that record, but any modifications can only be made by the owner of the record. Thus consistency and coherency problems are avoided.

Remote access of a record is essentially a two step process. When an application (such as an LCR) requests access to a particular data record, and that data record does not exist locally within the DIU, the DIU must *discover* the owner of that record (or if indeed it even exists). Discovery can be made by two methods; each DIU can have a static map of remote records and the address of the DIU which owns that record. If the target record can be located within that map then the owner is known. Static maps are for use with records across a serial link. The other form of discovery is dynamic, where a broadcast on the network will attempt to elicit a reponse from the owner. If no owner is found, then either the owner is incapacitated, or no owner exists.

Once the owner is discovered, the next step is accessing the record. All database record access occurs through an event server task which runs on each node. The local event server will communicate to the owner’s event server, and register an interest in the record; the owner then sends the current record data to the client. The client will store the record data in a local cache and inform the original requestor of the record’s availability. While ever the client is still interested in the record, the owner of the record will send a new copy whenever any fields within the record are updated, thus keeping the client’s cached copy in synch. Some hysteris is applied to the cacheing timeouts, so that intermittent interest will not cause undue network traffic.

LCR Integration

Database record access (both local and remote) is integrated with the SCL subsystem, so that LCRs can directly access the database fields within each object. Whenever a new class or data record is defined, the LCR compiler creates access macros that automatically define the field types and record fields in the new structure, thus no changes are required within the DIU resident code to download new data classes or access the data classes from LCRs.

This tight integration of the database subsystem and the LCR subsystem provides a seamless application environment, so that users writing control algorithms need not be concerned with decoding point database attributes to derive meaningful information.

LCRs can also themselves perform data processing by modifying local data records. Remote data modification can be achieved by initiating remote LCRs.

Local Control Routines and SCL

Initially, LCRs were developed by describing the control algorithm in a flowchart first, and then coding into C. A number of limitations arose as a result of using C as the primary implementation language. The biggest problem experienced was the learning curve and development effort that had to take place for substation designers and control engineers to learn how to program in C. Once operational, maintenance and debugging was a problem, since the development environment available on the embedded platform was limited. Robustness was also a problem, since the LCRs were effectively running as special forms of native application code, so it was possible for programming, configuration or design errors to cause CPU crashes. Since the interface to the LCR subsystem was narrow and limited, LCRs had to often perform much manipulation of data structures and pointers to obtain the desired information. Overall it was clear that the LCR subsystem was too difficult and fragile to provide a useable environment for developing more sophisticated applications, or to migrate into a distributed environment.

Early in 1991 Megadata designed and implemented a replacement LCR environment that was useful and powerful enough to replace the C based LCR subsystem. It was based on Megadata's Sequential Control Language (SCL), an interpreted language that incorporated sophisticated state control facilities, logical and mathematic functionality, and integration with the distributed database. The aim was to shield the substation designer and LCR implementor from the native machine, and allow the use of a high level robust environment for developing control applications. One goal was that no engineer should have to debug programs using low level machine monitors, but that a high degree of confidence in the correctness of the algorithm implementation should be obtained *before* the program is actually executed. This meant that the SCL program should undergo rigorous parsing via the SCL compiler, and that any errors occurring in the runtime program would occur as a result of faulty logic (as opposed to syntatic incorrectness, or invalid data or null pointers causing machine crashes). Since SCL was interpreted via a psuedo object code, it was highly robust. No DIU crashes could occur as a result of faulty SCL programs.

SCL is essentially a state control language, as opposed to C which is a procedural language. SCL differs from other control languages or programming environments such as PLC ladder logic or boolean algebra in that much more sophisticated applications can be supported via a rich set of operators and database access facilities. PLC programs are generally limited to dealing with simple boolean or analogue inputs and outputs. SCL programs can perform boolean algebra, as well as mathematic functions, employ timers for accurate timing of events, built in floating point and integer arithmetic, and automatically integrates with the distributed database. Database classes are accessible via simple SCL macros, allowing access or modification of any named field within the database record.

A key feature of SCL is that it can be extended by providing a set of C library routines implementing new functions and facilities. This allows a low level customisation of the SCL run-time support to add new facilities as required. SCL programs are state driven, which means control algorithms expressed as flow charts or state diagrams are easily and quickly transposed into SCL syntax. SCL allows interaction between LCRs by allowing LCRs to start or stop other LCRs, either locally within the same DIU, or remotely across the network.

Experience with SCL has been gained by PE over the last 18 months. It is a tribute to the flexibility of the language that not one line of C was required to be written by the substation designers to completely replace the older LCR subsystem. Since it is an interpreted language, highly efficient and compact programs result; one example is a transformer voltage regulation LCR, which in C was nearly 2000 lines of code (not including hundreds of lines of include headers). The equivalent SCL program is 207 lines of code, which is considerably easier to maintain and understand. Most of the saving has been obtained not just through the use of a control language, but primarily because all the elements of the SMS combine together to form a much more coherent and structured whole e.g the use of the object oriented database allows a transformer to be represented by a single database record, which contains the distilled and processed state as presented by the other database objects, each of which has performed the particular processing required for the plant item it models. Thus the LCR need only view the particular fields within the transformer record with which it is concerned.

The aim of providing a powerful control language is to allow the substation designers and configurators to leverage new and unthought-of applications into reality without having to deal with a poor development evironment. The goal of the SMS is to provide an application development environment for the substation, so that maximum benefit can be extracted from the existing hardware and software architecture without redesign or reimplementation of core software.

Man Machine Interface

Currently the STU MMI node employs the same hardware as the original system. A semi-graphic display generator provides a colour display, allowing operator control and viewing of alarm lists, one-line mimic displays and other functional displays. The display subsystem interfaces to the rest of the system via the distributed database, and new database class display handlers can be easily added to the picture display section. A simple special function panel provides easy operator interaction, and ten softkeys allow special functions to be added when required to particular displays.

Each DIU within the system contains an online monitor to allow display and monitoring of internal operation of the network and functions within each node. DIUs allow remote login to other DIUs on the network, making it easy for technicians to quickly find faults within the system. Display of LCR operation is possible, allowing internal variables accessible to each LCR to be displayed or modified.

Configuration

Due to the complex and intricate nature of the previous point database system, it was often time consuming and error prone to develop and test new configurations of databases. With the advent of an object oriented database, configuration becomes considerably simpler and more robust. The use of new configuration tools meant that the time to configure a substation was reduced to a third of the previous time taken, and greater confidence that the database was correct. Connecting the substation network to the development Ethernet network meant that new databases could be downloaded in a matter of seconds.

Reusability of LCRs and standard database elements such as circuit breakers, transformers etc meant that the task of maintaining existing databases became considerably easier. Another benefit was that due to the modularity of the database, changes made to the processing of particular plant items such as high voltage circuit breakers does not affect the operation or configuration of other plant items.

Master Station Interface

When the first STU sites were installed, PE did not have a centralised master station from which to monitor the substations. The installation of a Megadata control centre in 1988 allowed connection of the STUs to a low speed radio network, which allowed telemetry data to be returned to the master station. One benefit of this communications network was that new STU databases could be downloaded via the telemetry link. In the early system, however, the database was so large that it was considered to be too bandwidth consuming to download over the limited speed channel. The current system uses adaptive Lempel-Ziv compression to reduce the size of the database from a typical size of 160 Kbytes to around 70 Kbytes, so that downloading the database via the telemetry link becomes a viable option.

An interesting phenomenon occurs when an intelligent substation control system is connected to a master station that is designed for unintelligent RTUs. In essence the STU becomes a dual-ported SCADA system, where a local MMI contains displays and alarm lists, and the central master station also contains an alarm list and displays. Care must be taken that correct interlocking of controls occur, and that operator or master station overrides are correctly designed and implemented. The main problem is that the master station uses a simple telemetry protocol to retrieve the raw data from the STU. Currently underway is the implementation of a new PE master station employing Megadata's M•O•S•A•I•C product, which will allow much closer integration with the SMS database.

Current Developments

A number of developments are currently under way that extend the basic functionality of the SMS into new areas and performance requirements. Shortland Electricity has awarded Megadata a contract to install the SMS into a number of substations throughout Newcastle and surrounding country areas in NSW. Part of this contract will entail the phasing out of the older MD3000 MMI, replaced by a much more powerful MD1000 CPU based on a 25 Mhz 68020 containing up to 4 Megabytes of RAM, and 2 Megabytes of non-volatile database memory. This processor has an onboard 802.4 Token Bus controller along with a twisted pair RS-485 modem, allowing a high level of integration and performance to be achieved within a single DIU. The twisted pair network can operate at speeds of up to 2 Megabits within a local area, and can operate across dual media. Digital inputs will be scanned at 1 millisecond intervals, providing extremely accurate sequence of event recording in the the presence of system activity.

Prospect Electricity has a requirement to automate a number of larger transmission substations, and due to the critical nature of these sites, each substation will contain a duplicated MD1000 system acting as an MMI and communications host, so that failure of one of the MMI nodes will not cause loss of substation control from the master station. This duplication is achieved by operating a dedicated display console on a separate Ethernet, which is connected to both MMI nodes. One MMI node will operate as a hot-standby; failure of the online node will cause a failover to the standby node. The distributed database will ensure that system availability will be held to a maximum, with failure of any node not affecting the operation of any other node. The size of these substations are typically 3 to 5 times the number of I/O count of other PE zone substations, but the inherent scalability of a distributed system maintains consistent response times across a wide range of different application and customer requirements.

The State Electricity Commision of Victoria has awarded a contract to Megadata to install a simplified version of the SMS into a 500 Kv substation at Altona, where MD1000 DIUs will perform high speed scanning of telemetry input and transfer the data back to a M•O•S•A•I•C based master station via the 802.4 network and an Ethernet gateway. The benefits of having a communications and networking strategy that allowed easy and flexible interoperability with standard workstation components are clearly seen.

Future Directions

There are a number of areas where active research and development is being undertaken to expand the SMS product base. In keeping with the key understanding that the SMS is more of an application environment than a vertical product, these future areas fall into two broad categories; the development of the core product through utilisation of new hardware and software developments, and secondly the broadening of the applications operating on the SMS into new areas of substation control.

New Hardware

A number of new developments are planned to increase the performance or functionality of the core SMS product. The continuing trend towards high integration means that a cost effective single board DIU is possible that incorporates onboard networking and telemetry I/O; this would allow each item of plant equipment to have a DIU assigned to it, providing a truly modular system. Lower cost DIUs also makes it cost effective to place intelligent control in areas where it would have been too expensive previously, such as customer sites, small building substations etc.

Currently under test is a high speed serial controller card that would provide high performance (up to 2 Mbit) interconnection between disjoint networks, allowing high data throughput or fast response across an entire network. This will provide the hardware means by which substations can interoperate with other substation networks, and share data effectively.

Much discussion over the last few years has centred upon the possibility of direct measurement of primary quantities i.e eliminating current or voltage transducers. Apart from the cost savings, much more detailed information can be obtained through the use of high speed recording and signal analysis of the data. This would find applications in the fault recording and protection analysis areas. Megadata is using modern Digital Signal Processors to provide the high speed data processing required for such measurement. Given the right building blocks, many other applications open up in the feeder control and sectionalisation areas.

The phasing out of the older semi-graphic character displays opens up the possibility of using full graphics at the substation, without the requirement of a dedicated PC or workstation (which may be inherently less reliable than a completely embedded system). Full graphic systems have already been installed by Megadata at a number of manned substation sites.

New Software

Software improvement and upgrade is a continual process. Whilst it is not likely that the core elements of the SMS will radically change in the near future, significant improvements can be made to the functionality by value adding certain applications. Better simulation tools for executing and debugging LCRs, and graphical entry of control algorithms will make the task of application development much quicker and easier. Providing more network services is a relatively simple matter, such as the integration of standard Remote Procedure Calls (RPCs). As more and more networks are interconnected, and DIUs wish to access remote data, it is likely that the distributed database discovery method will migrate more to a distributed naming scheme, where proxy servers will monitor the whereabouts of data records, and provide clients with the address on the owner as required.

Whilst the current database architecture allows the creation of object hierarchies, it does so in an explicit manner. The integration of a MOSAIC style class database that incorporates multiple inheritance will allow a more natural creation of data models, and also provide a useful integration path for interconnecting the SMS with control centres.

Communications

The use of a standard networking protocol suite allows interconnection of substations through the use of a WAN. Newer communications technology such as the use of ISDN provide a cost effective yet fast and reliable method of substation interconnection. The provision of ISDN (or higher speed modems) at the substation also means technical staff can connect to the substation, download new databases, check the correct operation of applications, or retrieve history data. The ready availability of TCP/IP cognizant products ensures that the communications infrastructure is non-proprietary and low cost.

As more and more vendors support the OSI MAP standard, a migration strategy is in place to evolve the applications towards the use of MAP upper layer protocols. Initially dual protocol stacks can operate within an application layer bridge; eventually transport bridges can operate within each DIU. Since the DIU networking code is primarily based the freely available Berkeley Software Distribution TCP/IP code, it is relatively simple to add the OSI protocol stack support available in the latest BSD 4.4 release.

The SMS provides a good foundation for integrating proprietary equipment into the substation, such as intelligent protection relays, remote sectionaliser nodes, and other specialised equipment. It is relatively straight forward (but usually non-trivial) to communicate to these nodes using a defined protocol, and to merge the data obtained into the SMS database architecture.

Substation Applications

Prospect Electricity continue to discover new ways that DIUs can be used to run new applications within the substation. It is plain to see that PE have only just started to scratch the surface, and the potential for cost reduction and system robustness through automation is tremendous. Load management at large customer sites is a typical application, where other PLC or proprietary solutions are usually 5 to 10 times the cost, and the level of integration with the rest of the control network is low.

Megadata is discussing with PE many different applications that could be operated at the substation level, especially Energy Management systems suitable for distribution automation.

Whilst the SMS was designed and implemented as an flexible and powerful substation application environment, it can also be used in any plant control applications, or other electrical transmission applications such as feeder control, metering, sectionalisation, load management etc.

Conclusions

The architecture of the Megadata Substation Management System has been described in detail. The system as discussed has been installed and running in the field for over 12 months; earlier STUs have been installed for 7 years. An attempt has been made to show how developing a good foundation for a flexible distributed application environment (rather than delivering a system that can’t be extended) is a much better solution, as it will allow new applications to be developed as the need arises. Key to this strategy is the distributed object orientated database, an open communications protocol suite to allow flexible interoperability, and a powerful control language to allow implementation of sophisticated applications.

Much of the work now revolves around developing the applications that only now are being explored in any depth, such as cost effective load control, interconnection of substations, cost reduction through intelligent use of resources etc. The initial installations of the current applications have been successful in providing the customer with a large degree of substation automation, with a resulting saving in costs and increase in availability.

Longer term, the vendors of plant equipment must realise that more and more customers will require intelligent control of the plant items (such as circuit breakers, transformers etc.). Some manufacturers are placing PLC or other control equipment within the plant items, but usually this communicates to the outside world using a proprietary protocol. The trend towards interoperability and Open Systems that has revolutionised the general computing market must also carry through into the insular world of SCADA and control systems. Standards for the entire gamut of interconnection layers such as physical, protocol and applications must be developed that allow plant equipment to be connected into a standard control network, so that instead of networked DIUs being interfaced to the plant via telemetry I/O, the plant item itself is part of the network.

Controlling the future smart substation will then be simply a matter of plugging the plant items together with the standard networking (most likely fibre optic) connections, and then connecting the network to a central node that oversees all operations of the substations, and also acts as a communications gateway to the outside world.