Volume 10 Issue 1

May 1995

ICL

# Ingenuity

THE TECHNICAL JOURNAL

# Ingenuity™

ICL's Technical Journal

**Editor**

J.M.M. Pinkerton

ICL, Lovelace Road, Bracknell, Berkshire. RG12 8SN, England.

All correspondence and papers to be considered for publication should be addressed to the Editor.

The views expressed in the papers are those of the authors and do not necessarily represent ICL policy.

Published twice a year by Multimedia Publishing Unit, ICL, Bracknell and Manchester.

**1995 subscription rates (including postage & packing):**

| | UK and Europe | Rest of World |
|---|---|---|
| Annual subscription | £72 | $144 |
| Single issues | £43 | $86 |

**Ingenuity**

**Volume 10 Issue 1**

# Contents

*Front cover:* Customers using a "kiosk" (a specialised form of PC terminal for a multimedia application - in this case in a building society branch). Customers can interact easily with the system without assistance and are said to appreciate being able to get quotes for, say, deposits or mortgages without commitment or sales pressure. A paper on kiosks is expected in the next issue.

# Object databases and their role in multimedia information systems

**M. H. Kay**[1] **and Y. Izumida**[2]

[1] ICL Fellow, ICL Enterprise Technology, Bracknell, UK
[2] Fujitsu Limited, Kawasaki, Japan

**Abstract**

This paper describes why object database technology plays a vital role in the development of multimedia information systems, and indicates the features necessary in an object database to meet this requirement. These are illustrated by reference to ODB-II, Fujitsu's object database system. Some examples of multimedia applications implemented using ODB-II are described.

## 1. Introduction

Relational databases have become the accepted way of storing business data, but tables of rows and columns have their limitations. As people have moved towards graphical user interfaces, with richer ways of presenting information on the screen, the limitations of storing the data in tabular form become more apparent.

Over the last ten years many researchers have attempted to find richer information models with greater expressive power than the relational model, including work on knowledge bases, extended relational models, functional or semantic models, and object databases. Of all these models, it is the object databases that have advanced furthest in terms of commercial acceptance. Fujitsu's Jasmine project [Aoshima et al, 1990; Ishikawa et al, 1993] integrated many of the new ideas into a single system, and this technology is now marketed under the name ODB-II. ICL is working with Fujitsu to productise the technology for international markets.

Object databases are needed where the data doesn't fit neatly into tables. Examples include geographical information systems, engineering and configuration management databases, scientific databases (e.g. in molecular biochemistry), and intelligence analysis databases. Many of these applications turn out to be multimedia in nature.

What exactly is a multimedia database? A simple definition is one that contains media such as image, graphics, voice and video as well as conventional structured data. We will look at the implications of this both at the level of data modelling and semantics, and also at the level of the technical infrastructure needed.

The paper is arranged as follows. Section 2 gives an introduction to object database technology. Section 3 provides an overview of the ODB-II system. Section 4 is a description of the characteristics of multimedia databases. Section 5 contains an analysis of the requirements placed on the database technology by the use of multimedia, and Section 6 gives some examples of multimedia applications implemented using ODB-II.

## 2. Object database technology

The development of object database technology emerged from two separate research directions:

- In the mid 1980s it started to become clear to database researchers [Stonebraker, March 1990] that there were many application areas where the relational model of data was unsuited, because of the complexity and variety of the data structures. Examples include engineering databases, geographical and temporal databases, and some scientific databases. Stonebraker observed that systems in these application areas were still using hand-crafted data management rather than off-the-shelf database technology, simply because commercial database software could not handle the data complexity with anything approaching acceptable performance.
- At the same time the programming language community started to realise that there was something unsatisfactory about the fact that databases could only be accessed by embedding within the application a separate database sublanguage with a different type system. Researchers (see for example [Atkinson and Buneman, 1987]) started to develop the idea of *persistent programming*, in which a programmer can use the same constructs to reference long-term data on disc and short-term data in memory. A prerequisite for this is that the same type system is used in both cases.

Both communities turned towards object-oriented concepts as the way forward for creating a powerful and extensible type system that was equally relevant to the data-centred world of database definition and the algorithmic world of application design. There is now a good consensus in the industry as to the functionality required of an object database: as a database it must support the longevity, sharing, integrity, and accessibility of data, and as a container for objects it must provide a type system incorporating a hierarchy of subtypes, definition of object interfaces in terms of both state and behaviour, and polymorphic and encapsulated implementation of behaviour using methods.

Within this consensus on functionality, there is still a great deal of diversity in terms of syntax and also in terms of architecture. The most visible difference is in the nature of the database programming language: should it be an extension of Smalltalk and C++, as in the ODMG proposals [Cattell, 1994], or should it be derived from SQL, as argued by [Stonebraker et al, May 1990]. Behind this question of language is a debate about architecture: there are many significant differences between products in the way data is stored and in the division of work between client and server, which lead to completely different quality profiles in terms of performance, scaleability, resilience, and security [see Cattell, 1991]. So in contrast to the relational world, where to a first approximation all the products are much the same, in the object world all the products are very different. Increasingly the term *object database* is applied exclusively to those products with a persistent programming architecture, while products that incorporate relational operations at their heart are referred to as *object relational databases.*

ODB-II as a product derives from the Jasmine research project at Fujitsu laboratories [Aoshima et al, 1990; Ishikawa et al, 1993]. Like ICL's Raleigh prototype [Kay and Rivett, 1991] and HP's IRIS project [Fishman et al, 1989], this was strongly influenced by the functional data model [Shipman, 1981; Kulkarni and Atkinson, 1986] which provides its theoretical foundation. The object model provided by ODB-II includes the relational model as a subset, and the architecture has much in common with the established client-server architecture of relational systems; it can therefore be described as an object relational system. However, the object features are not a mere incremental add-on to the relational features: the data model and the native database programming language (ODQL) feel much more like an object system than a relational system. In particular, it is important to note that ODB-II is not implemented as an object layer on top of a relational layer: this is evident in the fact that navigational operations such as parts explosion can often perform ten times faster in ODB-II than in a relational database.

## 3. Overview of ODB-II

### 3.1 The object model

The ODB-II object model can be summarised by the following definitions and axioms. (Compare with [Kay and Rivett, 1991]):

Objects

- Each distinct thing of interest is an object
- Each object has distinct identity

**Entities and literals**

- There are two kinds of object, called literals and entities.
- The identity of a literal is its value. Examples of literals are the number 93.7 and the string "London". Literals are neither created nor deleted, they simply exist. The basic data types for literals are Integer, Decimal, Real, Boolean, ByteSequence, and String (the last two being variable-length).
- The identity of an entity is established by its creation-event. That is, an entity acquires a unique, non-reusable identifier when it is created, and retains this identifier until it is destroyed.
- Literals may contain references to entities. For example, if x and y are entities, the tuple [x, y] is a literal.

Classes

- Every object is an immediate instance of exactly one class. A class is a description of behaviour shared by a collection of objects.
- A class is itself an entity in the database, with operations to interrogate and update metadata.
- In general a class has one or more superclasses (the exception is the class *Object*). Classes inherit the characteristics of their superclasses.

Characteristics

- The behaviour and the state of an object are defined by the characteristics of its class. There are two kinds of characteristic, called properties and methods. Properties in turn are subdivided into attributes and relationships.
- An operation represents a service that clients can request. The target of the request may be an instance belonging to the class, the class itself, a set of instances belonging to the class, or a set of classes subordinate to the target class. The operation takes zero or more additional parameters, and returns a single object or a set of objects (entities or literals, instances or classes) as its result. An operation is implemented by an algorithm called its method. (Following common practice with other object systems, operations are often referred to simply as methods, but the distinction is useful and we will retain it here. In particular, a polymorphic operation is one that has several methods.)

- A property is a persistent value associated with an entity that clients may read and update. The value can be a single object or a set of objects; each such object can be a literal or a reference to an entity or class. A property whose value is a literal or set of literals is called an attribute; a property whose value is an entity or set of entities is called a relationship.
- There are two kinds of property, instance-level and class-level. For an instance-level property, each instance of the class has (in general) a different value for the property. For a class-level property, there is one value for the whole class. However, a subclass may have a different value for the property.
- A characteristic may be NIL, indicating that it has no current value.
- Characteristics are inherited by the subclasses of the class on which they are defined. They may be redefined on a subclass provided that they still conform to the constraints imposed by the superclass. The implementation of operations may be redefined arbitrarily (so that operations are polymorphic). Selection of an implementation (method) is done by searching for the most specific method based on the class of the actual target of the operation request.

Sets

- Sets can be used in ODB-II in most contexts where individual objects can be used, although technically, sets are not objects. In particular, the target, result, or parameters of an operation can be a set, and the value of a property can be a set.
- Sets are homogeneous, in that all members of the set must be objects of the same class. However, the members can belong to a subclass of this class.
- An ODB-II set is strictly speaking a list: it has the capability to contain duplicate members and to maintain the ordering of members. However, many operations on ODB-II sets (such as union and intersection) are designed to emulate the behaviour of mathematical sets.

As this discussion shows, the object model of ODB-II does not explicitly contain any multimedia primitives. The multimedia support in the product is provided (a) in the form of a class library built using this object model, and (b) in the way in which the underlying database engine is implemented.

## 3.2 The ODQL language

ODQL is a database programming language. Architecturally, it performs the same role as SQL in a relational database: that is, it is the language in which the application makes requests on the database server. Unlike SQL, however, ODQL is not just a query language, it is also a computationally-complete programming language with variables, assignment, conditionals and iteration. It is also used to write methods, which gives it an architecturally similar role to proprietary languages such as Oracle's PL/SQL.

The database aspects of the language and the processing aspects are inseparable from each other. The same types and operators are used throughout.

The syntax of ODQL is a pragmatic blend of constructs borrowed from SQL, C, and C++, though the philosophy behind it owes a lot to Daplex [Shipman, 1981]. The type system is the ODB-II object model, in the sense that the value of every ODQL variable or expression is an ODB-II object or set of objects.

Like SQL, but unlike traditional programming languages, ODQL provides the ability to add, delete, modify and interrogate metadata (class definitions) at run-time. At the same time, it is a compiled strongly-typed language. The unit of compilation is a method, and the system is able to check that the type definitions used by a method at compile-time are still valid at run-time: if not, the method must be recompiled. The source code of methods is held in the ODB-II database to facilitate this.

The flavour of the language can be seen from the following excerpts:

**Defining a class**

```
defineClass Employee super: Person
{       instance:
                Integer personnelNo;
                Location basedAt;
                Integer grade;
                Decimal [8, 2] spendingLimit();
        class:
                Integer nextPersonnelNo;
};
```

This code defines a class called Employee as a subclass of Person. As well as the characteristics inherited from Person, there are two instance-level attributes, personnel number and grade; an instance-level relationship to a location, an instance-level operation to discover the employee's individual spending limit, and a class-level attribute holding the next personnel number to be allocated.

**Defining a method**

The code of methods is written in C or C++ with embedded ODQL statements. ODQL statements are distinguished by an initial $ sign. The example below is written entirely in ODQL. It determines an employee's spending limit as a simple function of grade.

```
defineProcedure Decimal [8, 2]
            Employee::instance:spendingLimit()
{
      $if (self.grade > 10) {
            $return(117.50);
      } else {
            $return(5.00);
      };
};
```

**Executing a query**

A query can be executed against any set, in particular, the set of all instances of a given class, as in the example below. Note that instances of subclasses of Employee are included as well as the immediate instances of Employee. In this example emp is used as a range variable in the query, to refer to each employee in turn; the variable e plays a similar role in the scan statement which iterates over all members of the set ee. The query finds all employees based in London whose spending limit is greater than 100; this is followed by a scan of the result, which increases the grade of each such employee by one:

```
Employee set ee;
ee = emp from Employee emp
   where emp.basedAt.name = "London"
   and emp.spendingLimit() > 100;
scan ( ee, e ) { e.grade = e.grade + 1; };
```

## 4. Multimedia databases

### 4.1 The nature of multimedia information

In a database context, multimedia information can be loosely defined as information represented in one or more (some would say two or more) of the following forms: image, text, graphics, graphical animation, sound, and moving image.

Currently a great deal of interest, both in product marketing and in academic research, is centred on sound and video. This can easily distract us from the fact that the speed and capacity of computer hardware and digital networks have only just reached the stage where audio and video applications are technically viable, and there are still many situations where they are not commercially cost-effective. Also, information authors still have much to learn about how to exploit the potential of these digital media, and this can add greatly to the cost of the creative work in producing information content. For all these reasons, few business information systems today rely on sound or video, and even in areas where the advantage of these media is most apparent, such as public information kiosks and computer-based training, some of the most effective and successful titles are very restrained in their use of these media.

By contrast, static media such as image and graphics are now sufficiently well established that they are taken for granted at trade shows, but are only just reaching the stage of widespread exploitation in everyday commercial applications.

Multimedia information poses a number of challenges to database technology, including the following:

### Size of objects

Pieces of image and sound, and moving image in particular, are large. This requires efficient algorithms for storage allocation and management of recovery. Operations that reduce the information content, for example forming a thumbnail of an image or extracting the first few seconds of a piece of sound, should be executed close to the disc to avoid moving large amounts of unwanted data. Architecturally, it must be possible to transfer the data directly from the database subsystem to the presentation subsystem without copying it through each of the intermediate software layers (including, in particular, the application program). [See Gibbs et al, 1993].

### Isochroneity

Moving image and sound are isochronous media; that is, they have a time dimension associated with them and cannot be displayed statically. It is not acceptable to retrieve all the data from disc before presentation starts; in turn this means that the retrieval from disc must keep up with the speed of presentation. It may be necessary to synchronise two different pieces of media, for example a sound track with an animation sequence. [See Gibbs et al, 1994]

### Internal structure

Database people sometimes talk of multimedia information as *unstructured* information. Nothing could be further from the truth. Pieces of media are not monolithic objects, they have an internal structure. This structure can be very complex, even for static media (text, graphics, image) as evidenced by document architectures such as ODA [Campbell-Grant, 1987] and standard encodings such as SGML [Goldfarb, 1990]. What is true is that multimedia information is rarely constrained by a schema to quite the same extent as codified information. Although there will always be rules relating to a type of document (whether it be an article in *Ingenuity* or a recording of birdsong to go in an ornithological database), the effectiveness of multimedia information depends on allowing the author a certain degree of creativity.

Relational database systems have in recent years been extended to allow the storage of *blobs*, or *binary large objects*. Blobs are not really objects at all, since they are untyped. They simply contain unstructured binary data. This can of course be used to store images, sound, or even ODA, and this might appear to the user as a multimedia database. But because the database system has no knowledge of the data type, and therefore no ability to manipulate the information, this is really nothing more than an escape clause allowing the application to do the things that the database system cannot. The limitations of blobs are discussed in detail in [Gibbs et al, 1994]

### External structure

As well as their hierarchic internal structure, different pieces of media can have arbitrary relationships to each other. Examples of such relationships are:

A cites B
A supersedes B
A is a summary of B
A is derived from B
A is a critique of B
A mentions X, which is discussed in more detail in B

Such references are often presented in the form of hyperlinks between documents. A link may be to another document or to a component of a document. Clearly the issue of document identification is crucial, and the scheme of Universal Resource Locators used by the World-Wide Web [Berners-Lee et al, 1994] has recently received widespread acceptance.

Many systems in the past have relied heavily on the notion of a document as a unit of information: a document is often the smallest unit of update or retrieval and the largest target of a query. The concept of a document is of course based strongly on analogies with paper systems and is being challenged by the move to all-digital publishing: in a CD-ROM publication, it is not at all obvious where the document boundaries, if any, lie.

## 4.2 Query in multimedia information systems

The notion of query in a multimedia database is typically quite different from query in a relational database. In a relational database, we are concerned with establishing the facts: How many tomatoes did we sell last week? We like to assume that the database represents each piece of information once and that the information is factually correct. In a multimedia database, by contrast, we are concerned with discovering the documents that we need to read in order to establish the facts: find me the documents containing information about fungal diseases in tomatoes. Answering factual questions depends on the readers extracting semantic information from the document, and at the same time forming their own subjective assessment of its credibility.

Document retrieval has traditionally been the domain of text retrieval systems. There is a long tradition behind this technology, which has until recently been completely separate from mainstream commercial database technology. Originally the emphasis was simply on locating the presence of particular words occurring in the text; in more recent products an increasing amount of linguistic intelligence is applied to the text to identify its true subject matter. But the focus is still on assessing the relevance of the document to the user, not on extracting the information contained in the document and using it to answer factual questions.

Text retrieval technology can be readily extended to retrieve documents containing graphics, images, or other media. In fact it has been applied this way for many years, some examples dating back to the days of microfilm storage systems. The modern digital architecture is not essentially different: a textual description of each document is created, either manually or automatically, and text retrieval techniques are used to find first the textual description and from this the original document.

Some text retrieval systems have attempted to exploit the internal structure of documents. For example, the ICLFILE product (based on the work described in [Kay, 1985]) exploited ICL's Normalised Document Format which implemented an early subset of ODA. More recently products have appeared that exploit SGML mark-up in the document to identify its internal structure. Note that the operators needed to query this kind of structure are quite different from the operators of the relational calculus, which apply only to normalised tables.

Systems that attempt to use multimedia information to extract factual answers to queries (does this X-ray show evidence of tuberculosis?) exist, in live use as well as in the research literature. See for example [O'Docherty et al, 1990]. But in practice they are implemented by extracting the factual information from the media before input to the database. The facts can be seen as derived information, and as is always the case in database systems, derived data is stored when the costs of storing it are less than the costs of recalculating it. As a result, such systems impose no unusual burdens on the database technology.

## 4.3 Navigation and browsing

To supplement query (which we can define as content-based retrieval) multimedia information systems make extensive use of browsing (that is, ad-hoc retrieval of documents based on references from other documents).

Browsing is widely exploited in hypertext systems of which on-line HELP systems are the most widely-used example; more recently the World-Wide Web has introduced browsing on a global scale ("surfing the web"). The difficulty for authors of creating information links to guide the reader to the right place is widely recognised, and it is now generally accepted that some combination of query and browsing is needed.

A key issue is how to maintain the integrity of browsing references when the body of material is constantly changing.

Another requirement is time-based browsing, typically used to locate the required part of an audio or video sequence using operations such as fast-forward and pause. These operations introduce considerable architectural complexity especially in a wide-area context [see Laursen et al, 1994].

## 5. Requirements of multimedia databases

In this section we will examine the requirements placed on database technology if it is to handle multimedia information, and in particular we will try to identify why it is that research work on multimedia databases has almost universally adopted object databases as the supporting technology.

### 5.1 Query and navigation

We have seen that multimedia information systems need to provide access both through content-based query and through navigational browsing.

Early database systems, such as hierarchic and Codasyl databases, were navigational [Bachman, 1973]. Query facilities were added, for ad-hoc enquiries, as a layer on top.

The relational protagonists challenged this view, arguing that the programmer should only say what data was needed and not how it should be found. This led to the dominance of SQL as a declarative data access language. Where navigational access is needed, as in many engineering databases or geographical databases, it is implemented clumsily by a succession of queries using primary keys as pointers. This makes such applications very inefficient.

The early persistent-language object databases such as Ontos and GemStone brought a return to an architecture where query interfaces, if provided at all, are layered above navigational interfaces. [See Cattell, 1991]. By contrast, systems like ODB-II (other examples include UniSQL, Illustra, and HP's Odaptor — these systems are sometimes referred to as *object relational*) attempt a harmonisation of declarative and navigational access at the same level, within a unified model and language. Object databases thus support the combination of query and navigation that is needed for multimedia information.

The history of text-based technology is analogous. In this case the first generation of products were all query-based, using inverted indexes of search terms to locate documents by means of a search expression. Then hypertext systems started appearing, based entirely on browsing. Modern products for CD-ROM publishing generally support a blend of the two techniques.

### 5.2 Independence of storage format

With multimedia technology evolving at such a furious pace, it would be unrealistic to hope for a single set of standards to become universal. Different standards become popular in different communities either because of their technical properties or simply because of promotion by a dominant supplier or purchaser. As a result, the industry has learnt that information systems have to be constructed to cope with variety and change in both the standards and in the supporting technologies. [See Kay, 1993].

The object-oriented techniques of encapsulation and polymorphism are designed for exactly this purpose. Encapsulation hides the stored data behind a request interface; so, for example, one might request an 8-colour 2cm square thumbnail of an image in .BMP format, without knowing what format the image is actually stored in. Converters operate behind the scenes (as methods) to carry out whatever transformations are needed. Equally, polymorphism allows different implementations of these methods to coexist, so that different objects can be stored in different formats without the application needing to be aware of this. The application programming interface can be entirely independent of the storage format.

ODB-II is supplied with a multimedia class library that is organised on these lines. The relevant part of the hierarchy looks like this:

```
MMFile
        MMImageFile
                MMBitmap
                MMPixmap
                MMTiffFile
                etc...
        MMAudioFile
                MMSunAudio
                etc...
```

The class MMFile supports operations appropriate to any piece of media, for example import/export between the database and operating system filestore. MMImage supports operations appropriate to any still image, for example displaying the image on a particular window of a particular workstation. MMAudio similarly supports audio operations, such as playback and concatenation. MMBitmap, MMPixmap, etc., define the implementations of these operations for particular formats of stored image or audio.

## 5.3. Storage of large objects

Database systems were traditionally designed to handle millions of records each varying in size from a few bytes to a couple of kilobytes. By contrast, one second of high-quality video can occupy tens of megabytes.

Typically a large object will be stored as a chain of fragments. ODB-II chooses to manage this, not in some low-level storage manager, but within the methods of the multimedia class library. The fragments are each stored as separate objects. The size of the fragments is chosen to maximise utilisation of the underlying disc pages, which in turn can be of a user-selected size.

Recovery techniques for multimedia also deserve special attention. ODB-II uses a shadow paging technique, analogous to that first introduced in Postgres [Stonebraker, 1986]. For large objects this is more efficient than a conventional write-ahead log, since an updated object only needs to be written to disc once. At commit time, instead of writing another copy of the updated pages, a pointer is changed to make the new copy the current one.

It is not always appropriate to hold pieces of media physically within the database. Many utilities such as compression utilities and format converters expect to find the data in operating system files, and it is inconvenient to export them and re-import them every time such a utility is run. In any case, operating system filestore is usually quite efficient at managing modest numbers of large objects. The problems are of course in managing integrity: operating system filestore, at least in the 1970s technology in general use today, is not strongly typed or encapsulated, and is not subject to transaction-based recovery disciplines. In a client-server environment, however, these problems can often be tackled adequately at application level.

ODB-II therefore offers a choice of two ways of managing multimedia data, called external management and internal management. With external management, the data is held in operating system files, with the filename recorded in the ODB-II database; with internal management, the data is held internally to ODB-II. As with the question of data formats, the difference is hidden from application programs, which make requests on objects the same way in either case.

The way external management is designed also provides the potential to link into other external document stores, for example an image store managed by a document image processing system such as ICL's PowerVision, perhaps on optical media.

(Unlike the question of data formats, this is not implemented directly by subclassing: you will not find two subclasses of Media called ExternalMedia and InternalMedia. There are various reasons for this. One is the difficulty, even with multiple inheritance, of maintaining two orthogonal classifications of the same objects — here by storage location and by storage format. Another is the fact that an object cannot change its class without changing its identity; so it would not be possible to convert an external object to an internal object, or vice-versa.)

## 5.4 Modelling of time-based media

Conventional data modelling techniques can capture a great deal of structural and semantic detail about the nature of information, but they are not well suited to analysing the time dimension. Some of the complexities of modelling time-based information are discussed in [Gibbs et al, 1994]. They introduce the abstraction of a timed stream of media elements. Here the term media elements includes such things as video frames, audio samples, and events. The concept is general enough to embrace MIDI-style music (which consists of a stream of instructions to play particular notes at particular times), graphical animation, and timed text as used in subtitles or on an autocue. Various operations on streams are defined, in particular:

- **interpretation** operations which are largely concerned with understanding the details of how the media are encoded
- **composition** operations which specify spatial or temporal relationships between a number of streams to define a new composite stream: for example, adding subtitles to a video. Such operations have been extensively analysed by [Little and Ghafoor, 1990].
- **derivation** operations which transform streams in more complex ways, for example creating a transition or *wipe* between two video scenes. Other examples include colour separation, noise reduction, or MIDI synthesis (translating musical notation into sound).

The same authors in a separate paper [Gibbs et al, 1993] show how these modelling ideas can be expressed in object-oriented terms. They summarise the benefits of using an object approach as being (a) that details and variety of encoding techniques can be encapsulated, (b) that the complex composition and derivation relationships can be represented, and (c) that the active nature of an object database maps directly to the active nature of the information itself.

It is of course true that the current commercial products supporting audio-visual information do not exploit object databases. The activity of Oracle Corporation in promoting a wide-area video server product has misled some people into thinking that the job can be done with relational technology: in fact [see Laursen et al, 1994], Oracle have put most of their engineering effort into solving the challenging problems of achieving adequate performance and reliability of the network and the magnetic disc technology; the data itself is stored as a simple serial stream in its final transmission format, and is streamed straight from the discs through the network to the consumer's TV set-top box without going anywhere near a database of any kind. While this solution is technically very impressive, it does not address the more complex data management needs of, say, a news publisher collecting hundreds of video stories every day and repackaging the material in different ways for dozens of customers with different requirements and house-style. Currently such a user is likely to be using analogue technology, consisting largely of robotic juke-boxes handling physical video tapes. The broadcasting industry is moving to digital (magnetic disk) technology in a piecemeal way, replacing one part of the studio process at a time, and the opportunities for a database approach are only just starting to emerge.

## 6. Case studies

In this section we give some examples of multimedia information systems that have been implemented using the ODB-II system or its research prototype, Jasmine.

### 6.1 Mascot: publishing technical manuals

Mazda, a major Japanese motor manufacturer, has used ODB-II (actually its forerunner Jasmine) to streamline the process of publishing technical manuals (in paper form). The results of the pilot study were reported in [Kuwano et al, 1991].

The background to the requirement was the increasing variety of cars delivered by Mazda, the advance of technological innovation in car electronics, and the demand for ever-increasing quality of after-sales service. This meant the publications section had to deal with an increasing number of documents while shortening their production time and maintaining their quality. The total output was around 100,000 pages per year. However, there was a high level of re-use, deriving from the use of common components in different vehicles: as much as 70% of the content of a new manual could be made by cutting and pasting.

Mazda decided to store the master information from which the manuals were derived in an ODB-II database. This information consisted of text, image, and drawings. They needed a system that could handle these media, that could store hierarchic structures as well as complex cross-linkages and navigate them rapidly, and that could handle the management of multiple variants and translations of the same material. They also wanted to refer to information in external databases, for example a database of vehicle components held in relational form. For a year's production of sixty manuals of 1,500 pages each, the database was sized at 7.5 Gb: about 20,000 objects per manual.

The database was used only to hold the master information. For editing, information was extracted to a desk-top publishing package and later re-imported. The technical writers used embedded fields in the text to hold references to diagrams, images, or data values; once the text was in the database these could be interpreted as references to other objects.

The project concluded that ODB-II met all their requirements, giving excellent performance and programmer productivity. They concluded by recommending that ODB-II be used in other areas, including CAD and CASE repositories, and a move to CD-ROM publishing.

## 6.2 Kansai Electric Power Company

The Technical Research Center of the Kansai Electric Power Co., a major Japanese utility, have researched the use of object database technology and ATM networking to develop a multimedia-on-demand system, initially to provide a video kiosk for public relations purposes. The project was carried out in collaboration with Fujitsu Laboratories, and is described in [Iwamoto et al, 1995].

The purpose of the system was to provide members of the public with information relating to the technology of nuclear power generation, especially its safety aspects. As such it had to cater for a wide variety of users with different levels of interest and knowledge.

The system was implemented using a SUN server and a number of PC clients, in geographically separate locations (Kyoto, Osaka, and Nara), connected to the server using a B-ISDN network constructed using ATM switching nodes.

There were three major requirements on the database technology:

- Input and editing of information. All media were handled uniformly as digitised data. For video information, the MPEG1 and H.261 standards were used for encoding. Relationships between different pieces of information were recorded in the database. No attempt was made to record the meaning or content of the information, but this was seen as an important area for future work.
- Storage of information. ODB-II was chosen because of its ability to handle different kinds of information in a uniform way, and to handle relationships in a systematic way.
- Retrieval of information. Efficient end-user retrieval was seen as an important requirement. To resolve the inherent ambiguity in identifying pieces of media both at the input stage and on query, it was seen as important to support both query by conditional retrieval and also browsing.

The user interface to the system was provided by Fujitsu's IntelligentPad environment [Tanaka, 1993]. This provided a number of retrieval modes to meet the needs of different kinds of user:

- Spatial retrieval: clicking with the mouse on a pictorial representation of the nuclear power plant.
- Quiz: the user answers a quiz. Information is presented using movies, images, and voice.
- Guided tour: the system takes the user on a virtual tour of the plant; at each stage prompting the user with questions on some theme and using the answers to decide what information to present next.
- Keyword relationships: when information is displayed, keywords describing related information are also shown. The user can select a keyword to browse to related information.
- Image browsing: images can be displayed in reduced resolution, so the user can select the images he wants to see.
- Time-based browsing: movie sequences can be skipped forwards and backwards arbitrarily.

Although the system was implemented with a relatively small amount of data, the authors were able to conclude that using an object database gave considerable advantages, in particular the ability to manage all media in a uniform way, so that objects could be handled without considering the type of data.

The authors did report some outstanding problems, which are probably typical of many similar multimedia systems:

- The increasing trend to open systems allows a wide range of technology from different vendors to be exploited, but causes increasing difficulty in verifying that all the components work together.
- The user interface for the general public still needs further work to make it genuinely useable.
- Preparation of content is very time-consuming. Better authoring tools are needed.

## 6.3 Web publishing: the Oxford prototype

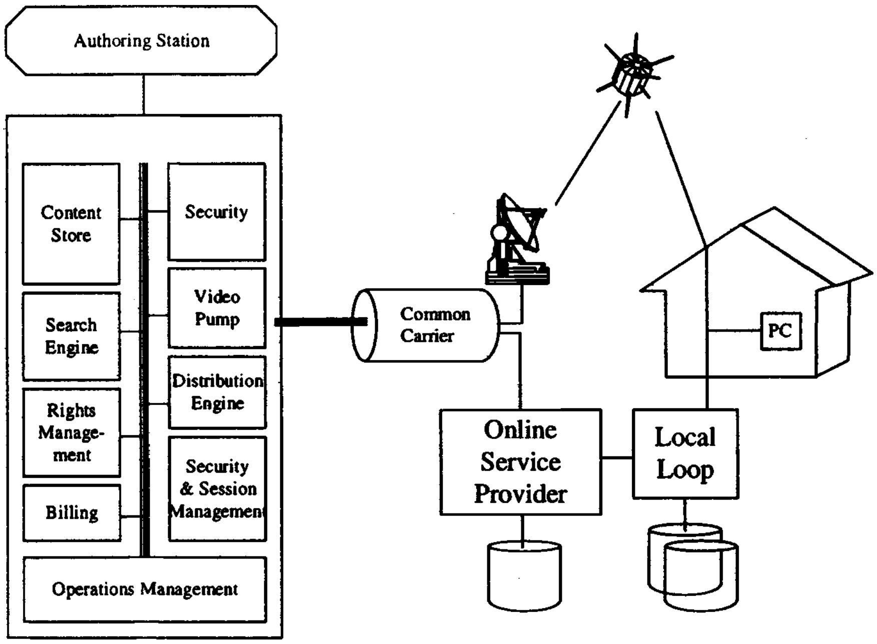
ICL has been developing collaborations with a number of large publishing groups exploring the potential opportunities and risks associated with on-line publishing.

The publishing industry is at a cross-roads: new technology is changing the nature of the business. The traditional bedtime book, of course, is not threatened, but it is widely recognised that some areas such as academic journal publishing cannot continue indefinitely in their current form. The key business issues are firstly, identifying exactly where the publisher adds value and thus avoiding the risk of being bypassed in the value chain; and secondly, finding means of payment that are seen as equitable. (*Ingenuity* itself has not been immune from these questions.) Like the software business, the publishing business suffers the problem that the costs of publishing are largely unrelated to the number of copies distributed and totally unrelated to the number of times they are used: this makes it hard to devise a charging model where price reflects both the value to the consumer and a fair reward for the supplier. There is also a great worry that electronic information, even when protected by copyright, will be even more widely pirated than paper information is. Management of security, accounting and billing, and royalty collection for copyright holders is thus a key component of any successful on-line publishing venture.

Many publishers currently feel safer with CD-ROM publishing than with on-line publishing. There is still a tangible product that is purchased for a unit price; it is relatively difficult to make illicit copies; and conventional distribution channels can be used. The problems are partly the lack of immediacy, and perhaps more importantly, the fact that on-line information is far more convenient to academic or commercial users, who want the information at their desks, not at a library on the other side of the campus.

The purpose of the Oxford prototype (so called because it used historical information about Oxford colleges and their alumni) was to show how an ODB-II information store could be integrated with World-Wide-Web technology to achieve the world-wide availability of information associated with the Web, coupled with a highly flexible framework for accounting and royalty management.

The architecture of the system is shown below. In the prototype, ODB-II was used to implement both the content store and the accounting system.



In the content store, pages of information were coded (manually with the aid of some simple scripting tools) as ODB-II objects. The objects were coded at the level of individual paragraphs, pictures, or hyperlinks, using a model that separated logical structure from layout structure rather in the manner of ODA. The user interface was delivered entirely through the Web client software product Netscape, running in a Microsoft Windows or Motif environment. To display a page of information, it must first be converted into the standard format used by the Web, HTML (a specialisation of SGML). This is done dynamically, and the conversion process takes into account knowledge of the user. For example, hyperlinks to a document in a different document series will be displayed as ordinary text if the user has not subscribed to that document series.

Charging can be highly flexible, using any combination of subscription mechanisms, pay-per-view, or payment for connection time. The content store notifies the accounting system of events that are potentially chargeable, and the accounting system invokes database methods to establish whether the events are permitted for that user and if so whether they are chargeable. This allows the mechanisms to be customised according to the needs of each publisher, and indeed to different categories of user.

There are several advantages in using an object database like ODB-II for the content store, rather than managing a collection of HTML documents directly in UNIX filestore as is done on a conventional Web server. These are largely the traditional advantages of a database: management of security and integrity, and support for indexing and query. For example, a traditional problem in any pool of hypertext documents is managing the integrity of the links, that is, ensuring that the destination of any link actually exists. Within an object database, it is easy to query the system to find out whether there are any dangling links, and to police the deletion of pages to prevent the situation arising.

## 7. Conclusions

Traditional database technology has largely been concerned with organising large collections of codified facts. With multimedia databases we are more concerned with providing access to large collections of documentary information. This imposes different requirements on the technology, an object-oriented approach is the only satisfactory solution. This will provide the stimulus for the acceptance of object database products such as ODB-II.

## Acknowledgements

The authors are grateful to many colleagues who provided information and comments. Especially Yoshinori Tanabe, who collected information from many parts of Fujitsu and in some cases translated it, and Hiroshi Ishikawa, who provided many pointers to the research literature. The Oxford Prototype system described here was developed by Mark Berner, Michael Hogan, Padraig O'Hiceadha and Andy McLeod.

## References

AOSHIMA, Y., IZUMIDA, Y., MAKINOUCHI, A., SUZUKI, F., and YAMANE, Y., *The C-based Database Programming Language Jasmine/C,* in Proc 16th VLDB Conf., Brisbane, 1990.

ATKINSON, M.P. and BUNEMAN, O.P., *Types and Persistence in Database Programming Languages,* ACM Comp. Surveys, 19(2), 1987.

BACHMAN, C.W., *The Programmer as Navigator,* Turing Award Lecture, CACM 16(11), November, 1973, pp.653-658.

BERNERS-LEE, T., et al, *The World-Wide Web.* CACM 37(8), 1994 pp.76-82.

CAMPBELL-GRANT, I., *Introducing ODA, ICL Tech J* 5(4), November, 1987, pp.729-742.

CATTELL, R.G.G, *Object Data Management: Object-Oriented and Extended Relational Database Systems.* Addison-Wesley. ISBN 0-201-53092-9, 1991.

CATTELL, R.G.G. (ed), *The Object Database Standard* ODMG-93 Morgan Kaufmann, San Mateo, CA, ISBN 1-55860-302-6, 1994.

FISHMAN, D.H., et al. *Overview of the IRIS DBMS.* In Kim and Lochovsky (eds), *Object-Oriented Concepts, Databases, and Applications.* Addison-Wesley. ISBN 0-201-14410-7. 1989.

GIBBS, S., BREITENEDER, C., and TSICHRITZIS, D., *Audio/Video Databases: An Object-Oriented Approach* in Proc 9th Intl. Conf. on Data Engineering, pp.381-383, 1993.

GIBBS, S., BREITENEDER, C., and TSICHRITZIS, D., *Data Modeling of Time-based Media.* Proc. ACM SIGMOD 94, Minneapolis, pp.91-102, June 1994.

GOLDFARB, C. F., *The SGML Handbook.* OUP, ISBN 0-19-853737-9, 1990.

ISHIKAWA, H., SUZUKI, F., KOZAKURA, F., MAKINOUCHI, A., MIYAGISHIMA, M., IZUMIDA, Y., AOSHIMA, M., and YAMANE, Y., The Model, Language, and Implementation of an Object-Oriented Multimedia Knowledge Base Management System. ACM Trans. on Database Systems, 18(1), pp.1-50, March, 1993.

ISHIKAWA, H., IZUMIDA, Y., and KAWATO, N., *An object-oriented Database: System and Applications* in IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing. May 1994.

IWAMOTO, H., MOGI, K., and TERADO, M., *Multimedia on Demand System: MMDS.* Fujitsu J, 46(1), pp.80-89. (In Japanese) January, 1995.

KAY, M.H., *Textmaster — a document retrieval system using CAFS-ISP. ICL Tech J*, 4(4), pp.455-467, November, 1985.

KAY, M.H., and RIVETT, P.J., *An Overview of the Raleigh Object-Oriented Database System.* ICL Tech J, 7(4), pp.780-798, November, 1991.

KAY, M.H., *The evolution of OPENframework.* ICL Tech J, 8(3), pp.365-382, May 1993.

KULKARNI, K.G., and ATKINSON, M.P., *EFDM: Extended Functional Data Model.* Comp. J. 29(1), 1986, pp.38-46.

KUWANO, N. et al, *Applications of Object-Oriented Databases to Publishing Systems* in Int. Symp. on Database Systems for Advanced Applications, Tokyo, April, 1991.

LAURSEN, A., et al, *Oracle Media Server: Providing Consumer-Based Interactive Access to Multimedia Data.* Proc ACM SIGMOD 94, Minneapolis, pp.470-477, June, 1994.

LITTLE, T.D.C, and GHAFOOR, A., *Synchronization and Storage Models for Multimedia Objects,* IEEE J. on Selected Areas of Communication, 8(3), pp.413-427, April 1990.

O'DOCHERTY, M.H., et al, *Advances in the Processing and Management of Multimedia Information*, ICL Tech J, 7(2), pp.271-287, November, 1990.

PRABHAKARAN, B., and RAGHAVAN, S.V., *Synchronization Models for Multimedia Presentation with user Participation.* ACM Multimedia, pp.157-166, 1993.

SHIPMAN, D., *The Functional Data Model and the Data Language DAPLEX.* ACM Trans. on Database Systems, 6(1), pp.140-173, March, 1981.

STONEBRAKER, M., and ROWE, L., *The Design of POSTGRES.* Proc 1986. ACM SIGMOD Conf. on Management of Data, Washington DC, May, 1986.

STONEBRAKER, M., *Introduction to the Special Issue on Database Prototype System.* IEEE Trans. on Knowledge and Data Engineering, 2(1), pp.1-3, March, 1990.

STONEBRAKER, M., et al, *Third-Generation Database Systems Manifesto,* in Proceedings of the First OODB Standardization Workshop, ANSI X3/SPARC/DBSSG/OODBTG. Atlantic City, NJ, May, 1990.

TANAKA, Y., *IntelligentPad*, Japan Computer Quarterly, No 92, 1993.

## Biographies

*Michael Kay*

Michael Kay gained a Ph.D. from the University of Cambridge in 1976 for research on database management systems. Soon afterwards he joined ICL, where he has worked on the development of successive generations of database software technology, including Codasyl systems (IDMSX), text retrieval systems (ICLFILE), and relational products (Ingres). His interest in object databases arose from work on data dictionary and repository systems, as described in previous papers in the ICL Technical Journal.

Dr. Kay is an ICL Fellow and a Fellow of the British Computer Society.

*Yoshio Izumida*

Yoshio Izumida received the B.S. degree in 1975 and M.S. degree in 1977 in electronic engineering from Yokohama National University, Yokohama, Japan. He joined Fujitsu Laboratories Ltd. in 1977 and is now a manager of the Database Special Project Department of Fujitsu Limited. His current research interests include object-oriented databases, multimedia databases, and parallel processing.

Mr. Izumida is a member of IEEE, the Association for Computing Machinery, the Information Processing Society of Japan, and the Institute of Electronics, Information and Communication Engineers.

# The ICL Multimedia Desktop Programme

**Ian R. Campbell-Grant**
ICL Fellow, ICL Enterprises, Bracknell, UK

Abstract

With an increasing range of available technologies and a greater rate of technical innovation, new opportunities arise to provide novel facilities for individual office employees, for example desktop video- and dataconferencing, hypermedia information nets etc. The efficiency and effectiveness of businesses can be significantly increased by such means, but doing so can require substantial investment. To make investment decisions sensibly, any business must investigate what technologies are most effective in its own case. By establishing pilot programmes the costs and benefits can be assessed.

This paper discusses such pilot programmes in ICL employing multimedia technology under Project Desktop. This involves cooperation between a number of distinct ICL businesses, each mounting one or more pilot schemes of its own. The aim is to use experience from the pilot programmes to facilitate wider use of the methodologies by all ICL businesses, thereby demonstrating their value to ICL customers. In view of the relative independence of ICL businesses, an important further aim is to identify the standards and guidelines necessary to allow them to interwork effectively.

The Desktop programme selected three main fields for particular attention: namely education, information and communications. Within these fields a particular set of applications likely to achieve business advantage was selected; concepts underlying these applications are explained. To evaluate the possibilities and likely benefits of the pilot programmes, detailed case studies were felt necessary; two of these case studies are summarised.

The types of technical issue that underlie meeting the business opportunities are identified and the available technical capabilities are described. A technical inventory lists those considered practical now, distinguishing those to be left to the future.

At the time of writing, the Desktop programme as summarised has been underway in ICL for about a year and the initial phases of pilots are established, allowing interim conclusions to be drawn.

## 1. Introduction

Two phrases widely used to characterise the way that the application of Information Technology may be evolved are *technology push* and *market pull*. Technology push is important because with the evolution of technology as well as quite new opportunities emerging (e.g. desktop videoconferencing) the costs of achieving capabilities can change and effect cost-benefit issues (e.g. the original windows implementations were expensive and cumbersome, the technology was not adopted until the price came down by orders of magnitude).

In businesses operating in an office environment, support at the individual employees desk has been mainly restricted to provision of, or support for, secretarial services (e.g. mail, document preparation). With the increasing range of available technologies and rate of technological innovation, new opportunities are emerging, for example desktop videoconferencing and dataconferencing, and hypermedia information nets. Use of such new technologies can result in a major change in the way individuals work and a company does business. The efficiency and effectiveness of businesses can be significantly increased but this often requires substantial investment.

To allow such investment decisions to be made, confidence is needed that the change will be beneficial to the business. One technique is to keep up to date with successful implementations in other businesses; however businesses are different and what is effective for one may not be effective for another. Often culture and personnel issues are as, or more important than, technical issues. To provide the necessary experience it is sensible for businesses exploiting IT to investigate and determine what technologies are most effective for them. One of the best ways to do this is to establish pilot trials. Key criteria for such pilot trials are that the pilot should be as close as practicable in detail to a wider implementation and that they are appropriately monitored. This is best achieved if the pilots are, even if only tentatively, justified by expected benefit to the business. Once the pilot is in place, achieved costs and benefits can be monitored, allowing informed conclusions on effectiveness for the business.

In such pilots the focus needs to be on the business application and not on technology per se. Thus the technologies applied should be established and should be available in deliverable, fully supported products. The pilots should avoid research or advanced development activities, as addressing both new business applications and new technology in the same programme is likely to fail.

This paper is concerned with a set of pilot programmes of this kind, concerned with the application of multimedia technology, being performed by ICL under the name *Project Desktop*.

This project is a co-operative activity between a number of the individual ICL businesses, each of which is mounting one or more pilot schemes of their own. The objective of each individual pilot is to establish practical business benefits to the particular business. The Desktop programme is not directed at devising new products but rather at determining how better to form and exploit combinations of those already available. Project Desktop aims to use the experience of the pilot programmes to encourage and facilitate re-use of solutions across the set of ICL businesses and so to establish proven opportunities for exploitation by ICL customers. In view of the devolved nature of the set of ICL businesses an additional key aspect of the Desktop programme is to identify those standards and guidelines which are necessary in order that technologies adopted by the various ICL businesses shall effectively interwork.

## 2. Multimedia

Multimedia essentially involves exploiting new technologies for people to have more effective communications by computer or with computers; hence information is represented in the most appropriate form which may include (but is not limited to) coloured images, hypertext, animations, video or sound. Multimedia provides for information to be used more effectively, for communication to be more efficient and for the form of representation to be that preferred by the recipient.

Multimedia generally exploits more aspects of human senses than conventional IT systems and to do this it requires higher bandwidth from technology. To get higher usability without losing the other important qualities such as performance requires enhanced system capabilities for communications and storage. Presenting information non-verbally or as hypertext involves some unfamiliar design skills for the originators of information.

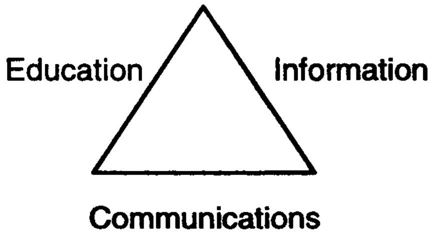
While multimedia has many new applications, for example in the consumer market, home shopping, or video on demand, there is also a potential for major applications to support the operations of business enterprises. In the case of ICL, in order to explore these opportunities the Desktop programme was planned to introduce office facilities enhanced by multimedia in such a way as to:

- improve the productivity and value of staff
- allow staff to advise customers based on personal experience of the advantages of new ways of working
- produce case studies to identify and describe the business benefits, based on the use of the OPEN*framework* methodology, [Kay, 1993]
- feedback to system developers and suppliers on the further facilities needed and their priority, so as to accelerate provision of business applications to ourselves and our customers.

## 3. Fields selected for attention

Owing to the rapid evolution of technology (and of the available set of products) businesses must focus on the new applications that can be realised, rather than particular products and what they can support today. This involves the definition of generic applications which can be regarded as user processes or scenarios. A pilot or set of pilot programmes is needed to identify how valuable these new applications will be to the company and whether they will be of widespread use or are applicable only in particular specialised areas.

For the ICL Desktop programme, which is concerned with the application of multimedia to increase the effectiveness of business enterprises, three main fields were selected for attention, namely: education, information and communications.



**Education** involves education both *of* and *by* staff. Education of staff involves delivery of education to the individual's desk. The opportunity for education at the desk can be used to develop the effectiveness of staff, providing the organisation encourages the individual to exploit this facility. Education by staff involves support for business presentations. Business presentations including multimedia can increase the effectiveness of messages, and thereby enhance business success.

**Information** involves the ability to access information from the desk; ideally staff should be able to refer to whatever information they want, wherever and whenever they need it. Business efficiency and the value of staff can be improved through excellent access to information (for example, product definitions or background information). In many cases such information should be recognised as a shared corporate asset. The quality of processes for the collection, organisation, presentation and maintenance of reliable information is critical. As well as information internal to the company, ready access can be provided to external information services. Information describing ICL and its products and services can be provided in electronic form direct to customers in order to enhance their interface to ICL.

**Communications** involves the ability to support business processes by improved interfaces to IT systems. This includes provision of asynchronous communications, such as electronic mail and document interchange; and synchronous communications such as desktop conferencing.

In addition, group working and virtual communities (individuals working closely together but geographically distributed) can be supported by shared facilities such as forums and electronic notice boards.

Within this framework, there are applications applying to most staff within a business enterprise; in addition there are particular specialist applications for particular specialist communities.

As in many cases, when defining any pilot or set of pilot programmes, there were different opportunities for investigation. In such a situation it was desirable that widely relevant applications were selected in the first phase that could be built upon existing facilities and be supported by practical and relatively low cost solutions. These initial applications were selected because benefits were real and implementation was feasible, but also because they were sufficiently demanding to provide useful progress in resolving the technical, organisational or business issues.

There were two main aspects to framing such a programme:

- identification of technical solutions to the provision of the applications selected
- analysis of potential business advantages, for various applications.

The project first undertook a requirement-collection phase and built a register of possible applications where multimedia could be used to better enable the business enterprise. For these applications, specific scenarios were analysed and case studies produced, identifying benefits and blockers and quantifying their prospective value (as far as possible).

For the part of the programme concerned with analysis of business advantages, ICL would promote the use of OPEN*framework* analysis techniques, used to good effect within Project Desktop, as summarised in section 6 of this paper.

## 4. Areas for initial study

After investigating a large number of possibilities the Desktop programme decided to select a particular set of areas likely to achieve business advantage through the application of multimedia technology:

Information

- information bases
- supporting multimedia infrastructure
- on-line and CD-ROM delivery of information

Communications

- enhanced electronic mail
- desktop video conferencing and data conferencing[1]
- support of group working

Education

- desktop learning
- presentations incorporating multimedia (audio/video)

Specialist communities

- legal
- home working
- reception

These areas are discussed in more detail in sections 5 and 6 of this paper.

## 5. Concepts

In developing the application of technology for business benefits, one has to take a view of the likely benefits to be derived from a set of concepts. Getting some agreement on what these concepts are is an important part of the further definition of any such pilot or set of pilot programmes.

The above areas for initial study, derived for the Desktop programme as a whole, may be expanded as follows.

### 5.1 Information

Information can be an important company asset, but only provided it is accurate and up-to-date, and stored in an easily accessible but controlled way. Multimedia adds value to information by enabling it to be structured for easier access, and to be presented for ease of use (a picture is worth a thousand words).

#### 5.1.1 Information Services

Business opportunities existed for the provision of and access to multimedia corporate and divisional information services. Already a number of information services existed, but these were principally text based with few, if any, internal linkages to facilitate browsing and

[1] desktop video conferencing provides person to person or multiway group conferencing, with a "videophone" connection; desktop data conferencing [Fuller, 1991] allows a variety of features including pre-prepared presentations, or collaborative real time editing on documents, visible to all participants in a conference.

searching for information. New forms of information services which began to exploit multimedia were starting to appear, for example, the World Wide Web (WWW) [Berners-Lee et al., 1994] and a prototypical ICL Web.

There was an opportunity to extend and exploit fully these multimedia information services, internally within ICL. These services could offer access by ICL staff to external services, or offer customers and suppliers access to ICL, allowing them to find out about the company, its people, and its products and services.

### 5.1.2 Information Topics

The multimedia information services needed to be structured for ease of reference according to topic and to assist the management in both gathering and maintaining the information.

The topics for which there was a demand included the following:

- marketing collateral
- product catalogues
- libraries of presentations
- site library information
- people directories
- manuals
- customer information
- technical reference information
- ICL business information

The most useful types of information had to be evaluated so as to plan the required structure and content of information servers and also to allow monitoring of the traffic on the network so that the required systems could be accurately sized .

Collecting accurate up-to-date information needs a federated activity including informed sources co-ordinated across various parts of the company. Even maintaining telephone numbers is a complex process. Central or nodal collection points are needed. Building such collections of information could be co-ordinated centrally but each part of the company should be responsible for its own portion of the information.

It would be desirable that everyone in ICL should, in due course, have access to an internal information base. Given suitable accounting facilities, anyone in the company who was expert on any special topic would be encouraged to make their expertise accessible within this information base. Multimedia information services might be offered on a commercial basis, for example users and/or suppliers of information could be charged for access by the service providers.

### 5.1.3 Information Media

To appeal to the widest possible audience for multimedia information services they should be provided on-line for interactive access or off-line on CD-ROM. Providing information services on CD-ROM would give users not having efficient on-line access (for example users working at home or away from the office) access to information (though this would not be so up-to-date as on-line information).

### 5.1.4 Information Management

In the longer term, once information bases were in place, opportunities for value added information services were perceived, possibly as an electronic research assistant to look for information on an intelligent basis, or as an electronic butler to organise and filter information once retrieved.

## 5.2 Interpersonal communication

Improving the way people communicate would improve the efficiency and effectiveness of ICL, reducing costs of exchanging information, by reducing the time taken, and by helping people to work together more readily. Various ways to achieve this are described in this sub-section.

### 5.2.1 Information exchange

Effective person-to-person communication leads to information exchange. There were business opportunities in ICL for improving the efficiency with which people exchanged the various information types whether by electronic mail or file transfer. It is believed that more advanced techniques, such as multimedia, would make communication of information more effective. Whether this is in fact the case needs to be established, and some Desktop pilots were specifically aimed to help determine this. [2]

Much time and effort was found to be wasted editing and reformatting information to display, print or reuse it. An opportunity existed with information exchanged within the company but, due to the wider variety of information representations in use, it was even greater with information exchanged with people outside ICL.

### 5.2.2 Meetings

A considerable amount of time and money was spent travelling to meetings. Some meetings did require personal contact, but many did not.

[2] Although electronic mail has clear benefits it can be a great time waster, e.g. one company has restricted internal electronic mail to be available only from 8:00-9:00, 1:00-2:00 and after 5:30. Although this is probably rather extreme it indicates advanced technology may not necessarily increase efficiency and that in some cases alternative solutions to email should be sought.

The current alternatives were either telephone conversations, though these were not always appropriate, or video conferencing using studios, but these studios were not always conveniently located or available. Where small numbers of people were involved desktop conferencing with at least audio and "white board"[3] facilities provided a more flexible and increasingly attractive alternative opportunity.

Where large numbers of people were involved for one-way briefings then desktop video conferencing with screen projection provided an opportunity to reduce travel costs.

### 5.2.3 Groups

Group working needed improvement, where the group is spread over several locations. Such groups included line management units, temporary working groups, and special or common interest groups. Clearly electronic mail provided one form of communication, but this needed to be extended to provide for storing and sharing group information.

## 5.3 Education

In view of the ever-changing business environment, on-going development of the skills of staff is called for to allow them to meet their continually changing goals.

Typically, education has been provided in formal courses and seminars where the student is part of a group. Flexible learning using work books and videos enables students to take courses at their own time and pace. Multimedia in the form of interactive text, audio and video can be used to enhance this education process. [Campbell-Grant, 1987]

Providing people with the framework in which to develop and improve their knowledge, skills and attitudes, is a key element of business success and involves the identification of:

- business goals
- personal objectives
- development needs
- education programmes

### 5.3.1 "Just-in-time" education

From an analysis using the above framework, multimedia technology was seen to facilitate the education of each employee (at least partially) at his or her desk.

[3] White board facilities are those allowing text and diagrams to be generated in electronic form and shared in real time with participants in a desktop conference.

Education at the desktop is improved by making it available as and when needed, and its cost is reduced. With education just-in-time:

- individuals can take more responsibility for achieving expectations and goals
- the organisation can be more responsive with greater emphasis on training as needed
- flexible scheduling allows costs of releasing people for training to be reduced
- people skills can be improved through more effective training

Standing in the way of achieving these benefits are:

- a lack of standards for the education infrastructure
- limited availability of electronic education programmes

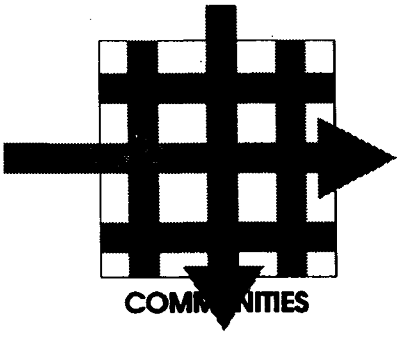
The major problem in achieving the success of such a programme is persuading staff to use this form of education. This is a cultural rather than a technical matter; in many cases training is treated by the individual as of minor importance. In particular, many middle-aged staff resist being educated, whereas staff in their 20s often thrive on IT; perhaps younger staff are more willing to learn because they have more recently received formal education.

### 5.3.2 Presentations

The multimedia content of business presentations for, say, marketing and sales could be increased, demanding the use of multimedia authoring and delivery tools. The key objective in providing multimedia enhanced presentations is to achieve better acceptance of the sales and marketing messages by making the presentations more appealing to the audience, by better conveying information and by increasing information retention.

## 5.4 Specialist communities

The preceding sub-sections have described concepts and techniques applicable generally to staff across the company. In considering pilot projects it was also worth considering specialist communities which might usefully be addressed.



Focusing on a particular community allows the complete range of IT support needed by that community to be considered. In the first phase of Desktop three specialist communities were considered.

### 5.4.1 Legal/Contracts Team

Legal/contracts staff have particular requirements for IT in day-to-day work. They include an electronic analogue of the complete customer file and electronic capture of all communications exchanged with customers.

Efficiency and effectiveness can be increased in preparing contracts and by making the customer file available to other staff by electronic means (e.g. taking this information to a meeting on a portable PC).

### 5.4.2 Home Workers

Tele-working can increase effectiveness of home-based workers and those working occasionally from home. Provision of relatively low bandwidth is sufficient to support email and World Wide Web mechanisms and information services can be provided by relatively inexpensive modems.

Using these same connections, multimedia information can be downloaded to a home PC, but cannot be supported in real time. Where high bandwidth is required, e.g. for conferencing, the solution currently available is ISDN. With the trend towards increasing home working, it is believed that high quality connection to colleagues (e.g. conferencing facilities) should substantially improve work effectiveness.

### 5.4.3 Reception Desk

Reception systems can be upgraded to enhance the image of the company presented to visitors, by providing multimedia imaging and communications This can include an image processing system for reception which produces instant photopasses for visitors. It can be used to capture information on visitors that can be held in a database. In addition, useful information can be provided for visitors, e.g. a description of the organisation together with photos of staff. Travel plans for visitors can be assisted by providing access to traffic reports, train and bus timetables and route planning capabilities and so on.

Whether the Security and Reception staff could cope with this major change to their working practices needed evaluation.

## 6. Case Studies

To evaluate the possibilities and likely benefits to be obtained from pilot programmes it was very often found necessary to go into much greater detail than was available by simply capturing and documenting concepts, as outlined in section 5 above. It could involve detailed case studies in an attempt to identify all significant pros and cons of a particular approach. To illustrate this, two case studies in the preparatory phase of Desktop are summarised below.

## 6.1 Example: Business Line (an in-house service)

Some key technicians believed that there would be substantial business benefit to be derived from providing professional staff (e.g. marketing staff, design staff) with access to on-line electronic information. However, despite this bias there was no solid foundation to justify the investment decisions that would be necessary. An OPEN*framework* case study was undertaken by the Desktop programme to provide such a justification.

In this study a full analysis of providing a wide variety of different types of information to a wide variety of different staff was not thought likely to lead to a clear benefits analysis. It was decided to analyse one typical application of such information in depth and that this focus should be the Business Line information service.

Business Line is a telephone help desk which provides a query answering service mainly for ICL consultants and customers. By providing a multimedia information service specifically tailored to these consultants and customers Business Line could give users direct access to information, and at the same time off-load most of the straightforward and repetitive queries. This would enable Business Line staff to concentrate on providing personal services which add greater value.

The analysis of the Business Line service took account of the user requirements, expressed in terms of the five OPEN*framework* qualities, and the business benefits that would result, expressed in terms of the four OPEN*framework* perspectives [Brunt & Hutt, 1993].

The key benefits to Business Line users were expected to be:

- more information readily available and up-to-date
- ability to browse to find the right information
- no congestion when new information becomes available
- no need for local information sources and associated gathering/ maintenance and storage costs

The key benefits to Business Line management would be:

- improved quality of service encapsulating a wider audiences
- release of personnel for more difficult queries

The main blockers were setting up and supporting the information services.

For users these are:

- possible upgrades to infrastructure, hardware, software and network (this may mean running at a degraded level until investment is recovered, e.g. transfers take longer, or some information such as audio or video cannot be presented)

For Business Line management the impact is on the service providers:

- initial training required on information structuring/set-up (Self-help groups exchanging advice and guidance on good practice and exemplar case studies may be required)
- maintaining large amounts of structural information (calling for investment in tools).

It was concluded that there were clear benefits for Business Line management and their customers (users) once the Information Access service was up and running. The disadvantages were the effort involved in creating a service with sufficient useful information, and in upgrading the users' platforms to provide access. However, starting with a small number of topics with limited types of content (for example text and graphics) and a selected user base, these difficulties could be overcome. The service could be extended incrementally.

## 6.2 Example: Desktop Conferencing

Again, some senior technicians believed that there would be substantial business benefit by providing access to desktop conferencing facilities. An OPEN*framework* case study was undertaken by the Desktop programme to see if the benefit justified the investment.

As in the Business Line case, the benefits of person-to-person communication were analysed taking account of the user requirements, expressed in terms of the five OPEN*framework* qualities, and the business benefits that would result, expressed in terms of the four OPEN*framework* perspectives.

The organisation studied was geographically dispersed. While the majority of the sales people were regionally based, many managers and consultants covered several, in some cases all, regions. Not only did this create problems with arranging meetings with clients, but also with internal meetings. A half day meeting could consume a whole day when travelling time was added.

The type of meeting where person-to-person communication is most valuable is one which:

- is short compared with the travelling time
- involves one or two people travelling large distances
- has few people present (ideally two but not more than four)
- does not require personal contact

Meetings in this category include:

- **sales tracking/forecast meetings,** enabling managers to reduce the time to formulate their sales forecasts and action plans by interacting with their sales people on forecasts

- **establishing deals**, enabling sales and marketing people to respond easily and to quickly reshape deals
- **people management**, enabling managers to see their people for objective and work reviews, and for work scheduling

The first requirement was to determine the real benefits from using multimedia technology to improve internal person-to-person communication. After a successful trial period this could be extended to a wider internal audience on a benefit/cost basis. The experience gained would be used to establish person-to-person communication between ICL and its clients, and also as an exemplar for selling person-to-person communication to client organisations.

It was concluded that there would be clear benefits for management and staff once a desktop conferencing service was available, and sufficient investment in kit had been made to allow it to be easier to use than spending a half day driving to a meeting. The disadvantage was the cost of setting up an adequate service. However, a small number of desktop conferencing kits at selected regional offices would give practical experience to those who could benefit most, enable any initial difficulties to be overcome, and quantify the benefits further. The service could then be extended incrementally.

## 7. The Technical View

As well as selecting areas for initial study, developing the concepts and performing case studies, there was a need to identify the technical possibilities, and to decide which capabilities were practical and which should be left for the future. The available capabilities, once analysed, would form a technical inventory. Given that the technical ability is fundamental to addressing the opportunities cost effectively, some guidelines for the approach were needed.

For Project Desktop, the following technical view was developed:

The various business opportunities were seen to call for four types of technical solution:

- real time conferencing (e.g. videoconferencing, chat services etc.)
- network based communications (e.g. email, file transfer)
- network based information access (e.g. on-line databases)
- physical information distribution (e.g. CD-ROM publishing)

There were two usage modes to consider:

- 1 to 1 (e.g. person-to-person videoconferencing)
- 1 to many (e.g. broadcast videoconferencing)

and, when dealing with information, two phases of usage:

- information search (i.e. finding the information required)
- information interaction (i.e. viewing/interacting with the information)

The four OPEN*framework* perspectives, ideally each of which should gain business benefits, are the *enterprise management, users, application developers* and *service providers*. Each of these has different key technical requirements:

**Enterprise managers**

Developments should be incremental to current installed systems, with upgrades dictated by business needs/benefits.

Enterprise-wide solutions rather than piecemeal solutions are desirable. Piecemeal solutions may already be developed for a number of opportunities. While these should deliver the required local business benefits, a single agreed technical approach would substantially increase the ultimate benefits.

Users

The piecemeal implementation of systems without a unifying framework may leave the user with multiple, non-cross-referenced information sources and multiple different access methods. Unless addressed this can prove a major blocker to productivity gains. Ideally users should have a single mechanism for search/browse functions for information.

Multimedia information should be presented at the highest level available in a user's local infrastructure (e.g. terminal, PC, sound card, etc.) - users should be able to *access* all information, but only *interact* at the level determined by their local capability.

The ICL mail network is excellent for normal business flow between users. However, if there is a predominance of email over information access systems this can waste time. A major problem with email is that the sender needs to know who will be interested in any particular item and therefore tends to broadcast information widely. The alternative of publishing information where interested parties can access it, as and when required, substantially reduces the volume of mail and simplifies individual filing requirements.

**Application developers** (e.g. Information creators)

Currently, the lack of a single set of document types and access mechanisms means that information providers have to provide and maintain their information in multiple media (often requiring multiple toolsets) and on multiple information databases. Similarly, application developers integrating information from various sources would benefit from consistency of information representation. Ideally there should be a single toolset to generate information types suitable for any form of distribution and easy conversion between information types.

Service providers

- Cost effective implementation and maintenance
- Conformance to corporate security requirements

The lack of an easily integrable set of solutions can make the cost of providing services high. Service providers can find it difficult to meet users' aspirations for access to multimedia information.

Common information formats would make it cost effective to reach a wider audience.

The types of information fall into two types which pose different requirements:

- time independent (e.g. text, pictures)
- time dependent (e.g. audio, video)

The technical view described above was taken as the basis for building the technical inventory described in the next section.

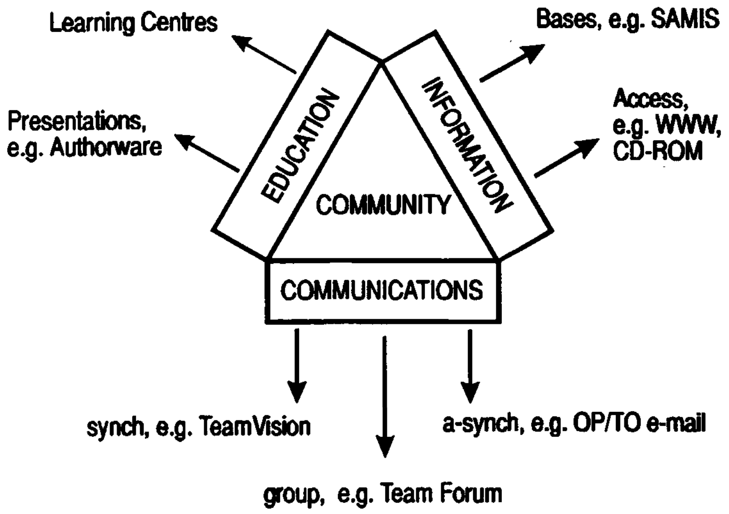
## 8. Technical Inventory

The Technical inventory surveyed the current technical solutions available, described pros and cons, and outlined the blockers to progress.

### 8.1 Real Time Conferencing

Desktop videoconferencing, such as ICL's TeamVision product, is becoming mature technology. The implementation of standards such as the ITU-TS[4] Recommendation H.320 allow interworking between different suppliers products for the "video" element.

[4] ITU-TS is the part of the International Telecommunications Union dealing with Technical Standards. This organisation has its headquarters in Geneva and defines standards for telecommunications (e.g. the international telephone service, the facsimile service). Until 1992 it was known as the "CCITT".



There were two major blockers to the widespread adoption of this technology. First, there were no widely agreed standards for dataconferencing - exchanging desktop information in real time (e.g. sharing windows, etc.). This problem could be overcome tactically by the adoption of a standard product, e.g. ICL TeamVision. The ITU-TS is developing a Recommendation for dataconferencing to be known as T.120 which is expected to be available in 1996. The second blocker was the requirement to pipe an ISDN connection to every desk. Installing this would cost approximately £400 per desk and therefore limit its use to critical staff only, or to "video-conferencing stations" acting like mini-suites. The correct technical solution is to carry the protocols out of the PC via the current LAN connection to a server, where the appropriate wide area connection can then be made. This is a non-trivial exercise and although most videoconferencing software providers are working on this type of connection, they are dependent on the LAN protocol suppliers also addressing the technical requirements.

Broadcast capability (i.e. one to many) is still under development. Various products are available in the marketplace, allowing multiple users to interact in real time using text and pictures, but the potential internal applications are more suited to desktop "work sharing" products (e.g. TeamVision).

### 8.2 Network Based Information Access

This is the widest area of application, and also has the largest number of techniques supporting it. There are two broad categories of system. In the first, users are able to "publish" information to a wider community via a moderated or unmoderated "forum". An example is the ICL

TeamForum product[5]. The information is usually fairly unstructured and indexed either by the user or the moderator. In the second, information "owners" publish a collection of fully indexed information to a user community. An example is ICL's SAMIS[6] information base.

From the user perspective, the source of the information is largely irrelevant. Currently users often require different "client" applications dependent on the format of the information. Ideally what is required is a universal search mechanism that can access information irrespective of source. Also, users would like a single toolset to access the information, irrespective of where or in what format it was originated.

Current systems do not provide a universality of access that allows easy proliferation of information.

A potential solution to this problem is to adopt the most flexible access/index mechanism currently available and implement it as a standard across all information bases. The World Wide Web mechanism is rapidly gaining industry adoption in this role because it offers:

- client applications for dumb screens, Windows PCs, X windows/UNIX workstations and Macintoshes
- server applications for UNIX, PCs and Macintoshes
- standard hypertext-based access mechanism, based on a Standard Generalised Markup Language (SGML) Document Type Definition Hypertext Markup Language (HTML)
- ability to cope transparently with multimedia data types
- ability to cross-refer transparently to other information in the network
- ability to provide a "file transfer" service within the same interface

Several prototype servers are now operating within ICL which prove the concept of uniform information access based on WWW mechanisms, such as allowing information to be "browsed".

Email is sometimes used for user-initiated information access by means of "mail-servers" to which users can email a precise information request and

---

[5] The ICL TeamForum product provides for sharing of application data (which can be documents, spreadsheets, presentations, graphics, etc.) between groups of users. The forum data can be replicated to any number of ICL Team Office or ICL OfficePower servers (using connections adhering to the ITU-TS X.400 Recommendations) such that populations of users across many servers can access the same data.
This population can form a virtual community - users can be anywhere in the world where X.400 services can be supplied (e.g. Japan, U.S., Europe).

[6] SAMIS - Sales and Marketing Information Service

the server application sends back the information automatically. This is simple to provide but not ideal as no support is provided for the information search phase. It is also not suitable for large files, videomedia for instance, which would soon swamp any client PC.

There are also significant issues surrounding support of videomedia. WWW mechanisms cope with videomedia by downloading the media files to the client where they are "played" locally. This has significant disadvantages due to loss of interactivity. The user must wait while the file downloads (assuming he has space on his PC to receive it) and any interactivity is strictly local after that.

The alternative is to spool the videomedia from the server. This however requires greater network bandwidth to ensure the user receives an uninterrupted flow of data. Handling this network bandwidth is possible by use of "structured cabling", already adopted in parts of ICL.

## 8.3 Managing Information Bases

There are benefits associated with maintaining reliable sets of information on various topics and there are also benefits in various communities of staff having access to that information. The client-server separation of access environments and interfaces from provision and maintenance of the information is key.

The information structure can be provided by a federation of good practitioners or 'profit making service facilities'. The provision and maintenance of an up-to-date information base on a particular topic, for example even something so apparently mundane as an accurate telephone list for a company, is a substantial process involving certain relevant people and sources of data. Other classes of information involve quite different specialists and processes. Standards are desirable to define good practice regarding quality, reliability, consistency and security of information.

Information should be prepared and held in a standard, multimedia aware form. There are already many content formats to consider and multimedia products are developing rapidly without much regard to the latent problem of compatibility. Corporate information needs a non-proprietary standard that can encompass the emerging content standards. HTML is the format used by Mosaic and the World Wide Web and is an obvious choice following their very rapid global adoption. HTML is a particular use of SGML and leaves open the possibility of migration to other uses of SGML or the complementary standard, ODA, [Campbell-Grant, 1987] according to market demands.

It is fairly straightforward to gateway HTML-based distribution services to existing databases such as the ICL SAMIS information base.

Using HTML means the freely available Mosaic information browser can be used on PC, X or Mac workstations; and as a fallback the text only

Lynx browser can be used on UNIX hosts. In general, multiple clients should be developed for access to the information so as to:

1) show what is available

2) display as much as possible in that environment.

## 8.4 Network-Based Communications

The commonest form of network-based communications is electronic mail (email). The ICL OfficePower and TeamOffice products allow easy attachment of multimedia documents to email. This is already bringing problems, however, as users now often receive documents they cannot read because they do not have the required client applications to open them.

This is being addressed to an extent (e.g. by support for browsing Word-for-Windows documents from OfficePower) but until a standard multimedia document exchange format (e.g. ODA) achieves critical mass this will always require technical solutions that have to be revised with each new product release. An achievable tactical solution would be to agree a limited set of document types and provide tactical solutions for these.

## 8.5 Networking Infrastructure

To support multimedia over networks, two aspects need to be considered - connections within one site, or campus using Local Area Networks (LANs); and, connections between sites using Wide Area Networks (WANs).

Within a campus, structured cabling should be implemented; this technology can deliver high bandwidths to the desk and is required to enable multimedia applications. It is currently being provided for a large number of ICL customers.

A choice should be made between Switched LAN technology, shared local LAN devices, or a combination to provide the necessary bandwidth to desks.

With WANs, current network infrastructures can often support low bandwidth multimedia applications such as off-line information services, email, and WWW mechanisms.

To support synchronous multimedia communications such as desktop conferencing and video conferencing, the ATM (Asynchronous Transfer Mode) technology has been developed for use over WANs. [Deignan, 1994]. However, ATM is not yet widely available and, in the short term, either ISDN services can be provided to the desk, or the campus network could support ISDN to the desk.

Support for home-working access to low multimedia bandwidth applications can be provided either by high specification modems providing high compression rates, or by ISDN. ISDN will be required for medium to high bandwidth applications such as conferencing.

### 8.6 Physical Distribution

For multimedia information, the most common form of physical distribution is via CD-ROM. The technology for CD-ROM publishing is well established. Most of the issues are centred on getting the information into a suitable format for publishing, and finding a suitable search/browse tool that can be used for many types of information.

Notionally, CD-ROMs can be mounted on a server in a juke-box to avoid all users requiring a local CD-ROM drive. This works for low levels of access, but, for instance, spooling videomedia material off a CD-ROM on a server to a number of clients is not practical. The information could, however, be copied to the server's local hard disc after which network distribution would be feasible (subject to the constraints on network bandwidth outlined above).

## 9. Standards

Having studied the requirements as described in sections 5 and 6 above, and technical capabilities, as described in sections 7 and 8 above, the next issue was how ICL as a distributed company could most effectively adopt this technology.

To achieve the benefits of efficient communications between staff in the various ICL companies, solutions adopted in the various companies must interoperate effectively. A key objective of Project Desktop was the definition of appropriate standards and conventions for this.

The definition or selection of such standards must be justified on a business basis and should be derived from practical experience, for example, with the Desktop pilots.

Examples of particular standards are the ITU-TS Recommendation H.320 for Desktop Conferencing and it is expected that the T.120 Recommendation will be proposed for data conferencing once this standard is established. Standards also cover the platforms and software required for various capabilities. For example:

- free use of different data types to represent the same information (e.g. audio) could result in different audio cards
- free use of applications could result in multiple incompatible documents, spreadsheets, graphics, etc. Use of a recommended set of products, with adequate conversion tools available, can avert many of these problems.

## 10. Interim Observations on Pilots

The ICL Desktop programme, as summarised in this paper, has been underway for just under a year and the initial phase of pilots have been established. As they are still underway at the time of writing, it is too early to give final conclusions. Interim conclusions are given for the pilots listed below.

Pilot projects underway within ICL include:

### 10.1 Information - On-line information base

The main justification for this pilot was to help staff keep up-to-date with the evolving ICL organisation and processes. It is important that they should be able to refer to up-to-date information on: the ICL organisation; on the allocation of responsibilities on commercial policies (to ensure these are accurately understood); and, on trading partners and subcontractors.

Initial information is on-line and in use by a substantial number of staff. An Open*framework* based paper has identified the various concerns to be addressed by the pilot.

### 10.2 Communications - Desktop Conferencing System

Using ICL's TeamVision product to hold meetings of established teams is proving a success. It is being used by a significant number of groups, both at a working level and at a senior management level, within the UK and internationally. In some cases, equipment is mounted on a trolley providing a mobile conferencing capability between groups (up to 3/4 persons at one end).

A definite knack has to be acquired to make successful use of desktop videoconferencing. There are a few good users, but some staff are resistant to technology (some people find it worse than dealing with answerphones). This seems connected with a greater "them and us" division than in a physical meeting.

The equipment has been connected to a big screen TV and works well. The use of this technology, e.g. for senior management briefings, would seem feasible.

### 10.3 Education - on demand training

Multimedia courses for demand training are in use, primarily at a number of dedicated learning centres. The biggest issues were to build awareness and establish regular usage. The learning centre systems provide a standard way for users to access courses. This system maintains a database of all logged-on use and statistics. In the early stages, usage by staff was low, primarily due to cultural issues. (At first many staff did not feel it right to use on-site learning centres except in their own time.) They required active promotion by personnel and support by management.

Staff using courses are mainly those advised to do so by personnel, mainly new joiners (e.g. a course on commitment to quality), secretaries (e.g. courses on making the most of the telephone, use of Windows), and selected groups involved in change, e.g. staff whose systems have been upgraded to use Windows. People who persevere find this form of training very useful and responsive to their needs.

### 10.4 Specialist Communities - reception desk

The first phase pilot has been completed, and the trial caught customer interest. However, security staff and (to a lesser extent) reception staff found it difficult to relate to and use the system effectively. The most important aspect was to understand the process of Reception and Security and their information requirements.

More comprehensive pilot systems taking account of these lessons are planned for later in 1995.

### 10.5 Specialised Communities - Legal/Contracts team

This is proving a good area for evaluating comprehensive Desktop services.

Legal and Contracts staff work in terms of "Folders"; all material connected with contracts is in electronic form. This pilot aims to provide Legal/Contracts staff with full electronic facilities to support the customer folder including information capture. Such an Electronic Folder has to contain everything its paper content had (e.g. email items, faxes, hand-written notes). Consequently, as well as image capture there is a need for general office facilities, such as direct fax into PC; X.400 and Internet Connections. Paper copies of contracts (and other correspondence) are still required, for legal reasons, but can just be filed, with no provision for routine access.

The first phase pilots are regarded as having gone well and to have been very successful. This technology allows the staff to be significantly more effective, (e.g. locating relevant information when handling queries is much easier and also there is much less chance of losing key letters/communications). Electronic folders can be carried in a lap top to meetings.

In the longer term, an information database could be supported, with several viewing stations for each building to realise some additional major benefits. In addition, textual information should be provided to the desktop, e.g. over OfficePower.

### 10.6 Specialised Communities - Home-workers/Teleworking

The home desktop has been technically proven to work to an acceptable technical standard. The initial phase of the pilots are therefore regarded as a success.

## 11. Conclusions

Interim observations on the pilots in the current phases of Project Desktop are included in section 10. Once these pilots are completed, it is intended that an overview will be published in a future issue of Ingenuity, together with the main conclusions of the project.

The sponsoring organisations had already identified requirements related to these applications. The Desktop programme has proved to be a useful catalyst to channel activities and apply them to the company as a whole.

This means the initial applications had to be considered from three aspects:

- providing the new service (technology, infrastructure)
- installing and supporting the application for users
- monitoring the results such that the company could learn from the experience.

In general, the project has already demonstrated its methodology to be well conceived. Results can be re-used by other companies in the ICL family and within solutions supplied to ICL customers.

## Acknowledgements

This work was only possible through committed team effort across ICL. I particularly wish to acknowledge the contributions made by the members of the whole team on *Project Desktop* and in particular John Collins, Peter Gardner, Andy McLeod, Bob Pennell, Roy Smethurst and Liz Tebbit for their help with this particular paper. I also wish to thank senior ICL management for their support for this programme, in particular John Davison, Andrew Boswell, John Elmore and Neil Allpress.

## References

BERNERS-LEE, T., CAILLIAU, R., LUOTONER, A., NIELSEN, H.F., and SECRET, A., "The World Wide Web", *Comm. ACM*. Vol. 37, No. 8, pp.76-82, August 1994.

CAMPBELL-GRANT, I.G., "Introducing ODA", *ICL Tech. J,* Vol. 5(4) pp.729-742, 1987.

COLLINS, J.R., and PRATT, J.M., "Multimedia Information used in Learning Organisations", *Ingenuity*. Vol. 10(1), pp.50-70, 1995.

DEIGNAN, F., "Asynchronous Transfer Mode - ATM", *Ingenuity* Vol. 9(2), pp.303-324, 1994.

FULLER, A.R., "Future Applications of ISDN to Information Technology", *ICL Tech. J.* Vol. 7(3), pp.501-511, 1991.

KAY, M.H., "The Evolution of the OPEN*framework* Systems Architecture", *ICL Tech.J.* Vol 8(3) pp.365-382, May, 1993.

BRUNT, R., and HUTT, A., "The Systems Architecture - An Introduction", ISBN 0-13-560186-X, 1992.

## APPENDIX

### A note on the World Wide Web

To explain the significance of WWW, here's a little story.

At a recent meeting of senior technicians there was this little conversation (suitably corrupted for propaganda reasons).

Hazel had been bemoaning the information explosion and how she couldn't cope with all this stuff she was being sent over the Internet. The one bright spot was Tricia. Hazel explained what an invaluable job Tricia was doing filtering masses of information in her specialist area and sending on to Hazel the bits that really mattered.

Penny was not so happy. She wasn't on Tricia's mailing list and was missing out on important information. Tricia said she couldn't send the stuff to everyone, otherwise she would just exacerbate this information explosion problem.

Well this is just the kind of thing which expert staff should be doing for their colleagues and all the other interested parties in the company. Its also the kind of thing that can be done much better using WWW.

The problem with email is that the sender needs to know who will be interested in any particular item, and even if he gets that right, everyone who is interested gets his own copy, and then has to decide where to keep it.

WWW is to email rather like video on demand is to broadcast media.

On WWW you don't send out the info. You just make it available. Those who are interested just read it in situ, they don't usually take copies, because the next time they want to refer to it, it will still be there and they can look again (unless it's out of date). Instead of thousands of people struggling to keep a filing system to manage knowledge of a specialist area, a few will organise the material, and this structured information is made available to all.

If Tricia filed the interesting material she received in her own WWW space, then anyone interested could look at it (though on an ICL server it would be just people in ICL, and this could be further restricted if necessary). No-one who wasn't interested would find it cluttering his mailbox (in fact even the interested parties don't get their mailbox cluttered). When Tricia finds a useful bit of information outside ICL, she doesn't need to copy it, she just puts a pointer in her web space, which anyone can click on to see the information.

In this environment, an objective could be that everyone in the company who is the company's expert on any special topic should make their expertise accessible over the internal WWW.

The key point is that the information should be provided and maintained by the information "owner". Its organisation and distribution on servers is secondary.

## Biography

*Ian R. Campbell-Grant*

Ian Campbell-Grant is an ICL Fellow, with responsibility for representing ICL at the highest technical level and for operating as an external technical authority. He has been with ICL for 29 years, and since 1981 has been instrumental in the development of application level standards for interconnection of ICL office products.

His current role is focused on OPEN*framework*, where he is an accredited OPEN*framework* practitioner and Company Architect for Multimedia, responsible for ensuring that the ICL portfolio relates appropriately to multimedia technology, so as to meet business needs most efficiently. As an OPEN*framework* consultant he is leading the corporate project *Project Desktop* (this paper) to apply advanced IT in the office environment, and in particular multimedia, to the benefit of ICL as a business enterprise.

Ian also played a major part in developing international open system standards. He led the European Computer Manufacturers Association (ECMA) group that produced the first version of the "Open Document Architecture" (ODA) standard, and now leads the work within the European Workshop for Open Systems (EWOS) on "Structured Multimedia Information". As reflected by these activities, he is recognised globally as an authority on multimedia and Open Information architectures.

He holds a bachelors degree in mathematics and two masters degrees, one in computer science from the Massachusetts Institute of Technology (MIT) and the other in mathematical statistics. In addition he holds an electrical engineering degree from MIT.

# Multimedia Information used in Learning Organisations

**John R. Collins and John M. Pratt**

Peritas Multimedia Systems, Windsor, UK

Abstract

The proposition is that all companies need to become *learning organisations* and institute *lifelong learning* for their employees. The concepts of *self-managed* and *just-in-time learning* are discussed and related to the process of learning as the business environment radically changes.

The paper shows how abstract business needs can be met through exploiting technology-based systems. It also shows a way of relating the different models for business and technology.

It is suggested that learning is more effective if delivered through multimedia techniques. The aspects of multimedia which are considered provide information of value in the learning process to organisations. The need to identify and manage components of multimedia information are discussed.

## 1. Introduction

This paper is a direct result of the Commission of the European Community research and development third framework sub-programme Developing European Learning through Technological Advance (DELTA). The DELTA programme had more than 30 projects on the exploitation of technology for education, learning and training (ELT) most of which emphasised the use of multimedia information. The whole DELTA programme was based on the assumption that some elements of the learning process can be aided by operating over geographically large distances using technology-based systems, such as satellite television transmissions in Greece and Portugal.

The paper proposes that all companies should be considered as learning organisations. By this we mean that employees needing to assimilate new information readily can now be helped by technological services as their future roles/functions change.

Organisations concerned with ELT can be considered as *learning businesses*, by which we mean enterprises whose core business is the provision of ELT. They include not only whole organisations such as (distance) teaching institutes and training companies, but also business units and project groups whose function is to deliver ELT within a larger enterprise. They would include ICL, or any other conglomerate, as well as smaller businesses using services from supplier organisations.

Learning businesses are involved with information in many forms and the use of multimedia information in many aspects of the learning process is now widely accepted as well as exploited. We use the term multimedia information to include:

- the intersection or integration of text
- graphics
- moving video and sound with colour
- significant artistic and presentational content
- real-time interaction and availability on a local PC system.

Usually we refer to "multimedia objects" as being a discrete (or contained) set of data which may include any one or more of these attributes.

Many organisations and educational professionals (as mentioned in the DELTA CD-ROM [Delta, 1994] and many learned publications) take the view that using multimedia information can increase retention rates and reduce learning times and therefore costs. Other benefits are touched on in this paper as well as case study reports from learning centres [Peritas, 1995]. These centres are being established by many organisations, including ICL, and provide concentrated facilities where individuals can gain access to information about the ELT process and more especially details of multimedia-based courseware for use on PCs. Courseware means all the elements of a particular ELT package and usually includes software and documentation.

Peritas has been a partner in three projects in the DELTA programme;

- *CTA - Common Training Architecture* which has produced the CTA Handbook [CTA, 1994]. The CTA is a tool to help learning businesses to procure technology-based systems which precisely meet their business requirements now, and which can be evolved readily to respond to business or technology changes in the future. It suggests that ELT needs can be described in business terms and related to standards. It proposes some example *scenarios* as well as profiles, and describes how these can be related to technology-based systems. It was especially developed to include the needs for multimedia objects in information technology and communications (ITC) systems.

  CTA aims to support the design and implementation of technology-based systems by promoting interoperability, portability, ease of use and re-usability. Recommendations are provided in technical volumes as to the standards and functional profiles which are identified as guidelines.

- *CTA-II - Common Training Architecture Phase II.* This project focuses on the integration, testing and validation of learning system implementations related to the recommendations of the CTA. The work includes advice on technical standards and their validation. In particular the project is building an Integration Information Database, based on ICL OPEN*framework* ideas for the Information Knowledge Database, for ELT systems.
- *EAST - Educational Access and Support Tools.* This project modelled the ELT processes, especially the learning phases of pre-learning through post-learning, and developed some prototype tools which have been evaluated in real learning environments. The people involved in these processes are described as *learning actors* and include learners, mentors and tutors, training managers and administrators as well as suppliers of services.

An essential aim of the three projects was to ensure that users' needs were properly captured and specified before more detailed technical work was committed. The CTA Project in its Architecture and Standards work reused a number of other models. These included the Open Distributed Processing (ODP) [ANSA, 1991] model, that of the European Foundation for Quality Management (EFQM), [EFQM, 1992] and the European Computer Manufacturers Association (ECMA) Computer Aided Software Engineering Environment (CASEE) reference framework model [ECMA, 1990].

All the DELTA work was substantially based on OPEN*framework* for understanding and expressing business and technology models. However, we particularly emphasised the practical aspects of using an architectural approach as a tool to help businesses. The successful results have demonstrated the practical benefits of using OPEN*framework*.

The business and technical models are entirely consistent with the OPEN*framework* techniques and ICL's decision to use EFQM as its *strategic quality method* (SQM). The essence of the work on the three CEC funded projects was to develop OPEN*framework*-based models and specifications as a framework and a process supporting exploitation of technology for ELT within organisations. We shall describe the CTA and EAST projects in more detail to show the relationship with multimedia information.

CTA is a framework and a process which organisations may use to help understand and solve ELT needs. It comprises a comprehensive set of models and other elements which, together, provide a way of describing all the business, pedagogic and technological needs of the ELT community in a structured way. Decisions can then be made on the technology-based systems most likely to meet those needs.

To help learning organisations use the CTA, six scenarios have been proposed which describe ELT situations and events using a number of criteria. For example one scenario concerns employee training including training at work, at a training centre or at home.

The CTA framework, in its basic form, is best shown in a simple diagram; see Figure 1.

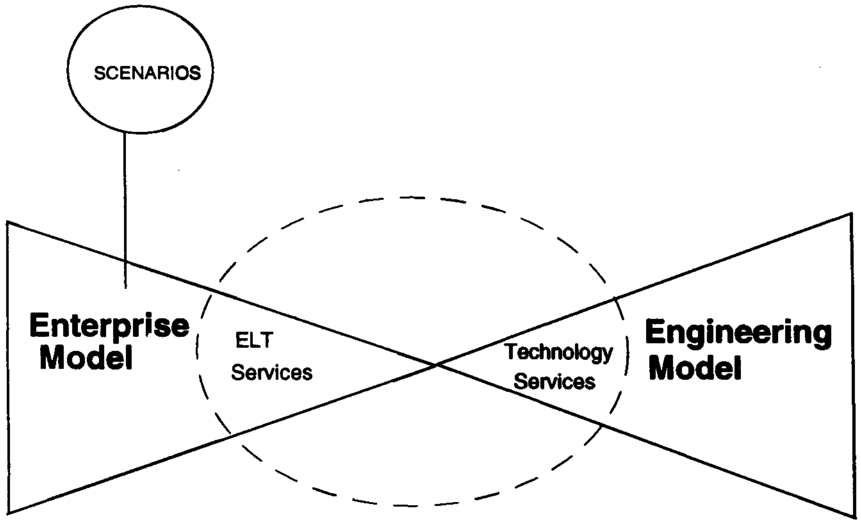


Figure 1 The CTA Framework

This diagram has many elements which are not shown, but which, to show the completeness of the CTA, will be touched on in what follows:

The Enterprise model includes the following elements:

- description of the learning business and its needs
- selection of a CTA scenario
- description of the pedagogic needs and ELT aspects
- outline of the expected learning situation
- identification of the important ELT criteria or qualities
- identification of any ELT policies to be taken into account.

*Services* are the means by which the output from, or results of, the Enterprise model are defined. When defined these ELT services become the essential feed into the technology services which in turn are then defined in the Engineering model.

The Engineering model of the CTA focuses on three technical areas, namely communications, information storage and retrieval and user interface.

This model includes:

- technology services in these areas
- relevant reference models
- relevant standards
- specific qualities for maintainability, etc.
- appropriate functional profiles
- example working configurations
- criteria for interoperability, portability and reuse.

It is even more important to recognise and take into account that many previously separate business areas are now merging because of the significance of multimedia material. For instance the integration of IT (or computing) with telecommunications has already started and the audio/video production businesses are now being added.

The dotted oval, in figure 1, represents the coherence and completeness of all the models and elements of the CTA. The CTA Project found it necessary to show that the use of CTA is not only iterative but could be entered at any point.

Further details will be provided in some of the following sections.

## 2. Business Need

The vision of the ICL Learning Organisation can be stated as:

> "to confer competitive advantage on the autonomous companies of the ICL Group by delivering practical ELT support to maximise the return on the assets of its people and their knowledge."

It is suggested that ICL is already a learning organisation because it has:

- an investors in people scheme certification in the UK, [UKIIP, 1994]
- investing in people practices and procedures, [ICLIIP, 1994]
- quality certification to BS5750/ISO9000, [ISO9001, 1994]
- quality practice to the SQM (EFQM) model; [ICLSQM, 1993]

However, there is a further need to identify how ICL can exploit its culture and practice as a role model of a learning organisation by using the latest flexible and distance learning technology in conjunction with its existing systems. Clearly such a learning organisation model, with all the expressions of service implicit in a practical implementation, would lend itself to the identification of separate services and therefore products.

Consider a few examples where multimedia information is of benefit to the delivery of technology-based systems in the new learning environment:

- the Peritas Self-Development Centre [Peritas, 1995] provides a variety of techniques for *diagnosis or needs analysis*. Tools are provided to help employees understand how business needs can be related to competencies and thus to their skills and experience.
- employees are now encouraged to move between jobs much more frequently than in the past, and thus would benefit from *just-in-time* training. This training can be *on-demand* from their desktop PC, whether at work or home. This is no different from manufacturing, except that the employee is the recipient of just-in-time training, when necessary.
- the concept of just-in-time training also means that one can take only that part of a training course or package which is important now. Thus courseware should be developed to ensure it is flexible and modular in construction. Learning centres are also showing that distance learning is a preferred option for many users. Providing information about multimedia-based courseware material, as well as the course itself, at the desktop using the ubiquitous PC has many benefits in saving time.
- providing easy access to experts within, as well as from outside, the company in a tutoring, mentoring role as well as technical support service role could be significantly enhanced by access to video conferencing. In addition, because so much learning material is now video-based there is a need to transmit very large multimedia objects.

One way in which the general need could be described is as a people development environment, meeting in part a growing demand for self-managed learning, where individuals are responsible for their own training and learning. It has already been suggested that much of the training and learning in the future will be provided through learning centres. The authors' concept of a learning centre ranges from a learner's home-based PC through to a technology-based training room set up at work.

Using some of the models previously described and based on the above discussion we can develop a learning organisation model, based on the ODP model. This considers IT architectures from 5 viewpoints. The notion of differing viewpoints is vital when using multimedia as it is itself a combination of many technologies and visualisations. The relative positioning of the EFQM, CTA and EAST models emphasise their use alongside the headings of ODP.

Figure 2 shows how the learning organisation can be considered as a combination of models and views.

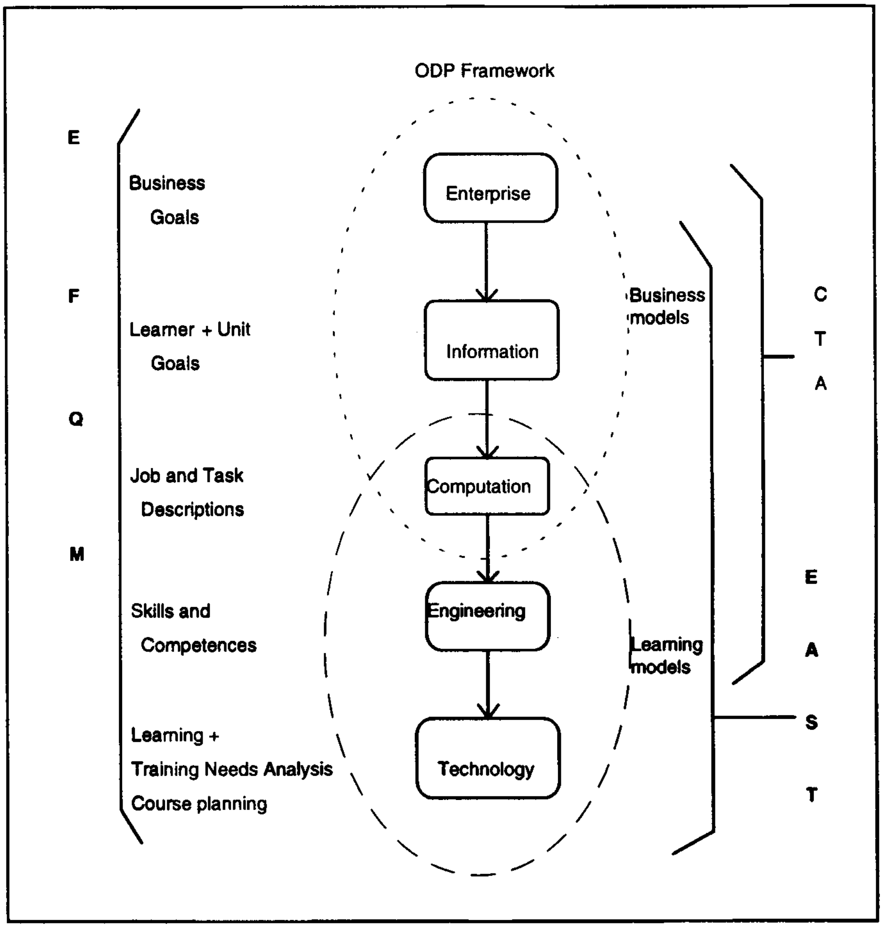


Figure 2 Model of a Learning Organisation

This learning organisation model could be realised by distinguishing the following main stages:

## 2.1 Business Goals

Model the enterprise to identify the key results from business units, and the resources to be deployed. Determine the key results areas for each employee and the methods of measurement. Identify the competencies required for these key results and tasks.

### 2.2 Personal Objectives
Agree with each employee key goals, objectives and deliverables, as well as methods of achievement. Implement a structured and regular review process with recognition of change built-in. Provide a means for employees to enhance their competencies.

### 2.3 Development Needs
Provide methods and tools to help each employee relate their competencies and strengths to the business and their personal goals.

Provide employees with the ability to analyse their knowledge, skills and attitudes against the goals and competencies required.

Provide support, from peers and others, to maximise the understanding of each employee in achieving their goals. Identify their ELT and experience needs.

### 2.4 Learning Solutions
Develop learning plans for each employee, with regular feedback, review and monitoring.

Provide solutions which may be initiated and (in part) implemented from the employee's desk.

A possible result of creating such a model or architecture is a set of associated methods that can be used to advise other enterprises how to "create their learning organisation".

Many of the tools and much of the learning material used in the future will be based on multimedia objects or even provided as multimedia material altogether. Many courses are now provided on CD-ROM and require that video and sound can be output from a multimedia PC.

This whole area is considered in the OPEN*framework* books: "Preparing the Organisation" [Hutt, 1993 & 1994]. These books cover:

- the what and why of learning organisations
- basic learning concept
- how to make business changes
- the mechanics of change
- establishing acceptance of change
- basic and further learning improvements.

Clearly there are many organisations, of all sizes, which are in a position similar to that of ICL in regard to implementation of a learning organisation. Hence there is considerable potential for a business which focuses on helping other organisations to become learning organisations.

An area needing further research is *valuing of learning services*. This implies that a measurement system should be in place to enable employees in an enterprise to value (many) aspects of the learning business. More employees are becoming responsible for delivering valuable services to enterprises as consultants. These consultants need considerable support from the organisation they represent as well as their own knowledge and experience. Such support from management consultancy companies is often provided by a corporate methodology. For ICL this support can be provided by the OPEN*framework* methodology and the considerable amounts of information available, by way of schemas, generic models and case studies etc., in the toolsets. There is significant benefit to be gained from reuse of such information.

## 3. Exploitation of Technology

Much has been written (and more will be) about the "information super highway" and its implications for distance working. The concept that working includes learning is becoming widely accepted. In addition there is now general acceptance that lifelong learning is a way forward for society. Exploitation of IT systems and telecommunications will change dramatically the way in which people work and learn. For example during the last two years the authors have been using ISDN to work alternately from their homes a few days each week. The practical experience and knowledge gained from this mode of working has been invaluable in the projects and other related work.

A major conclusion was the need to manage information if working is to be effective. While it is true that access to enormous amounts of information is invaluable to the authors' work, we recognise that too much information can be counter-productive. "Information overload" is occupying the minds of ICL employees responsible for access to information through the Internet. The problem is to find just that information which is valuable in the context of current work, a problem compounded by the fact that multimedia information is different and needs more powerful PC systems. This is for two main reasons. For fullest exploitation, multimedia objects include sound and moving video. The data storage required by such objects is usually very large, files of many Mbytes being needed for a few minutes presentation. Furthermore, since interaction between learner and a course running on a PC or similar system is vital for the best exploitation of multimedia, it is necessary that these large objects can be moved very quickly between locations, especially where servers are involved.

As an example, consider the technology exploited in prototype EAST systems in a number of user evaluation trials in five countries across Europe. The scope of the EAST system may be understood by reference to figure 3. Here training server and support tools have been defined as a complete system together with the actors, thus creating a total service environment. Using the ODP viewpoints, the system is defined at the enterprise level in terms of the goals of the project and, at the computation level, as a hierarchy of processes. The individual tools which could provide these processes have been defined in a separate associated specification on "Support Tools" [EAST, 1993].

Figure 3 below expresses the EAST architecture at the computation level and shows the conceptual transition of the learner through the learning phases formulated in the requirement stage of the project. Each phase transforms the learner, in terms of the state of his/her knowledge, and therefore the learner is depicted as a passive object being passed between phases. In addition, each phase will employ services provided by the central support, and these services include the time of the tutor and the attention of the mentor, as well as conventional infrastructure services. All services are scheduled by a central support process.

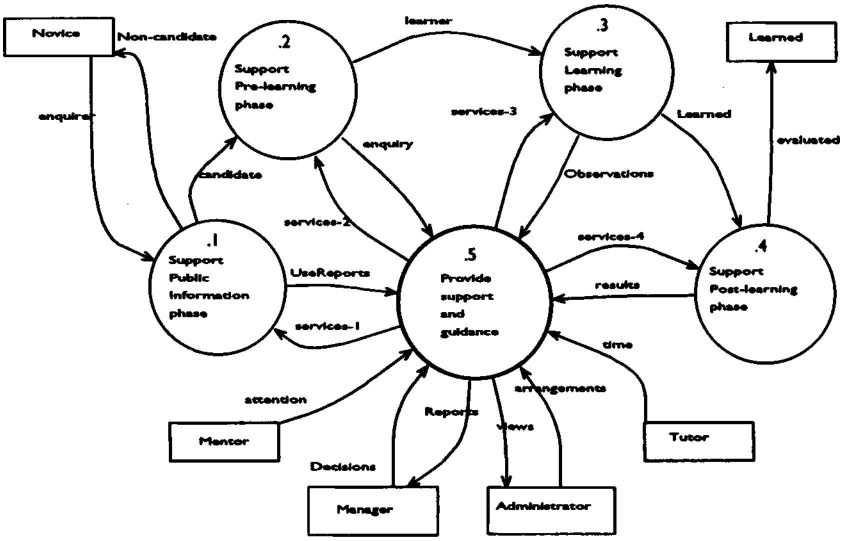


Figure 3 EAST Functional Model

The Public Information phase is concerned mainly with providing public access to the catalogue, and enabling the catalogue to be maintained.

The Pre-Learning Phase consists of three distinct but interrelated applications/resources, namely:

- a User profiler
- a Training Needs Analysis tool
- a Course Planner.

The Learning Phase includes the essential functions of running, tutoring and monitoring the course.

The Post Learning Phase includes the evaluation of the course, both from the learner's and the course provider's point of view.

The central process 'Provide Support and Guidance', numbered 5 in figure 3, embraces all the processes which allocate and co-ordinate the continuing resources, including the tutors and mentors. The allocation of these actors to the learning phases is expressed as the provision of 'service', which in detail will be appropriate to each phase. The process contains all the administration functions, and has the means to identify, schedule and account for all types of resources.

Tools included in the system:

- catalogue of learning material browser
- learner profile to show details of individuals
- technology-based multimedia learning material.

These tools are part of a comprehensive set designed to work in a client-server environment based on PCs and Windows. The requirement for power, networking and communication as well as storage can be extremely demanding. If there are many learners at remote sites with tutors and mentors then the information management aspects become significant.

Technology already used in other DELTA projects and therefore likely to be used in learning centres include:

- PCs and/or other workstations
- PC and/or UNIX and probably larger servers
- CD-ROM and other disc storage devices
- local and remote high speed/bandwidth networks
- satellite and similar communication systems
- television and radio broadcast systems
- mobile systems.

Further details and information can be obtained from the DELTA CD-ROM [DELTA, 1994].

## 4. Exploiting Multimedia Information

Multimedia information is used in a learning organisation to support a learning process, rather than as an end in itself. In other words a learning organisation aims to use multimedia to express many distinct aspects of ELT. The organisation gains from the semantics conveyed by multimedia objects, as much as from the multimedia objects per se. Multimedia enables authors of training courses to motivate the learner in many subtle ways. However the expressiveness it provides demands good management if large overheads in design and production are to be avoided.

Such management can be described in terms of the following types of activity:

- classification
- segmentation
- protection
- composition
- synchronisation
- abstracting.

These activities will be described in turn, indicating the aspect of multimedia information that supports each.

## 4.1 Classification

Multimedia-based material used for education, learning or training contains a variety of types of information distinguished as ;

| | |
|---|---|
| • domain | (knowledge, concepts, facts, scope) |
| • goal | (topics selected for instruction) |
| • instruction | (actual learning functions) |
| • user model | (characterising the user) |
| • presentation | (the material used) |
| • layout | (arranging material over space/time) |
| • management | (of material use and production). |

These types of information will often have been expressed simultaneously by the author of the material, and combined in a composite object. For instance a training course in a selected domain will consist of the topics selected for instruction, which in turn are presented using one or more sets of material, arranged in space and time to encourage learning by the user. In other words, learning material is conventionally multi-type as well as being delivered on multimedia.

It is difficult to classify such material as any one type, which in turn can prevent re-use of sections of the material for other purposes, and can also inhibit searching for the material by distinguishable attributes. Furthermore, if authors have to design all the types, then they need to be expert in many techniques (domain, pedagogy, graphic design etc.). Ideally these aspects of learning material should be disentangled and made more explicit and distinguishable, with potential benefits in terms of production costs. Thus, by classifying and describing components according to the types listed above, a learning organisation could aim to simplify the task of authors and, hopefully, allow reuse of components.

Turning to the use of multimedia to carry such information, we can analyse how these types are supported by the content and structure of multimedia objects.

### 4.1.1 Domain

This type can be expressed by means of a network (directed graph) of logical facts, rules and constraints. Such information is often not an explicit part of multimedia, but is implicit in its content. Where diagrams or images of real world structures are contained in multimedia objects, then any means of interacting with such diagrams must take account of the domain knowledge to constrain and guide the interaction. For example, moving objects in a virtual 3D space must be based on laws of motion within that space to support sensible interaction. A user who is experiencing such movement is being influenced directly by the domain rules, even if they are not visible. Information which controls the scope of the interaction is also of this nature; it is expressed as logical and numerical constraints.

Therefore interactive multimedia can be said to contain domain specific logic, which should ideally be explicit, rather than implicit in the content.

### 4.1.2 Goal

This type of information classifies the topics upon which the learning process is focusing. It is pre-defined by tutors as the learning objectives of the user, and by its nature is a set of measurable goals to be achieved.

Multimedia-based learning material supports this type by containing pre-defined states (such as having executed all modules of a course) which relate to the goals. These states are often implicit in the sequencing mechanism of the material, but if the goals are explicit then tutors can more easily observe and modify the progress of learning.

### 4.1.3 Instruction

This type contains examples, questions and/or answers, arranged in a predetermined sequence, which instruct the learner and test the knowledge gained; types are explicit in the content and sequencing of multimedia learning material.

Many kinds of media may be used as selected by the author and tutor. Note that both input and output of information may be wanted. For example, language instruction naturally expects the learner to speak as well as listen. Therefore a learning support system needs to be able to record and identify incoming multimedia material (such as spoken answers), as well as presenting it. In the simplest case this will imply yes/no answers given by the learner, but in more complex cases it may even be audio/video recordings of the learner.

### 4.1.4 User Model

This is a statement of what is known about the user, which in turn may determine the form or sequence of material presented. Ideally the material should provide differing functionality according to the type and skill of user (designer, mentor, tutor or learner). The types of user in an organisation would need to be classified in advance, so that the material can be adapted (or adapt itself) to the type of user. For example, learners could be characterised according to their level of attainment. The implication is that the author of the material needs to have a model of the target organisation, at least in terms of types of user.

Multimedia learning material refers to such a model according to predetermined criteria, and such references are expressed in a language convenient to the author.

### 4.1.5 Presentation

Presentation is the conventional multimedia content, that is text, voice, sound and vision objects. Note however that the material should adapt to a range of constraints (e.g. window size, natural language) determined by the layout mechanism described below.

The material must therefore be defined in such a way that it can be adapted to fit such constraints. Examples of this are scaleable fonts, geometric graphics, and scripts translatable from one natural language to another. In other words, the contents of multimedia objects should distinguish between what is the content, and how it is presented.

### 4.1.6 Layout

Layout expresses precisely *where* on the screen and *when* in a time sequence the material is observed by the user. Generally, it is independent of the rest of the content, and is often set dynamically by the user. However, an author may wish to predetermine or constrain a particular layout, in order to convey some instructional point. For example, an object may be moved to show its relation to some other object. In such cases layout information becomes an explicit part of the multimedia learning material, to be controlled and used as required by the instructional sequence.

### 4.1.7 Management

Management ensures that the material is available when required by the user. Multimedia learning material should include relevant information (such as identity, ownership, status) which supports the management of the material. It is not desirable to dissociate management information from the content of the multimedia objects, since this undermines the integrity of the information. Each unit of material should include at least a unique identifier.

Thus classification identifies how the content of the multimedia objects expresses all the types of information needed by a learning organisation. Current practice is to use all such types as required without making explicit distinction between them, but this leads to excessive custom design of learning material. To improve this situation the 'Common Training Architecture' has at its core a "Common Information Space" [CTA, 1994a] which includes the above classification of types. This should encourage projects to distinguish the types, and so be more efficient.

## 4.2 Segmentation

This activity takes multimedia-based material and divides it into separate functional components, which can then be managed more easily. The activity needs to occur as the material is designed, and could bring benefits by way of:

- lower storage costs (by avoiding duplication of components across modules)
- Lower communication costs (by moving only components which are needed, when they are needed).

This policy is slowly evolving in the office automation market, where large monolithic applications are now being decomposed into smaller re-usable functions. So, for example, one spell checker function can be used across an office suite of word processor, spreadsheet, and database. The same policy is needed for learning material.

Often an instructional activity will need to call on functions such as word processors commonly provided in the environment. The important concept here is that common functions are called by the interacting object, rather than asking the user first to activate a function (such as word processor tool) and then to select material (such as a file to edit).

## 4.3 Protection against unauthorised copying

Publishers currently have difficulty controlling distribution over a wide market. Hence legitimate users must pay a higher share of production costs, in the extreme case commissioning bespoke design and production. If publishers had more feedback and recompense for their skill, then it is possible that a wider variety of material would be produced, and at more affordable costs. This in turn would help learning organisations to obtain appropriate material at more reasonable prices.

Such protection requires that multimedia objects are packaged with appropriate access control, to prevent unauthorised use. The packaged material no longer has a visible, passive, file structure, but can only be accessed by execution. This practice is already evident in the distribution of 'locked' material on CD-ROMs given away on the covers of magazines.

## 4.4 Composition

Generally multimedia learning material is demanding in terms of storage, communications and machine performance. Although technology is advancing rapidly to provide better solutions, a user organisation will always gain if it can compose learning material as efficiently as possible from local and remote components, avoiding unnecessary movement or storage space. At one extreme the material is composed by authors and then delivered (broadcast) in one physical unit such as a CD-ROM or video tape. This is efficient for delivery, but offers poor response to changes. At the other extreme material is held as a large collection of small components in a random access store, which is expensive. Instructional material requires a compromise between these extremes, by allowing sections of broadcast material to be mixed under the control of mutable control information.

The lowest cost distribution option (video tape or wide area broadcasts) is slow to access in a random order, and possibly too generalised. However tape can be indexed digitally, (as in video editing machines) and therefore could be managed to support a complex course structure. Publishers could make more use of such delivery mechanisms, including guidance on the selection of sections and modules. Alternatively CD-ROM can be used in the same way for lower volumes with the advantage of random access. In both these cases the initial publication needs to allow for future changes and additions.

The interests of the learning organisation are best served by mixing the high volume (generalised) material, with a low volume of expensive specialised material and amendments.

## 4.5 Synchronisation

For maximum learning potential multiple streams of material may need to be synchronised. Indeed, this may include multiple video streams, or video synchronised to information structures. The synchronisation of delivery may need to be as fine as 'lip synch' between speech and mouth movements. This in turn requires that the multimedia learning material itself is indexed in some way to support synchronisation.

Synchronisation of remote delivery of material is particularly difficult, because of the time delays imposed by the remote delivery. Communication suppliers have some solutions to this in special cases (such as direct broadcast), but in the more general communication situation the timing cannot be predicted. Even a stand-alone PC can have difficulty in predicting the timing of delivery of material. Therefore the material itself needs to include explicit indexing and timing information, which can then be used directly to provide synchronisation on delivery.

## 4.6 Abstracting

The learning organisation is concerned to maintain the effectiveness of its employees, and therefore needs to be sure that little time is wasted on browsing. Either the learner is supported by very effective and fast browsing mechanisms, or needs to be encouraged to consult a mentor who has pre-digested the available material.

The learner needs to be able to navigate easily amongst many sources of knowledge and material, and requires an immediate response to selection of a key phrase or highlight, followed by detailed retrieval of the associated information. Naturally, if the material is based on multimedia objects, then the problem arises of providing an appropriate 'clip', which is short but meaningful. It is not adequate to start running the whole of the material, as this is time consuming. It would be more useful if the catalogue material always included specially prepared clips for such browsing.

A mentor may be either within the organisation, or in a professional support organisation. Apart from providing text and voice communication, the mentor may also wish to provide additional clips of material, and to send these to the learner.

## 5. Illustrative Example

As an example, a particular item of multimedia learning material will now be analysed, chosen from "The Office Professional" course published in the ChangeMaster series by Peritas Multimedia [Peritas, 1994].

The course aims to help office staff organise their time more effectively while coping with unplanned events. It is published on CD-ROM, along with a work book, and includes video clips and sequenced instruction.

### 5.1 Classification of contents

#### 5.1.1 Domain

This is a conventional British commercial office. The staff portrayed have typical roles (supervisor, assistant) and perform the business processes mentioned in the commentary. Although this is unsurprising to us who work in such an environment, this domain may not be suitable for learners from other backgrounds. The structure of the domain could be made explicit in terms of interrelated roles, processes and rules.

#### 5.1.2 Objectives

The learning goals are stated in descriptive text, and are implicit on the sequencing of the material into modules, each of which focuses on a subset of the overall goal. The attainment of the sub goals is recorded for each learner in an associated data file.

#### 5.1.3 Instruction

This uses video, voice and text to set up a domain situation, followed by questions to the learner on the preferred actions which should be followed. The sequence of instructions is set by the course designer to respond to the learners answers, bringing in additional material as required.

#### 5.1.4 User Model

A learner is first assessed by means of questions on attitude, skill and experience. This is recorded so that the subsequent instruction sequence can adapt itself to the learner.

#### 5.1.5 Presentation
This uses simple menu selection from questions, with additional video and sound clips overlaid as required. The graphic design is fixed by the author, and uses attractive fonts and colours. The language selected is English.

#### 5.1.6 Layout
No provision is made for the learner to change the given layout. Commercial standard IBM-PC compatible technology is assumed.

#### 5.1.7 Management
The material is packaged as a CD-ROM plus a supporting ring binder with worksheets and notes. This package is physically copied and distributed by the supplier as required, but does not have a unique serial number. On initial use the course creates an electronic record of the user on the PC on which it is running, but allows multiple users to execute the material on multiple machines.

### 5.2 Segmentation
The course as delivered is compiled and no segmentation of the instruction is provided for any other use. The video clips and images are stored as distinct items and can be reused. The course designer has more flexibility as the source material is held as small segments which may easily be reused as proformas (such as menu selection screens, or simple interaction sequences).

### 5.3 Protection
No protection is included with the course.

### 5.4 Composition
The course consists of a control program and a large set of clips. These clips can be managed separately and, for example, a new clip can be delivered and inserted into the control program by recompilation.

### 5.5 Synchronisation
The instructions are sequential and no synchronisation is needed. Synchronisation of sound/vision is needed within the video clips, where it is implicit in the recorded material.

### 5.6 Abstraction
The course is described in associated brochures, but the course itself does not contain any abstraction which could be separately browsed.

Summary of example:

By studying the course, using the classification proposed in this paper, we can see how it provides the distinct features, and that this can then lead to better comprehension of what it does and does not provide. For example, the fact that it trains users in task organisation is independent of the language in which it is delivered. Such classification can then cater for more efficient reuse of the various aspects of the course.

## 6. Conclusions

Multimedia information will enable learning organisations to be more effective provided that relevant explicit information is included in the material.

Authors and publishers need to include in their material means of distinguishing the types of information, and descriptive clips. They should declare the models they have of types of user, and protect their own interests by packaging the material with access controls.

The learning organisation needs to assess the value of generalised material, and to be prepared to purchase advice from external mentors before commissioning bespoke material.

Delivery mechanisms are already effective, but need more sophistication in synchronisation and assembly of modular material.

A complete glossary of terms and abbreviations is provided in the annex to the CTA General Overview volume (white book) of the CTA Handbook [CTA, 1994]. It is also planned that the CTA Handbook Series will be included in a later edition of the ARCHITEXT CD-ROM.

## References

ANSA, "ANSA The Architecture. Open Distributed Processing Phase Three 1992-1995". ANSA Work-Programme Booklet, APM Ltd., Cambridge, 22nd June 1991.

ARCHITEXT CD-ROM, The OPEN*framework* Systems Architecture series of books in hypertext format, Published by ICL's Multimedia Solutions, Enterprise Technology Division.

CTA, "The CTA Handbook", CTA Overview. ISBN 90-358-1370-7, 1994.

CTA, "The CTA Handbook, Common Information Space", ISBN 90-358-1380-4, 1994.

DELTA, "Telematics for Flexible and Distance Learning," CD-ROM DELTA from CEC DGXIII C3, Brussels, 1994.

EAST, "Function Specification of Support Tools, Project D2016 East, CEC Brussels, Deliverable D05, 24th February 1993.

ECMA, Technical Report TR/55. "Reference Model for Frameworks of Software Engineering Environments" 1990. Also available from the US Department of Commerce, National Institute of Standards and Technology as NIST special publication 500-201.

EFQM, "Total Quality Management. The European Model for Self-Appraisal", ISBN 90-5236-035-9, 1992.

HUTT, A.T.F., "Preparing the Organisation, Issue 1", November 1993.

HUTT, A.T.F., "Preparing the Organisation Workbook, Draft 0.1", OPEN*framework* Series, February 1994.

ICLIIP, The standard ICL manuals and processes used for the Investing in People practices throughout the Group.

ICLSQM, Statements of Direction: "Quality The ICL Way". Reference GQP 207, 1993-1997.

ISO9001, "British Standard EN ISO 9001: 94" published as international standard, ISBN 0-580-23439-8, 1994 (appropriate to ICL work and for which ICL as a Group is certified. Peritas is a BSI registered firm under reference FS20944).

PERITAS, "The Office Professional", a CD-ROM from the ChangeMaster Series, Peritas Multimedia in association with Echelon Training, 1994.

PERITAS, The Peritas Self Development Centre, Beaumont, Windsor, opened January 1995.

UKIIP, "The UK government certification scheme for Investors in People", 1994.

## Further Reading

The OPEN*framework* Systems Architecture Series, including the OPENWay Toolset.

Newsletters of the British Association for Open Learning.

Newsletters of the European Association of Distance Teaching Universities.

The COMETT European Community action programme for education and training for technology. 1987 through 1995.

Concerted Action on Learning and Pedagogic Research, The Tavistock Institute of Human Relations. 1993 through 1994.

TRIBUNE Bulletin from the CEC DELTA Project Tribune, 1992 through 1994.

MAYO, A.J. and LANK, E., "The Power of Learning". The Institute of Personnel and Development Management, ISBN 0-85292-565-4, 1994.

## Biographies

*John R. Collins*

John Collins has worked in the computer industry since 1964. For the last 15 years he has managed the transfer of technology from research and development to production of state of the art systems. He was responsible for the delivery of Ada and Systems Engineering life-cycle development architectures, strategies, methods and tools. He was an active member of the ICL engineering improvement programme during its formative years and led the unit which introduced an integrated object-oriented analysis and design method and tool into ICL. He has most recently managed the business unit responsible for multimedia software tools in ICL and gained accreditation to ISO9001 on a number of occasions.

Currently he is responsible for the transfer of flexible and distance learning technology working on EEC programmes.

*John M Pratt*

John Pratt has been in the computer industry since 1961, specialising in the design of control and communication systems. For the past four years he has worked as a Principal Consultant and was primarily responsible as the Design Authority for a Systems Engineering Department. Prior to that he worked at the European Computer Industry Research Centre, Munich, for three years, as the Man Machine Interface Group Leader.

Currently he is Principal Architect, Peritas Multimedia Systems responsible for the technical integrity of training support systems.

# The Software Paradigm[*]

**Brian Warboys**

**Senior ICL Fellow, Department of Computer Science, University of Manchester, M13 9PL**

**Abstract**

The nature of modern computer systems is such that a single paradigm is an insufficient model. This paper attempts to show that what is required is for projects to create a design framework which tolerates change (and failure) in the design process. Within this *framework* the appropriate paradigm should be selected for appropriate design components.

## 1. Introduction

Two case studies will be used throughout to illustrate and illuminate the issues. They are arbitrarily selected from numerous possibilities. They happened to be on my desk when I wrote this paper.

The first is based on the report (Computer, 1993) on the series of fatal accidents which occurred between June 1985 and January 1987 from fatal doses of radiation from the Therac-25 computerised radio therapy machine. References to this case study are labelled '**Therac.n.**'

The second study is based on the report of the inquiry into the chaos caused in the London Ambulance Service when the new Computer Aided Despatch System (CAD) went operational on the night of 26 October, 1992. References to this case study are labelled '**LAS.n**'.

My point in using these case studies is not to resort to "scare mongering" but to illustrate the difficulties with practical examples. Software Engineering is not an academic exercise. The detailed attention to semantics cannot be ignored. If it is, it can have disastrous consequences for practical systems.

---

[*] A paper prepared for the MOD DSAC seminar on: 'Practical Limits of Automation in CIS', London, November, 1994

William Blake (1794) wrote: "He who would do good to another must do it in minute particulars General Good is the plea of the scoundrel, hypocrite and flatterer. For Art and Science cannot exist but in minutely organised particulars".

> LAS.1 .. *on 4 November 1992 the system did fail. This was caused by a minor programming error that caused the system to 'crash'. The automatic change over to the back up system had not been adequately tested, thus the whole system was brought down.*

> Therac. 1 *In the code, the Class3 variable is incremented by one in each pass through Set-UP Test. Since the Class3 variable is 1 byte, it can only contain a maximum value of 255 decimal. Thus, on every 256th pass through ... the variable overflows and has a zero value. The overexposure occurred when the operator hit the 'set' button at the precise moment that Class3 rolled over to zero. ... described the technical 'fix' implemented for this software flaw as simple.*

## 2. On the nature of Software

Software is "soft"; it is generic, formal and extremely malleable. Its application as a *process* tool has social, psychological and management effects. We are effectively moving away from software programmes as calculators towards viewing software systems as *partners.* This is the basic meaning of the expression "Programming in the Large" introduced to describe the practice of building large systems containing many bought-in components. We might better describe modern approaches where the emphasis is on "programming the business" as "Programming in the Huge".

However the conventional issues with software still remain. It is thus, at a technology level, a precise formulation, whilst being, at a human system level, the means of producing a very imprecise guidance tool. Real world performance issues are an obvious example:

> LAS.2 ... *the computer system itself did not fail in a technical sense. Response times did on occasions become unacceptable, but overall the system did what it had been designed to do.*

> Therac.2 *They determined that data-entry speed during editing was the key factor in producing the error condition: If the prescription data was edited at a fast pace (as is natural for someone who has repeated the procedure a large number of times), the overdose occurred.*

The conflicts between its implementation needs and its ultimate system usage represent one of the difficulties. Another is the challenge of controlling the conflict between software, being on the one hand malleable and one the other manageable as a development activity.

An astonishing fact is that, in spite of decades of evidence to the contrary, "management" and software designers are still eternally optimistic about software development.

> LAS.3 *The Chief Executive's report also states 'there is no evidence to suggest that the full system software, when commissioned, will not prove reliable'.*
>
> Therac.3 *Given the complex nature of this software and the basic multitasking design, it is difficult to understand how any part of the code could be labelled 'straightforward' or how confidence could be achieved that 'no execution paths' exist for particular types of software behaviour.*

This situation is exaggerated by the trend towards end-user programming of software applications. Here the "programmer" is thinking much more about programming the *business* rather than programming the machine and, unfortunately, our research over the last 25 years has tended to focus on the problems of the latter.

Djikstra asked, as long ago as 1973, at the IBM Newcastle Conference on the Teaching of Computer Science [1], "Is Computer Science nearing its completion? Is computing practice settling down in a way beyond recovery? Or are, as a result of current circumstances, university professors tired and discouraged?".

Already signs are that the wider problem is perhaps too difficult to solve. So what does an exploration of "The Software Paradigm" reveal about the limits on the application of this most malleable of technologies?

## 3. Software Crisis or Software Affliction

Dasgupta in his book on "Design Theory and Computer Science" [2] introduces the notion that we are suffering not from a crisis but from an affliction. Software Engineers have been talking about a "Software Crisis" virtually since the term "Software Engineering" was coined at the Nato Conferences of 1968/1969 [3]. Crisis is defined as a "turning point", in particular the turning point of a disease when it becomes clear whether the patient will live or die. Yet we have had a crisis for nearly 30 years so the term is misplaced. What we really have is a chronic affliction, that is something which lasts for a very long time.

Some of the causes of this affliction lie with:

- the attention to detail that is required
- the conflicts between a malleable and manageable technology
- the evolutionary nature of the software design process
- the evolutionary nature of software itself
- the differing emphasis on the development of products versus the support for human processes.

## 4. On paradigms

A paradigm is usually defined as a pattern or a model. The term is associated closely with the work of Kuhn which led to the formulation of the notion of Kuhnian Paradigms [4]. He identified them as having two related aspects:

- A disciplinary matrix - essentially a network of beliefs, techniques and theorems. They have three main properties:
    i. symbolic generalisations: general formal assertions that are later taken for granted and employed without question (e.g. Ohm's Law)
    ii. model beliefs: a commitment to a belief in a model to which the relevant domain conforms, (e.g. Bohr-Rutherford model of the atom)
    iii. values: e.g. scientists believe that prediction should be quantitative rather than qualitative
- Exemplars: shared examples that illustrate the properties of the paradigm.

## 5. A superficial review of the most influential Software Paradigms

In the beginning there was:

### 5.1 The Algorithmic Paradigm (AP)

This is based on the notion that software design problems are well-structured. It is characterised as the execution of a domain-specific algorithm which, given a set of requirements, generates a design satisfying them in a finite number of steps. This is classical design automation. It views computing as an algorithmic style and is essentially language-based. Typical examples are mathematical models of airflows, CFD problems and language processors. Algorithms are problem solving systems that do not have explicit access to external knowledge. Rather the knowledge is contained in the algorithm itself. More precisely, control strategy and knowledge about the task domain are intertwined and together "define" the algorithm.

Significantly, the regularity of this paradigm does guarantee the avoidance of errors, only too prevalent, which arise when an 'ad-hoc' approach is taken to well-structured problem domains.

> LAS.4 ... *an example of such problems was the failure to identify every 53rd vehicle of the fleet.*

It was soon clear that the class of software which was producible this way was, although important, very limited.

So then:

## 5.2 The Analysis-Synthesis-Evaluation Paradigm (ASE)

Essentially, given a set of requirements, the software design process principally involves:

- analysis of requirements
- one or more stages of synthesis principally driven by decomposition
- evaluation

The best known example is the Waterfall Model of the Software Lifecycle. It derives from the desire to make software development "scientific"; essentially to define an engineering *Method*. At the Nato conference in 1968 [3] Ross observed that "The most deadly thing in software is the concept, which almost universally seems to be followed, that you are going to specify what you are going to do, and then do it. The projects that are called successful, have met their specifications. But these specifications were, in the main, based upon the designers ignorance before they started the job."

> LAS.5 *It should be noted that the SO quotation for the CAD development was only £35,000 - a clear indication that they had almost certainly underestimated the complexity of the requirement... the bid ... was some £700K cheaper....*

The Ross quote was the first time to my knowledge (note it was 1968 though!) that it had been suggested that software development was a learning process: that software evolves.

Therein lies the fallacy with this paradigm. Essentially Software Design:

- problems are incomplete
- requirements may be inconsistent
- acquires a "life of its own", second order requirements arising from the design process itself.

This leads us to the concept of "Bounded Rationality" expounded by Simon in 1976 [5]. In essence he pointed out that in such systems there are constraints on the cognitive and information-processing capabilities of the decision-making agent which means that the agent is not independent in a way which could possibly lead to a normal rational process.

> LAS.6 *It should be said that in an ideal world it would be difficult to fault the concept of the design. It was ambitious but, if it could be achieved, there is little doubt that major efficiency gains could be made.*

Thus the ASE paradigm, leading to the classical design approach of decomposition, is not widely applicable to software development. Decomposition essentially identifies possible choices of components considered independently. This produces unbounded problems as the components interact and generate second order requirements.

The ASE paradigm leads the designer to assume that his problem is well-structured. The characteristics of such systems are that they are empirical, that is in their solutions, possible transitions are observable. This tendency is exaggerated by modern quality procedures which emphasise the need for formal inspections at the end of each development phase resulting in a "freezing" of the outputs of that stage.

The problem with modern "guidance" systems as distinct from simple tools is that they are essentially non-deterministic. There are all manner of second-order effects, the most significant being that the use of these systems changes our behaviour and hence our requirements of the system.

This rapid and inevitable evolution means that the ASE paradigm is too rigid for all of our needs although useful for some aspects of the system.

## 5.3 The Formal Design Paradigm (FD)

Since decomposition was clearly limited, the alternative of abstraction with subsequent refinement was introduced. Abstraction provides us with a more flexible tool. Further, mathematics provides a tool for abstraction and further as Hoare reasoned in 1986 at his inaugural lecture at Oxford [6]:

i. computers are mathematical machines (behaviour is mathematically defined)

ii. programs are mathematical expressions (describe precisely what)

iii. a programming language is a mathematical theory (a formal system for programming)

iv. programming is a mathematical activity.

Thus software design becomes a mathematical proposition or theory that solves the problem as represented by the specification of the requirements. Hence, in the FD paradigm, the design process is an activity which exploits the traditional methods of mathematical reasoning. It has been frequently justified by quoting Djikstra's remark that: "Testing can show the presence of errors but never their absence".

However, like the ASE paradigm it assumes that requirements are known and that essentially the problem is well-structured. It is in essence a refinement of the ASE paradigm.

As applications of software became more strategic so grew the desire for decision-support systems with more "intelligence".

Thus we arrive at:

## 5.4 The Artificial Intelligence Paradigm (AI)

This derives from the view that software design problems are not well-structured. Thus, rather than define a rigid set of requirements, the system should absorb domain-specific knowledge to an extent that it can provide a set of possible answers which satisfy the constraints implied by that knowledge. The design process involves:

- a symbolic representation of the problem (the problem space) structured in terms of:
  - i. initial problem state
  - ii. goal or desired problem state
  - iii. all other states being reached or considered in attempting to reach the goal state from the initial state
- transitions from one state to another being affected by applying one of a finite set of operators
- the result of applying operators being, in effect, a search for the solution through the problem space.

The paradigm is explicitly founded on the concepts of *search*, *knowledge* and *heuristics*, in that the search is determined by knowledge of the problem domain and by a collation of general heuristics. The rules are partly domain specific and partly domain independent. The more the problem solving system relies on domain specific knowledge the "stronger" is the problem solving method itself. "Expert Systems" are instances of such "strong" methods. In the case of the Algorithmic Paradigm, the algorithmic design styles correspond to the domain independent heuristics of AI. However, whenever domain specific knowledge does occur in algorithms, it tends to be of greater scope and granularity than that encoded in the rule-based type of knowledge seen in AI systems. In such situations the algorithmic design system converges to a solution with virtually no search of the problem space.

Again an interesting class of problems can be addressed in this style but the rule-base tends to become unstructured and in very large systems difficult to digest. In addition, many components are dealing with well-structured problems and can easily be developed through algorithmic approaches.

This leads us to the obvious conclusion that systems are hybrid in nature and that the appropriate paradigm is one which recognises this fact. Thus:

## 5.5 The Theory of Evolutionary Design Paradigm (TED)

This is also referred to by Dasgupta [2] as "The Theory of Plausible Design" but I prefer the emphasis on evolution. This takes as its starting point that the software design process is evolutionary. The requirement is to create a design framework which accommodates change (and failure) and utilises one of the previous paradigms at the appropriate places. Basically it is a collection of tentative hypotheses such that:

- one can attempt to provide evidence in their favour to establish the "plausibility" of the design
- belief in the design's plausibility may have to be revised

TED is based on the view of design as an "empirical scientific" activity in stark contrast to the Formal Paradigm which views design as a "mathematical modeling" activity. Essentially TED maintains that design in mixed paradigm systems can consist only of establishing constraints on the implementation. Collectively these constraints represent the design.

The approach means, in practice, that designers are forced to resort to satisfactory rather than optimal designs, thus recognising both the evolutionary and error-prone nature of software development. It is worth noting that, even given that there will be elements which clearly can be formulated as well-structured problems, their optimal solutions may turn out to be intractable. It is interesting to observe that in both our case studies "perfection" was required and, being unachievable, led to problems which were major contributing factors.

> **LAS.7** *However its success would depend on the near 100 percent accuracy and reliability of the technology in its totality. Anything less could result in serious disruption to LAS operations.*
>
> **Therac.4** *A common mistake in engineering, in this case and many others, is to put too much confidence in software.*

Under such conditions it is fruitful to view the act of software design as an evolutionary process and the design itself, at any stage of its development, as a tentative solution to the problem posed. The adequacy of the design is solely determined according to whether it meets the requirements prevailing at that stage of the design process. Thus the design is not only tentative at intermediate stages of its development but also when the designers (or their managers!) see fit to terminate or freeze the design.

In other words, according to the evolutionary model, a design at any stage of its development (including the final stage) is:

- an evolutionary offspring of an earlier design form
- likely to evolve further in the future

Thus software design problem solving is a special case of the process of scientific discovery and suggests that we recognise in our design management systems:

- the strong link between natural and artificial sciences
- the use of testable hypotheses as a method (use of prototyping and simulation)
- the nature of evolutionary systems and hence the need for an incremental development approach (a working product (or subset) which evolves on a daily basis).

Any description of a modern IT supported business system shows the requirement for an approach which has a complex and multi-disciplinary nature. They are systems comprising organisational structure, doctrines, procedures, rules, personnel, information, communications, software, equipment, etc., etc.

The scope of such systems clearly indicates the 'socio-technical' nature of modern IT systems and the hybrid nature of the scientific basis which underpins it.

In building such software systems we are attempting to apply the style of the software paradigm(s) to real world problems, but recognising at the same time that the problem cannot be transformed into a basic calculation problem.

## 6. Conclusions

The advantages and shortfalls of the various paradigms outlined above are, I hope, evident. They all contain useful notions and, as the various quotations from the case studies clearly illustrate, they should not be ignored.

However, the TED paradigm suggests that we need to construct a design framework for modern systems which allows for the embedding of the appropriate paradigm to meet a specific design component requirement. This framework must also, from the beginning, cope with the problem of the evolution of the system. As systems move from a task-oriented "do this" approach to a process-oriented "achieve this" approach, so the need for smooth evolution will grow. As more of the human process is codified so the need will grow for the process to change as it is used.

Further, in practice, the problem is how to apply the disciplines of systems engineering at a real world level and how to include the behaviour of the human user in the system design.

At this level the system needs to be evaluated against three criteria:

i. organisational adoptability (how easy is it for the organisation to adopt)

> LAS.8 ... *the inability of the system to cope easily with certain established working practices (e.g. the taking of a vehicle different from the one allocated by the system).*

> **LAS.9** ... *the impact of CAD upon the existing communications infrastructure was never properly and systematically understood.*

ii. community adoptability (how much does the benefit depend on everybody else adopting it)

> **LAS.10** *It was recognised that a system such as this would be a first' ... (other similar ambulance systems were rejected - my paraphrasing) ... there is no confidence in the system.*

iii. agent of change (how likely is it to facilitate some new approach or make an existing one obsolete)

> **LAS.11** *Management were misguided or naive in believing that computer systems in themselves could bring about such changes in human practices.*

LAS. 12 *The changes to CAC operation ... made it extremely difficult for staff to intervene and correct the system.*

Such questions are a clear indication of the hybrid nature of the design task. In such an environment "design by constraint and evolution" is a very attractive option.

## References

[1] DIJKSTRA, E.W. "Selected Writings on Computing: A Personal Perspective", Springer-Verlag, 1982.

[2] DASGUPTA, S. "Design Theory and Computer Science", Cambridge University Press, 1991.

[3] NAUR, P. and RANDELL, B. "Software Engineering: Report on 1968 NATO Conference", Nato, 1969.

[4] KUHN, T.S.

# Single Sign-on Systems

**Tom Parker**

**ICL Enterprises, Winnersh, Berkshire, UK**

Abstract

The term *single sign-on* means what it says: it allows a user of a distributed system, who may potentially be using a variety of different application services spread over different end-systems, to sign on, i.e. authenticate himself to the distributed system as a whole only once. The results of that authentication are then propagated automatically to the different end-systems as required. In reality of course this simple concept conceals a whole gamut of complications and security implications. This paper aims to help the reader to understand: the topic: the potential benefits to, and requirements for, usability and manageability, and the threats to, and consequent requirements for, security. It also summarises what is happening in this new, leading-edge technology and where it is going.

## 1. Introduction

More and more of today's systems are distributed. Typically, they consist of multiple individual computer end-systems servicing the requirements of multiple users, each user sitting at an intelligent workstation and requiring access to applications, possibly distributed ones, on a number of different end-systems.

The workstations and end-systems are all connected by communications links, potentially extending over large distances and passing through environments outside the control of the systems administrators. Users do not necessarily always log in from their own office *home* workstation. There is an increasingly mobile user population, accessing distributed systems while off-site, even to the extent of signing on and obtaining secure access using a laptop in a hotel room. It is these and similar systems that form the natural home for single sign-on products.

If we are to provide security for such systems there are a number of general threats that we need to defend against. The main areas of concern covered in this paper are masquerade, and attempts to obtain access privileges beyond those granted.

An attacker may use maliciously introduced software in user workstations or servers, or may actively or passively wiretap communications links.

When a user uses the distributed system, we need to be able to manage that user's access rights, and accountability for use of those rights, in a way which is co-ordinated across all of the system's components. We cannot simply cobble together the different security features of the individual components from which the system has been constructed; few of these will have been designed in the context of distributed working, and they will often handle the same requirement in different ways with different user and management images. What is needed are products that look at security from a whole system point of view. Single sign-on products are a class of products that do just this. Single sign-on is a security feature that has only relatively recently become available in real products, but which adds substantially to the quality and usability of security across distributed systems.

## 2. How Does it Work?

This section outlines the different ways in which single sign-on products have been designed to do their job. The advantages and disadvantages of the different approaches are also outlined.

### 2.1 Scripting

Scripting is a technique whereby individual target end-system sign-on protocols, normally conducted by the user, are conducted automatically by the single sign-on product instead. It does this by interpreting a program (the script) which simulates the user depressing keyboard keys, and reacting to individual end-system sign-on prompts. The product itself holds and manages the different sets of authentication information required by the end-systems. It then extracts this information from its data base and inserts it in the data stream simulated to be from the user, at the appropriate points. If needed, the script can be programmed to prompt the user to enter data at chosen points in the script.

Scripts can be pre-produced by the vendor for specific known end-system sign-on protocols, or can be left to the customer to write. The target is unaware that the user is not physically entering all or any of the data via a keyboard, but because all of this can happen automatically the user still retains the single sign-on image. The user no longer needs to know the authentication information required to conduct these individual sign-ons, indeed it might no longer be in a form that can be entered on a keyboard; the user just knows the authentication information needed to sign on to the single sign-on product itself.

## 2.2 Advantages and disadvantages

Scripting can be viewed as a simulation of single sign-on by hiding the actual execution of real multiple sign-ons from the user. Each scripted sign-on is only as secure as the old style of sign-on method, and none of the other potential benefits of the functional extensions described later in this paper can be made available to these end-systems.

However, scripting is useful as a usability improvement, and as a transition aid to a proper more secure solution. It does make possible a relatively painless gradual move from multiple individual end-system sign-on to fully functional single sign-on, and enables alien proprietary end-systems of different types to be brought quickly under the single sign-on umbrella. With scripting there is no need to modify target end-system code, and the approach is a simple one. If the script interpreter is sufficiently general, scripting can also provide a non-security benefit by continuing to take the user inside the application to a point tailored to that user's specific needs.

## 2.3 Authentication and Access Tickets

The user authenticates to a remote authentication server (a component of the single sign-on product) and receives a data token or certificate in return. This can subsequently be used to prove the user's authenticity (we shall refer to all of these forms of returned protected data structures simply as *tickets*, though their content and method of protection vary widely from product to product).

The user selects a target application server to access, and the single sign-on product in the workstation uses the above authentication ticket to obtain an access ticket from a remote security server suitable for use in accessing the target. (This is again part of the single sign-on product) In some cases different access tickets may be obtained for different target servers, in others the same access ticket can be used at multiple targets. In some products the original ticket is directly used at a target, the middle step being missed out.

Finally, special code in each selected target receives the access ticket, validates it and establishes an authorised connection.

All of the tickets are protected cryptographically against theft, manipulation and misuse. The access ticket can be accompanied by a second ticket containing cryptographic keying information to be used to establish a key between the user's workstation and the target. This information can also be part of the ticket itself. The key is used to protect operational and security data exchanges.

Different products use different terminology for these components and tickets.

## 2.4 Advantages and Disadvantages of using tickets

This approach, including the use of the intermediate ticketing step, is the basis on which all genuine fully functional single sign-on solutions are constructed. It is the approach taken in the international standards world, both formal and de facto, and is the one that is likely to have the most enduring long-term future in an Open Systems context. The sign-on logic in individual end-systems is replaced by a single sign-on component which validates tickets presented to it, and security depends on the technology under which the tickets are protected and controlled. When cryptographic techniques have been used, the protection is, in practice, easily strong enough to satisfy most commercial and unclassified government needs, and the use of such tickets opens up the functional extension possibilities described in section 4. Some of these extensions can greatly enhance the overall security of the system.

A disadvantage of ticket-based implementations is that individual end-systems need to be converted to accept tickets before they can be brought into the single-sign on regime. Different end-system types need different code installed, and communications protocols need to be modified to carry additional cryptographic and ticketing information. This means that in a heterogeneous system it is possible that, in the short term at least, it will not be possible to include all end-systems under a ticket-based single sign-on regime. The usability and adaptability of products based on this approach will depend on their strategy for solving this problem.

### 2.5 Delegation

A server receiving an access ticket may be permitted to use the ticket itself with respect to other servers, acting as the user's delegate or proxy. Whether it can do this depends on the control values placed in the ticket.

## 3. Extended Single Sign-on Functionality

When single sign-on is supported, the presence of a separate security service in which to hold and manage a user's authentication information provides an opportunity to hold other useful user-related security information. This section describes how, by supporting the management and transmission of this information, single sign-on products can provide valuable additional security functions. It should be noted that the scripting approach described above cannot be used to support any of the extended functions described here.

### 3.1 Initiator and Target Access Control Information (ACI)

Access control decisions are made on the basis of:

- security attributes associated with the user (known in ISO terminology as Initiator ACI [ISO 10181-3]),
- security attributes associated with the target object being accessed (Target ACI),
- the kind of access being requested (for example, read or write)
- the context within which the access is being requested (for example, the time of day).

In many situations the Initiator ACI can be simply the identity of the user, and the corresponding Target ACI a conventional Access Control List (ACL), specifying for each incoming user identity the actions available to that user. But there are other situations where an Initiator ACI reflects the user's group memberships, position in an organisation or security clearance more appropriately. For groups or jobs, the corresponding Target ACI might be ACLs containing actions permitted by users as members of those groups or in those jobs. In the case of security clearance, the corresponding Target ACI would be the security classification of the object to be accessed.

What has this to do with single sign-on?

When there is only one central computer, all of the ACI can be held by its operating system – everything is in the same place. The user signs on, and the resulting authenticated identity is used in the central computer to obtain the ACI that applies to that user.

In distributed systems supporting single sign-on, it is possible to store the Initiator ACI at a security server accessed during the sign-on process, and have it presented to the target system when the user accesses it. This can have significant management and usability benefits.

## 3.2 Role-Based Access Control (RBAC).

Many business applications would like to control access on the basis of the job a person is doing in the organisation while at the same time preserving individual accountability. RBAC provides a solution to this. In RBAC schemes the user's identity need play no part in actual access control decisions (though individual accountability would still be required, below and section 3.3).

RBAC uses an Initiator ACI attribute to represent a user's role in an organisation. This *role* attribute can be thought of as a special case of group membership attribute, and it is directly used in system access control decisions. When users change jobs, once the administrator has updated their roles, the next time they access the system they automatically get the privileges associated with the new job, and no management action is required in individual server end-systems. Similar considerations apply when a new employee joins the organisation in a particular job.

Sometimes role is linked to organisational affiliation. For example, a pay clerk in department 1 may be able to perform the same functions as a pay clerk in department 2, but on different data.

Some general papers on roles are [ROLES 1] and [ROLES 2].

## 3.3 Accountability

We can extend the idea of carrying Initiator ACI to the end-system, to allow the end-system to be presented with other user-related information such as the audit policy that should apply to this user's use of the end-system, or a human understandable *Audit Identity* under which the user is to be made accountable, separate from the identity used for access control purposes. The idea is that the target end-system simply puts the received audit identity in its audit trail records without needing to understand it.

This feature is useful for a number of reasons:

- it solves the common problem that end-system user names are not really individual user names but rather are names used by a number of users. Names like *Root* and *System* spring to mind as being the worst examples of the type. Nevertheless such a name (or its system representation) is the only identity recognised for access control purposes and is often the one used in audit trails. If a separate and individual identity is used for audit purposes, individual accountability is brought in from the cold.
- it enables the system to support a policy under which a user can be authorised to act on behalf of another user (say while that user is away on leave) but will remain accountable via his or her own Audit Identity.
- it enables a user to be known by different identities, for access control purposes, while maintaining a single identity for accountability reasons. Different access identities often arise in real systems for historical reasons.
- it enables a user's access to an end-system to be controlled on a basis other than individual identity, for example, simply as a function of job role or other group membership, but still retaining individual accountability. The user need not be known as an individual in the end-system, simplifying the management of access control in that system.

## 4. Cryptographic Protection of Single Sign-on Information

In section 2 implementations were described that performed single sign-on by issuing tickets which were subsequently used as evidence of authenticity. Options built on the use of tickets were described in section 3. The security of these implementations and their enhancement options depends heavily on the protection given to the tickets and to the operations authorised by them. Again, it should be noted that scripting does not lend itself to the forms of protection discussed here.

In a distributed system, in the absence of physical security, communications links need to be protected by means of cryptographic techniques. These can be used for preserving both the integrity and confidentiality of data.

When single sign-on is supported, cryptography is also needed to protect security data while it actually passes through individual computer systems. In particular the single sign-on security server needs to protect its guarantee of the user's identity (and possibly other security attributes) during its journey to a target end-system. For example, if that journey causes it to pass through the user's workstation (in most implementations this is what happens) the user may try to tamper with it. All attempts to steal, modify or otherwise misuse tickets must also be thwarted. In particular, if the communications link between the user and the end-system is susceptible to attack, it is important to make sure that any accesses made by that user to the end-system are securely linked to the sign-on.

Cryptography can also be used to help users ensure that they really are communicating with the genuine application server.

So, whereas in the past cryptography was put into a distributed system mainly when the customer needed it to protect user data, now, with the advent of single sign-on, vendors are needing to install cryptographic features to satisfy the security needs of their own products. Cryptography is needed for system functions whether the customer requires it for user data or not, and it needs to be strong.

In most countries of the world, government restrictions are applied to the use of cryptography, though in general, governments of different countries are less concerned about using cryptography for integrity purposes than they are about confidentiality. So, single sign-on products that minimise the need for confidentiality have a wider potential market, provided that they can demonstrate that the cryptographic features they provide cannot be used for other than their intended purpose.

## 5. Open Systems and Single Sign-on

Single sign-on has been the subject of a great deal of activity in the international standards world, both in the context of de jure standards from organisations like the European Computer Manufacturers' Association (ECMA) and the International Standards Organisation (ISO), and de facto standards like the Open Software Foundation (OSF) and the Internet Engineering Task Force (IETF).

The de jure standards have tended to look at single sign-on within a larger context of a distributed security architecture. They consider this to be necessary because of the need to integrate authentication, access control and cryptography in order to provide proper distributed system security. In contrast, the de facto standards have grown from an implementation satisfying a specific single sign-on requirement and have been extended to cater for the needs of a wider community. This difference in approach is by no means unusual, indeed it is entirely characteristic!

### 5.1 The Generic Security Services Application Program Interface (GSS-API)

Before discussing individual Open Systems developments in terms of specific ways of supporting single sign-on, it is useful to look at how this functionality is activated. The single most important Open Systems development within this context is the GSS-API, a de facto standard interface through which single sign-on and other distributed security services can be accessed in a way which hides from the API user all details of the mechanism by which these functions have been implemented. The GSS-API is widely accepted. It has been taken up by OSF, IETF, ISO, Project SESAME (described below), X/OPEN, POSIX and a number of individual vendors. The basic GSS-API is defined in [GSS-API 1] and the security attribute and delegation extensions in [GSS-API 2].

## 5.2 ECMA and Project SESAME

Most of the de jure Open Systems standards work in this area has been performed in ECMA. It has drawn on, and contributed to, the generic ISO Security Frameworks developed in [ISO 10181].

An ECMA Technical Report, [ECMA-TR46, 1988], concentrates on the application layer and describes a security framework in terms of application functions necessary to build secure open systems. Despite its age, this report is still valid and useful, providing good background to the single sign-on requirement. The continuation of this report, [ECMA-138], defines the abstract security services for use in a distributed system, including single sign-on services providing authentication and access control functionality. This Standard is now largely superseded in detail by more current work (see next paragraph), though the principles and overall approach still reflect current thinking. The concept of the Priviledge Attribute Certificate (PAC) developed in this Standard has become accepted world-wide.

In December, 1994 ECMA completed standard, [ECMA-219], which defines the functionality and the protocols for a distributed security service providing single sign-on, distributing access rights to human and application users and providing the related cryptographic services needed to protect this information.

To counter the de jure standards take-up and practicality problems, project Secure European System for Applications in a Multivendor Environment (SESAME) was formed. SESAME is part funded by the Commission of the European Communities (CEC) to exploit the work of ECMA. This work is now resulting in real security technology suitable for incorporation in a product. The SESAME partners in this work are Bull, ICL and Siemens Nixdorf Informationssysteme AG (SNI). SESAME's functionality is outlined in section 7.

## 5.3 The International Standards Organisation (ISO)

The International Standards Organisation is divided into Security Committees (SCs) covering general areas of work, within which working groups (WGs) concentrate on specific topics. There are two main ISO groups working in areas which impact single sign-on products:

- In ISO/IEC SC27/WG1 there is a sub-group dealing with *Security Information Objects*, attempting to standardise the syntax and semantics associated with authentication tickets, access control tickets and security attributes among others.

- In ISO/IEC SC21/WG8 there is a recently started work item known as Security Association Management and Support (SAMS), in which a generic single sign-on architecture is being developed, along with specific solution mechanisms. It is adopting the GSS-API and the ECMA work, along with other specific technology solutions. See [ISO, 1994].

### 5.4 Kerberos and OSF DCE

In 1986, the Massachusetts Institute of Technology (MIT) developed a set of single sign-on components called Kerberos (named after the three headed dog which in Greek mythology guards the entrance to the underworld). Kerberos was designed and built for a specific environment – a distributed system of UNIX machines on the MIT university campus. The implementation project was called Project Athena.

Kerberos was the earliest implementation of a secure single sign-on capability in which full cryptographic protection of both security exchanges and operational user exchanges was provided. The practicality, uniqueness and availability of Kerberos at that time caused it to be taken up by the Open Software Foundation (OSF) as the basis for the security technology of their Distributed Computing Environment (DCE), supported and developed by vendor members of OSF.

### 5.5 Other Activity

X/OPEN, POSIX and the Object Management Group (OMG) have working groups on distributed system security, including the support of single sign-on. This work follows the philosophy of the activities already described above, increasing the scope of its acceptance.

## 6. Available Solutions: Products, Prototypes and Vendors' Plans

This section contains outline descriptions of a selection of some products available on the market today. It does not claim to be complete, but gives a flavour of the kind of technology available. It also describes the functions provided in the main single sign-on *construction kits* of SESAME and Kerberos. The products/kits are listed in alphabetical order.

Only the main features are highlighted in the narrative descriptions below. The features described constitute the author's best understanding of these products, but the author assumes no responsibility for the accuracy of the information in this section.

### 6.1 Access Manager from ICL

Access Manager is a single sign-on and security management product supporting role-based access control which provides secure scripted single sign-on to a variety of target platform types. In 1995 these functions will be supplemented by the addition of SESAME technology in a way that enables users to bridge gradually from piecemeal legacy security to full networked security. Access Manager integrates with PC security products to provide local workstation security, and with other network security products such as Netware 4 (q.v.).

Interworking with DCE security is possible, preserving the single sign-on image. Later integration of AM and DCE security management is planned.

## 6.2 Bespoke Solutions from CKS

C&K Software Ltd has a range of security products which can be combined in a variety of ways to provide specific packaged single sign-on solutions orientated around scripting to IBM systems.

## 6.3 Control SA from 4th Dimension Software

This is part of a stratified suite of products that automate IT operations. It supports a proprietary secure single sign-on technology based on Kerberos, but incorporating some public key technology. A proprietary "USA-API" is used by the 4D code to access resident security system code. For legacy systems, target host passwords are still used, and the product includes capabilities to synchronise password changes.

## 6.4 DECathena from DEC

DEC has developed a security superstructure for Kerberos V4, and turned this component into a product called DECathena. Kerberos V4 differs from V5 in that it does not support UNIX groups in the access tickets, and does not support any delegation.[1]

## 6.5 ISM from Bull

ISM Security Services is a Bull single sign-on product running on the Bull Integrated System Management Platform. It supports:

- consistent and coherent administration to control users and their associated access rights
- a centralised user authentication server (authenticating users by means of a password or a smart card)
- single sign-on to legacy applications via script or simple API
- workstation local security with audit capability
- a GSS-API library supporting the OSF DCE security mechanism to secure client/server applications (AIX and PC/DOS/Windows).

It is planned to add SESAME technology later.

## 6.6 Kerberos

Kerberos is a set of freely available components that run on UNIX platforms, see [Kohl and Neuman, 1993]. The version of Kerberos current in early 1995 is V5. The V5 implementation components provide the following single sign-on functions or features:

[1] For more on the differences between Kerberos V4 & V5 see section 6.6. Ed.

- a simple authentication service providing single sign-on
- a key distribution service integrated with the distribution of access tickets
- access tickets containing the user's identity and UNIX group membership, protected by DES encryption
- cryptographic protection of security exchanges by DES encryption
- optional integrity and confidentiality protection of user exchanges with end-systems using DES
- can be accessed through the GSS-API.

## 6.7 NetSP from IBM

NetSP is a product that uses an authentication scheme based on a cryptographic hash function. By confining the cryptographic capabilities of the system in this way, IBM has avoided problems of exportability, since only reversible cryptographic capabilities are subject to export or usage controls. The scheme is known as KryptoKnight and is described in [Molva et al, 1992]. NetSP supports the base GSS-API but does not offer any access control extensions. Version V1L2 also supports legacy single sign-on to LU6.2 RACF hosts and (for OS/2 only) to LANserver and Novell Netware servers.

## 6.8 Netware 4 from Novell

The latest version of Novell's network operating system has been enhanced to support single sign-on. It uses RSA public key cryptography to support single sign-on via a challenge-response protocol.

## 6.9 OSF DCE

Single sign-on is provided in OSF's Distributed Computing Environment via an enhanced version of Kerberos V5. The DCE security features are basically those described for Kerberos V5, with added access control functions implemented in target end-systems. Access to the security services is through remote procedure call, not GSS-API, though separate GSS-API support is about to be implemented for non RPC users.

## 6.10 SESAME

Version V3 of the SESAME Technology components provides the following single sign-on functions:

- Kerberos V5 authentication or strong authentication using public key technology
- an ECMA style Privilege Attribute Service supplying ECMA PACs containing Initiator ACI and user Audit Identities, signed using public key technology

- a Kerberos V5 Key Distribution Service that can be used independently of PAC acquisition and distribution, optionally supplemented by public key-based key distribution across security domains. The option of pure public key-based key distribution within or between security domains is also provided
- repluggable (by vendors) crypto-algorithms to cater for different functional and legal requirements
- can be accessed through the GSS-API.

SESAME V3 is freely available for non-commercial purposes, and can be used commercially under licence. For a description of SESAME V3 see [Parker and Pinkas, 1994].

Like Kerberos, SESAME is a component that must be ported into vendor environments as appropriate as part of the task of incorporating it into product.

### 6.11 Simple Public Key Mechanism (SPKM) from BNR

Bell Northern Research have proposed a strong but simple single sign-on technology to support authentication under the GSS-API [Adams, 1994]. It uses public key technology only, conforming to ISO/ITU-T Directory standards [ISO, 9594].

### 6.12 SSO Toolkit with PC-Guard from Airtech

The Single Sign-on Toolkit product is one of a family of security products from Airtech (Mergent in the USA). Specific features are as follows:

- It uses the PC-Guard product to perform the initial sign-on. PC-Guard supports local password-based sign-on from PCs and has APIs to support smart card and swipe card sign-ons. Airtech will provide bespoke routines for these functions on request.
- Having signed on to PC-Guard, SSO Toolkit uses scripting to automate sign-on to a variety of different end-system types. Scripts exist for Novell File servers, LAN Manager, Banyan Vines and DEC Pathworks.

### 6.13 Workgroup Manager from ICL

Workgroup Manager is part of ICL's integrated TeamWARE product line, which also includes the TeamOFFICE office system. Workgroup Manager provides single sign-on in the context of local area networks managed by LAN Manager or Windows NT. This includes support for UNIX servers under a LAN Manager for UNIX regime.

## 7. Multiple Single Sign-ons?

It can be seen that there are plenty of different single sign-on solutions around. So much so that there is the real danger that large distributed systems built from smaller legacy networks will find themselves supporting multiple single sign-on mechanisms. What was single sign-on now becomes multiple sign-on again! Also, different implementations of the same technology may support different profiles of that technology, and interworking between a client supporting one profile and a server supporting another may not be straightforward. There are two main (non-exclusive) approaches being taken to solve these problems:

- Invent a higher level sign-on co-ordination function, which takes one of the methods of sign-on as the prime method, and automatically handles the sign-on protocols to each of the underlying mechanisms. This is quite a difficult task, particularly since co-ordinated security management is also a requirement. Some work on this is now starting in OSF.
- Enhance existing mechanisms so that mechanism interworking is supported. For example, an initiator supporting SESAME technology will, in future, be able to send a Kerberos ticket to a Kerberos target or an SPKM ticket to an SPKM target. This requires the ability to find out which mechanisms are supported by each network component, and negotiate a mutually acceptable one. It also still leaves a requirement for overall co-ordinated security management. Enhancements to the GSS-API to include mechanism negotiation are under discussion in a number of security groups.

Watch out for new products, or extensions to the products identified above, that perform a sign-on co-ordination function and/or support mechanism negotiation.

## Acknowledgements

This paper is a revised version of that printed in Information Security Monitor for December 1994 published by Legal Studies Publishing Ltd. The material has been developed, and updated, from a more detailed report on the topic produced under the sponsorship of the CCTA, the Government Centre for Information Systems. The full report can be obtained from: The Library, CCTA, Rosebery Court, St. Andrew's Business Park, Norwich, NR7 0HS Telephone: +44.1603.704704.

## References

ADAMS, C; "The Simple Public Key GSS-API Mechanism", Internet Draft, draft-ietf-cat-spkmgss-01, October 1994.

ECMA, 1988; "Security in Open Systems – A Security Framework", ECMA Tech. Report, July 1988.

ECMA, 1989; "Security in Open Systems – Data Elements and Service Definitions", ECMA Std 138, December 1989.

ECMA, 1995; "Authentication and Privilege Attribute Security Application with Related Key Distribution Functions", ECMA Std. 219, January 1995.

FERRAIOLO, D & KUHN, R; "Role Based Access Controls", Proc. 15th Nat. Comp. Security Conference. October 1992.

GSS-API 1; "Generic Security Service API (GSS-API) Base", X/OPEN Preliminary Specification, Document Number P308.

GSS-API 2; "Generic Security Service API (GSS-API) Security Attribute and Delegation Extensions", X/OPEN Snapshot, Document. No. S307.

ISO 10181-3; "Information Technology – Security Frameworks in Open Systems – Part 3: Access Control." ISO/IEC Draft International Standard. 10181– Part 3.

ISO 10181; "Information Technology – Security Frameworks", ISO/IEC Draft International Standard, 10181, Parts 1-8.

ISO 9594; "Information Processing Systems – Open Systems Interconnection – The Directory", ISO/IEC Draft International Standard 9594-8 Part 8 Authentication Framework.

ISO 1993; "Security Association Management and Support", ISO/IEC JTC1/SC21/WG8, 3rd Working Draft of ULS Group, Orlando, December 1994.

KOHL, J & NEUMAN, C, "The Kerberos: Network Authentication Service V5", Internet RFC 1510, September 1993.

LAWRENCE, L.G.; "The Role of Roles", Computers & Security, Vol. 12, No.1, February 1993.

MOLVA, R, TSUDIK, G, VanHERREWEGHEN, E, & ZATTI, S; "KryptoKnight Authentication and Key Distribution System", ESORICS conference Toulouse, 1992.

PARKER, T.A. & PINKAS, D; "SESAME V3 – Overview" January 1995.

PARKER, T.A., "Standards for secure interfaces to distributed applications", *ICL Tech. J.* Vol 7 Issue 1 pp.141-153, May 1990.

## Biography

*Tom Parker*

Tom Parker graduated from Downing College Cambridge with a degree in Mathematics in 1963. He joined ICL in 1971 to work on early design of ICL's proprietary VME Operating System, and started to specialise in computer security in 1978. He is now a Principal Consultant on this topic within the company.

He was responsible for the inception and much of the design of the VME High Security Option, which is now certified by the British Government at the UK equivalent of B1 on the American DoD Security Evaluation Criteria scale.

He has been closely involved with security standardisation work in ECMA since 1986, and was editor of the current standard ECMA-219 on distributed system security. The ECMA work has since been adopted by the CEC as the basis of project SESAME, a development project from which product quality distributed system security components are being produced. He is the ICL SESAME architect, and led negotiations with the Open Software Foundation (OSF) which resulted in their acceptance of the SESAME architecture as a foundation for their future strategy for security in their Distributed Computing Environment (DCE). The SESAME technology is being built into ICL's Access Manager product, an activity with which he is closely involved.

The ECMA work is also in the process of being adopted by ISO. He is an active member of ISO, attending their SC21 security meetings and in the past chairing the SC27 working group on Security Information Objects.

He has published a number of technical and conference papers on security and has lectured in many countries world wide.

In 1991 he was made an ICL Fellow, the highest technical accolade the company can award. There are ten Fellows throughout the company.

# Why is it difficult producing safety-critical software?

**B. A. Wichmann**

**National Physical Laboratory, Teddington, Middlesex, TW11 0LW, UK**

**Abstract**

Where safety is concerned, quality must not only be attained, but must be *seen* to be provided. This is not easy with software, as objectively assessing it is either expensive or unconvincing. This article surveys the topic in a manner that should be of interest to those producing quality software, even if safety is not a concern.

## 1. Overall design strategy

As with all software, the design stage is crucial. In the case of safety-critical software, the key issue is the impact that the design has upon the level of criticality of the software. For instance, in the case of flight-critical avionics software, a single software fault should not make an aeroplane unsafe.

The systems architecture can reduce the impact of single software faults, thus giving greater confidence in the system's reliability. However, depending upon the application, it may not be possible to reduce the reliance upon software. Some examples from current practice are as follows:

- In the case of fly-by-wire with an unstable aircraft, computer control is essential in the military sector. Voting between several independent computers can help reduce the risk of a single error causing an accident.
- In the case of a kidney dialysis machine, one certification body advises that the functions of control and protection should be separated and implemented on different processor types. Such diversity again reduces the risk of a software error causing a problem.

- The Sizewell Primary Protection System is computer-controlled and is safety-critical since the secondary hardware-only system cannot cater for all the possible safety hazards. However, the existence of a secondary system means that the reliability required from the software is comparatively modest.
- Modern railway signalling systems are replacing a hardware interlock by a software-controlled interlock. Hence these systems are very critical, which is reflected in the specification of the standards to which they are produced.

One can see two trends, operating in different directions. Firstly, the demand for additional functionality which encourages the use of computers; secondly the concern about the difficulty of ensuring safety, which militates against the use of complex software.

Modern standards for safety-critical software classify the software according to the criticality, which is based upon a hazard analysis of the entire system. The increasing use of computers makes many systems *safety-related* rather than safety-critical. For instance, railway signalling is clearly safety-critical, but a passenger information system is still safety-related since information presented could result in a safety hazard if too many people attempted to reach an over-crowded platform.

Experience of the relevant application domain is clearly vital to providing an appropriate design backed with a good hazard analysis. The specification of the software, including reliability requirements, must be drawn up in such a manner that the application, system and software experts can all agree. A vital part of this process is a 'safety argument' to justify the use of the system. Here, the overall design needs to be analysed to ensure the overall system safety objectives are satisfied. This task can sometimes be assisted by a machine proof that a model of the entire system satisfies the logical requirements for safety. This has been undertaken with the Prover tool [Säflund, 1994] for nuclear, railway signalling and avionics applications.

One can see from the above that hazard analysis is a key step in establishing the context for the software and the potential consequences of failure. For details of this process, see [Henley, 1981], [Lewis, 1987]. In critical cases, senior management should approve a hazard analysis report prior to the approval of the software development process.

v *Have you performed a hazard analysis on your system?*

A difficult design choice can be deciding the degree to which an operator can override aspects of the computer system. Some operator actions can improve safety, but 'silly' operator mistakes should not jeopardise safety.

In the case of the London Ambulance Service incident, it seems clear that the safety-critical nature of the operation was not fully understood by management, which resulted in inadequate control procedures [London, 1993].

Many systems are critical, but not life-critical. The risk of an environmental disaster is a common concern with chemical plants, while with financial systems, large economic losses may be possible. In the case of one New York bank, an index error in a COBOL program caused the computer system to borrow many millions of dollars automatically from the Federal Reserve overnight. Apart from repairing the software, the staff had to explain the significant interest charges made!

The final hazard analysis may not be possible until some design decisions have been taken, which may well result in a less than optimal solution.

The desire to obtain benefits from computerisation may result in designs that are controversial. For instance, in the case of the A320 Airbus, five independent computers control vital functions; the problem here is that the computers work to a common specification, which could result in a common-mode failure. On the other hand, with much conventional engineering, an apparent safety factor of five would give all the confidence one could reasonably expect.

v *How confident are you of the system design?*

## 2. Is quality assurance impossible?

The complexity of software systems implies that many conventional quality assurance techniques are only partially effective. For instance, the draft IEC standard lists 69 methods to underpin software [IEC, 1991]. A careful choice of complementary methods is needed, but no one universal solution can be recommended. We consider some key issues below.

### 2.1 Statistical testing

The conceptually simplest method of determining the reliability of a computer system is by statistical testing, that is, testing the software with a statistically representative sample of its operational use.

There are two essential problems with this approach:

- For complex software, or software operating in a complex environment, it may not be feasible to produce a statistically representative sample.
- It has been shown that the amount of testing required to meet very high reliability targets for some safety applications may not be feasible [Littlewood, 1993].

Interestingly, the Canadian Nuclear Standard [Canada, 1990] does have a requirement for this form of testing — which is feasible owing to the nature of the application to nuclear protection systems. (The changes in the parameters monitored by sensors are restricted by the physics of the reactor. This implies that test cases can be generated to test the software based upon statistical variations within the physical constraints.)

## 2.2 Conventional testing

There is no doubt that conventional testing is effective in locating faults and in giving greater confidence in software. Unfortunately, the conventional (software engineering) testing standards are not sufficiently precise to provide the level of assurance that is often needed. The civil avionics standard DO-178B [RCTA, 1992], [Wichmann, 1994a] does define a suitable level for testing (depending upon the criticality of the software), and hence this standard can be recommended in this context.

## 2.3 Stress testing

One form of testing with which NPL has had some success is stress testing. Even if statistically representative data cannot be constructed, it may be possible to produce complex test cases by special means. NPL has produced a number of tools to stress-test compilers in ways that are complementary to the conventional black-box testing of validation suites [Davies, 1989], [Austin, 1991].

The difficulty here is the effort required to generate appropriate test cases. The NPL generators use a pseudo-random number generator to allow many options to be chosen at random, reducing the risk of an unusual combination being overlooked.

## 2.4 Static analysis

Performing a static analysis of the source code of software can reveal bugs which conventional testing would be unlikely to detect. Unfortunately, the strength of static analysis methods varies and quantifying the assurance benefits is not easy [Wichmann, 1995]. Some essential properties of a system can be data-dependent and therefore not capable of validation from the source text alone.

## 2.5 Accredited testing

Formalised testing by test laboratories is widely accepted in many industries. For software, ERA Technology Ltd has recently been accredited by NAMAS for testing software in the safety area. ERA's scope consists of five procedures, which involve both static and dynamic testing. Further details of this are to be found in [Wichmann, 1995].

The advantage of this approach is that the formalisation gives a high assurance to the testing process. If specific testing procedures could be agreed and made publicly available via high quality standards, then accredited testing could make a major contribution to the assurance process.

## 2.6 Formal Methods

The use of mathematically based methods caused some controversy with the publication of the Interim Defence Standard 00-55 [MoD, 1991]. Analysis tools that are based upon mathematical principles do not seem to cause difficulty, but the specification languages such as VDM-SL and Z do appear to be unacceptable to some.

Although 00-55 espouses Formal Methods, the civil avionics standard takes a very negative approach [RCTA, 1992]. Effective development does require an agreement between the experts with domain expertise as well as software engineers. Hence any use of Formal Methods must address the communication problem which undoubtedly exists with notations including unusual symbols.

The potential benefits from Formal Methods are significant, since the ambiguity in natural language is a major problem [Hill, 1972].

### 2.7 Language-based checks

A programming language can provide a means for expressing and checking the validity of some aspects of the design. The strong type checking in a language like Ada has clear advantages for assurance [Wichmann, 1994b]. Moreover, given a suitable basis like Ada, subsets can be defined to aid understanding and mechanical checking (including formal proof). The SPARK Ada subset seems to be particularly effective in the area of safety [Wichmann, 1994c].

All of the above techniques are useful, but none provides a complete answer. Practical, effective development must involve a judicious balance of techniques.

v *If the system failed, are you confident that you undertook all reasonable precautions in the software development?*

## 3. The Software Engineering context

The software engineer typically assumes a perfect environment — in spite of knowing that this is unobtainable in practice. For instance, errors in processor chips are negligible in practice and are therefore ignored, but this cannot be appropriate in the most critical applications. For some information and suggestions about this problem, see [Wichmann, 1993].

Similarly, errors in compilers are often discounted. The work undertaken to check for such faults for the Sizewell Primary Protection System indicates that compiler faults can be expected about once every 50,000 lines of source code [Pavey, 1993].

Similar concerns must be recorded against the use of operating systems, but the author has no reliability figures that can be used to indicate the magnitude of the problem.

The author would expect future systems to depend increasingly upon sophisticated validation and verification tools used to check vital properties. However, some of these tools are many megabytes in size and would be extremely difficult to validate. In consequence, errors in such tools must be taken into account in any hazard analysis. Schemes like that for compiler validation are needed to give increased confidence in such tools.

## 4. Quality Assurance standards

The ISO-9000 series of standards provides a standardised framework for the management of quality. Since the ISO-9001 standard [ISO, 1987] is generic for all services, the requirements made are very general and lack specific technical requirements needed to assure quality for software. The TickIT scheme guidance to the application of ISO-9001 to software development does not contain additional requirements [DTI, 1992], and hence cannot provide a high enough level of assurance for safety-critical software.

The requirement in this area is clear: a quality management system (which could follow the ISO 9000 model), and specific technical measures for which auditing requirements are specified. DO-178B specifies both in one standard, while the IEC draft standard [IEC, 1991] rests upon ISO 9000 and provides a catalogue of appropriate technical measures.

A different approach to quality has been formulated by the Software Engineering Institute in the USA, based upon process improvement [Paulk, 1991]. Again, the lack of specific technical requirements makes this approach ineffective for critical software.

v *What independent evidence have you that your software process is appropriate?*

## 5. Assessment

To determine if a safety-critical system involving software is 'fit for purpose' is very hard. Currently, we lack objective criteria. Accredited testing could provide a means of producing most of the technical data required for assessment, but at the conclusion of the process subjective judgement is still likely to be required.

Some regulatory procedures require that certain technical measures are taken without regard to their effectiveness or the criticality of the system. This is inappropriate since some procedures will be applied which are of little benefit (and may give a false sense of security). Other procedures will not be undertaken in spite of some good evidence of their effectiveness. Good engineering trade-offs are required.

In the case of the civil avionics standard DO-178B, objective assessment would seem reasonable and effective. However, no other internationally agreed standard appears to be adequate in this respect. This implies that assessment is very largely subjective, which in turn implies that establishing an effective single market in this area within Europe is fraught with difficulty (if not danger to the public).

v *Since Intel could make an error with something as well understood as division, how confident are you of your software?*

## 6. Conclusions

Conventional quality assurance is necessary but not sufficient for safety systems involving software. However, quality assurance via ISO 9000 is only now becoming widely accepted in the UK for software, and therefore common practice for software development is typically much less rigorous than is necessary for critical systems.

Effective standards are required to aid the assurance process, but these are either absent or lack support for many industrial sectors. In consequence, the following points should be considered for critical system development:

- The system design should involve hazard analysis in which the consequences of a software failure are quantified. Subsequent development should only be approved when senior management is satisfied that the risks are acceptable, thus demonstrating appropriate risk management as recommended by the Engineering Council [Eng, 1993].
- Careful consideration must be given to selecting a suitable set of technical measures to underpin the software process, as illustrated with the 7 methods considered in section 2 above.
- Even when regulations do not require it, independent assessment, testing or advice should be sought to increase the confidence in the system, its implementation, or its design.

The *italicised* items in this paper can be used as an informal check-list.

## References

AUSTIN, S.M., WILKINS, D.R., WICHMANN, B.A. — "An Ada Program Test Generator". TriAda Conference Proceedings. ACM. October, 1991.

CANADA, "Standard for Software Engineering of Safety-critical Software". AECL CANDU. 2251 Speakman Drive, Mississauga, Ontario, Canada, L5K 1B2. Revision 0, December, 1990.

DAVIES, M., WICHMANN, B.A.M "Experience with a compiler testing tool". NPL Report DITC 138/89. March, 1989.

DTI, "Guide to Software Quality Management System Construction and Certification using EN 29001". Department of Trade and Industry. ISBN 0-9519309-0-7. February 1992.

ENG, "Guidelines on Risk Issues". The Engineering Council. ISBN 0-9516611-7-5, February 1993.

HENLEY, E.J., KUMAMOTO "Reliability Engineering and Risk Assessment", Prentice Hall, 1981.

HILL, I.D., "Wouldn't it be nice if we could write computer programs in ordinary English — or would it?", The Computer Bulletin, Vol. 16, 1972.

IEC, IEC/SC65A/(Secretariat 122) "Software for computers in the application of industrial safety-related systems". Draft for comment, December 1991.

ISO, "Quality systems — Model for quality assurance in production and installation", ISO 9001:1987.

LEWIS, E.E., "Introduction to Reliability Engineering", John Wiley, 1987.

LITTLEWOOD, B., STRIGINI, L.,. "Validation of Ultra-High Dependability for Software-based Systems". *Comm ACM*. Vol. 36, No 11, pp.69-80.

LONDON, "Report of the Inquiry into the London Ambulance Service, 1993". Communications Directorate, South West Thames Regional Health Authority, 40 Eastbourne Terrace, London W2 3QR.

PAULK, M.C., CURTIS, B., CRISSIS, M.B., "Capability Maturity Model for Software". CMU/SEI-91-TR-91. August 1991.

PAVEY, D.J., WINSBORROW, L.A., "Demonstrating Equivalence of Source Code and PROM Contents". *Computer Journal*. Vol. 36, No 7. pp.654-667. 1993.

MoD, Interim Defence Standard 00-55, "The Procurement of Safety-critical Software in Defence Equipment", Ministry of Defence, (Part1: Requirements; Part2: Guidance). April 1991.

RCTA, "Software Considerations in Airborne Systems and Equipment Certification". Issued in the USA by the Requirements and Technical Concepts for Aviation (document RTCA SC167/DO-178B) and in Europe by the European Organization for Civil Aviation Electronics (EUROCAE document ED-12B), December 1992.

SÄFLUND, M., "Modelling and Formally Verifying Systems and Software in Industrial Applications". The Second International Conference on Reliability, Maintainability and Safety (ICRMS'94 - Available from NPL). 1994.

WICHMANN, B.A., "Microprocessor design faults". Microprocessors and Microsystems, Vol. 17, No 7, pp.399-401. 1993.

WICHMANN, B.A. [1994a]., "A Review of a Safety-Critical Software Standard". NPL. June 1994.

WICHMANN, B.A. [1994b]., "Producing Critical Systems — The Ada 9X Solution". Technology and Assessment of Safety-critical Systems. Edited by F Redmill and T Anderson. Springer-Verlag. 1994.

WICHMANN, B.A. [1994b]., "Strategy on the Use of SPARK". NPL Report 227/94. June 1994.

WICHMANN, B.A., CANNING, A.A., WINSBORROW, L.A., WARD, N.J., MARSH, D.W.R., "An Industrial Perspective on Static Analysis". Software Engineering Journal, pp.69-75, March 1995.

## Acknowledgements

The National Physical Laboratory runs the NAMAS accreditation service which has recently approved the first software test laboratory operating in the area of safety. In addition, NPL provides consultancy on a wide range of matters impacting on the area, including Formal Methods, security, Ada, testing and usability. The Software Engineering Group currently markets the tool Prover in the UK.

## Biography

*Brian Wichmann*

Brian Wichmann has worked at the National Physical Laboratory since 1964. In the past, he was responsible for the validation suite used to check Pascal compilers worldwide, and was a designer of the numerical features in the Ada programming language.

He is currently Chairman of the British Computer Society Safety Critical Systems Task Force, and is also a Visiting Professor at The Open University.

# Experiences using the Ingres Search Accelerator for a Large Property Management Database System

**Sarabjot S. Anand, David A. Bell and John G. Hughes**
University of Ulster, Northern Ireland

**Abstract**

The property management database system under development at the Northern Ireland Housing Executive (NIHE) is a large relational database system. The application system has a high expected transaction processing rate approximately 37,000 transactions per day (most of them accessing multiple tables) from about 250 on-line users. Performance is of critical importance to its success. In this paper we consider the effect of the Ingres Search Accelerator on the transaction processing efficiency of the system. The performance enhancement brought about by SCAFS (ICL's current version of the well-known Content Addressable File Store, CAFS [Babb, 1979, 1985], [Coularis et al., 1972] the heart of the Ingres Search Accelerator) is assessed for different file organisations. Recommendations on how the performance of SCAFS can be improved by tuning certain parameters are provided. We also provide a rough guideline as to when the Ingres Query Optimizer "decides" to use SCAFS for different file organisations and point out deficiencies in this decision-making process. We conclude by recommending techniques that may be employed to enhance the role of the Ingres Search Accelerator in Ingres database systems.

## 1. Introduction

The Property Rent Accounting and Waiting List (PRAWL) database system under development at the Northern Ireland Housing Executive (NIHE) is large - NIHE is the largest "landlord" in Europe. We refer to it in the rest of the paper as the NIHE database system. The system is being implemented using Ingres/Star on 11 ICL DRS6000 machines. The approximate size of the PRAWL database is 20GB spread over 5 NIHE regions. The expected work load on the system from about 250 on-line users is 37,000 multi-table transactions per day.

A database system of this size and transaction rate clearly requires considerable attention to performance. One 'performance enhancer' available in the applications environment is the Ingres Search Accelerator (ISA) released by ICL in partnership with Ingres Corp. in 1991, claimed by its designers to reduce the time for searches on Ingres tables, thus enhancing the overall performance of the database system. In this paper we investigate whether the ISA lives up to its name and try to gauge its usefulness to the PRAWL database system.

The chief potential benefit of using SCAFS is clearly to enhance performance of a database application. Now performance is usually assessed in terms of response time, throughput or utilization of system resources. Simulation, analytic or measurement studies can be used to evaluate these indices. The present paper reports on a performance measurement study of SCAFS using mainly the responsiveness and, to a lesser extent, utilization indices.

Leung et al. [Leung et al., 1985] studied the efficiency of CAFS 800 but that analytic study is significantly different from ours for two basic reasons:

- SCAFS is very different from the CAFS 800 and so many of the objectives and parameters of the study carried out by Leung et al, are no longer valid, for example, degree of amplification of the drive and cell size, which were characteristics of the EDS60 discs but are no longer meaningful. The parameters that affect the performance of SCAFS are more at the design level of the database, for example, file organizations and indexes. The performance of SCAFS in a multi-user environment is affected by scheduling on the SCAFS board and the number of locations at which the table being searched is located. These are the parameters that we take into consideration, keeping in mind the fact that the NIHE database system is a large multi-user system with a high transaction processing rate.
- Our tests were performed on real data from the property management system database of the NIHE, using measurement rather than the analytic model of the previous study.

Our objective in the present study is two-fold. First we are looking for reasons why the performance of the newly-installed NIHE database system was lower than expected, despite the availability of SCAFS. There is a lack of practical guidance in the literature for development of large relational systems and we hope that by sharing our experience in this paper other users may benefit.

Our second objective is still pragmatic, but it has a more scientific thrust – to gain insights into the applicability of a backend filter architecture (SCAFS here) for different design profiles and to study how other database system modules, for example, the query optimizer, could utilize this information.

Firstly we quantify the precise improvement in performance brought about by SCAFS over direct access search devices for different file organisations. We study the effect of different file organizations and query result sizes, i.e. the number of 'hit' tuples, through a series of simple queries made on the NIHE database.

Secondly, during the investigation of SCAFS, we found that certain SCAFS and INGRES parameters affect the SCAFS performance. So we report on their effects. In particular, we assess the effect of the SCAFS scheduling parameter and the number of locations over which a table being searched is divided.

Thirdly, we give a rough guide to when the Ingres Query Optimizer decides to employ the Ingres Search Accelerator, and critically examine cases where the decision made by the Optimizer may not be the 'best' in practice.

The rest of the paper is arranged as follows. Section 2 describes the NIHE application whose performance enhancement is our prime concern. In section 3 we discuss the improvement in performance brought about by using SCAFS for scanning tables which use different file organizations and also consider the effect of compressing data. This is followed in section 4 by a discussion of the effect of the SCAFS Scheduling Parameter on the performance of SCAFS. Section 5 discusses the effect of data placement on the performance of SCAFS. A rough guide is presented in section 6 to how the Ingres Optimizer decides on using or not using SCAFS when executing a particular query. Section 7 briefly describes a novel approach to using SCAFS within a transaction-intensive database application and section 8 concludes the paper and discusses on-going and future work involving SCAFS being undertaken in the authors' laboratory.

## 2. The NIHE Application

The Northern Ireland Housing Executive, established in 1971 by the Housing Executive Act (Northern Ireland), is the largest body of its kind in western Europe. The organisation is geographically distributed around 70 locations throughout Northern Ireland.

In 1987, the NIHE reviewed the state of their computerisation. The vital role of computerisation was recognised and a decision made to enhance the present system.

Computerisation in the NIHE had started in the 1970s with a number of batch processing systems, mainly for financial accounting. In the 1980s a distributed network connecting each of the district offices of the Housing Executive was developed for on-line queries to their "tenant management" systems. The survey carried out in 1987 highlighted the gap between the NIHE's then current management performance in terms of innovation in policy development and its information technology resources and a decision was made to improve its use of information technology.

The NIHE decided to develop a ten-year systems strategy detailing the contribution of information technology to corporate aims and objectives. The scope of the review included identifying applications required by the NIHE, determining their purpose, ranking them in an order of priority and identifying the most appropriate hardware and software strategy needed to deliver them. The result was a decision to procure a new integrated computer-based tenant management system encompassing a property database, rent accounting and waiting lists. The core component of this tenant management system is the large property management database. The expected size of the database is approx. 20 gigabytes and the expected number of transactions per day is 37,000, executed by approximately 250 concurrent on-line users.

Transaction complexity of an application is normally measured by the number of tables accessed per transaction. For the NIHE the transaction profile is as follows: 4% of the transactions are queries accessing 1-3 tables, 59% access 4-6 tables, and 1% access 7 or more tables; 29% are updates accessing 1-3 tables, 6% are updates accessing 4-6 tables and 1% are updates accessing 7 or more tables. Clearly this is a more complex transaction profile than normal OLTP applications like banking and airline reservations that normally have a higher transaction rate but lower complexity of transaction.

The following specific objectives were outlined for the new tenant management system, in addition to the usual consistency and security objectives [NIHE, 1988]:

- to increase efficiency in the performance of Property, Rent Accounting and Waiting List tasks
- to improve the quality and level of information to managers, staff and customers
- to improve control
- to provide "value for money"
- to Minimise batch updates

In collaboration with ICL, the NIHE are currently developing a distributed property management database system called PRAWL, which will provide the Housing Executive with the facilities outlined above. The system is being developed in INGRES/STAR and uses ICL's DRS6000 machines. A large relational database system like PRAWL poses enormous problems to the implementors who need to look to the database research community for assistance. In this paper we discuss our investigation of the possible use of ICL's SCAFS database filter machines to enhance the performance of the NIHE system. Phase 1 of the project is now complete and is on-line. Phase 2 of the application is now under development. While the tests reported in this paper are based on Phase 1, it has also provided useful insights into performance enhancement for Phase 2.

## 3. SCAFS Vs DASD

In our experiments to quantify the improvement in performance over the conventional Direct Access Storage Devices (DASD)) approach brought about by the use of SCAFS, we considered the response time (CPU time + I/O time) of a query and the CPU time required for the query's execution. Performance indices were evaluated by measurement in the property management database.

Queries used in the experiments were made on an existing table from the application with 113,922 tuples. Table 1 shows the size of the table for the four different file organisation types tested. The table was stored on a 1.2 GB disk with a data transfer speed of 1.9 Mb per second. As SCAFS "can scan data as fast as the data can be read off the disk" [INGRES, 1991], the time taken by SCAFS to scan a table should be dependent only on the size of the table. Table 1 also shows the theoretical scan time ( = file size in Mb /1.9) required by SCAFS to scan the tables. Note that the difference in file sizes is due to the different fill factors and number of overflow pages of the table for different file structures.

Table 1 File Sizes and Theoretical Scan Time of Experimental Data

| File Organisation Type | File Size (Mb) | Theoretical Scan Time (sec) |
|---|---|---|
| Heap | 28.482 | 14.99 |
| Hash | 79.204 | 41.68 |
| ISAM | 88.338 | 46.49 |
| BTree | 36.574 | 19.249 |

Table 2 Effect of SCAFS on Response Time

| <u>No SCAFS</u> SCAFS | Uncompressed 0 | 60000 | Compressed 0 | 60000 |
|---|---|---|---|---|
| Heap | 2.94 | 2 | 3.167 | 1.52 |
| Hash | 2.667 | 2.296 | 2 | 2.57 |
| ISAM | 3.877 | 3.344 | 2.67 | 2.2 |
| BTree | 3.42 | 2.14 | 2.33 | 2.54 |

Tables 2 and 3 show the effect of SCAFS on response time and CPU time and Table 4 shows the effect of data compression on SCAFS, for queries with no 'hit' tuples and 60,000 'hit' tuples respectively. The entries in these tables show the ratios indicated in the top, left-hand corner. As can be seen from Table 2, for uncompressed data the advantage of using SCAFS decreases for the larger number of 'hit' tuples. This is understandable because as the number of 'hit' tuples increases the I/O channel traffic increases, reducing the advantage of using SCAFS — the main gain only being in the CPU idle time while the table is being searched by SCAFS. For compressed data we find that, for hash and BTree file organisations, the effectiveness of SCAFS increases as the number of 'hit' tuples increases.

From Table 3 we find that the savings in CPU time using SCAFS are enormous for queries with the small number of 'hit' tuples but for the larger number of 'hit' tuples, the saving reduces and is sometimes even less than two-fold when the number of 'hit' tuples is approx. 60,000. Once again this is understandable because with a larger number of 'hit' tuples, the CPU is sent more data from SCAFS, increasing its load.

From Table 4 we find that for hash and BTree file organizations the benefit of using compressed data in terms of response time increases for the larger number of 'hit' tuples. Also for hash file organisation, using compressed data increases the CPU time required for queries with the larger number of 'hit' tuples. This highlights the fact that the advantage of using SCAFS varies with the file organization type and great care needs to be taken in deciding when to use SCAFS for different file organizations (see section 6).

Table 3 Effect of SCAFS on CPU Time

| No SCAFS / SCAFS | Uncompressed 0 | 60000 | Compressed 0 | 60000 |
|---|---|---|---|---|
| Heap | 106.33 | 1.62 | 95.875 | 1.42 |
| Hash | 97.34 | 2.91 | 95.428 | 1.718 |
| ISAM | 127.4 | 3.29 | 85.7 | 1.51 |
| BTree | 97.34 | 2.02 | 125 | 2.54 |

Table 4 Effect of Data Compression on SCAFS Performance

| Uncompressed / Compressed | Response Time 0 | 60000 | CPU Time 0 | 60000 |
|---|---|---|---|---|
| Heap | 2.834 | 1.2 | 1.5 | 1.106 |
| Hash | 1.5 | 1.928 | 1.857 | 0.94 |
| ISAM | 5.44 | 2.44 | 2.5 | 1.14 |
| BTree | 1.05 | 1.458 | 1.89 | 1.09 |

The graphs in figures 1(a - d) show more comprehensively, how the response time and CPU time for queries varies as the number of 'hit' tuples varies from 0 to 60000 for heap and ISAM file organizations. The graphs in figure 1(e-h) show the variation in response time and CPU time when using compressed data. We limit our discussion here to heap and ISAM file organizations. For a detailed discussion including BTree and Hash file organizations we refer the interested reader to [Anand et al, 1994a].

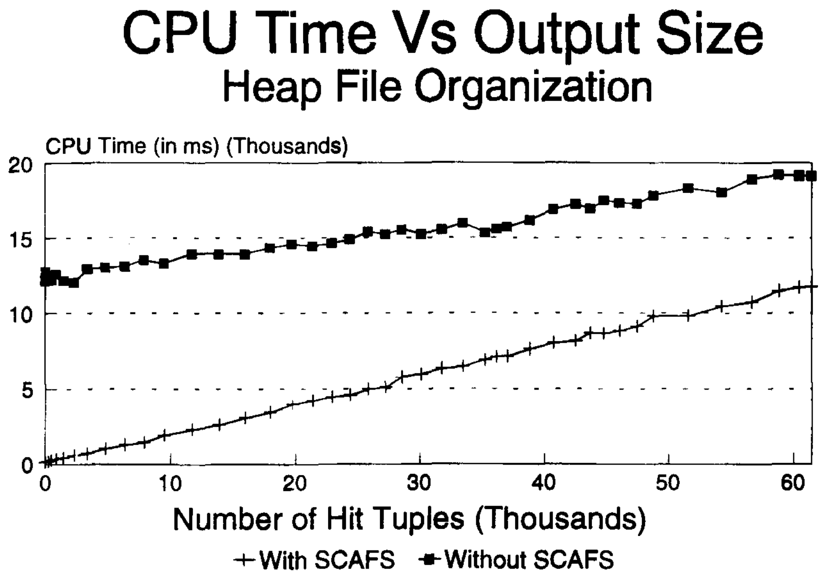


Figure 1(a)

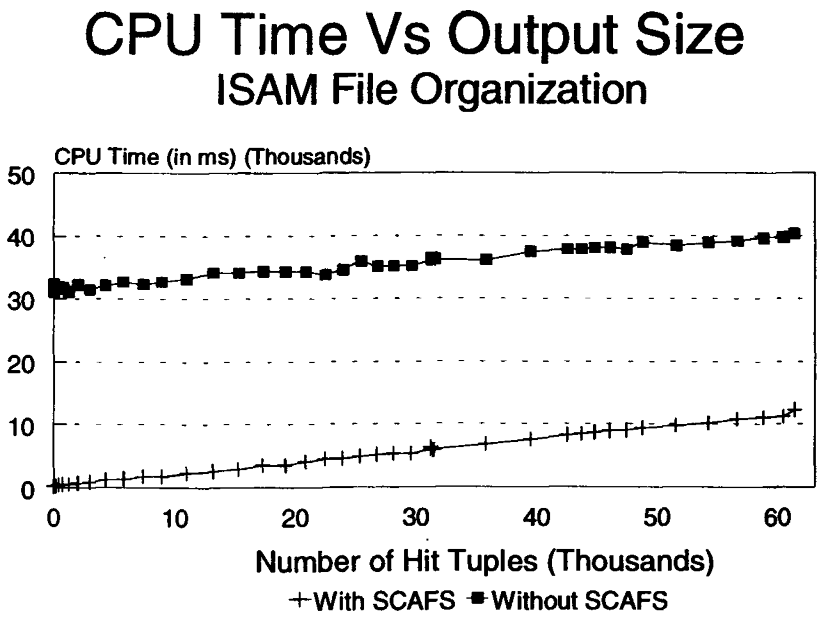


Figure 1(b)

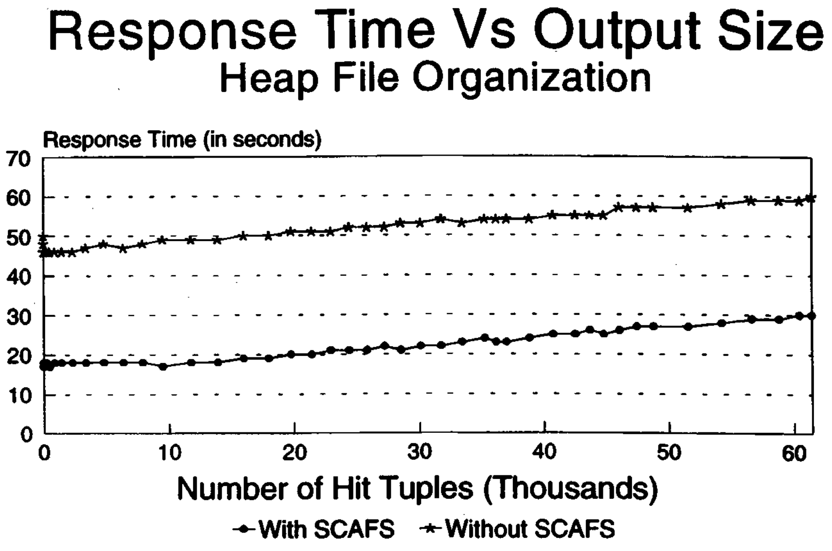


Figure 1(c)

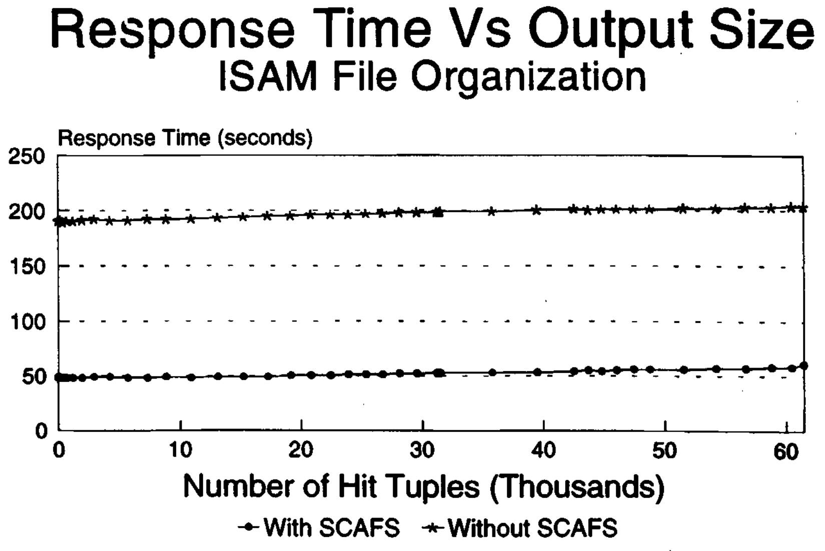


Figure 1(d)

While the graphs in figures 1(a - h) are used primarily to quantify SCAFS benefits, we can see that the benefits in CPU time and response time of using SCAFS are inversely proportional to the number of tuples satisfying the query. So SCAFS is most effective in this sense when the query result size is small. Compressing data does not affect the benefits due to SCAFS. The variation in the benefit of SCAFS for different file structures can be attributed to characteristics like fill factor, number of overflow pages etc.

These results provided valuable insights for the application and for the broader exploration of the potential of SCAFS. The results discussed in this section serve to emphasise the fact that the benefit of SCAFS depends on the file organization of the table being searched and great care must be taken when employing SCAFS.

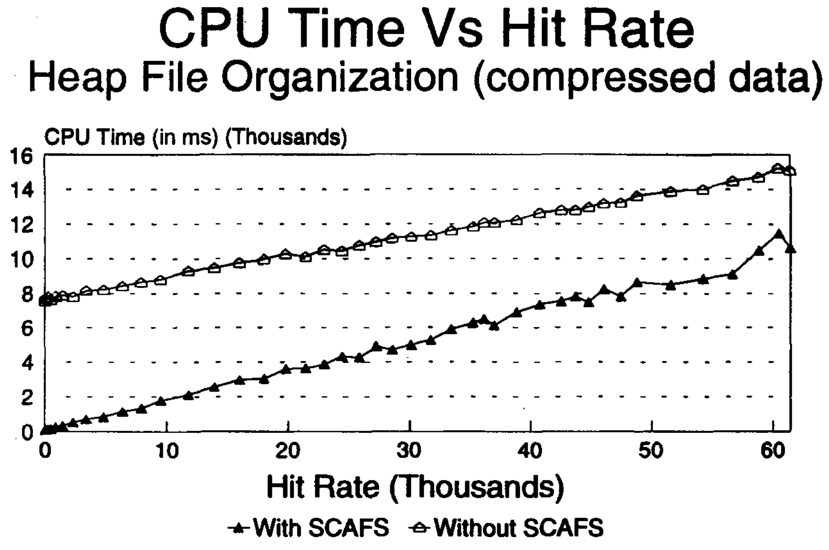


Figure 1(e)

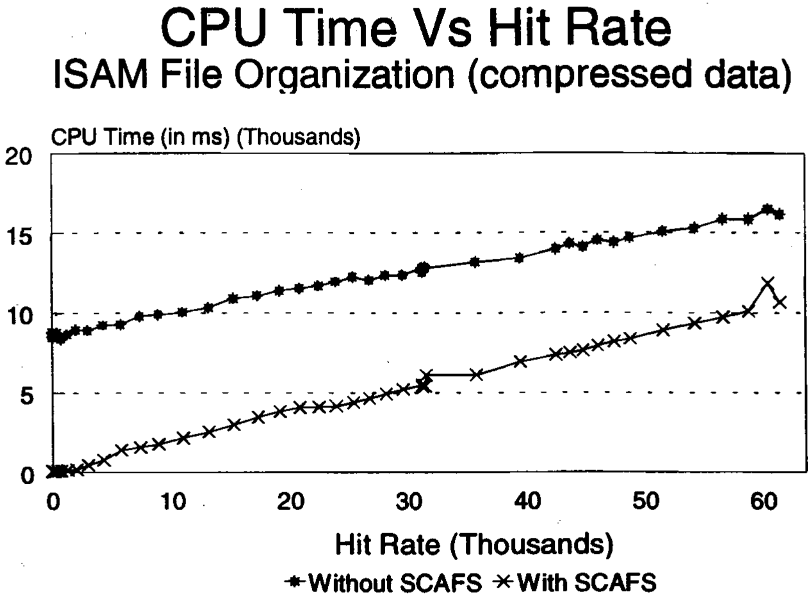


Figure 1(f)

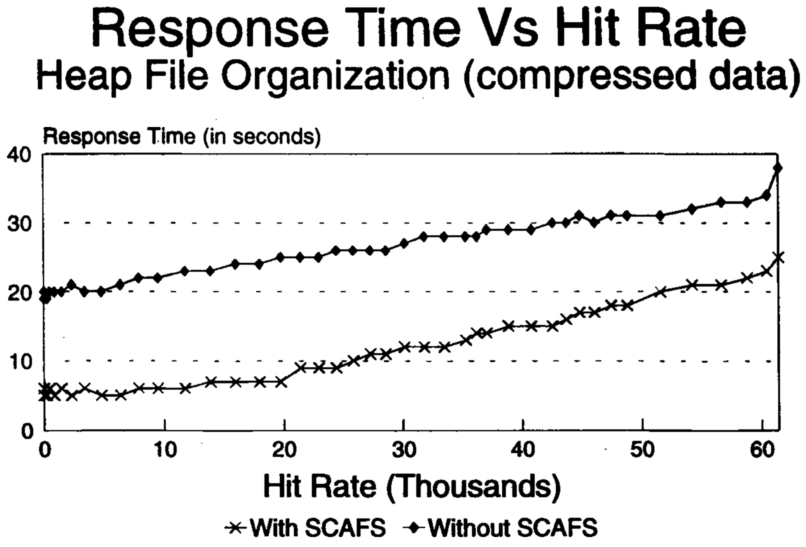


Figure 1(g)

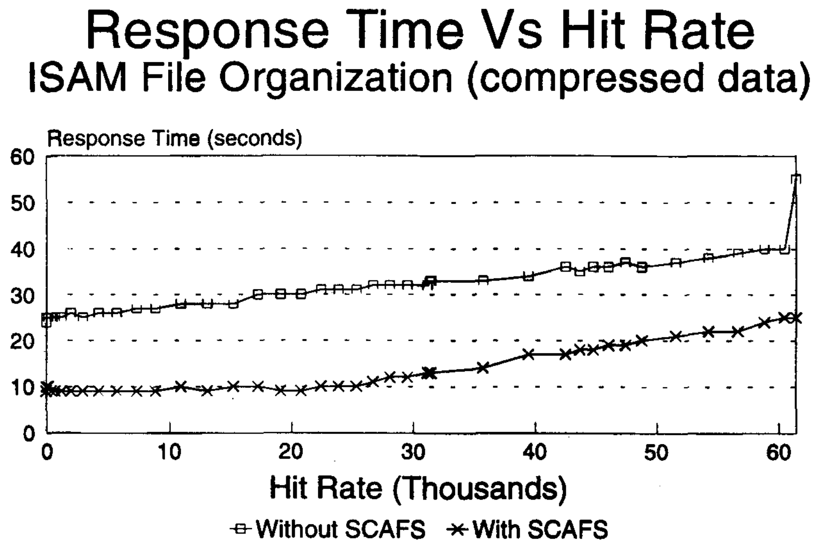


Figure 1(h)

## 4. The Effect of Scheduling on the SCAFS Board

In this section we investigate the effect of different scheduling policies on the performance of SCAFS. We first discuss how the value of the SCAFSSCHED kernel parameter effects scheduling on the SCAFS board. We then study the effect of different SCAFSSCHED values ranging from 1 to 40,000.

For a small value of SCAFSSCHED, the scheduling is fairer in that all the queries get a small time on the board and are not waiting for too long a time in the queue, but the individual response times for each query increase. For very small values of SCAFSSCHED we may find that there is a large system overhead due to scheduling and the advantage of using SCAFS is lost due to this overhead.

For large values of SCAFSSCHED, queries will be waiting in the queue for long periods of time though results for the queries will start being received much sooner. For a very large value of SCAFSSCHED, we may find that it is equivalent to running the queries in serial order 3 at a time. The total time taken to receive the results of all the queries will remain the same.

These observations led us to seek more detailed insights into the effect of this parameter. For a more detailed discussion of the SCAFSSCHED parameter see [Anand et al., 1994a].

The graphs in figure 2(a-h) show how SCAFSSCHED affects the execution on 52, 39, 26 and 13 queries on a single location table of heap file organization. As the value of SCAFSSCHED is increased, the time reduces between all the queries having started and the time the first results are received. From the graph in figure 2 (a) we see that there is not much difference in the performance of SCAFS for SCAFSSCHED values of 20,000 and 40,000. Thus there must exist a value for SCAFSSCHED with the property that there is no increase in performance for any values greater than it. This value of SCAFSSCHED is probably dependent on the time SCAFS takes to scan the table being queried i.e. the size and file organization of the table being scanned (see section 3). Also, the graphs for a constant value of SCAFSSCHED with varying query load are of very similar shapes and average query response times per query. Therefore the performance of SCAFS for a particular value of SCAFSSCHED does not seem to depend on the query load.

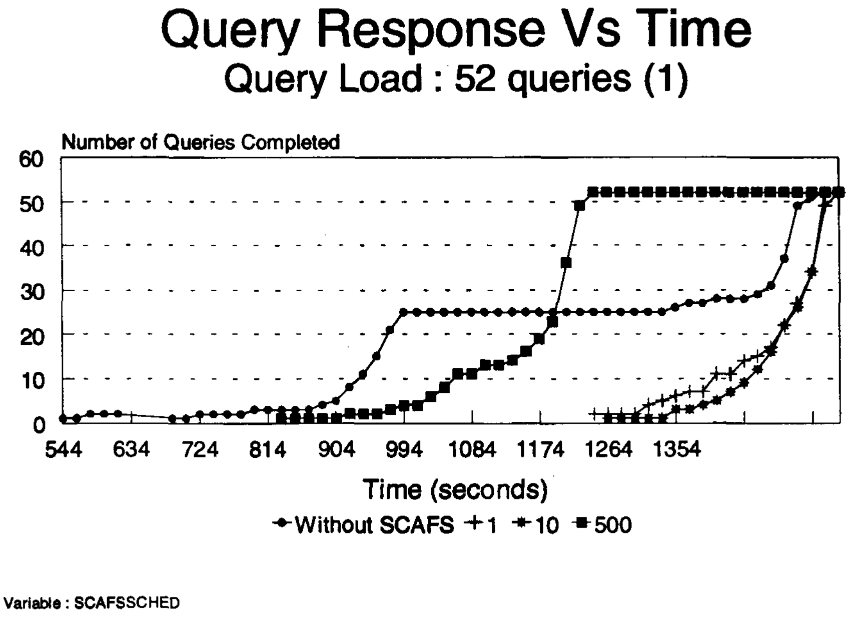
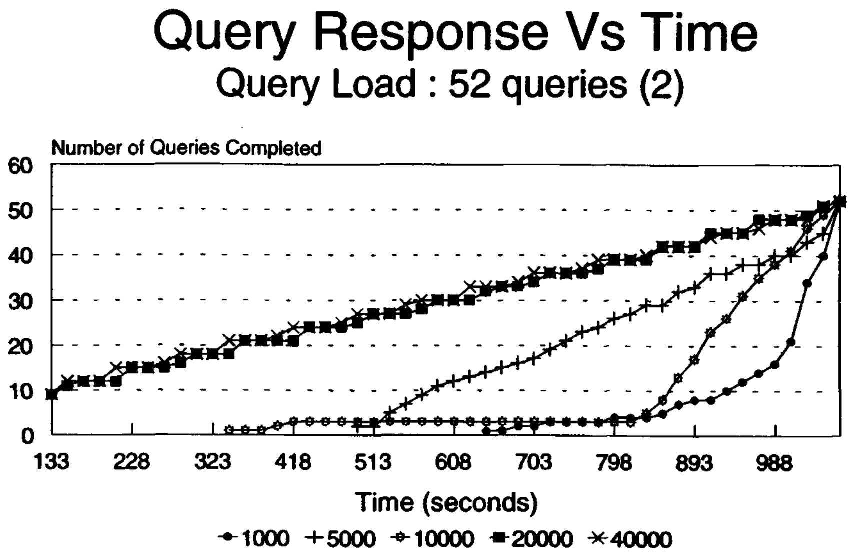
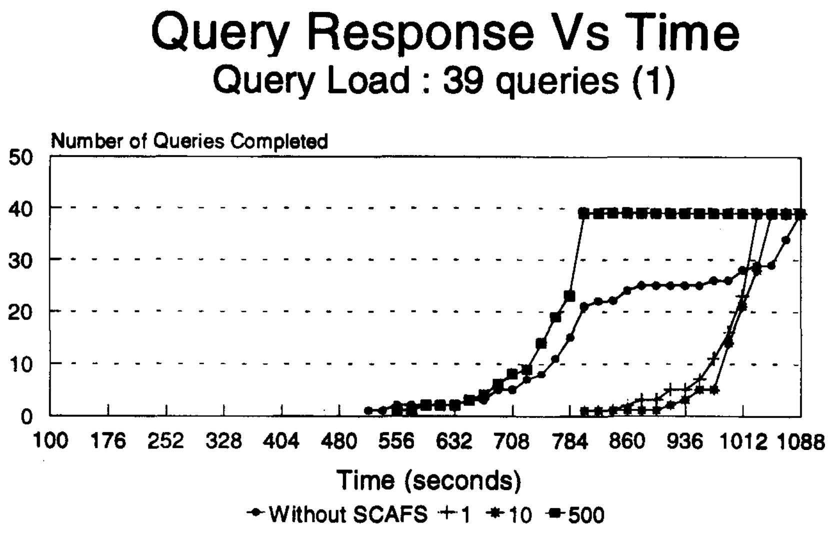


Figure 2(a)



Variable : SCAFSSCHED

Figure 2(b)



Varaiable : SCAFSSCHED

Figure 2(c)

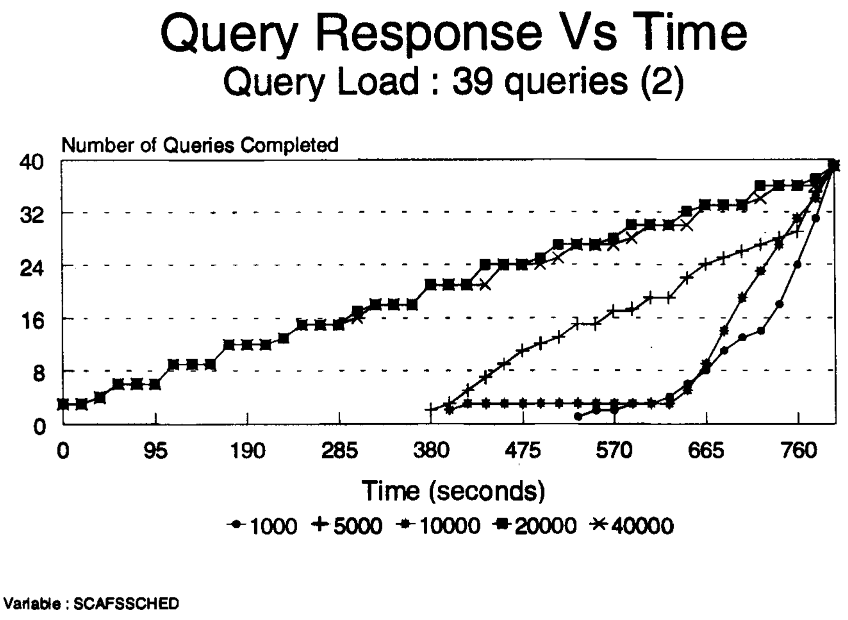


Figure 2(d)

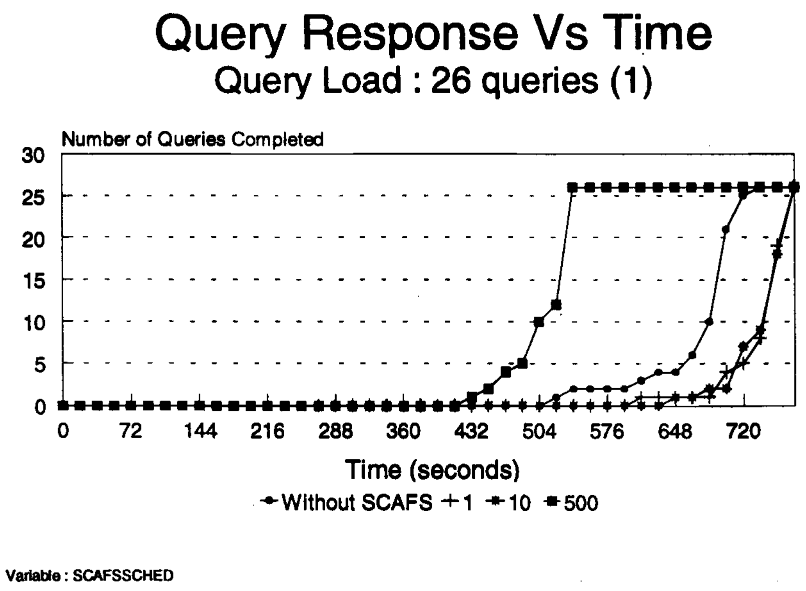


Figure 2(e)

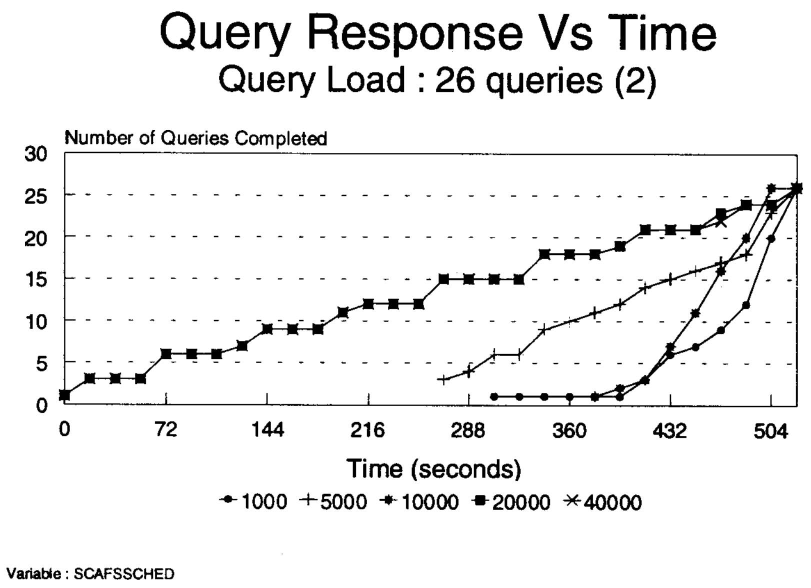


Figure 2(f)

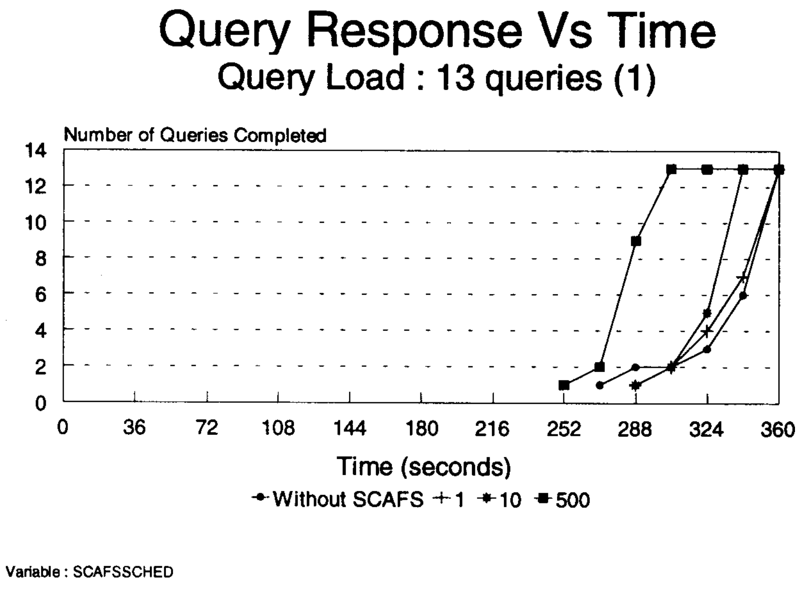


Figure 2(g)

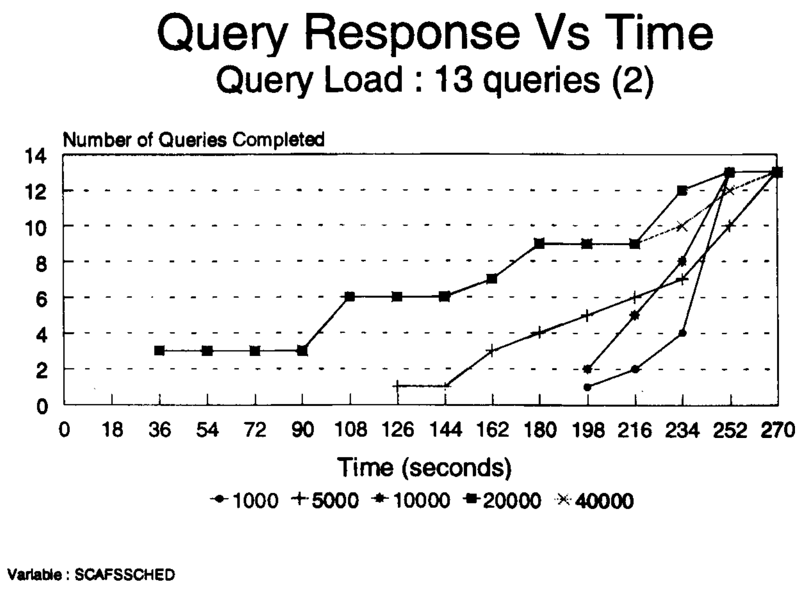


Figure 2(h)

The graphs in figure 3 show the total response time for a varying number of queries on a single location table of heap file organization for different values of the SCAFSSCHED parameter. As can be seen from the graph in figure 3(b), for SCAFSSCHED values ranging from 1,000 to 40,000 there is no appreciable change in the total response time for the queries. However, if the value of SCAFSSCHED is reduced to less than 500 there is an increase in the total response time of the queries (figure 3(a)). For a SCAFSSCHED value of 1 or 10 the total response times are equal to those when not using SCAFS. This degradation is due to the system being overloaded by swapping on the SCAFS board. For SCAFSSCHED value 500 the total response times are comparable to those for higher SCAFSSCHED values. Thus, increasing the value of SCAFSSCHED to a value greater than 500 has little effect on the total time taken by SCAFS to execute all the queries.

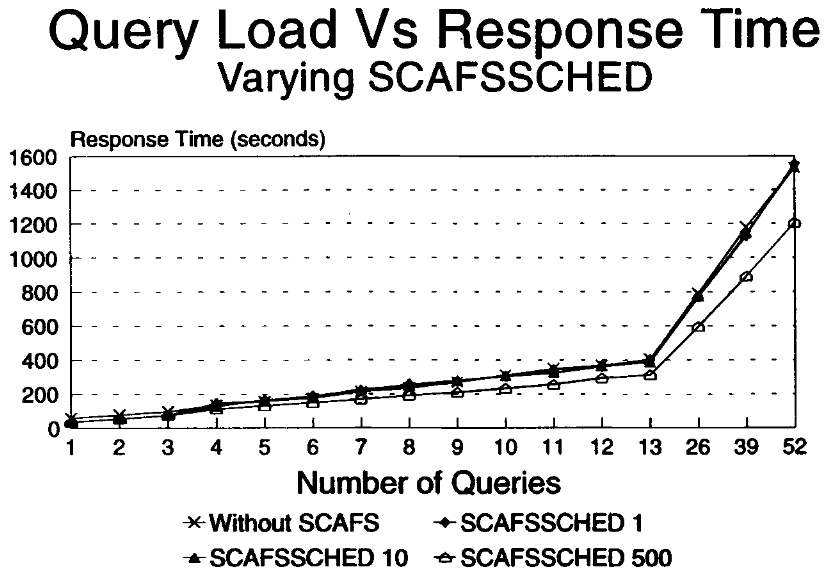


Figure 3(a)

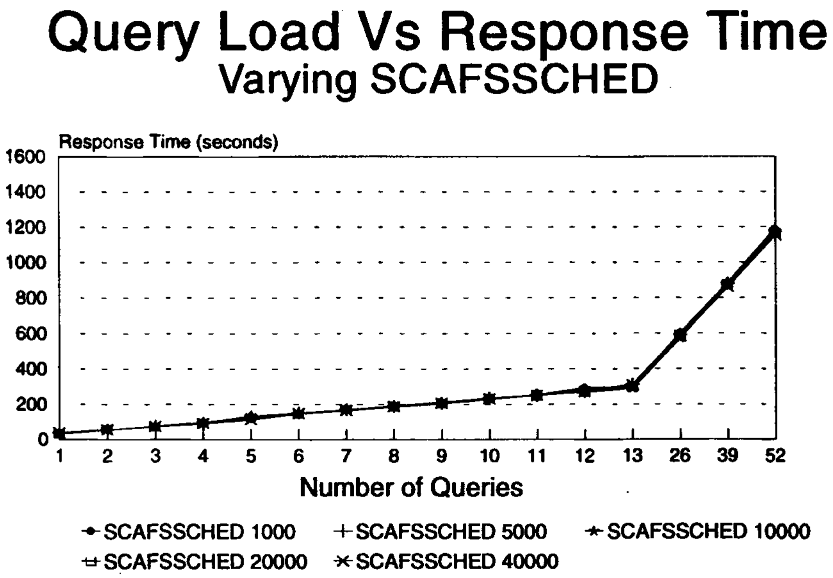


Figure 3(b)

These results show how important this parameter is in multi-user environments, such as the NIHE database system. Clearly, our experimental results, displayed in figures 2 and 3, validate our predictions on how the SCAFSSCHED parameter would affect the performance of SCAFS and therefore, the performance of the database system.

## 5. The Effect of Table Locations

When more than one query is being executed at the same time on data lying on the same disk, there is disk contention which increases the execution time of the individual queries. At any point in time the maximum number of queries searching data in parallel using SCAFS is three, so the maximum disk contention occurs when all three queries are searching data on the same disk.

Disk contention can be reduced by splitting data over a number of locations. Locations could be on the same disk, the same Small Computer System Interface (SCSI) channel or separate SCSI channels. For SCAFS to be employed all the table locations must be accessible to SCAFS.

Splitting data over more than one location on the same disk does not reduce disk contention as there are still three queries concurrently searching for data on the same disk. In fact disk contention may be increased as the disk head would spend more time on seeks.

If data is split over more than one disk on the same SCSI channel, then there is a possibility that at some time the queries on the SCAFS board are searching data on separate disks, reducing the disk contention.

If data from a table is split over disks on separate SCSI channels, we increase the parallelism in the execution of the queries. The disk contention remains the same. There is now a maximum of 3*n queries that could be running simultaneously (where n is the number of SCSI channels that the data is spread over), three on each SCAFS board.

As the advantages of splitting data over different disks on the same SCSI channel, i.e. reducing disk contention, and over different SCSI channels, i.e. increasing parallelism, are independent of each other, a blend of both methods would be expected to give the best results.

For this part of our performance study we looked at how the total response time varies for different work loads on tables with different numbers of locations. The results are presented in figure 4(a-c). As can be seen in figure 4(a), a table with two locations on separate SCSI channels gives lower response times than when the table is stored at a single location, at two locations on the same disk or at two locations on separate disks but sharing the same SCSI. The graph in figure 4(b), shows that the response times for a table with 3 locations on separate SCSI channels are lower than that for 2 locations (figure 4(a)) but the response times for a table with 4 locations on separate SCSI channels are comparable to those for a table with 2 locations on separate SCSIs channels. A table with four locations, two on each SCSI channel performs better but still not as well as the table with three locations on separate SCSI channels. In figure 4(c), we see that response times for a table with 5 locations are higher than those using three locations on separate SCSI. The table with 6 locations, 2 on each SCSI, gives the lowest response times but the improvement over the 3 location table is negligible (see figure 4(b)).

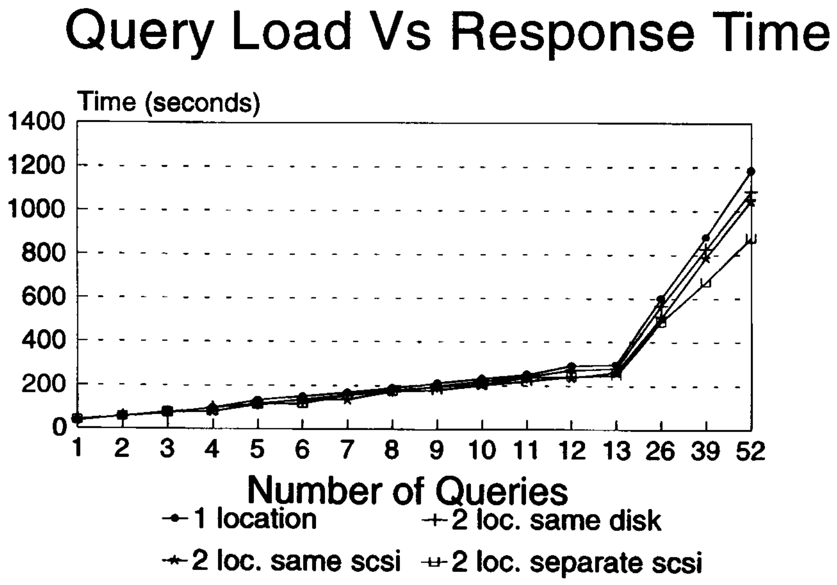


Figure 4(a)

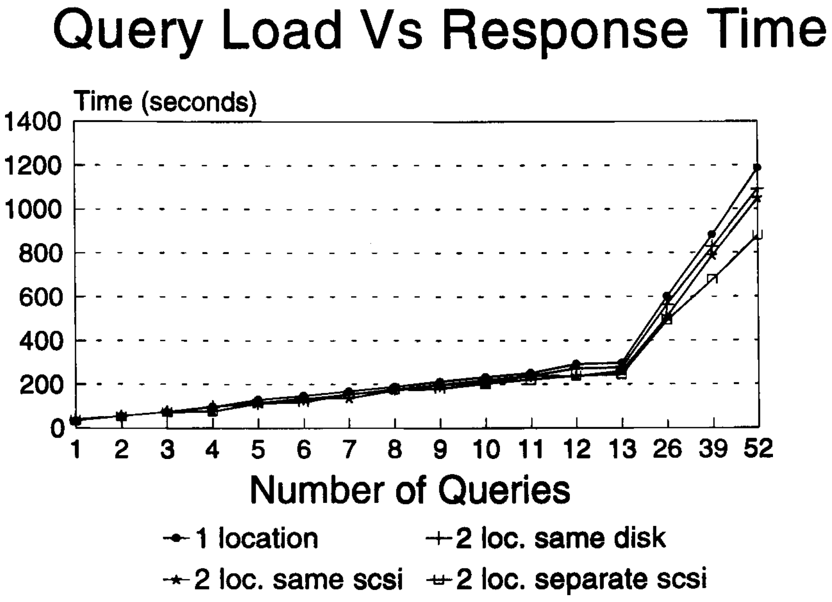


Figure 4(b)

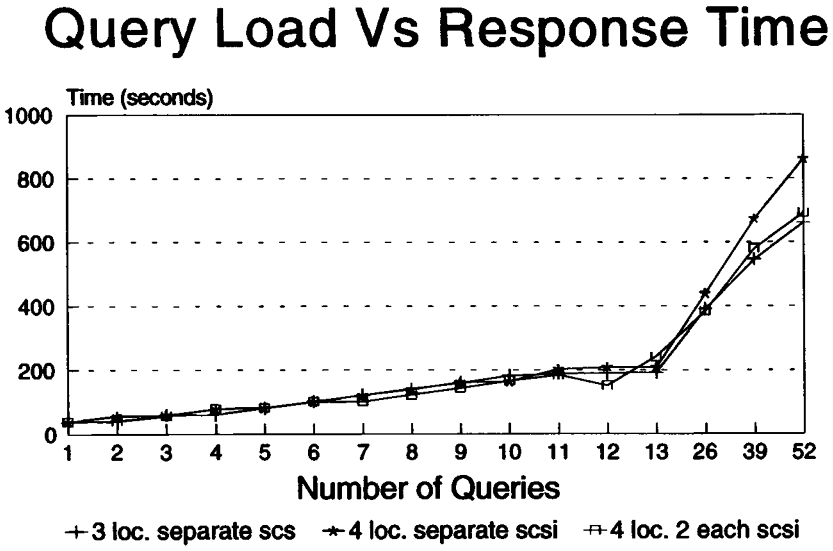


Figure 4(c)

The above results indicate that using 3 separate SCSI channels simultaneously gives the best results for our application and configuration. The reason for this could well be some constraint imposed by the hardware, for example, size of main memory. More SCSI channels being used simultaneously would mean more queries being run in parallel and an increase in memory requirements.

The above results also indicate that spreading a table over separate SCSI channels reduces response times to a much greater extent than spreading a table over separate disks on the same SCSI channel. The queries used in the test were staggered by 2 seconds each. If these queries were staggered by a longer time, having two locations on the same SCSI could have more effect on response times. Also the SCAFSSCHED parameter for the tests had a fixed value of 1000 ms. For larger values of SCAFSSCHED the spreading of data over locations on the same SCSI channel may have a greater effect.

## 6. The Ingres Query Optimizer and The Ingres Search Accelerator

Whether or not SCAFS should be used in the processing of a particular query is solely the decision of the Ingres Query Optimizer. During our investigation of this decision making process we found that at present SCAFS is not being used to its full potential by Ingres. We present here a few guidelines that have been found to be quite accurate in predicting whether SCAFS is going to be employed by the Ingres Query Optimizer.

### 6.1 Heap File Organization

- SCAFS is always employed for searching heap tables.

### 6.2 Hash File Organization

- SCAFS is used to search hash files except when the *where* clause of the query contains equality predicates on all key fields 'anded' together.
- When searching on secondary indexes on the hash table, if the expected number of 'hit' tuples is >= 13.82% of the whole table, SCAFS is employed.
- When searching on secondary indexes, if the *where* clause uses 'not like', 'not between' or 'like' with a % as the first letter in the like-pattern, SCAFS is always used.
- If the *where* clause has two predicates on secondary indexes anded together, SCAFS is employed only if
    - o over 3.35% approx. of the whole table is expected as the result to the query and
    - o each of the predicates, individually, expects to 'hit' 13.82% of the whole table

- If the *where* clause has an 'or', SCAFS is always used.

## 6.3 ISAM File Organization

- If an ISAM file is being searched on its primary index, SCAFS is used only if the *where* clause uses a like-pattern with a % as its first character.
- When searching on secondary indexes on the ISAM table, if the expected number of 'hit' tuples is >= 9.75% of the whole table, then SCAFS is employed.
- When searching on secondary indexes, if the *where* clause uses 'not like', 'not between' or 'like' with a % as the first letter in the like-pattern, then SCAFS is always used.
- If the *where* clause has two predicates on secondary indexes anded together, SCAFS is employed only if
  - over 1.8% approx. of the whole table is expected as the result to the query, and
  - each of the predicates, individually, expects to 'hit' 9.75% of the whole table
- If the *where* clause has an 'or', then SCAFS is always used.

## 6.4 BTree File Organization

- If a file organized as a BTree is being searched on its primary index, SCAFS is only used if the *where* clause uses a like-pattern with a % as its first character.
- When searching on secondary indexes on the BTree, if the expected number of 'hit' tuples is >= 8.74% of the whole table, SCAFS is employed.
- When searching on secondary indexes, if the *where* clause uses 'not like', 'not between' or 'like' with a % as the first letter in the like-pattern, SCAFS is always used.
- If the *where* clause has two predicates on secondary indexes anded together, SCAFS is employed only if
  - over 1.575% approx. of the whole table is expected as the result to the query, and
  - each of the predicates, individually, expects to 'hit' 8.74% of the whole table
- If the *where* clause has an 'or', SCAFS is always used.

The conclusion we arrived at after testing the Ingres Query Optimizer was that the optimizer only considers the use of SCAFS after it has made the decision to scan the table. Thus, it does not take any of the benefits of using SCAFS into account when making the decision whether SCAFS should be employed or not. This results in queries being executed without SCAFS even though it would be more efficient to use SCAFS.

## 7. Increasing Throughput using SCAFS

The results in the above sections can be used to create a kind of query pre-processor that can rewrite a query into a semantically equivalent query that forces the Ingres Query Optimizer to process the query using SCAFS. Such a pre-processor linked with Ingres could have great potential for use in high frequency on-line transaction processing database applications that are CPU bound.

The authors have earlier [Anand et al, 1993, 1994] suggested an architecture for such a query pre-processor. In CPU-bound applications when the CPU or I/O channels are very busy it would be useful to be able to channel some of the load to SCAFS so as to relieve the CPU and I/O channels of some of their load. In doing so we may be increasing the individual query response time but we would be increasing the overall throughput of the system which is clearly more critical in transaction intensive database applications. Most literature on SCAFS [INGRES] stresses the ability of SCAFS to do fast table scans. However, we believe that the use of SCAFS in the way presented in this section would provide enhanced benefits for existing database applications like the NIHE application.

Rules discovered using data mining techniques (Transformation Knowledge) [Anand et al, 1995, 1995a] can be used to reformulate queries into semantically equivalent queries using the guidelines provided in section 6 (Domain Knowledge) so that the Ingres Optimizer is provoked into choosing to use SCAFS to execute the query thereby relieving the CPU and I/O channels of the extra load. Thus, it would be possible for the system to handle a larger number of queries concurrently.

The basic principle of the query pre-processor is given below. For a more detailed discussion on the query processor readers may refer to [Anand et al., 1994].

```
if (SCAFS is not busy)
{
  if (CPU is busy)
  or (I/O traffic is heavy)
  or(cost_of_query(SCAFS) - cost_of_query(¬SCAFS) < threshold)
  {
  Reformulate Query using Domain Knowledge
  and Transformation Knowledge to use SCAFS
  }
else
  {
  Reformulate Query using Domain Knowledge
  and Transformation Knowledge to reduce the
  cost of execution of Query
  }
}
```

## 8. Conclusions and Future Work

A number of conclusions and observations emerge from this study and although they should be considered anecdotal rather than scientific at this stage, we believe they give useful insights into the utilization of SCAFS for large Ingres Databases.

- The improvement in response time and CPU time brought about by SCAFS depends on the organization of the file and the number of 'hit' tuples for the query.
- The time taken by SCAFS to scan a table can be reduced by using a small index or some form of hashing that would reduce the amount of data to be scanned. Especially for large databases it would be useful to be able to use sparse indexes along with SCAFS to reduce the amount of data to be scanned without putting an undue load on the CPU. [Wiles, 1985] showed how secondary indexes could be used in conjunction with CAFS-ISP, the version of CAFS for the VME

environment. Such an approach could be useful in the case of SCAFS as well.

- Tuning the SCAFSSCHED parameter for a specific application can affect the performance of the system considerably.
- The number of table locations over which the table being searched is spread affects the performance of SCAFS for high work loads and can have a considerable effect on the performance of the system.
- The Ingres Query Optimizer does not take into account the full power of SCAFS when deciding when SCAFS should be used to search an Ingres table.

A fairly simple tool that predicts the optimal value of the SCAFS scheduling parameter and the number of table locations over which the table being searched should be spread, given a work load, could be useful at database design time.

We are working on these issues at present and also recommend further research into the use of secondary access paths for associative memories. Earlier studies into the use of secondary indexes in conjunction with CAFS-ISP [Wiles, 1985] have shown that there is benefit in using secondary indexes on ISAM files in conjunction with CAFS.

We believe that these results will provide Ingres Database designers with useful insights into the prediction and improvement of the performance of databases under development.

## Acknowledgements

We would like to thank the ICL SCAFS team at Bracknell and the members of the project team at the University of Ulster and the Northern Ireland Housing Executive for their help and valuable suggestions in performing the above tests on SCAFS. The research was funded by the Industrial Research and Technology Unit, Northern Ireland and the industrial partners were ICL plc and Kainos Software Ltd.

Thanks are due to Morgan Kaufmann Publishers of San Francisco to reprint this slightly edited version of the same paper published by them in the report of the Very Large Database Conference 1994.

## References

ANAND, S.S., BELL, D.A., HUGHES, J.G., "Using Semantic Knowledge to Enhance the Performance of Specialized Database Hardware". Workshop of the FSTTCS'93 Conference, IIT Bombay, December, 1993.

ANAND, S.S., BELL, D.A., HUGHES, J.G., "Database Mining in the Architecture of a Semantic Pre-processor for State-Aware Query Optimization". Proc. of the AAAI-94 Workshop on Knowledge Discovery in Databases, Seattle, July, 1994.

ANAND, S.S., BELL, D.A., HUGHES, J.G., "Performance Enhancement using SCAFS in a Large, Transaction Intensive Database System". Informatics Research Reports Issue No. 9, University of Ulster, Nov. 1994a.

ANAND, S. S. SHAPCOTT, C. M., BELL, D. A., HUGHES, J. G., "Data Mining in Parallel". Proc. of WOTUG'95, April, 1995.

ANAND, S.S., BELL, D.A., HUGHES, J.G., "Evidence-Based Knowledge Discovery in Databases". IEE Colloquium on Knowledge Discovery in Databases, February, 1995a.

BABB., Ed, "Implementing a Relational Database by means of Specialized Hardware". ACM Transactions of Database Systems, Vol. 4, No. 1, March, 1979.

BABB., Ed, "CAFS file - correlation unit". *ICL Tech. J.* November, 1985.

COULARIS, G.F., EVANS, J.M., MITCHELL, R.W. "Towards content addressing in databases". *The Computer Journal*, Vol. 15, No. 2, 1972.

INGRES. The Ingres Search Accelerator User's Guide, 1991.

LEUNG, C.H.C., WONG, K.S. "File processing efficiency on the content addressable file store". Proc. VLDB Conference, 1985.

NIHE. "Property, Rent Accounting, Waiting List - Operational Requirements Vol. 1 - Statement of User Requirements", 1988.

WILES, P.R. "Using Secondary Indexes for Large CAFS Databases". *ICL Tech. J.* Vol.4 Issue 4 pp.419-440, November, 1985.

## Biographies

### *Sarabjot S. Anand*

Sarabjot S. Anand received his B. A. (Honours) degree in Mathematics from Hindu College, University of Delhi in 1990 and the MSc in Engineering Computation from the Queen's University of Belfast in 1991. He joined the Faculty of Informatics at the University of Ulster in 1992 as a Research Assistant where he was involved in research into Database Performance Enhancement. At present he is working in the field of Knowledge Discovery in Database/ Data Mining. He is a member of the ACM and the IEEE.

Mail address: ssanand@uk.ac.ulster.ujva

### *David A. Bell*

David A. Bell has been Professor of Computing (since 1986) and Head of the School of Information and Software Engineering at the University of Ulster. He received his BSc in Mathematics in 1969, MSc in Programming Languages in 1976 and PhD in Database Technology in 1983 from the Queen's University of Belfast.

He has research interests in the areas of database technology and artificial intelligence. He has authored, co-authored and edited 10 books and over 150 papers in these subject areas. He has attracted a number of EU and national research grants.

He was chairman of the VLDB Conference in 1993 and is on the program committees of a number of other conferences.

He is a Fellow of the BCS and a member of the ACM.

Mail address: dbell@uk.ac.ulster.ujvax

### *John G. Hughes*

Professor John G. Hughes is the Dean of the Faculty of Informatics and Director of the Northern Ireland Knowledge Engineering Laboratory (NIKEL) at the University of Ulster. He received his BSc (1st Class) and PhD in Mathematics from the Queen's University of Belfast in 1975 and 1978 respectively and lectured in Computer Science at Queens from 1980-84 and 1986-88. From 1984-86 he worked at the International Atomic Energy Agency in Vienna.

He has research interests in the areas of database technology, artificial intelligence and knowledge engineering. He has published three books and over seventy research papers in these subject areas and has been involved in a variety of collaborative research and development projects with the industry which have attracted funding to the University of Ulster.

He is a member of the IEEE

Mail address: jgh@uk.ac.ulster.ujvax

# RAID

**Steve Hilditch**

**High Performance Systems, ICL, Manchester, UK**

**Abstract**

Redundant Arrays of Inexpensive Discs (RAID) are becoming used increasingly to enhance data integrity, data availability and I/O performance. This paper outlines the benefits, basic concepts and current products and introduces a new RAID design which enhances disc throughput.

## 1. Historical Introduction

Since RAID was introduced as a concept by Patterson, Gibson and Katz [Patterson et al., 1988] [Katz et al., 1989] it has taken a firm hold of peripherals vendors with some RAID-dedicated companies now in existence. RAID (Redundant Array of Inexpensive Discs) was originally designed to make large numbers of small discs into a high performance disc subsystem. However, it has become primarily a means of increasing data availability.

Consider the following quote from the abstract of [Katz et al., 1989]:

> *"... new developments driven by advances in small-diameter (5.25-in. and 3.5 in.) disk drives, promise very high I/O bandwidth if large numbers of devices can be organised into arrays of disks."*

The claim that larger numbers of smaller discs would out-perform smaller numbers of larger discs has been conclusively demonstrated. The hope in 1989 was that larger numbers of smaller discs would be roughly of the same price as smaller numbers of larger discs. However, this hope has not been fulfilled since disc prices are currently determined mostly by economies of scale. Therefore, RAID salesmen have latterly concentrated on emphasising RAID data availability, although most vendors still claim large performance numbers. The acronym RAID has similarly been re-styled by some to stand for Redundant Array of *Independent* Discs.

RAID 100 has been designed to put performance back into RAID. RAID 100 increases the disc subsystem throughput (number of disc transfers per second) performance when reading by up to 50%.

Since the earliest days of [Katz, 1989], RAID has come in various architectural designs called "RAID levels". Katz et al, proposed seven RAID levels (RAID 0 to RAID 6). Others have added further RAID levels, e.g. RAID 0+1 (also known as RAID 10), RAID 53 and RAID 7. In this paper we propose a new RAID 100. Recently, a RAID Advisory Board has been set up by most of the major peripherals players *"to foster an orderly development of RAID technology and introduction of RAID-related products into the marketplace"*. It has produced "The RAIDBook: a source book for RAID technology" [RAID, 1993] which defines RAID 0 to 6 as well as RAID 10 and RAID 53.

## 2. RAID Benefits

There are three main benefits of RAID disc subsystems: data availability, performance and increased disc connectivity.

### 2.1 RAID For Increased Data Availability

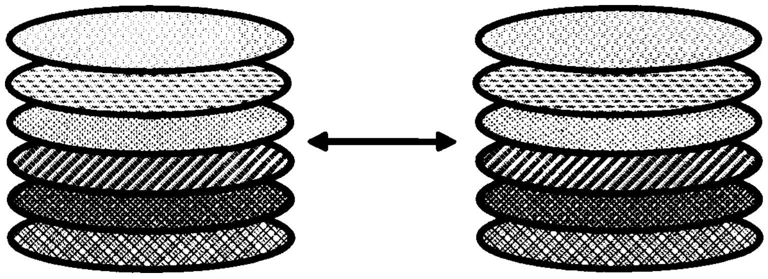


Figure 1 Data mirroring using two discs

The major benefit of RAID is in increasing system and data availability. RAID, in the form of disc mirroring (or plexing), has been available for some years see Figure 1. Other forms of RAID also increase data availability by storing each datum on more than one disc simultaneously. After disc failure, or even after failure of I/O controller cards, customer data is still available without any system crash or service interruption. Furthermore, with disc hot swap support, failed discs can be replaced on-line and original data copied onto the replacement disc. Thus any failure of the disc subsystem can be recovered and serviced without interruption to end user workload.

The unit of availability used in this paper is Mean Time Between Failure (MTBF). The manufacturers' quoted MTBF for a single disc is usually 500,000 hours, or about 57 years. We shall show below that the MTBF of a RAID disc subsystem is considerably longer than this. The MTBF failure rates given for RAID disc subsystems should be combined with cabinet, cable, disc controller and power supply failure rates to give an overall system MTBF. The large numbers below indicate that data security and the discs' mechanical reliability can be increased to the extent that other parts of the system will become the weak links in terms of reliability.

For the purposes of MTBF calculation we shall make the assumption that discs fail at random with a probability of one failure every 500,000 hours and that one disc's failure time is independent of the failure times of other discs. This assumption is not strictly true but gives a fair approximation.

## 2.2 RAID for Performance

In certain circumstances, RAID can increase performance, but in general RAID's increased availability decreases the performance of the disc subsystem. There are two cases in which RAID increases performance:

- it avoids disc bottlenecks
- it increases the disc transfer rate (megabytes per second).

*Have you ever seen one of your discs thrashing while others are idle?* Striping within most RAID designs avoids this possibility by scattering data across a number of discs. Striping means that activity involving one section of data will involve more than one disc. Therefore, overloading single discs is avoided.

*Do you need to do large disc searches such as those done by the ICL Search accelerator?* RAID can increase the transfer rate possible from a single disc through striping. If current disc technologies allow transfer rates of up to 4 MB per second, then striping can increase this up to saturation of disc channel bandwidth (SCSI-2 is either 10 or 20MB per second).

RAID disc subsystem throughput (number of transfers per second) is generally better when reading than when writing (2 to 4 times the throughput).

RAID disc subsystem price/performance, measured in cost per transfer per second, is normally worse than non-RAID disc subsystems, although read price/performance is better than write price/performance. The new RAID level 100 proposed in this document is the first RAID level actually to beat the price/performance of a non-RAID subsystem.

## 2.3 RAID for Increased Disc Connectivity

Hardware RAID controllers can increase disc connectivity by making more than one disc appear to be a single disc. Therefore the total disc capacity of the system can be increased while keeping the number of disc channels and I/O adapters fixed. This benefit will, however, become less important in the near future when serially connected discs take over from standard SCSI channels. Serial channels (IBM's SSA or Fibre Channel Ring Topology) will not only have a greater bandwidth but also a much increased connectivity.

# 3. RAID Concepts

## 3.1 Striping

Striping groups more than one disc together to appear as one logical disc and scatters the data amongst the internal discs in a round-robin fashion. The amount of data written contiguously to each disc in turn is called a "chunk". The collection of contiguous chunks, one for each disc, is called a "stripe". The picture below illustrates a 4 disc striping with 3 stripes and with each chunk equal to one third of a disc. Data written contiguously from the lowest address to the highest will be stored in chunks ordered as in Figure 2.

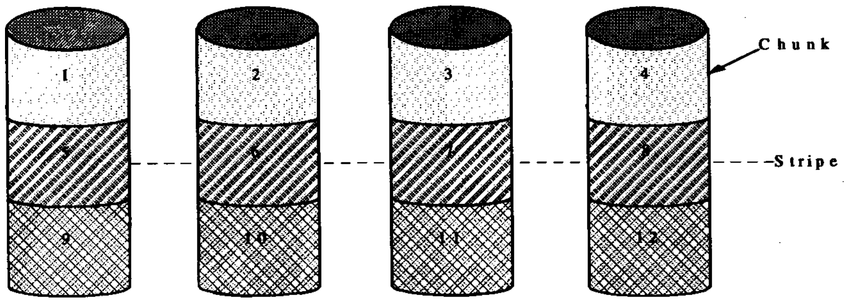


Figure 2 Data striping

It should be noticed that the number of chunks on a disc is usually large (between 1,000 and 100,000). The stripe size varies according to the RAID level and according to the desired performance characteristics.

Striping can be used to enhance performance in two cases: to remove individual disc bottlenecks and to increase the overall transfer rate. Choosing a thick stripe size (about 100 Kbytes) randomises data location on multiple discs so that the possibility of one disc being hit often and others standing idle is avoided. Choosing a thin stripe size (512 bytes or less per chunk) enables all discs within the RAID array to respond to each transfer, thus increasing the data transfer rate.

RAID 53 actually employs two levels of striping within the same array; the discs are arranged as a 2-dimensional array. RAID 100 employs striping within individual discs. See the next section for more details on RAID levels.

All standard RAID systems use the same location on each disc for the chunks in a stripe, as implied by the above diagram. The proposed RAID level 100 changes this convention for the purposes of performance.

## 3.2 Redundant Discs

RAID's increased data integrity and availability are achieved by the use of data redundancy on extra discs which can be thought of as containing parity data.

The simplest form of redundancy is disc mirroring or duplexing (or simply plexing) used in RAID levels 1, 10 and 100. The number of discs required is double that required to store the data. Each data disc has its mirrored pair containing exactly the same data. If either disc should fail its mirror contains all the data and preserves both data integrity and data availability.

Other forms of RAID use one extra disc for an array of discs. The extra disc capacity is used for storing parity information in the form of the exclusive OR (XOR) of the data on the other discs. For instance, if the data in each of the chunks of a stripe is bit-wise XOR-ed and stored on a parity chunk on a separate disc, then the properties of XOR imply that each chunk is the XOR of all the other chunks (including the parity chunk). Therefore if a disc were to fail, each chunk's data can be recovered by XOR-ing the data from the other chunks in the stripe. Figure 3 indicates redundancy as found in RAID levels 3, 4, 5 and 53. The different RAID levels are described in more detail in the next section.

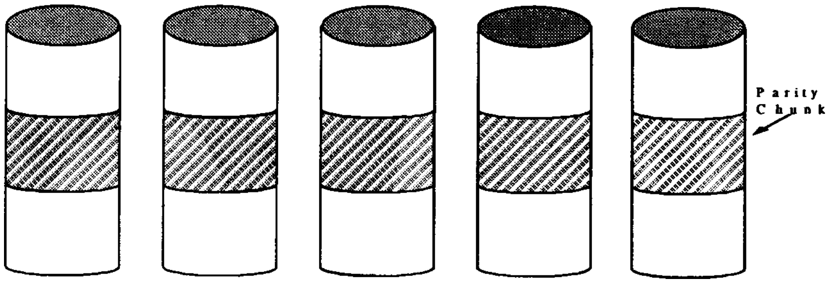


Figure 3 Striping with one parity chunk per stripe

The parity chunks might all be stored on the same dedicated parity disc, as with RAID 3, 4 and 53, or may be stored on different discs according to the stripe number as with RAID 5.

Mirroring (RAID level 1) can be thought of as a degenerate case of RAID 3, 4, 5 and 53 where there is only one data disc and one extra disc for parity. There is no striping, it degenerates completely.

## 3.3 Redundant Disc Access Paths

An associated concept to redundant discs is redundant disc access paths. Disc redundancy means that the failure of any single disc in a disc array does not affect data integrity or data availability. Attention then turns to the second most unreliable disc subsystem component: the I/O adapter or controller. Making the disc access path redundant means that the failure of any single I/O adapter or controller does not affect data availability.

If the RAID functionality is performed by software, as with the *ICL Disk Manager - VxVM* product, *GOLDRUSH* plexing and the *Nile Mirror Disk* facility, then the software can be configured to use different I/O adapters/controllers for each disc in a mirror. Therefore, software mirroring can easily provide redundant disc access paths. This can be drawn schematically as in Figure 4.

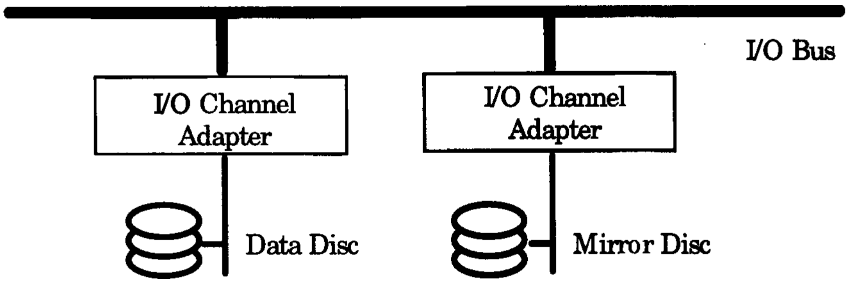


Figure 4 Redundant channel adapters and redundant discs

If the RAID functionality is performed by a hardware RAID controller, or a modified I/O adapter, then redundant disc access paths are more difficult to achieve. Firstly, it may be considered necessary to duplicate the RAID controller itself. This would mean that both RAID controllers would need equal access to the same array of discs. Since dual-ported discs are not common, the best configuration can be drawn as in Figure 5.

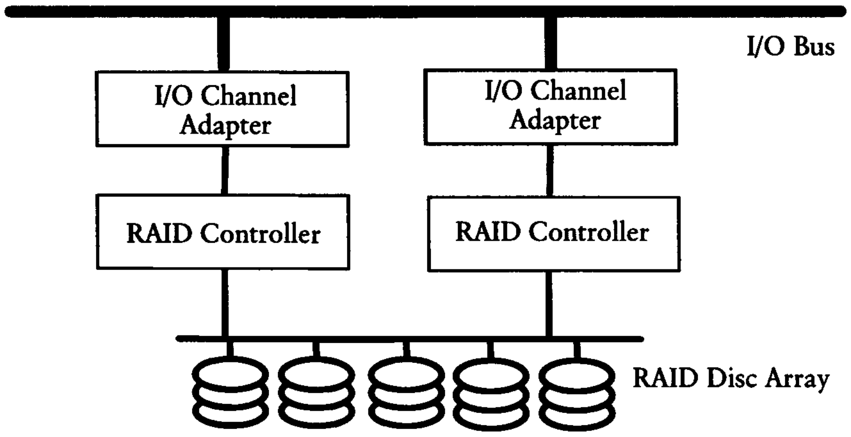


Figure 5 Redundant discs, adapters and RAID controllers

Figure 5 shows disc channels which are dual hosted i.e. there are two masters on the SCSI. Note also that there are redundant adapter cards.

Since both RAID controllers manage parity, perform recovery after disc failure and perhaps buffer disc transfers in volatile memory, redundant RAID controllers are more difficult to implement, especially since care must be taken to be able to cope with each possible failure scenario.

Redundant access paths to discs must be supported by the disc device drivers within the operating system. They must be aware of two sets of device numbers for the two RAID controllers and be able to switch automatically after loss of one access path.

### 3.4 Disc Hot Swap

RAID disc subsystems' promise of increased availability leads many vendors to implement on-line disc replacement to eliminate not only failure downtime but also service downtime. True disc hot swap means being able to run normal customer workloads uninterrupted while failed discs are removed, new discs inserted and the original data built up on replacement discs. Disc hot swap is obviously desirable as the number of discs within computer systems increases, since more discs are likely to fail on average. Many vendors including ICL are now offering disc hot swap. The ICL platforms supporting disc hot swap are *GOLDRUSH*, Nile, ***team*****servers** and ***super*****servers**. Some RAID manufacturers also offer hot swap of redundant RAID controllers.

## 4. RAID Levels

In this section we briefly describe each of the standard RAID levels with their advantages and disadvantages.

### 4.1 RAID 0: Striping

RAID 0 is striping data across a one-dimensional array. RAID 0 has no data redundancy so it does not improve data integrity or data availability. RAID 0 can be used to enhance performance in two cases: to remove individual disc bottlenecks and to increase the overall transfer rate. See the "Striping" subsection above for details. If discs have a mean time between failure of 500,000 hours this means one failure every 57 years. Therefore, the mean time between failure of a 4 disc subsystem is 57/4 = 14 years. RAID 0 is supported by ICL *Power***ARRAY** hardware RAID controllers, *ICL Disk Manager* software RAID controller, *NILE Virtual Disk* software RAID controller and Windows NT *Disk Administrator*.

### 4.2 RAID 1: Mirroring

RAID 1 is simple mirroring of data between two discs. Data integrity and data availability are enhanced greatly since the chance of both discs failing simultaneously is minute. If discs have a mean time between failure of 500,000 hours, this means one failure every 57 years. If the time to replace a disc is 6 hours then the mean time between failure of a mirrored pair is 57/2 x 500,000/6 years = 2,400,000 years. This calculation assumes that discs fail at random which is not entirely true, but the estimate should give the reader the gist of the increased availability. Consequently, disc subsystems containing 4 data discs and 4 mirror pairs would have a mean time between failure of 600,000 years.

RAID 1 costs the price of an extra set of discs which are not cheap. It should be considered the top-end, mission-critical choice of RAID.

RAID 1 on write involves both discs being updated which means that write transfers take slightly longer to complete and take roughly twice as much disc I/O bandwidth. RAID 1 on read involves reading from only one disc. The disc used for reading can be chosen so as to equalise the workload over both discs. RAID 1 read performance should be able to achieve nearly twice the throughput of a single disc (measured in the number of transfers per second).

RAID 1 is supported by ICL *Power***ARRAY** hardware RAID controllers, *ICL Disk Manager* software RAID controller, *NILE Mirror Disk* software RAID controller and Windows NT *Disk Administrator.*

### 4.3 RAID 10 (RAID 0+1): RAID 0 X RAID 1 Two-Dimensional Array

RAID 10, also known as RAID 0+1, arranges the discs in a two-dimensional array where each disc has an identical mirror and where the total data is striped across the mirrored pairs.

RAID 10 is as expensive as RAID 1, but has both the high integrity and the high availability of RAID 1 and the performance enhancements of RAID 0. RAID 10 can, therefore, be said to be the choice of the customer with mission-critical data.

### 4.4 RAID 2: Using Discs to Implement Hamming

RAID 2 arranges the discs in a one-dimensional array. For each byte or sequence of bytes, one bit is written to each disc and Hamming bits are written to one or more discs (depending on the Hamming code chosen). The simplest Hamming code is just parity which degenerates to RAID 3 as described below. Hamming codes have been seen as unnecessarily complicated compared with other RAID levels, so RAID 2 has been neglected.

### 4.5 RAID 3: Sector-Based Striping With Dedicated Parity

RAID 3 arranges the discs in a one-dimensional array. One disc is reserved for parity data only. Data is striped across the remaining discs using thin stripe size (either 128 byte or 512 byte chunks). Since only one extra disc must be bought for each array of data discs, RAID 3 is a cheaper RAID option than RAID 1 or RAID 10. Its mean time between failure of more than one disc per array can be calculated as above. For a 5 disc RAID 3 array, 4 data discs and one parity, with 6 hours to replace a faulty disc, the mean time between failure is 17/5 x 500,000/(6 x 4) years = 71,000 years. For a 9 disc RAID 3 array, 8 data discs and one parity, the mean time between failure is 17/9 x 500,000/(6 x 8) years = 20,000 years.

Since the stripe size is small, typically the same size as or less than normal disc transfers, every disc is involved in most disc transfers. This means that the data transfer rate is roughly equal to the number of data discs times the transfer rate of a single disc. This also means that the throughput of a RAID 3 array (number of transfers per second) is roughly equal to the throughput of a single disc.

Its performance characteristics mean that RAID 3 is chosen for large data transfers and is inappropriate for large numbers of small transfers (OLTP for example).

## 4.6 RAID 4: Large-Block-Based Striping With Dedicated Parity

RAID 4, like RAID 3, arranges the discs in a one-dimensional array with one disc reserved for parity data. Data is striped across the remaining discs using thick stripe size (chunks of at least 16 Kbytes size). Similarly to RAID 3, RAID 4 is a cheaper RAID option than RAID 1 or RAID 10. Its mean time between failure of more than one disc per array can be calculated as with RAID 3. For a 5 disc RAID 4 array, 4 data discs and one parity, with 6 hours to replace a faulty disc, the mean time between failure is 71,000 years. For a 9 disc RAID 4 array, 8 data discs and one parity, the mean time between failure is 20,000 years.

Since the stripe size is large, typically of larger size than normal disc transfers, only one disc is involved in most read transfers. On the other hand, a write transfer can be completed by reading and writing to two discs, the data disc and the parity disc. The data transfer rate is roughly equal to a single disc, except on very large transfers (at least the same size as a stripe) when the transfer rate is that of a single disc times the number of data discs. The read throughput (number of transfers per second) is roughly equal to the throughput of a single disc times the number of data discs. The write throughput is roughly equal to the throughput of a single disc, since the parity disc becomes a bottleneck.

Its write throughput compared with RAID 5 means that RAID 4 is rarely chosen for disc arrays. RAID 4 is appropriate for tape arrays since tapes are mostly used for backup and restore, which involve large data transfers. In fact, The Pyramid *Nile* product has a RAID 4 tape array called *FastTRAC*.

## 4.7 RAID 5: Large-Block-Based Striping With Rotated Parity

RAID 5, like RAID 3 & 4, arranges the discs in a one-dimensional array. However, parity is stored on different discs according to the stripe number. Data is striped across the remaining space on all the discs using thick stripe size (chunks of at least 16 Kbytes in size). Similarly to RAID 3 & 4, RAID 5 is a cheaper RAID option than RAID 1 or RAID 10. Its mean time between failure of more than one disc per array can be calculated as with RAID 3. For a 5 disc RAID 5 array, with 6 hours to replace a faulty disc, the mean time between failure is 71,000 years. For a 9 disc RAID 5 array, the mean time between failure is 20,000 years.

Since the stripe size is large, typically of larger size than normal disc transfers, only one disc is involved in most read transfers. On the other hand, a write transfer can be completed by reading and writing to two discs, the data disc and the disc holding parity for that stripe. The data transfer rate is roughly equal to a single disc, except on very large transfers (at least the same size as a stripe) when the transfer rate is that of a single disc times the total number of discs. The read throughput (number of transfers per second) is roughly equal to the throughput of a single disc times the total number of discs. The write throughput is roughly equal to the throughput of a single disc times the total number of discs divided by 4.

Its comparative cheapness, compared with RAID 1 & 10, and its comparative throughput compared with RAID 3 & 4, means that RAID 5 has become a very popular RAID level.

RAID 5 is supported by ICL *Power***ARRAY** hardware RAID controllers, *NILE Virtual Disk* software RAID controller and Windows NT *Disk Administrator*.

## 4.8 RAID 6: RAID 5 With a Second Independent Parity Scheme

RAID 6, like RAID 5 arranges the discs in a one-dimensional array with two independent parities stored on different discs which vary according to the stripe number. Data is striped across the remaining space on all the discs using thick stripe size (chunks of at least 16 Kbytes in size). RAID 6 is a cheaper RAID option than RAID 1 or RAID 10. Its mean time between failure of more than one disc per array can be calculated similarly to RAID 3. For a 6 disc RAID 6 array, with 6 hours to replace a faulty disc, the mean time between failure is 17/6 x 500,000/(6 x 5) x 500,000/(6 x 4) years = 980 million years. For a 10 disc RAID 6 array, the mean time between failure is 17/10 X 500,000/(6 x 9) X 500,000/(6 x 8) years = 160 million years.

The data transfer rate is roughly equal to a single disc, except on very large transfers (at least the same size as a stripe) when the transfer rate is that of a single disc times the total number of discs. The read throughput (number of transfers per second) is roughly equal to the throughput of a single disc times the total number of discs. The write throughput is roughly equal to the throughput of a single disc times the total number of discs divided by 6, since 3 read operations and 3 write operations must be performed for each write transfer.

The levels of availability achievable with RAID 6 seem out of proportion to those of the rest of a typical computer system. Also, its relative complexity and its poor relative performance mean that RAID 6 has been little used.

## 4.9 RAID 53: RAID 0 X RAID 3 Two-Dimensional Array

RAID 53 arranges the discs in a two-dimensional array. A one-dimensional array of discs is reserved for parity data only. Discs are partitioned into RAID 3 groupings: with dedicated parity discs and where data is striped across the remaining discs using thin stripe size (either 128 byte or 512 byte chunks). Data is striped in a RAID 0 fashion (thick stripe size) across the RAID 3 disc arrays.

Since only one extra disc must be bought for each group of data discs, RAID 53 is a cheaper RAID option than RAID 1 or RAID 10. Its mean time between failures is similar to a collection of RAID 3 arrays: with a 15 disc RAID 53 array, 12 data discs and 3 parity, with 6 hours to replace a faulty disc, the mean time between failures is 17/5/3 x 500,000/(6 x 4) years = 24,000 years. For a 30 disc RAID 53 array, 24 data discs and 6 parity, the mean time between failures is 17/5/6 x 500,000/(6 x 4) years = 12,000 years.

Since the stripe size is small in each RAID 3 group, every disc is involved in most disc transfers. This means that the data transfer rate of each RAID 3 group is roughly equal to the number of data discs times the transfer rate of a single disc, and the transfer rate of the RAID 53 array is equal to the transfer rate of a single disc times the number of data discs in each RAID 3 group.

This also means that the throughput of each RAID 3 array (number of transfers per second) is roughly equal to a single disc, and the throughput of a RAID 53 array is roughly the throughput of a single disc times the number of RAID 3 groups.

Its performance characteristics mean that RAID 53 is between RAID 3 and RAID 5 in performance: fairly good on transfer rate and fairly good on throughput. The ICL *Power***ARRAY Plus** hardware RAID controller is an implementation of RAID 53.

## 5. A New RAID Level: RAID 100

In this section we propose a new RAID level, RAID 100, which is similar to RAID 1 mirroring but improves read transfer performance by clever positioning of data and choice of disc from which to read. Like RAID 1, it can also be used with RAID 0 in a large array of an even number of discs. RAID 100 is the subject of a patent application by ICL and is being considered for use within ICL's RAID products.

### 5.1 Data Location

RAID 100 uses two discs for each disc's capacity of data. Each disc contains the same information but stored in different places. In fact, half of the data is located on the outermost cylinders of the first disc and the innermost cylinders of the second disc. The rest of the data is located on the innermost cylinders of the first disc and the outermost cylinders of the second disc.

For reasons of performance, thick striping is used to locate consecutive chunks alternately between the innermost cylinders and the outermost cylinders: the read transfer algorithm dictates which disc is used for reading and care must be taken to avoid one disc becoming a read bottleneck.

### 5.2 Read Transfer Algorithm

The read transfer algorithm is simply that in the absence of disc failure, data is always read from the outermost cylinders of a disc. Since each data chunk is located on the outermost cylinders of one disc, this means that the disc chosen for the transfer is fixed according to the parity of the chunk number.

### 5.3 Write Transfer Algorithm

As with RAID 1, updated data must be written to both discs, which will involve writing to the outermost cylinders of one disc and the innermost cylinders of the other disc.

### 5.4 RAID 100 Example

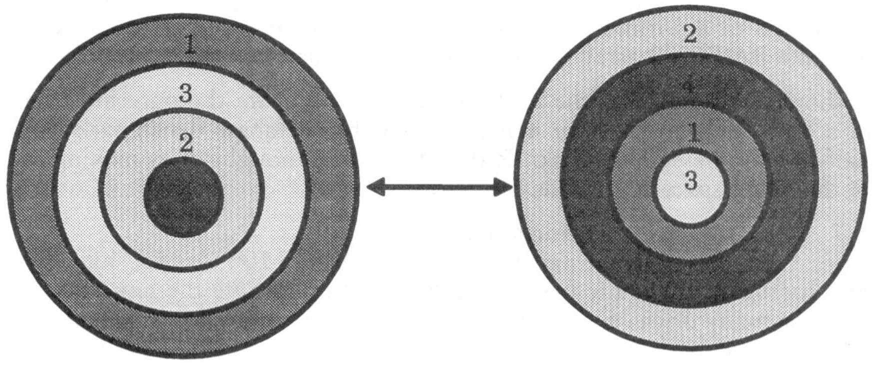


Figure 6 A simplified RAID 100 data layout

Figure 6 shows a simplified diagram of RAID 100. The data is divided into 4 chunks labelled 1-4 and stored as shown on the two discs. Chunks 1 and 3 are always read from the left disc and chunks 2 and 4 are always read from the right disc.

### 5.5 RAID 100 Resilience

The resilience of a RAID 100 array is similar to that of a RAID 1 array, and when used with RAID 0 is similar to the resilience of a RAID 10 array. If the time to replace a disc is 6 hours then the mean time between failure of a RAID 100 disc pair is 57/2 x 500,000/6 years = 2,400,000 years. Consequently, a RAID 100 disc subsystem containing 8 discs with RAID 0 striping on top would have a mean time between failure of 600,000 years.

### 5.6 RAID 100 Performance

The transfer rate of RAID 100 (megabytes per second) will be approximately that of a single disc on write. On large read transfers, at least twice the size of the chosen chunk, both discs will respond to the transfer and therefore the read transfer rate of RAID 100 will be roughly twice that of a normal disc for large transfers.

The throughput of RAID 100 with 100% read transfers will be greater than twice the throughput of a single disc. This is because data is always read from the outermost cylinders of the discs and therefore the seek time of the disc arms will be shorter. For example, today's typical high performance 3.5 inch disc rotating at 7200 rpm will have an average rotational latency of 4.2 milliseconds. If the average seek time is 10 milliseconds then with RAID 100 it would be approximately 5 milliseconds since less than half of the cylinders need to be traversed. The total time to complete a disc transfer is usually about 1 millisecond + the average seek time + the average rotational latency. For a normal disc this would be about 15.2 milliseconds, for a RAID 100 disc the average read transfer time would be about 10.2 milliseconds which is an improvement of 49% on the performance of a single disc. The RAID 100 disc array contains two discs each of which has the read throughput of about 1.49 times that of a normal disc. Therefore the read throughput of two discs within a RAID 100 array is about that of three discs!

It should be noted that the trick of reading only from the outermost half of a disc has been used for many years within database benchmark runs, half the disc being left blank. RAID 100 can be thought of as making use of this spare space to provide resilience.

### 5.7 RAID 100 Price/Performance

Since the throughput performance of RAID 100 is so high, its price/performance is the best of the RAID levels despite the extra cost of twice as many discs. See the Conclusions section below for a comparison of price/performance.

## 6. Conclusions

We have seen that the early high performance aspirations of RAID were not fully realised and RAID vendors utilised the increased availability and data integrity of RAID (levels other than 0). With RAID 100 we have attempted to restore high performance to RAID disc subsystems.

We provide below a comparison of RAID levels, showing availability, transfer rate performance, throughput performance and price/throughput where the unit of price is assumed to be £1,000 for a standard disc and the cost of the RAID controller is assumed negligible. The standard disc is assumed to be a 7200 rpm, 10 millisecond average seek time disc, with 1 unit megabytes per second transfer rate, 66 transfers per second sustained throughput (a 1 millisecond overhead is assumed for each transfer).

For comparison purposes we assume that 4 discs worth of data must be stored, which means that RAID 1 and RAID 100 will need 8 component discs, RAID 0 will need 4 discs and the other RAID levels will need 5 component discs. Table 1 shows a comparison of the important RAID levels. Average throughput is estimated using the rule of thumb "70% of all transfers are read transfers and 30% write".

Table 1: A comparison of some RAID levels

| RAID Level | Price £ | MTBF yrs.est. | Transfer Rate | Read Thr'put | Write Thr'put | Average Thr'put | Price/Av. Thr'put |
|---|---|---|---|---|---|---|---|
| None | 4,000 | 14 | 1 | 264 | 264 | 264 | 15 |
| 0 | 4,000 | 14 | 1 | 264 | 264 | 264 | 15 |
| 1 | 8,000 | 600,000 | 1 | 528 | 264 | 449 | 18 |
| 3 | 5,000 | 71,000 | 4 | 66 | 66 | 66 | 76 |
| 4 | 5,000 | 71,000 | 1 | 264 | 66 | 205 | 24 |
| 5 | 5,000 | 71,000 | 1 | 330 | 83 | 261 | 19 |
| 100 | 8,000 | 600,000 | 1 | 787 | 264 | 630 | 13 |

## References

KATZ, R.H., GIBSON, G.A., & PATTERSON, D.A., "Disk System Architectures for High Performance Computing" Proceedings of the IEEE, Vol. 77 No. 12, December 1989.

PATTERSON, D.A., GIBSON, G.A., & KATZ, R.H., "A Case for Redundant Arrays of Inexpensive Disks", Proceedings of the ACM SIGMOD Conference, Santa Clara, CA, 1988.

"The RAIDBook: A Source Book for RAID Technology" The RAID Advisory Board, Lino Lakes MN 55014-1296 June 9, 1993.

## Biography

*Steve Hilditch*

Steve Hilditch had a PhD in pure Mathematics (Algebraic Topology) and a BA in Theology before turning to Computers in 1987. From 1987 until 1991 he worked as a research associate and part-time lecturer at Manchester University on joint projects with ICL, High Performance Systems: Flagship, EDS and *GOLDRUSH*. During this time he completed his MSc in Computer Science. After a year learning French in Montpellier, he joined ICL, Bracknell. He now works as a systems designer in the UNIX Data Centre Systems product stream within ICL, Corporate Systems. Since joining ICL full-time, he has filed 13 patents in both hardware and software.

# Improving Configuration Management for Complex Open Systems

**M.J. Blin**

Université **Paris Dauphine, Place du Marechal de Lattre de Tassigny, Paris 16$^e$, France**

**J. Lisicki and I.G. Puddy**

**ICL-France**

**Abstract**

This paper describes the practical experience of version and configuration management of ISS400 studied by Paris-Dauphine University and ICL jointly to understand the requirements and the current issues in order to identify improvements. We examined the software and hardware structure, procedures to create new versions of packages and of files, system integration, customisation and management of customer distributed system versions. Two currently used tools, PVCS and PCMS, and their adequacy to the requirements were analysed and compared with a conceptual model already researched by the University.

The conclusion is that the current ISS400 version and configuration management can be radically improved in order to facilitate complex object versions and dependency links, inter-working of multiple teams (development, installation, maintenance,) plus the management of a lot of configurations containing hardware, software, documentation and services. This is essential especially for systems to be adapted for a large number of differing customer requirements such as languages, hardware. The goals are to improve the productivity of teams and product quality.

One aspect addressed by the proposed model is the automatic management of the inter-configuration and the inter-objects consistency constraints (set-up, control and dynamic updates). This is currently performed manually without any help from the facilities, causing possible inconsistencies within the whole system.

## 1. Introduction

A *complex system* can often convey different meanings and implications for each reader. The definition used for this paper is that a complex system is a mixture of various elements comprising hardware, operating system(s) environment, the base application(s), specific modifications required by the end user, connected peripherals, cabling, interfaces etc. all of which interact. Each element has well defined versions related to its maturity level, to customer variants and to the business processes it is designed to support and work with. To provide a system or business solution for one end user or a general style solution adapted to meet specific requirements, a *system version* is generated (figure 1). This means that one particular version of each relevant element is picked out.

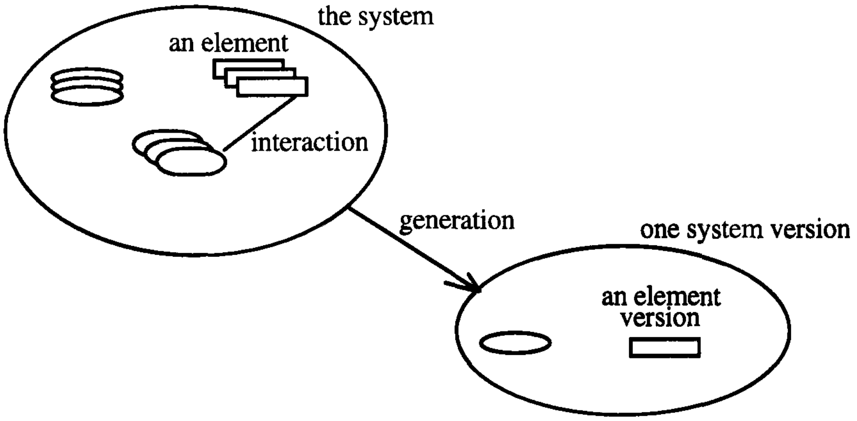


Figure 1 System and system versions

Subsequent changes or modifications to one or more elements - either to provide enhanced functionalities or to adapt the elements to new countries or to new languages - result in new system versions.

Because of business constraints and the complexity of the system, system management operations are distributed over several teams and possibly several sites. Each team works on a part of the system called a *sub-system*.

To create a new version of a system, each team creates a new version of the sub-system it manages. The resulting new sub-system versions are then integrated to constitute a whole and consistent system version.

All the operations of managing the evolution, installation and maintenance of a complex system are called in this paper *system management*. They include distributed version control and distributed configuration management.

A lot of published work is concerned with version control and configuration management. A synopsis of configuration management is provided in [Buckley, 1992], [Fieux, 1990] and [Feldman, 1991]. Much of these papers focus on version control and configuration management for software development and has led to several configuration management tools presently used by software suppliers [Belkhatir, Estublier, 1990], [Cohen, 1990], [Combes, 1990], [SQL, 1992], [Sage, 1990], [Hô Xich-Tuê, 1990], [Kruchten, 1990]. More recently, solutions were proposed for CAD [Katz, et al., 1986] [Kim, et al., 1989] [Katz, 1990] to control versions of designed elements. Object versioning and configuration management in object-oriented databases [Agrawal, et al., 1991], [Kafer, Schoning, 1992], [Sciore, 1994] are also being studied. Researchers in software analysis and design methods have also tried to answer the question of how to include version control and configuration management in analysis and design models, methods and CASE tools [Cohen, et al., 1988] [Malher, Lampen, 1988].

As far as we know, there is no published work in the area of system management. The management and the maintenance of installed complex system versions needs to deal with a great number of configurations which share a lot of element versions. Element versions have multiple dependencies and links between them. When new versions of a system are created due to the evolution of customer environments or to the maintenance process, new dependencies and new links are added or replace old ones. In addition, the management and the creation of system versions are made by several teams with a cooperative process. Thus, we built a model well adapted to the management of versions and of configurations of complex systems. Its adequacy to the set problem was verified in applying it to an existing complex system (the ICL ISS400 Retail Point of Sale System).

The work progressed as follow: first, to understand the system management requirements, the ISS400 system was studied and the reasons for difficulties in version and configuration management were examined[1] ; then, the ISS400 world was generalised and formalised to achieve a description of the real world to be modelled ; to propose improvements of current processes, the SEIM (System Evolution and Integration Management) conceptual model was elaborated from works of Paris-Dauphine University researchers [Cellary, Jomier, 1990], [Blin, et al., 1992] and applied to ISS400 ; the advantages and the limitations of this model for managing ISS400 system were analysed. This approach is summarized in figure 2.

[1] The reader may consult the report of a post-graduate student, R. Boudouda "Propositions pour gérer les versions et les configurations dans un système informatique complexe; application au cas ISS400", Lamsade/Paris-Dauphine University, D.E.A. 127, December 1993.

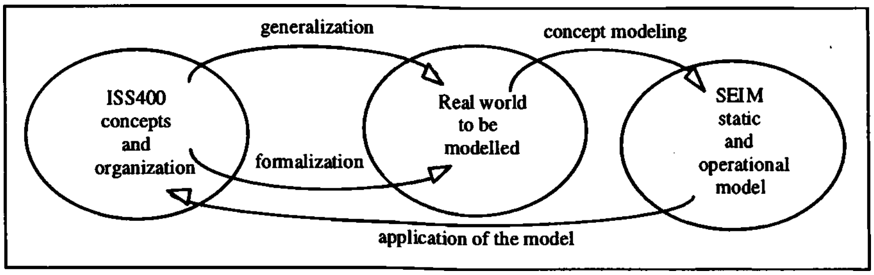

generalization
concept modeling
ISS400 concepts and organization
formalization
Real world to be modelled
SEIM static and operational model
application of the model

Figure 2 The approach used to elaborate and validate the SEIM model

This paper presents in section two the ISS400 system, its organization and the current processes in version and configuration management. Section three describes the formalised and generalised real world to be modelled and the SEIM conceptual model. Section four studies its application to ISS400. Section five expounds the improvements provided by a configuration management system based on the SEIM conceptual model and proposes some additional useful work.

## 2. ICL ISS400 Retail Point of Sale System

This section details the problems and the practices in version and configuration management posed by ISS400. Section 2.1 presents the main aims of ISS400 system and their consequences for version and configuration management. In section 2.2, the challenges of the ISS400 development teams are considered plus the consequences for the internal structure of the system and the process of creation of system versions. Section 2.3 discusses the current tools used for version and configuration management with their limitations.

### 2.1 The main aims of ISS400 system

The main principle behind ISS400 is to provide a single system capable of being enhanced and adapted to make it suitable for all types of retail outlet and business operation such as speciality retail outlets, hypermarkets, department stores etc. The aim is to produce a specific customer system version from a library of components, through a process of customisation and component selection.

The library comprises over fifteen thousand different code modules, each with about ten versions, together with a wide choice of hardware components and many interdependencies between code and hardware modules. Thus an immediate need arises for a configuration management method to generate a specific customer system version that works first time.

The ISS400 production and delivery involve the following categories of elements: hardware, operating system and environment software, base software, application software, documentation, services.

The hardware elements include: servers, workstations, tills, networks, power supply units, etc. For example, a till is composed of: main board, secondary boards, keyboard, displays, printers, scanners, cables and interface black boxes.

The software elements include several types of files: source code, data, executable code, screen formats etc.

The documentation includes external customer manuals and internal development specifications: requirements, design, test plans etc.

The service modules include installation, support, consultancy and training. Each service module requires specific skills and support documentation (e.g. installation notes, training materials).

Each element may be subdivided and several breakdown levels are needed. Apart from the breakdown structure, the elements have several *versions* related to their history, and several *variants* related to the language, country market or customer specific requirements. Versions are related to important changes. Minor changes lead to minor versions called *releases.* A variant is created from a version or from another variant and can have releases. Links between versions, variants and releases are recorded in informal text files.

For example in figure 3, the element named "TA" (which means *Till Application*) has an initial version identified "TA_IT_10.0". This element has subsequent releases due to bug fixing identified as "TA_IT_10.1" and "TA_IT_10.2" . Finally, a new version "TA_IT_11.0" has been created due to important functional changes. In the meantime, a variant identified "TA_XX_10.0" has been created from the initial version for the specific customer "XX". This variant subsequently evolves into releases "TA_XX_10.1" and "TA_XX_10.2". If required, a manual merge will be done between version "TA_IT_11.0" and release "TA_XX_10.2" to create a new variant "TA_XX_11.2" which will include both generic element improvements and customer specifics. Links between versions and releases created from version "TA_IT_10.0" are recorded in the informal text file "TA_IT.ho". Those between variant and releases created from the variant "TA_XX_10.0" are recorded in the text file "TA_XX.ho".

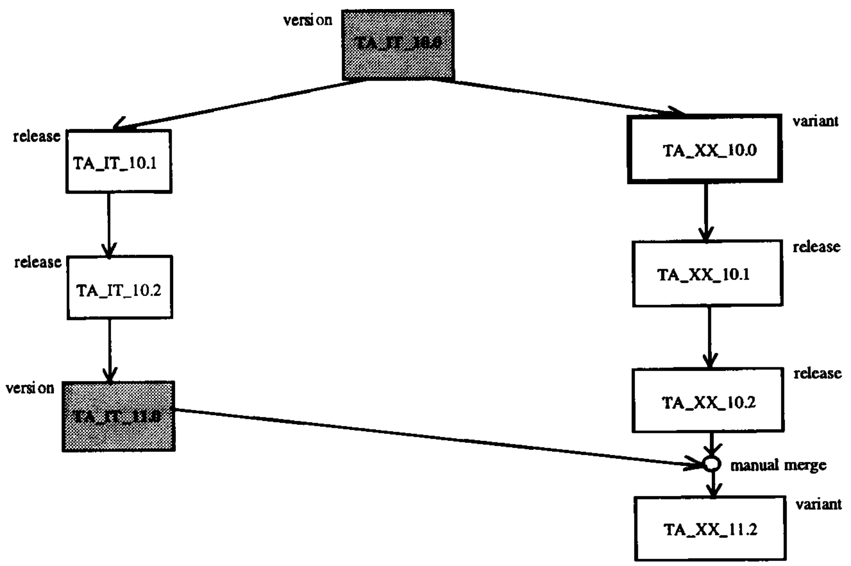


Figure 3 Example of versions, variants, and releases of an element of ISS400 system.

Usually, variants are handled by separate customer specific teams and versions are managed by a central development team.

The element versions may have multiple dependency constraints which are particularly strong between the hardware, base software and application software. The hardware elements are in rapid evolution driven by the state of the market. If a hardware element version is modified, the related software element versions have also to be changed. In addition, the software elements change under new customer requirements and due to bug fixes. The documentation and services are impacted also by the hardware and software evolution.

A system version for a specific customer involves some complex subset of element versions. Generally, a specific system version is called a *baseline* if it is considered as a reference by the system manager for creating new system versions for new customers. From a baseline, new system versions can be created to introduce successive minor changes. Eventually, a new baseline will be created if important changes have to be made. All the links between successive system versions and between successive baselines are recorded in informal text files called *release notices*. For example, in figure 4, from the baseline identified "V1L5.7", two specific system versions are created for customers "XX" and "YY". Subsequent minor changes for customer "XX" lead to new system versions identified "XX_01_V1L5.7" and "XX_02_V1L5.7". In the meantime, a new baseline "V1L6.0" is created from "V1L5.7" including major functional improvements. As the customer "XX" wants to take advantage of the new improvements, the modifications introduced in the system version "XX_02_V1L5.7" are added manually to the baseline "V1L6.0" to produce the new system version "XX_02_V1L6.0". The customer "YY" does not want to include improvements of the baseline "V1L6.0". The links between the successive system versions created for the customer "XX" and those between baselines are recorded in two different release notices.

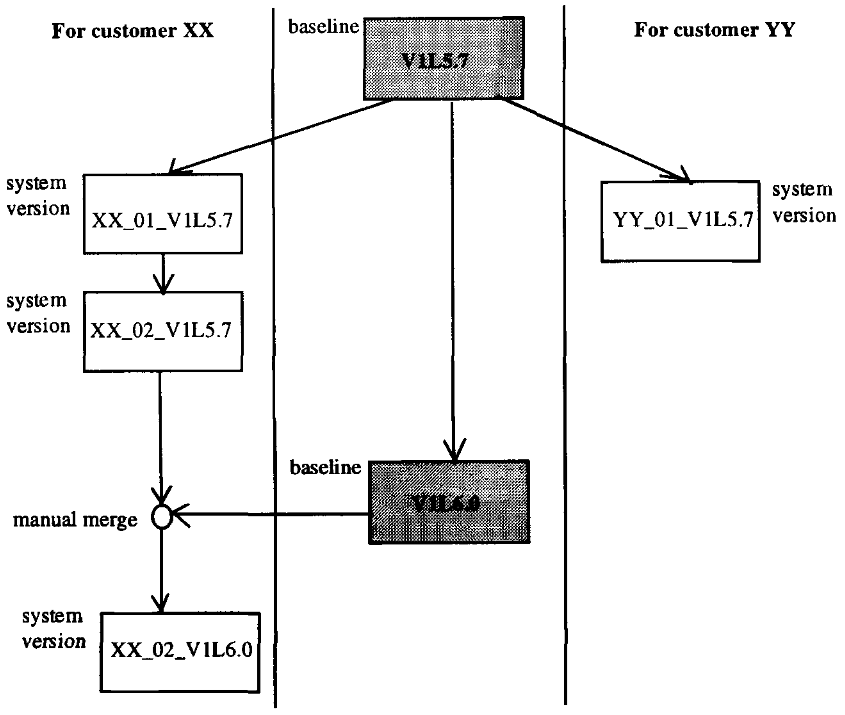


Figure 4 ISS400 baselines and system versions

Customer system versions are handled by customer specific teams. Baselines are managed by the central development team.

## 2.2 Challenges

Today, the ISS400 development teams have to meet two challenges. First, environments continually change: new code modules, new PC models and specific peripherals are created, bugs are corrected, etc. It is essential to integrate changes in the existing customer solutions without modifying the overall system behaviour. Secondly, to meet other specific requirements and to provide ongoing maintenance, it must be possible to change or modify any component within the system or to add new components with the minimum of effort.

The implications of changing are an integration and validation exercise of the whole system to ensure consistency and integrity of the solution. More importantly, during an installation (or rollout period) which could span several months, maintaining detailed knowledge about which components can work together is essential. Maintenance, for example, is often carried out by a different organisation which perhaps does not have the same information relating to the specific components (for example, which driver to use for a new hardware device). This results in a potential problem of interfacing.

To reduce the number of elements to modify when changes occur, a well adapted structure for ISS400 system (in four layers) has been chosen (figure 5). First, all application software is independent of the operating system and of the hardware. An intermediate layer (base software) ensures this independence. For example, currently, ISS400 designers investigate the possibility of a switch from OS/2 operating system to NT. Modifications are limited just to the base software level. Secondly, as can be seen in figure 5, dependencies between elements exist only between elements at the same level or at neighbouring levels. New customer requirements or technology driven changes (for example, new PCs) lead to new system versions with the same structure.

The different layers are handled by different teams. Due to the general structure of ISS400 system, the global evolution effort and the mandatory interactions between teams are minimized.

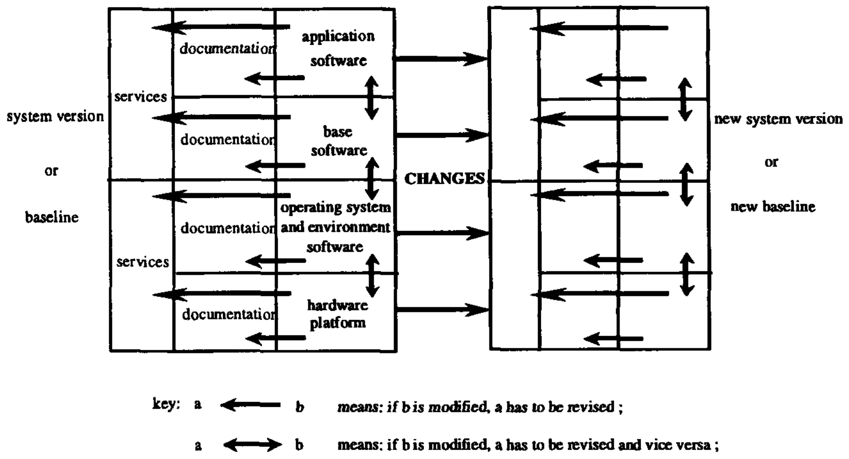

system version
or
baseline
services
documentation
application software
base software
operating system and environment software
hardware platform
CHANGES
new system version
or
new baseline
key: a ← b means: if b is modified, a has to be revised ;
a ↔ b means: if b is modified, a has to be revised and vice versa ;

Figure 5 ISS400 baselines and system versions structure

To create a new system version for a new or an existing customer (figure 6), each team creates a version of a related specific part of the system either by reuse, selection and customization of existing element versions or by development of new element versions. Each of these parts is verified in unit tests. The whole system is integrated and validated before installation on the pilot site.

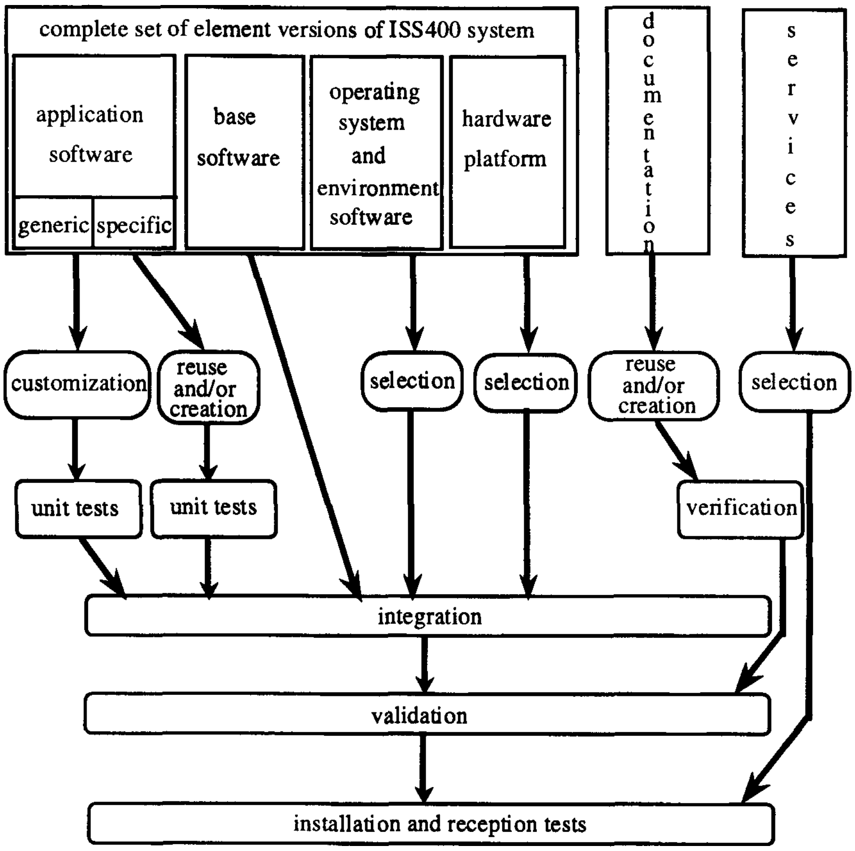


Figure 6 The process for creating a new system version

## 2.3 Current Version and Configuration Management Processes used for ISS400

Each element category (hardware, software, documentation, service) is handled *separately*, using separate methods and tools. Global coordination is done using standard management techniques (development plans and configuration documents, memoranda, meetings, release notes, etc.) rather than by a formalised process supported by a single software tool.

Dependencies are very common inside each category of elements and between elements of different categories, and their handling is tedious. This is the major source of potential incompatibilities.

This section focuses on handling software elements only, as it is a good example of the current practice used by the ISS400 team. The hardware, documentation and service configuration management practices are less formalised.

The basic software elements to be handled are: files and file packages. A file package is a group of files which are stored together related to a specific functionality. A higher level grouping is achieved through the sub-system level, e.g. Till Application sub-system or Electronic Funds Transfer sub-system.

Control of the current software version uses an industry standard configuration management tool PVCS [Sage, 1990]. The PVCS library is currently under conversion to PCMS [SQL, 1992].

PVCS is a simple tool handling file versions. It works like a librarian creating different versions of individual files using a "check-in/check-out" mechanism. The basic operations are PUT and GET a file into or from the library. For each operation a log record is maintained. In order to save disc space, only the increments are stored. The storage is in a file system directory structure defined by the user.

A common label can be assigned to a set of files of a given revision. This label allocation is used to group files and to handle versions and customer variants. As PVCS does not handle the package breakdown structure, another level of grouping is added using file lists in order to organise the files into packages. These file lists, called ".fil", representing packages, are controlled under the PVCS version mechanism. For some packages, a two level hierarchy is constructed in this way. The "file lists" mechanism is complemented by a complex directory structure representing the storage area of the packages, customers variants and baselines.

The configurations and the dependencies between files, packages and hardware elements are managed "manually", that is to say without any help from PVCS, using text files called release notes and handover notes. A release note is a document listing the packages and their versions used to build a complete system, including the major hardware/software dependencies. A handover note is a document attached to a package (or to a group of packages) listing the history of the package versions and the inter-dependencies with other related packages.

PCMS is a richer tool than PVCS. It allows breakdown structures to be represented and provides configuration management. It is implemented on top of an ORACLE relational database. The breakdown mechanism is based on a hierarchical structure of elements called *DESIGN-PARTS*. In ISS400, the DESIGN-PARTS implement sub-systems and packages. Each DESIGN-PART is assigned a version history. The DESIGN-PART variants are also supported by PCMS used to represent the customer variants. A DESIGN-PART has another level of breakdown into elements called *PRODUCT-ITEMS*. The PRODUCT-ITEMS are in fact source files.

Configuration management is implemented in PCMS through the *BASELINE* facility. A PCMS *BASELINE* is an inventory of the state of a system at the DESIGN-PART level or at the PRODUCT-ITEM level. It could be a complete released system, a test configuration, or a specific configuration subset. A BASELINE can be extracted from the database.

DESIGN-PART change management is done using a state transition mechanism; the possible main states are - open: approved: implemented: available: tested etc.

Finally, PCMS offers a human process support through a scenario facility, e.g. get the files you need, edit files, check the files, return the items, create new baselines, etc.

So, PCMS compared with PVCS gives the user a real advantage. But no support is provided to handle the dependencies between the DESIGN-PARTS, the DESIGN-PARTS variants, the PRODUCT-ITEMS and the BASELINES.

## 3. The SEIM conceptual model

The SEIM (System Evolution and Integration Management) conceptual model provides a solution for version control and configuration management of complex systems. Its main assignments are:

- to represent and to manage elements of different natures that may be composed of other elements and that may have dependency links between them,
- to represent and to manage element versions with dependency constraints between them,
- to represent and to manage versions of a whole system or versions of parts of a system with composition constraints,
- to assist the integration process,
- to manage customer system versions during installation and maintenance processes.

We explained the ISS400 concepts and organization in section 2. In section 3.1 the concepts are generalised and formalised to achieve a description of the real world to be modelled. In sections 3.2 and 3.3 the SEIM conceptual model is described in its static and operational aspects. Modelling of dependencies between element versions is presented briefly in section 3.4 and section 3.5 suggests a model for control activities.

### 3.1 The real world to be modelled

The teams who handle the system manipulate:

- elements which are indissociable units of information about hardware, software, services and documentation,

- links between elements which may have different meanings like composition, inclusion, usage, control or dependency,
- element assemblies which are an element, a subset of links involving it and all the other elements used by these chosen links, for example an element and all its components,
- element contents which constitute versions, variants and releases,
- links between contents of different elements which convey dependencies between contents of elements,
- contents of element assemblies which are an element content, a subset of links involving this element content and all the other element contents used by these chosen links, for example a content of an element and of all of its components,
- versions of a system or of parts of a system which are sets of element contents and links between them.

Each element, element assembly and system version has an identifier. The different contents of an element have, generally, the same structure. All the teams together manage only one system. So, a system has no identifier.

Figure 7 shows an oversimplified ISS400 system which contains two software elements "Till Application" and "FORTE"[2] and three hardware elements "Point Of Sale", "CPU board" and "Pin Pad". "FORTE" is used by "Till Application". "CPU board" and "Pin Pad" are components of "Point Of Sale". An element assembly, identified "Point Of Sale", has been defined comprising the component "Point Of Sale", related composition links and the component elements "CPU board" and "Pin Pad". The system is managed by two teams, the first one is responsible for the software part of the system and the second one for the hardware part. Each team has created a version of its part of the system. The version of the software part, identified "V1L5.3", contains the version "TA_IT_10.0" of the element "Till Application" and the version "3.1" of the element "FORTE". The version of the hardware part, identified "Strasbourg Store", contains the version "9520/150 R3" of the element "Point Of Sale", the version "R3" of the element "CPU board" and the version "DASSAULT LCM 103" of the element "Pin Pad".

[2] Nothing in common with the international hotel group of that name.

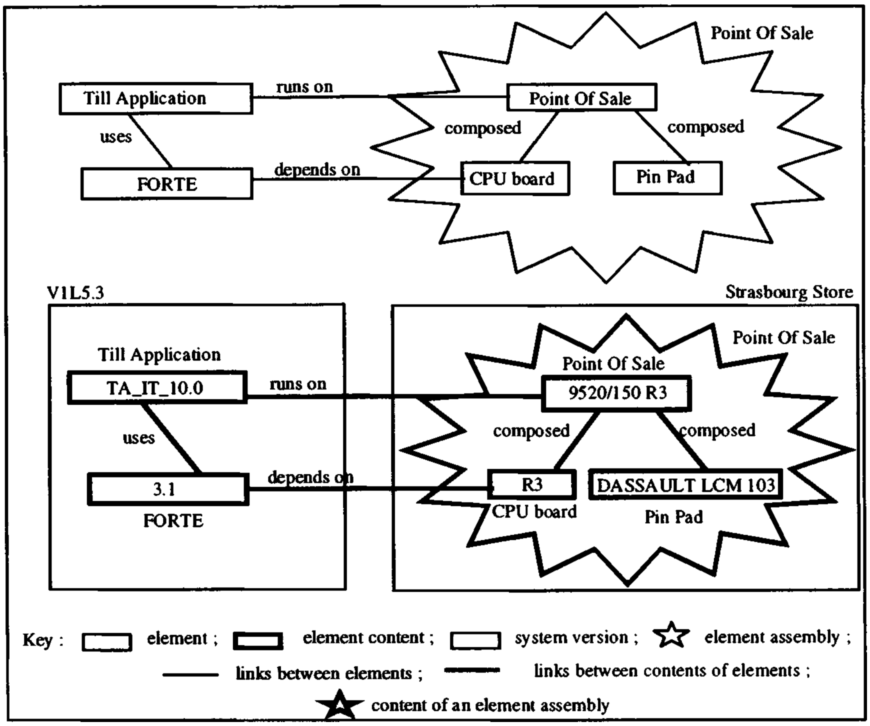


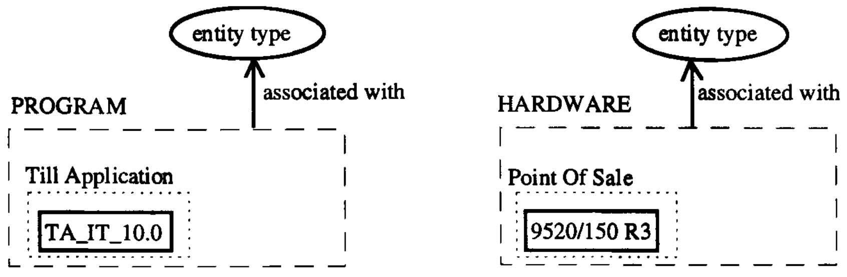
Figure 7 The real world to model

## 3.2 The SEIM conceptual data model

The SEIM conceptual data model is an improved entity-relationship model [Chen, 1976]. It contains the entity-relationship model concepts of *entity* and *entity class*, *association* and *association class*, and *aggregation*. It adds the concepts of entity, association and aggregation version and the concept of configuration and configuration class. On top of that, it associates a *type* with entity, association and configuration classes.

A *version of an entity* is a value. An *entity* is a set of entity versions associated with an identifier. An *entity class* is a set of entities associated with an identifier and with a type which defines the structure of the versions of each entity belonging to the entity class, a set of constraints and a set of possible operations on the entities and on the entity versions. Several entity classes may be associated with the same type. Each entity belongs to one and only one entity class.

Figure 8 represents two entity classes identified "PROGRAM" and "HARDWARE". The entity class "PROGRAM" has one entity identified "Till Application" which has itself one version "TA_IT_10.0". The entity class "HARDWARE" also has one entity identified "Point Of Sale" which has itself one version "9520/150 R3".



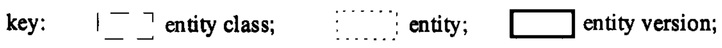


Figure 8 Entity classes, entities, entity versions

A *version of an association* is a named link between versions of different entities. An *association* is the set of all the association versions with the same name existing between versions of entities. An association is identified by the name of the association versions composing the association and the identifiers of the entities linked. An *association class* is the set of all the associations with the same name existing between entities of two different classes or of the same class. An association class is identified by the name of the associations composing the association class and the identifiers of the entity classes. It is associated with a type which defines the possible classes of the linked entities and constraints on the associations and on the association versions like cardinality constraints or nature of the dependencies between versions of linked entities. Several association classes may be associated with the same type. Each association belongs to one and only one association class.

Figure 9 represents an association class "runs on" between entity classes "PROGRAM" and "HARDWARE". This association class contains one association between the entity "Till Application" of the class "PROGRAM" and the entity "Point Of Sale" of the class "HARDWARE". This association has one version between the version "TA_IT_10.0" of the entity "Till Application" and the version "9520/150 R3" of the entity "Point Of Sale".

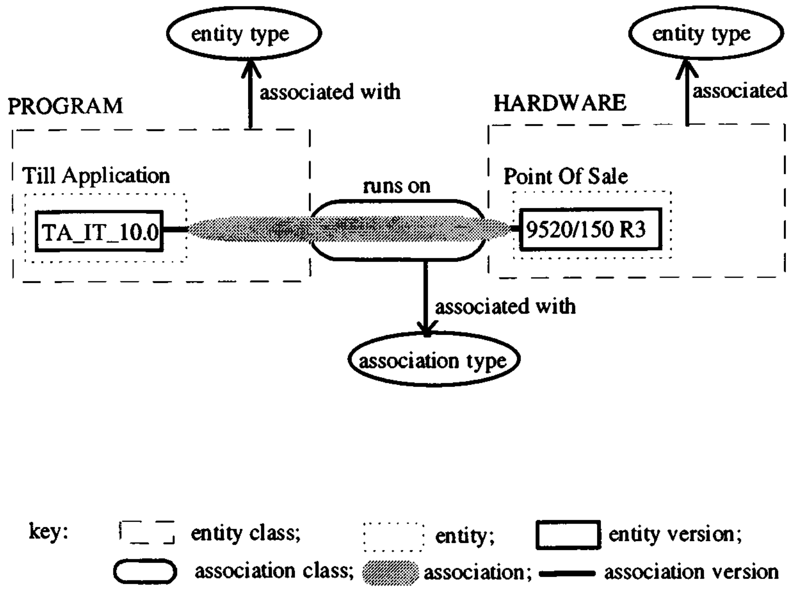


Figure 9 Association classes, associations, association versions

An *aggregation class* is a graph where each node is an entity class and each edge is an association class. It is associated with an identifier. An *aggregation* is a graph where each node is an entity and each edge is an association. It is associated with an identifier and belongs to one and only one aggregation class. An aggregation can be seen as an instance of the aggregation class. An *aggregation version* is a graph where each node is an entity version and each edge is an association version. The concepts of aggregation class, aggregation and aggregation version allow one easily to manipulate linked entity classes, linked entities and linked entity versions with a single operation like displaying or deleting an aggregation version.

Figure 10 represents two entity classes "HARDWARE" and "HARDWARE Component". The first one contains one entity "Point Of Sale" which has itself one version "9520/150 R3", the second one has two entities "CPU Board" and "Pin Pad". The entity "CPU Board" has one version "R3". The entity "Pin Pad" also has one version "DASSAULT LCM 103". An association class "is composed" links the two entity classes. An association links the entity "Point Of Sale" to the entities "CPU Board" and "Pin Pad". An association version links the version "9520/150 R3" of the entity "Point Of Sale" to the version "R3" of the entity "CPU Board" and to the version "DASSAULT LCM 103" of the entity "Pin Pad". An aggregation class "Hardware composition" has been defined. Its nodes represents the entity classes "HARDWARE" and "HARDWARE Component". Its edge represents the association class "is composed". The aggregation "Point Of Sale" contains the entities "Point Of Sale", "CPU Board" and "Pin Pad" on its nodes and the instance of the association class "is composed" linking these entities, on its edges. An aggregation version contains a version of each entity on its nodes and the association version linking the entity versions on its edges.

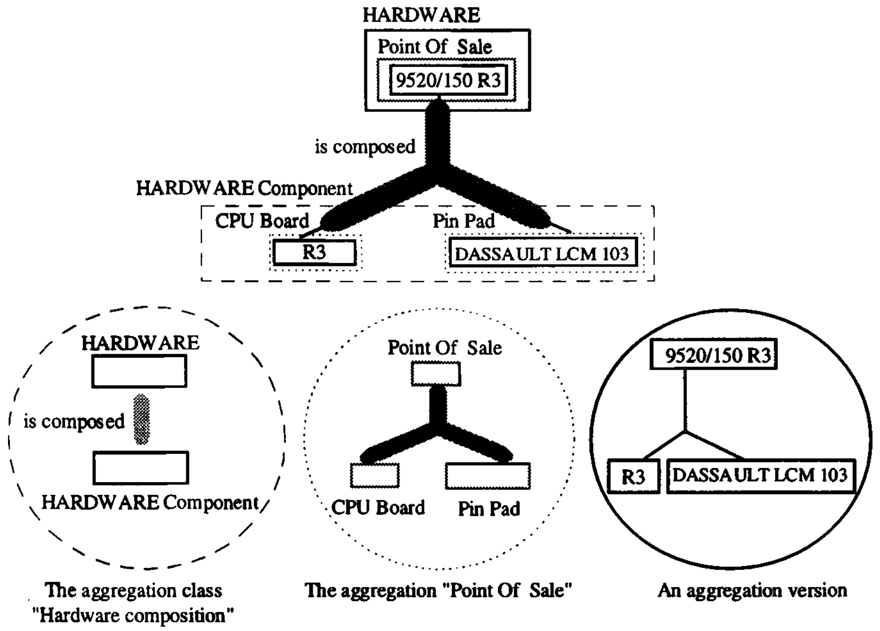


Figure 10 Aggregation classes, aggregations, aggregation versions

A *configuration* is a set of entity versions with, at most, one version per entity. It is associated with an identifier and one or several values of attributes which provide customer information on the configuration such as author or creation date. A *configuration class* is a set of configurations associated with an identifier and with a type which defines the structure of the attributes of each configuration belonging to the class, constraints on the composition of the configurations and a set of possible operations on the configurations. Several configuration classes may be associated with the same type. Each configuration belongs to one and only one configuration class.

Figure 11 shows two configurations classes "hardware configurations" and "site configurations". The first class contains one configuration "X hardware". The second class contains one configuration "X site".

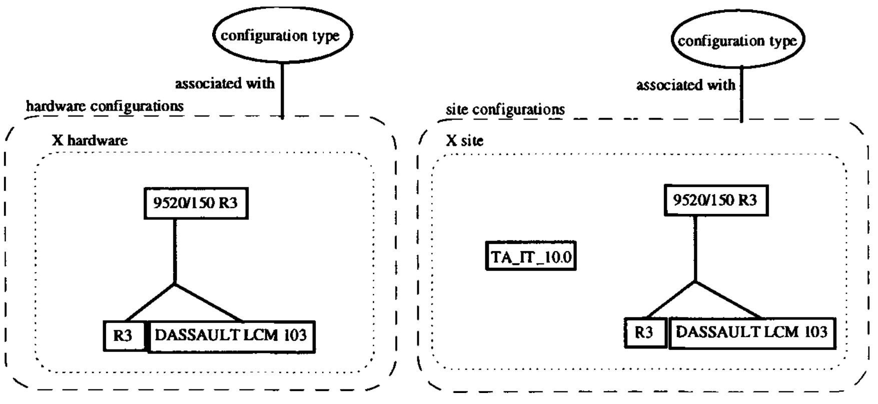


Figure 11 Configuration classes and configurations

The modelled world is a set of entities and a set of configurations such that each entity version belongs to one and only one configuration. Entity versions in different configurations may have the same value. For example, in figure 11, the versions of the entity "Point Of Sale" in the configurations "X hardware" and "X site" have the same value "9520/150 R3".

## 3.3 The operational model

The SEIM operational model offers base operations on entities and entity versions, on associations and association versions, on aggregations and aggregation versions and on configurations. In addition, it proposes a process model to create and to manage system versions and to maintain them in a cooperative manner.

### 3.3.1 Base operations on entities and entity versions, on associations and association versions, on aggregations and aggregation versions and on configurations

The configuration management system provides a set of base operations on entities and entity versions, on associations and association versions, on aggregations and aggregation versions and on configurations that can be referred in the entity, association and configuration types created by the users. The main operations are:

- on the entities: creation of an entity in an entity class, deletion of an entity, finding all the versions of an entity, finding all the entities of a class,
- on the entity versions: creation, deletion, modification of the value of an entity version in a configuration, comparing two versions of an entity in two configurations,

- on the associations: creation of an association in an association class, deletion of an association, finding all the versions of an association, finding all the associations of a class,
- on the association versions: creation, deletion, modification of an association version in a configuration, comparing two versions of an association in two configurations,
- on the aggregations: creation of an aggregation in an aggregation class, deletion of an aggregation, finding all the versions of an aggregation, finding all the aggregations of a class,
- on the aggregation versions: comparing two versions of an aggregation in two configurations,
- on the configurations: creation of a configuration in a class, deletion of a configuration, finding all the configurations of a class, displaying the content of a configuration, validating a configuration (verifying all the constraints defined on the content of the configuration), finding all the configurations containing a version of an entity with a specific value, comparing two configurations.

More complex operations explained in the following section allow the creation of a new configuration from one or several existing ones. These operations are derivation, extraction, integration and merge.

### 3.3.2 Creating, modifying and managing configurations

The operational model implicitly provides a type of configuration and a configuration class called *reference configurations.* The configurations of this class may contain a version of any existing entity. The only possible operation on reference configurations is displaying their content. A reference configuration cannot be modified (i.e. any entity version contained in a reference configuration cannot be modified or deleted), and the configuration cannot be deleted.

Suppose all the teams in charge of creating the entity versions composing the first versions of the system have terminated their work. The integration team (called designer/integrator in figure 12) is going to create the reference configurations representing each of the first versions of the system. After defining all the types and all the classes of entities, it will create one or several reference configurations by using the operation of creating a new configuration or the operation of deriving a new configuration from an existing one. The second operation allows a new configuration to be created containing entity versions with the same values as in the existing configuration. The user can, then, modify entity versions in the new configuration. Each reference configuration represents one complete and validated version of the system. It is available for all the teams which have to work on the system (the different software teams, the different hardware teams, the installation team, the documentation team) for creating versions of the system for new customers or for maintaining existing versions.

These different teams are not interested in all the entities composing the system. Thus, different types and classes of configurations will be defined to describe configurations of interest to each of them. To work on a version of the system, a team will create a new configuration in a specific class from the reference configuration corresponding to the interesting version of the system. This operation is called *extraction.* The new configuration created will contain only entities belonging to entity classes allowed by the configuration type (figure 12).

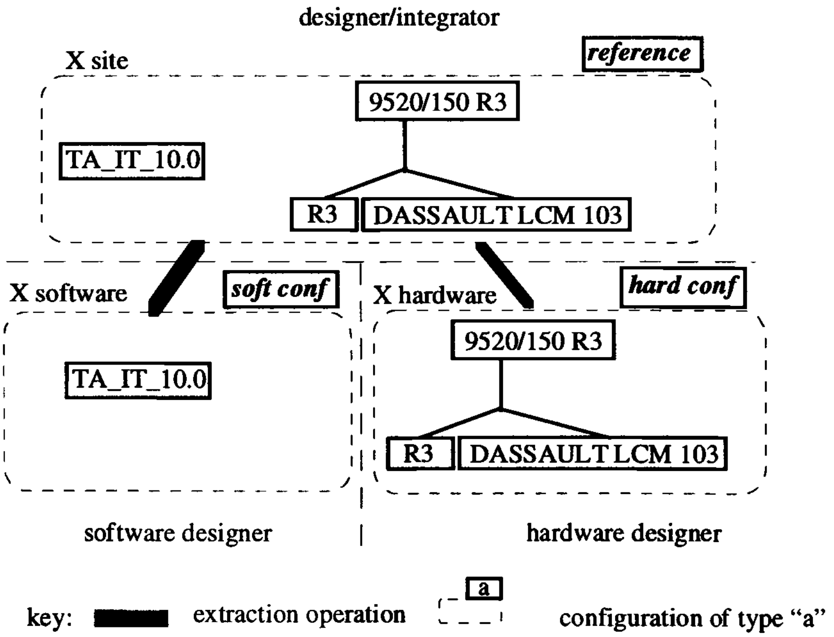


Figure 12 Configurations created by extraction

After creating a configuration by the extraction operation, a team may modify this configuration in adding entity versions, in modifying the value of entity versions or in deleting entity versions. It may also derive a new configuration.

### 3.3.3 Integrating configurations

To maintain a version of the system or to create a new system version for a new customer, each team builds the part of the system version coming within its remit and an integrator (called designer/integrator in figure 13) puts together the different parts. For that, each team builds a configuration representing the part of the version of the system it has to construct using the process explained in the section 3.3.2. Perhaps, all parts of the version of the system do not have to be changed. Then, an integration operation takes place between the original reference configuration $R_j$ and one or several configurations $C_i$ of other classes. Its result is a new reference configuration $R_k$ containing updated parts of the version of the system plus parts not modified. The general integration process is represented figure 13.

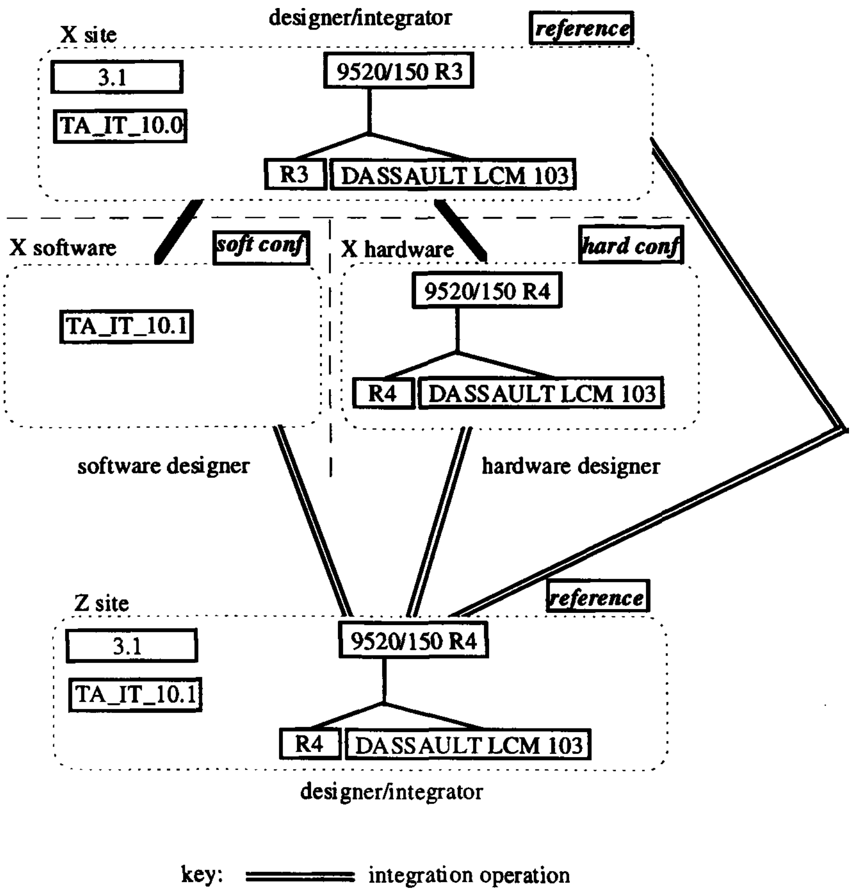


Figure 13 Configurations created by integration

## 3.4 Modelling dependencies between versions of different entities.

Complex systems contain a great number of dependencies between entities and between versions of different entities. During maintenance of a system, new entity and new entity versions are created and the number of dependencies grows. When a great number of entities exists, each with several versions, it is very difficult to manage all the dependencies manually. The difficulty is increased when the dependencies concern entity versions managed by different teams, for instance, between software and hardware. When a team creates a new configuration or modify a configuration, it is its responsibility to choose the right versions of entities to put together. To validate a configuration resulting from an integration operation, dependencies between versions of entities belonging to different prime configurations have to be solved.

Therefore, we studied a model of dependencies between versions of different entities so that the dependencies can be easily and automatically verified during the maintenance process.

At the beginning of the 90s, relational database management systems (DBMS) were improved by a trigger mechanism which allows the execution of specific programs in response to events. The active rule concept, then, generalised the trigger mechanism. It uses the Event-Condition-Action (E-C-A) formalism: when the event "E" happens, if the condition "C" is verified, the action "A" is executed. Most of the applications of the active rules are in DBMS [AFCET, 1993] for executing integrity constraints and calculating derived attributes. [Belkhatir, Estublier, 1990] introduce active rules in a software engineering environment to define an action to execute when a specific event arrives. These rules are connected to object types or to association types and are applied to every object or every association of the type.

To model dependencies in complex systems, we propose *behaviour rules* inspired by E-C-A rules. Behaviour rules are connected to entity types or to entities and concern operations executed on entities and entity versions. A rule can be executed before or after the execution of the related operation. The syntax of a behaviour rule is:

```
("BEFORE" | "AFTER"), <operation1>, [(".", entity
identifier)], [<conditions>], <message>, <"@",
association type>, <time>, [(".", (entity identifier|
configuration identifier)]
```

[ ] means optional.

```
<operation1> ::= (operation name, execution
parameters) | <operation1>, "|", (operation name,
execution parameters).
```

Each pair *(operation name, execution parameters)* is defined in the entity type concerned by the rule.

| | |
|---|---|
| *<conditions>* | expresses conditions on the value of the entity version before or after (depending on the first word of the rule) the execution of *<operation1>* |
| *<message>* | is a name and the execution parameters of an operation to be executed on the entity referred by *<association type>* if *<conditions>* is verified. |

| | |
|---|---|
| *<time>::=* | *"IMMEDIAT"\| (<operation2>,["."", (entity identifier \|configuration identifier) )* |
| | *IMMEDIAT* means that the operation indicated in*<message>* will be executed immediately. |
| *<operation2>* | is a name and the execution parameters of an operation. This option means that the operation indicated in *<message>* will be executed when *<operation2>* is called. *<operation2>* may concern a specific entity or a specific configuration |

Here are two modelled dependencies:

1) This rule is connected to the entity "FORTE". It requires that, after the execution of an operation "modify" on a version of this entity, if the attribute "id-release" of the entity version is modified, the operation "modify" has to be executed on the entity version referred by the association of type "depends_on" (in figure 7, a version of the entity "CPU board") at integration time.

```
AFTER modify ( ).FORTE, if id-release.FORTE # id-
release.@bak, modify (id-release.FORTE), @depends_
on, integration
```

2) This rule is connected to the entity "PA". It commands that, after the creation or the modification of a version of the entity "PA", if the version of this entity (identified by the attribute "id-version") is "PA_7.04" and if any version of the entity "3GLIF" exists in the configuration (3GLIF = "nil"), then the version "3GLIF_3.19" of the entity "3GLIF" will be put in the configuration. This rule will be executed at the validation of the configuration. Of course, this rule has to be completed by another one defining actions to execute if there already exists a version of the entity "3GLIF" in the configuration.

```
AFTER creation().PA | modify().PA, if id-version.PA =
"PA_7.04" and if 3GLIF = "nil", copier.PA (id-version
= "3GLIF_3.19"), validation
```

## 3.5 Extension of behaviour rules for modelling process control activities.

The verification of dependencies between versions of different entities sometimes requires some specific actions. For example, to verify the first rule in the last section, each time a new configuration of type "development" is created, the operation "savevalue" has to be executed on the version of each source program. This operation, for example, creates a new entity version linked by an association of type "bak" for each version of entity of type "source", in the configuration. The new entity version will have the same value as the original one. This action can be modelled by the following rule:

```
        AFTER creation ( ) |derivation ( ) |extraction ( ), ,
                    savevalue ( ).source, IMMEDIAT
```

The rule is connected to the configuration type "development". There is no condition for execution of the operation "savevalue".

Rules attached to configuration types have the following general syntax:

```
                ("BEFORE"|"AFTER"), <operation1>, [(".",
              configuration identifier)], [<conditions>],
         <message>,  [(".", (entity identifier| configuration
         identifier| entity type| configuration type)], <time>
```

## 4. Application of the SEIM model to ISS400

In the last paragraph, the SEIM conceptual model for version and configuration management has been described. We show, in this paragraph, the application of the model to ISS400. ISS400 is composed of four kinds of components: software, hardware, documentation and services. We considered each of these parts separately and for each ISS400 concept, we studied its modelling with the SEIM model. In this paper, we present the modelling only of hardware and software parts. Modelling of documentation and services do not introduce any specific difficulty.

Table 1 Modelling of ISS400 hardware concepts

| **ISS400 hardware concepts** | **Modelling** |
|---|---|
| Package, for example a Pos terminal | represented by an aggregation composed of an entity of type "package", entities of type "element" or "module" and associations of type "composition". The root of the aggregation is the entity of type "package". This entity type defines three attributes called *id-version, id-release, id-variant.* |
| Element, for example a keyboard | represented by an aggregation composed of an entity of type "element", entities of type "module" or "component" and associations of type "composition". The root of the aggregation is the entity of type "element". This entity type defines the three attributes *id-version, id-release, id-variant.* |

Table 1 cont'd. Modelling of ISS400 hardware concepts

| **ISS400 hardware concepts** | **Modelling** |
|---|---|
| Module, for example a cable | represented by an aggregation composed of an entity of type "module", entities of type "component" and associations of type "composition". The root of the aggregation is the entity of type "module". This entity type defines the three attributes *id-version, id-release, id-variant.* |
| Component, for example a plug | represented by an entity of type "component" which defines the three attributes *id-version, id-release, id-variant.* |
| Version of a package, of an element, of a module (to introduce a functional change) | represented by an aggregation version. The value of the attribute *id-version* of the root entity version identifies the version from the user point of view. |
| Version of a component | represented by an entity version and its attribute *id-version* as above |
| Variant of a version of a package, of an element, of a module (to introduce specific changes depending on the country, on the language, on the customer) | represented by an aggregation version. The value of the attribute *id-variant* of the root entity version identifies the variant from the user point of view. It is composed of three parts: country, language, customer. |
| Variant of a version of a component | represented by an entity version and its attribute *id-variant* as above |
| Release of a version or of a variant of a package, of an element, of a module (to correct bugs) | represented by an aggregation version. The value of the attribute *id-release* of the root entity version identifies the release from the user point of view. |
| Release of a version or of a variant of a component | represented by an entity version and its attribute *id-release* as above |

Table 2 Modelling of ISS400 software concepts

| **ISS400 hardware concepts** | **Modelling** |
|---|---|
| Hardware system or sub-system | represented by a configuration of type "hardware" or of another specific type. Attributes of the configuration allow the user to record information about the system or the sub-system. |
| Package for example "Till application" | represented by an aggregation composed of an entity of type "package", entities of type "object" or "library" or "data structure file" or "code file" and associations of type "composition". The entity type "package" defines three attributes called *id-version, id-release, id-variant.* |
| Object for example "ADMIN" (one the "Till application" object) | represented by an aggregation composed of an entity of type "object", entities of type "data structure file" or "code file" and associations of type "composition". The entity type "object" defines the three attributes *id-version, id-release, id-variant.* |
| Library for example "ADI" (network access library) | represented by an aggregation composed of an entity of type "library", entities of type "code file" and associations of type "composition". The entity type "library" defines the three attributes *id-version, id-release, id-variant.* |
| Data structure file for example "DISCOUNT.DF" | represented by an entity of type "data structure file" which defines the three attributes *id-version, id-release, id-variant.* |
| Code file for example "DSCMENU.P" | represented by an entity of type "code file" which defines the three attributes *id-version, id-release, id-variant.* |

Table 2 cont'd. Modelling of ISS400 software concepts

| **ISS400 hardware concepts** | **Modelling** |
|---|---|
| Version of a package, of an object, of a library (to introduce a functional change)<br><br>Version of a data structure file, of a code file<br><br>Variant of a version of a package, of an object, of a library (to introduce specific changes depending on the country, on the language, on the customer<br><br>variant of a version of a data structure file, of a code file<br><br>release of a version or of a variant of a package, of an object, of a library (to correct bugs)<br><br>release of a version or of a variant of a data structure file, of a code file | as for hardware |
| software system or sub-system | represented by a configuration of type "software" or of another specific type. Attributes of the configuration allow the user to record information about the system or the sub-system. |

Examples of operations that users need are:

- on entities

  create, print, delete and modify attributes, constraints and operations of an entity

- on aggregations

  create and delete an aggregation,
  print the identifiers of all the entities of an aggregation,
  find to which aggregation a specific entity belongs, for instance to find which packages contain a specific file

- on entity versions

  create, modify and delete an entity version in a configuration,
  print the value of an entity version in a configuration

- on aggregation versions

  find to which aggregation version a specific entity version belongs, for instance to find which package versions contain a specific file version,

  for each entity version of an aggregation, to print its entity identifier and the value of the attributes "id-version", "id-variant" and "id-release"

- on configurations

  create, merge, compare configurations,

  print the identifiers and the content of attributes id-version, id-release, id-variant of each entity version contained in a configuration.

## 5. What next?

This paper has presented version and configuration management issues in complex systems, especially in ISS400 system, and has proposed a model satisfying the needs of teams in charge of integrating, maintaining and managing different versions of a system. This model provides means to define detailed and aggregated elements and element versions, to manage a great number of configurations which share many element versions, to create different parts of a system version independently and to integrate them to constitute a complete system version, to represent dependencies between versions of different elements and to model the process control activities.

A version and configuration management system based on the SEIM model could be used simultaneously by teams which handle detailed component elements, for example, the hardware team which manipulates very small elements like cables and plugs, and teams only interested in aggregate elements, for example, the installation team which manipulates big elements like a server or a till without wanting to know what they are composed of. This allows simple updating of element versions since each element version exists only once in the common database used by all the teams.

The model represents separately elements and links between elements, element versions and links between element versions. This allows one to modify links without modifying elements or element versions. For example, in a configuration α, the till contains a complementary screen, so in this configuration, the version of the element "till" is linked to a version of the element "complementary screen". In a configuration β, the till has no complementary screen. So, in the configuration β, no version of the element "complementary screen" and no link exist but the value of the element "till" remains the same. Another example comes from integration and installation configurations. An installation configuration is created from an integration configuration but some links which do not exist in the integration configuration are added in the installation configuration such as links between tills and an HSI channel and between tills and their specific driver.

A prototype tool which implements the SEIM model is in progress.

Future work concerns:

- Constraints. Constraints can be connected to the appropriate level of the model: entity type, entity, entity version, association type, association, association version, configuration type, configuration. Solutions are provided to represent dependencies between versions of different elements. But other types of rules have to be studied to represent dependencies between configurations or configuration composition constraints. The System maintenance process also obeys work rules, for example, an integration procedure has to be preceded by a validation operation of configurations concerned and has to be followed by an operation which parameterizes the customer system version or, when a new version of a particular hardware is created by the hardware team, the software team has to be informed because it has to create an adapted version of a specific program.

  Future work will clarify the different types of rules and will extend the rule grammar to express all types of constraints to manage. The rule verifying process which considers delayed verifications and verifications which may or must not block the work process will also be studied.

- The mathematical formalization of the proposed model. It will allow a synthetic definition of the different concepts and operations and setting up systematic model validation.

- Use of the model to represent and to manage complex systems of other areas like branch banking and manufacturing.

## References

AFCET. *Actes de la journée "Bases de donnée actives"*, Paris, 8/12/1993.

AGRAWAL, R., BUROFF, S., GEHANI, N. SHASHA, D., in *Object Versioning in Ode.* Proceedings of IEEE 7th International Conference on Data Engineering, pp.446-455, 1991.

BELKHATIR, N., ESTUBLIER, J. *Un outil pour la gestion des logiciels : ADELE2.* in Bigre 71, IRISA, Campus de Beaulieu F-35042 Rennes France, pp.13-29, November, 1990.

BLIN, M.J., CELLARY, W., JOMIER G., in *A Model of Configurations for Hardware/Software Deliveries.* Proceedings of the International Conference on Software Engineering, pp.795-806.Toulouse, France, December, 1992.

BUCKLEY, F.J., *Implementing Configuration Management: Hardware, Software and Firmware.* IEEE Computer Society Press, 1992.

CELLARY, W., JOMIER, G. *Consistency of versions in object-oriented databases,* in Proceedings of the International Conference on Very Large Databases, Brisbane, Australia, pp.432-441, 1990.

CHEN, P.P. *The Entity-Relationship Model - Toward a unified view of Data.* ACM Transactions on Database System Vol. 1 No. 1, pp.9-34, 1976.

COHEN, G. *MCM: un gestionnaire de Configurations personnalisable.* in Bigre 71, IRISA, Campus de Beaulieu F-35042 Rennes, France, pp.57-64, November, 1991.

COHEN, E., SONI, D., GLUECKER, R., HASLING, W., SCHWANKE, R., WAGNER, M. *Version management in Gypsy,* in Proceedings of the ACM-SIGSOFT-SIGPLAN Symposium on Practical Software Development Environments, Boston, MA, pp.201-215, 1988.

COMBES, J.F., *ANDROMEDE: système de gestion et de développement de logiciel*. ESLOG 26 rue Paul bert 92140 Clamart, France, 1990.

FELDMAN, S.I., *Software Configuration Management: Past Uses and Future Challenges,* in Proceedings of the Third European Software Engineering Conference, ESEC'91, Milan, Italy, pp.1-22, October, 1991.

FIEUX, B., *Gestion de configurations: Historique et besoins utilisateurs.* in Bigre 71, IRISA, Campus de Beaulieu F-35042 Rennes, France, pp.7-12, November, 1990.

HÔ XICH-TUÊ. *SPMS+: un outil pour gérer le "developmental baseline",* in Bigre 71, IRISA, Campus de Beaulieu F-35042 Rennes, France, pp.65-70, November, 1990.

KAFER, W., SCHONING, H. *Mapping a Version Model to Complex-Object Data Model.* Proceedings of IEEE 8th International Conference on Data Engineering, Arizona, pp.348-357, 1992.

KATZ, R. *Toward an unified framework for version modeling in engineering databases.* ACM Computing Surveys, 22(4), pp.375-408, 1990.

KATZ, R., CHANG, E., BHATEJA, R. *Version modelling concepts for computer-aided design databases,* in Proceedings of the ACM-SIGMOD Conference, Washington D.C., 1986, pp.379-386, 1986.

KIM, W., BERTINO, E., GARZA, J. *Composite Objects Revisited.* in Proceedings of the ACM-SIGMOD Conference, Portland, OR, pp.337-347, 1989.

KRUCHTEN, P. *Gestion de Configurations dans l'atelier Rational.* in Bigre 71, IRISA, Campus de Beaulieu F-35042 Rennes, France, pp.45-56, 1990.

MALHER, A., LAMPEN, A. *An integrated toolset for engineering software configurations,* in Proceedings of the ACM-SIGSOFT-SIGPLAN Symposium on Practical Software Development Environments, Boston, MA, pp.191-200, 1988.

SAGE, *Polytron Version Control System Fundamentals Manuel d'utilisation PVCS sous MSDOS et OS/2*, Sage Software Inc., 1990.

SCIORE, E. *Versioning and Configuration Management in an Object-Oriented Data Model.* VLDB Journal, 3, pp.77-106, 1994.

SQL, *PCMS Quick Reference Guide for Unix and VMS Operating Systems.* SQL Software Ltd., June, 1992.

## Biographies

*Marie-José Blin*

Marie-José Blin teaches software engineering at Paris-Dauphine University and belongs to the Lamsade research laboratory of this University. She works in the "Database and software engineering" team and is particularly interested in version and configuration management and object-oriented software development methods and CASE.

Mail: blin@lamsade.dauphine.fr
Université Paris Dauphine, Place du Marechal de Lattre de Tassigny, Paris 16$^{e}$, France.

*Ian Puddy TD, BSc, CEng, MBCS, FISMM, GIMA*

Ian Puddy has spent most of his career in the last 20 years with ICL working across Europe mainly in Sales and Marketing. He was part of the team which initial established ICL in Retail. More recently he was directly responsible for the early ISS400 customer projects in Europe. Today he is based in France and responsible for business development within Industry Systems.

Mail: I.G.Puddy@icl1992.wins.icl.co.uk

*Justyn Lisicki*

After scientific studies, Justyn Lisicki has worked in information technology for several years, mainly in professional service area. Within ICL France, he manages software development teams and projects, and was involved in quality improvement process. He is interested in software engineering techniques and tools.

Mail: J.Lisicki@fra0104.wins.icl.co.uk

**SystemWise**

Systems Integration Information on CD

**Ingenuity**

The Technical Journal

**CustomWise**

Delight your customers with this new service to put your documentation on screen

**WebWise**

Maximise your business benefits with these six services to access the Internet and WorldWideWeb

**Multimedia** *Solutions*

ICL

**Telephone the Help Desk**

0181 565 7993

**Kiosks**

Customer-activated terminals to make your business more accessible

**CD***a*

Archive your system output to CD and save on storage and production costs

**Webkit**

Internet made easy with this simple package

**Architext**

The OPENframework Architecture Series on CD

# Index of Technical Journal Papers by Issue

# Ingenuity

## The ICL Technical Journal

## Guidance for Authors

## 1. Content

**Ingenuity**, the ICL Technical Journal, has an international circulation. It publishes high standard papers that have some relevance to ICL's business. It is aimed at the general technical community and in particular at ICL's users and customers. It is intended for readers who have an interest in the information technology field in general but who may not be informed on the aspect covered by a particular paper. To be acceptable, papers on more specialised aspects of design or application must include some suitable introductory material or reference.

**Ingenuity** will usually not reprint papers already published but this does not necessarily exclude papers presented at conferences. It is not necessary for the material to be entirely new or original. Papers will not reveal matter relating to unannounced products of any of the ICL Group companies.

Letters to the Editor and book reviews may also be published.

## 2. Authors

Within the framework defined in paragraph 1, the Editor will be happy to consider a paper by any author or group of authors, whether or not an employee of a company in the ICL Group. All papers are judged on their merit, irrespective of origin.

## 3. Length

There is no fixed upper or lower limit, but a useful working range is 4000-6000 words; it may be difficult to accommodate a long paper in a particular issue. Authors should always keep brevity in mind but should not sacrifice necessary fullness of explanation to this.

## 4. Abstract

All papers should have an Abstract of not more than 200 words, suitable for the various abstracting journals to use without alteration.

## 5. Presentation

### 5.1 Printed (typed) copy

Two copies of the manuscript, typed 1½/2 line spacing on one side only of A4 paper, with right and left margins of at least 2.5cms, and the pages numbered in sequence, should be sent to the Editor. Particular care should be taken to ensure that mathematical symbols and expressions, and any special characters such as Greek letters, are clear. Any detailed mathematical treatment should be put in an Appendix so that only essential results need be referred to in the text.

### 5.2 Disk version

Authors are requested to submit a magnetic disk version of their copy in addition to the manuscript. All artwork and diagrams to be supplied in their original source format. The Editor will be glad to provide detailed advice on the format of the text on the disk.

### 5.3 Diagrams

Line diagrams will, if necessary, be redrawn and professionally lettered for publication, so it is essential that they are clear. Axes of graphs should be labelled with the relevant variables and, where this is desirable, marked off with their values. All diagrams should have a caption and be numbered for reference in the text and the text marked to show where each should be placed - e.g. "Figure 5 here". Authors should check that all diagrams are actually referred to in the text and that all diagrams referred to are supplied. Since diagrams are always separated from their text in the production process, these should be presented each on a separate sheet and, *most important*, each sheet must carry the author's name and the title of the paper. The diagram captions and numbers should be listed on a separate sheet which also should give the author's name and the title of the paper.

### 5.4 Tables

As with diagrams, these should all be given captions and reference numbers; adequate row and column headings should be given, also the relevant units for all the quantities tabulated.

### 5.5 References

Authors are asked to use the Author/Date system, in which the author(s) and the date of the publication are given in the text, and all the references are listed in alphabetical order of author at the end.

> e.g. in the text: "...further details are given in [Henderson, 1986]"

with the corresponding entry in the reference list:

> HENDERSON, P. Functional Programming Formal Specification and Rapid Prototyping. IEEE Trans. on Software Engineering SE*12, 2, 241-250, 1986.

Where there are more than two authors it is usual to give the text reference as "[X et al ...]".

Authors should check that all text references are listed; references to works not quoted in the text should be listed under a heading such as *Bibliography* or *Further reading*.

### 5.6 Style

A note is available from the Editor summarising the main points of style - punctuation, spelling, use of initials and acronyms etc. - preferred for Journal papers.

## 6. Referees

The Editor may refer papers to independent referees for comment. If the referee recommends revisions to the draft, the author will be asked to make those revisions. Referees are anonymous. Minor editorial corrections, as for example to conform to the **Ingenuity** general style for spelling or notation, will be made by the Editor.

## 7. Proofs, Offprints

Printed proofs are sent to authors for correction before publication. Authors receive either a hard copy master of their paper or a Word for Windows version in either V2 or V6 on magnetic media.

## 8. Copyright

Copyright of papers published in **Ingenuity** rests with ICL unless specifically agreed otherwise before publication. Publications may be reproduced with the Editor's permission, which will normally be granted, and with due acknowledgement.

Ingenuity May 1995