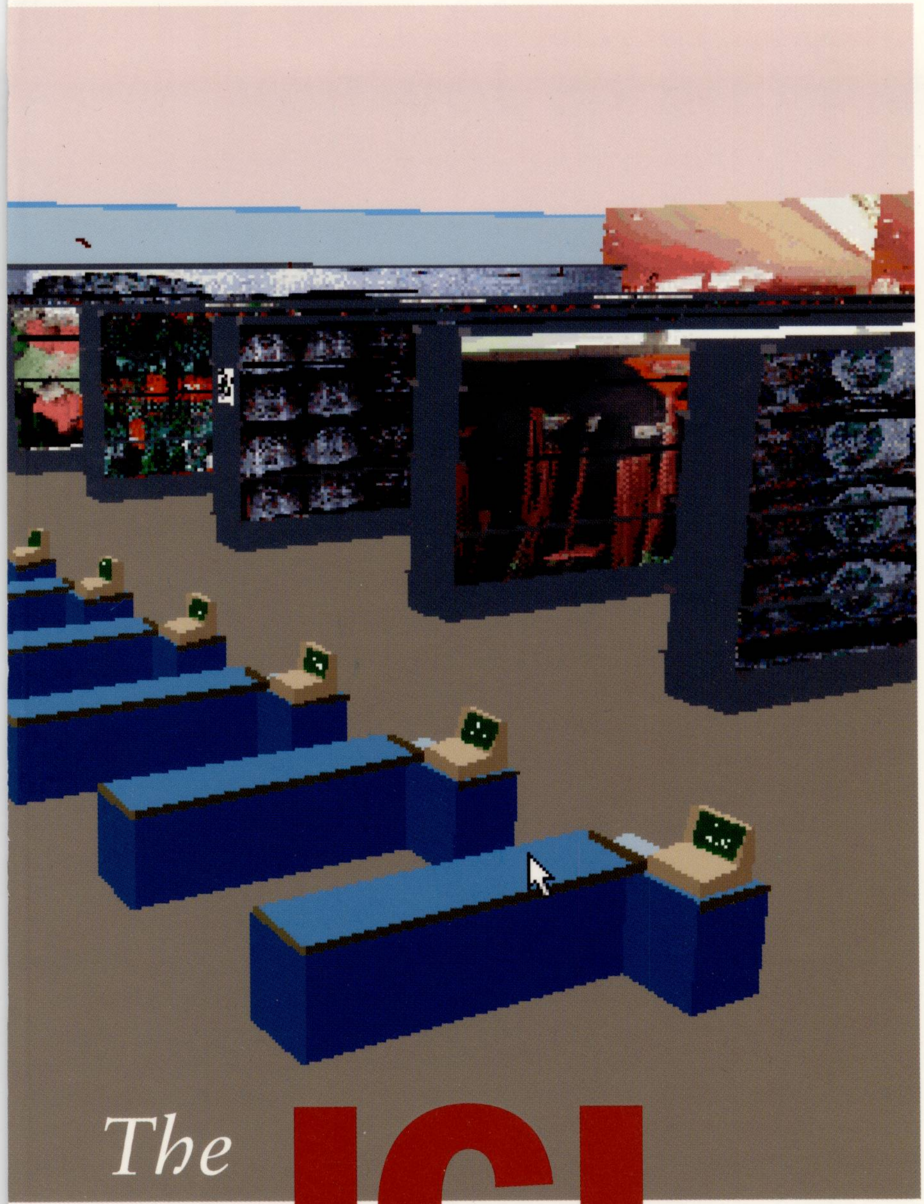
Volume 11 Issue 1 May 1996 Edition



ICL

The ICL Systems Journal

ICL Systems Journal

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

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

Published twice a year by Research and Advanced Technnology, ICL, Bracknell.

1996 subscription rates (including postage & packing):

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

ISSN 1364-310X

ICL Systems Journal

Volume 11 Issue 1

Contents

Editorial

V.A.J.Maller

The Internet and how it is used 1

Matthew Jeoffroy

An Architecture for a Business Data Warehouse 23

M.H. Derbyshire

Virtual Reality as an Aid to Data Visualization 48

Doug Urquhart

Re-engineering the Hardware of CAFS 71

Richard Illman

An Innovative Solution for the Interconnection of Future Component Packaging 84

Michael Hendriksen, Steve Hodgkinson, Neil Taylor and Bob Trussell

Development of Practical Verification Tools 106

D.J. King and R.D. Arthan

Coupling ORACLE with ECLiPSe **123**

S.K. Das and J. Dicker

Integrating the Object Database System ODB-II with Object Request Brokers **145**

Cormac McKenna

SAMSON and the Management of SESAME **158**

Barry A Byrne

Book Review **179**

Previous Issues **181**

Guidance for Authors **191**

***Front cover*:** Virtual Reality Interior of a hypothetical Store. See the paper," Virtual Reality as an Aid to Data Visualization," in this issue.

Editorial

The ICL Technical Journal has been published biannually since November, 1978. In May, 1994, the name was changed to, "Ingenuity, the ICL Technical Journal."

In 1990, Jack Howlett, the founding editor, handed over to Dr John Pinkerton, who continued in the role until the beginning of this year. He, in turn, has passed the baton to me and I hope that I shall be able to maintain the high standard established by my distinguished predecessors, which is, without question, a difficult act to follow.

Many in ICL will already know me (at least, I hope I am remembered!). I joined the company in January, 1962 and was actively involved in Research and Development until I was enticed away to join Thorn EMI as Technical Director of Thorn EMI Information Technology in the mid-eighties. Some years later, after I had left Thorn EMI and established my own consultancy business, I became, once again, involved in ICL's business as a consultant. There is an old adage in the affairs of ICL: once an ICLer, always an ICLer. The result was that I was offered, and accepted, an ICL sponsored chair at Loughborough University and have now become, in addition, the editor of the Journal.

When I became editor, the support structure, quite coincidentally, was being changed. At the beginning of this year, in addition to the editorial function, the responsibility for production, marketing and distribution was placed under Bill O'Riordan, the manager of Research and Advanced Technology. This reorganization gave the new editor the authority to review and, if necessary, to recommend changes covering all aspects of producing the Journal. One of the first actions was to re-consider the title, since it was clear that Ingenuity was not a popular name. So the editor decided to revert to the original title but, at a subsequent meeting of the editorial board, it was unanimously decided that, in the light of ICL's current business activities, a more appropriate title would be, "ICL Systems Journal." The current edition, therefore, is the first to be published under this new title.

The aims of the Journal remain similar to what they always have been, namely, to inform ICL's customers, the IT community, research workers and ICL staff about the company's achievements in the research, development and implementation of software and hardware technologies, the development of systems methodologies, as well as the integration and management of advanced systems in demanding applications environments.

The editorial policy will continue to be to produce a journal of high quality that will be read and, indeed, deemed essential reading not only by all those closely associated with ICL's business but also by those who are not. At all costs, the Journal will avoid becoming that quintessentially academic phenomenon: the write-only journal!

Publication of the Journal will continue to be paper based, but current abstracts and recent back numbers will become available on the World Wide Web.

Last, but by no means least, I should like to express my gratitude to the retiring editor, John Pinkerton, for the enormous help he has given me during my first six months as editor. I am now beginning to believe that I can swim without corks! I should also like to pay a personal tribute to him as one of the pioneers of electronic computing. John worked on radar during the Second World War and, after completing his PhD at Cambridge during the immediate post war period, joined J. Lyons & Co. where he achieved the distinction of designing the first computer used for commercial data processing, the LEO (Lyons Electronic Office). LEO Computers, as the group became called, eventually became part of English Electric Leo Marconi, which merged with ICT in 1968 to become ICL. John continued to work for ICL until his retirement and then undertook the task of editing the Journal. I am delighted that John, as well as the first editor, Jack Howlett, have agreed to remain on the editorial board and thus the Journal will continue to have the benefit of their wisdom and experience.

V.A.J. Maller

The Internet and how it is used

Matthew Jeoffroy

School of Information Systems
Kingston University, Surrey

"Like the PC, the Internet is a tidal wave. It will wash over the computer industry and many others, drowning those who do not learn to swim in its waves." Bill Gates, Microsoft Corporation.

Abstract

The ARPAnet has now evolved into a global communications medium called the Internet. This network of networks is the current darling of the media; whether it remains so will depend upon public opinion and usage of the services it provides. Figures about the Internet user-base would suggest continued success for the near future, although questions remain about its sustainability. Along with the growth of the network itself has been an increase in the number of options for connecting to this network, and an increase in development of tools and services available to its users. On the Internet, huge amounts of data exist and information retrieval is as important as the data itself; many start-up companies are addressing this issue and equally as many are tackling the difficult questions of security, the results of which will reverberate globally through all businesses planning an Internet venture. To become more than an experiment in global trading, today's Internet must grow and adapt to its increasingly business-oriented environment; only sustained positive results will facilitate this. This paper serves as a gentle introduction to Internet technology and discusses general concepts surrounding its use.

1. Introduction

One cannot read the popular or even the unpopular press, without being told that huge changes are on the horizon in the way we conduct our day-to-day existence. Not only will our entertainment, consumer and work requirements be fulfilled without our having to leave home, but the number of services available there will increase dramatically. Possibly of more significance will be the way we will interact with these services and how they, in turn, will interact with us.

Will it be a pipeline to the world or the pipe dream of an over-zealous US Senator? Whatever the case, the infrastructure to allow this to happen is not yet in place. We have major data backbones being put in place, (NSFnet/NREN, SuperJANET and a host of commercially-orientated trunks), but it will be several years before every home becomes part of the ubiquitous fibre-optic "Information Super-highway."

1.2 Origins of the Internet

What we do have at the moment is the Internet, a 25 year old network that is owned by nobody and everybody, which grew almost organically from the connection of two US Department of Defence computers into a network and which is beaten in size and interconnectivity only by the world's telephone network. The Internet has become the testbed for numerous multimedia experiments, both research and commercially orientated. Those who are fortunate enough to have access to this network are possibly witnessing the birth of "telelife" and the removal of the spatial and temporal nature of communications.

In 1969, the Internet, as it is now called, was known as the ARPANET, (the Advanced Research Projects Agency NETwork), which was managed by the US Department of Defense.

The development of the Transmission Control Protocol/Internet Protocol (TCP/IP) in the mid 1970s by Vinton C Cerf of Stanford University and Robert Kahn has enabled the interoperability between different computer systems to appear almost seamless to the user. By 1983 it had replaced the Network Control Protocol (NCP) as the delivery mechanism on ARPANET.

A milestone was reached in 1986 when the National Science Foundation (NSF) concluded it needed a high speed data backbone to link five super computer data centres and, at the same time, to provide access to them for research workers across the country by linking the NSFnet to all the regional networks; the Internet was outgrowing its original goals and environment.

By 1990 ARPANET was phased out and the transition to NSFnet-based Internet was complete. From this point commercial networks have emerged and linked into the wider Internet giving us the pervasive worldwide infrastructure that we have today: something of an electronic Petri-dish spawning new technologies and businesses.

1.3 How the Internet delivers messages

Every machine connected to the Internet has its own unique address. Technically this address is known as the machines Internet Protocol (IP) number. This number consists of four separate numbers, each somewhere in the range of 000-256, separated by dots. For example, the IP number for the main Internet server at the White House in the US is 198.137.241.30.

Any hardware-hardware, software-software or hardware-software interaction is facilitated through protocols (rules) and, as has been mentioned, the main communication protocol used on the Internet is TCP/IP.

Every message to be sent over the Internet is split up initially into many small packets so as to aid transmission speed and to allow each packet to take a different route through the network to reach its destination. Generically each packet will consist of a sending address, a destination address, the packet size and the sequential position of the packet in relation to the other packets making up the complete message. To traverse the network special hardware is required to send data packets from machine to machine until the destination is reached. These machines are called routers and are dedicated to the task of routing data packets through the network and deciding the best route for each data packet to take, based on availability and loading of virtual routes along the way [Quarterman & Carl-Mitchell, 1994]. By using this dynamic routing of packets of information, the Internet can maintain a high level of robustness and circumvent any problems or areas of "broken" network. Thus data goes where it is supposed to go, achieving one of the original aims of the network.

1.4 The Internet in use

The Internet is no different from many other telecommunications based information systems [Swatman, 1993] and is built on the same necessary foundations of IT and Telecoms. Where it differs from its counterparts is in its ability to offer global connectivity and flexibility, the keys to its popularity.

The Internet will work for the user in all but the most extreme cases. However, in order to get anything out of the Internet it is essential to get connected to it in an appropriate way; there are two main approaches:

- by a permanent connection: usually found in workplaces where the local area network (LAN) is connected to a dedicated leased line from a main telecoms operator. Even if the company has a corporate subscription with an Internet Service Provider (ISP), the ISP will still require the telecoms operator to install the leased line at the companies premises. Dedicated leased lines are not cheap and will cost between £10 000 and £250 000 per annum

- by a temporary connection: usually found in homes. Minimum requirements for this type of connection are obviously a computer, a modem, a phone line, together with a subscription to an ISP. Dial-up connections are relatively cheap and will cost around £15 per month plus 'phone charges on top.

Before entering this electronic labyrinth there are a number of key decisions to be made.

What type of service is required?

Is it just an email account or is access needed to all the services to which one can connect (WWW, Gopher, WAIS etc.)? While email is a core service, usually provided without being asked for, more and more users of the Internet are requiring the full range of Internet services mentioned above to be available in the connectivity portfolio so that they have immediate control over whether to use them or not: not all ISPs will offer full, unlimited access to Internet services.

Who can offer the service required?

Does the ISP have a Point of Presence (POP) located nearby? This is important for any users planning to spend an extended period of time connected to the Internet via a dial-up connection because they are in fact making a 'phone call. For example, it would be silly for a potential user living in London to subscribe to an ISP with a POP in Edinburgh because long distance telephone charges would apply for every session. It would make more sense for our user to subscribe to an ISP with a POP in London so that all sessions are charged at a local rate.

Is the service reliable, fast, viable, and secure? Can one get connected at both peak and off-peak times?

What is the cost?

Subscription charges to ISPs will vary *minimally* in cost (usually there is a monthly subscription charge) but sometimes *maximally* in services provided and even *accessibility* of services.

somewhere in the range of 000-256, separated by dots. For example, the IP number for the main Internet server at the White House in the US is 198.137.241.30.

Any hardware-hardware, software-software or hardware-software interaction is facilitated through protocols (rules) and, as has been mentioned, the main communication protocol used on the Internet is TCP/IP.

Every message to be sent over the Internet is split up initially into many small packets so as to aid transmission speed and to allow each packet to take a different route through the network to reach its destination. Generically each packet will consist of a sending address, a destination address, the packet size and the sequential position of the packet in relation to the other packets making up the complete message. To traverse the network special hardware is required to send data packets from machine to machine until the destination is reached. These machines are called routers and are dedicated to the task of routing data packets through the network and deciding the best route for each data packet to take, based on availability and loading of virtual routes along the way [Quarterman & Carl-Mitchell, 1994]. By using this dynamic routing of packets of information, the Internet can maintain a high level of robustness and circumvent any problems or areas of "broken" network. Thus data goes where it is supposed to go, achieving one of the original aims of the network.

1.4 The Internet in use

The Internet is no different from many other telecommunications based information systems [Swatman, 1993] and is built on the same necessary foundations of IT and Telecoms. Where it differs from its counterparts is in its ability to offer global connectivity and flexibility, the keys to its popularity.

The Internet will work for the user in all but the most extreme cases. However, in order to get anything out of the Internet it is essential to get connected to it in an appropriate way; there are two main approaches:

- by a permanent connection: usually found in workplaces where the local area network (LAN) is connected to a dedicated leased line from a main telecoms operator. Even if the company has a corporate subscription with an Internet Service Provider (ISP), the ISP will still require the telecoms operator to install the leased line at the companies premises. Dedicated leased lines are not cheap and will cost between £10 000 and £250 000 per annum

- by a temporary connection: usually found in homes. Minimum requirements for this type of connection are obviously a computer, a modem, a phone line, together with a subscription to an ISP. Dial-up connections are relatively cheap and will cost around £15 per month plus 'phone charges on top.

Before entering this electronic labyrinth there are a number of key decisions to be made.

What type of service is required?

Is it just an email account or is access needed to all the services to which one can connect (WWW, Gopher, WAIS etc.)? While email is a core service, usually provided without being asked for, more and more users of the Internet are requiring the full range of Internet services mentioned above to be available in the connectivity portfolio so that they have immediate control over whether to use them or not: not all ISPs will offer full, unlimited access to Internet services.

Who can offer the service required?

Does the ISP have a Point of Presence (POP) located nearby? This is important for any users planning to spend an extended period of time connected to the Internet via a dial-up connection because they are in fact making a 'phone call. For example, it would be silly for a potential user living in London to subscribe to an ISP with a POP in Edinburgh because long distance telephone charges would apply for every session. It would make more sense for our user to subscribe to an ISP with a POP in London so that all sessions are charged at a local rate.

Is the service reliable, fast, viable, and secure? Can one get connected at both peak and off-peak times?

What is the cost?

Subscription charges to ISPs will vary *minimally* in cost (usually there is a monthly subscription charge) but sometimes *maximally* in services provided and even *accessibility* of services.

What is the ISP's background?

> Anyone wishing to get connected via this route would be wise to look at the networking and access infrastructure of the ISP they want to connect to; whether the ISP is continually investing in that infrastructure, what level of service can be expected for the subscription fee and, based on the current levels of service, whether the ISP will still be in business in a few years time.

Users of a dial-in service do not need a permanent IP number assigned to their machine; instead, the ISP's server will assign a temporary IP number to them from a 'pool' of IP numbers held on that server. This is known as dynamic IP allocation. As soon as a user finishes the session the temporary IP number assigned is put back in the pool.

For the business user who requires a *presence* (site) as opposed merely to Internet access the connectivity questions will be roughly the same as those above, but there will be several additional issues to consider

Are strategic and contingency plans in place?

> Global business practices using the Internet are in a volatile state. Any plans for serious business use must be amenable to changing circumstances which may arise faster than one would expect in normal business practice (if there is such a thing).

Once plans are decided is it preferable to:

- go to an Internet Service Provider (ISP) who will house and maintain the site at about £30/£40 per month rental (per 5MByte of storage). A good link will be required between the site and that of the ISP for updating the information on the site and collecting mail; one must be prepared for certain restrictions on common gateway interfaces (cgis), data capture, databases etc.
- make in-house provision: install own server and maintain both it and the leased line connection. Although costs are falling, a 64Kbit/sec line with IP will cost £12,000 and a 2Mbit/sec line with IP will cost £250,000
- employ service providers: there are many and their number is increasing; for example, PIPEX, EasyNet, EUNet, Cityscape, Demon, BBC, Pavilion, Compuserve etc.
- run servers under different operating systems: Macintosh, Unix and Windows NT are the most popular platforms. A one-off purchase cost for a bundled hardware and software server package will cost between £5,000 and £20,000?

Who will be responsible for server maintenance?

The server and everything that goes with it will need constant attention in respect of updating of information, security considerations, file transfer, email, server software updates, custom-made scripts etc. Server maintenance is a full-time job requiring competent technical skills.

How to maintain and update pages on a regular basis?

If a site is to be open and successful regular mainteance of the data is required. This requires resources and the Internet culture is perceived as the provision of services that are easily used and eessentially free.

Should one get a permanent IP number and a domain name[1]?

If a permanent service on the Internet is to be provided via a dedicated leased line connection[2] to the network, then a machine with a permanent IP number will be needed. The simplest way of doing this would be to let the ISP assign the IP number and register a domain name. If the ISP cannot or will not perform this service then it may be worth looking elsewhere for service provision unless they have a good reason for not providing the service. At a minimum, what the ISP should be able to provide is the address of a regional, national or international IP network number registry. *Regional* and *national* registries will vary but *internationally* the registry is known as InterNIC (hostmaster@internic.net). An annual fee of about £100 is payable to the InterNIC for name registration. [Quarterman & Carl-Mitchell, 1994].

1.5 The InterNIC

Any new networks created on the Internet must be validated by an Internet Network Information Centre (InterNIC). The *international* InterNIC is funded by the National Science Foundation (NSF) to offer services specifically to the US National Science Foundation community

[1] A domain name is a name given to your organization's network which maps on to your network IP number. It is a far easier way of remembering the address of a machine.

[2] There are other ways of enabling this type of service such as having an ISDN connection but this has certain drawbacks such as cost and accessibility/ease of connection.

(NSFnet) and generally to the Internet community. Services include information about software and services (*is.internic.net*), directory, database and WHOIS[3] services (*ds.internic.net*) and domain name and IP network registration services (*rs.internic.net*). Users may access this service via Telnet, WAIS, Gopher or WWW. American Telephone and Telegraph (AT&T) currently operate the InterNIC. [Quarterman & Carl-itchell, 1994], [Comer, D.E, 1995]

2. Internet Applications

Having gone to the time and trouble of getting connected to the Internet it is worth exploring the services available; some are closely related cousins while others perform intrinsically and vitally different functions.

Most people are familiar with electronic mail (email), how it works and for what it has been used. Each tool discussed next, even if used in isolation, provides the user with access to vast data repositories, which collectively enable the user to access a whole world of information. The underlying functionality of most tools exploits the *client-server architecture.* In this mode of working the *client* exists as a piece of software that will usually reside in the user's desktop computer. The *servers* in this model are the machines that receive requests from the clients for almost any type of data, fulfil that request and pass the results back to the user. For further information on the client-server architecture and its applications see [Comer & Stevens, 1993].

In order to use most of the Internet tools it is necessary to understand the *uniform resource locator* (url). A url is like an address in Cyberspace. It is the way that the client locates documents requested. It consists of protocol type, a machine name, a directory structure and file, i.e.

http://infosys.kingston.ac.uk/isschool/welcome.html

There are around 12 million publicly accessible urls now in existence.

[3] WHOIS databases store information about users. The WHOIS server held by the InterNIC grew to impractical proportions requiring a distributed database architecture. Internet registered sites may or may not have a WHOIS database, preferring to use X.500 services.

2.1 World Wide Web[4]

The *World Wide Web* (WWW) is a distributed hypertext environment accessible over the Internet. Coupled with the introduction of the graphical *browser,* the WWW is largely responsible for the popularity of the Internet as it is today. The WWW has been compared frequently to an on-line interactive magazine or CD-ROM; similarly its contents may be retrieved using a browsing (surfing) method or by using specific information retrieval (IR) services developed to aid the user (these will be discussed in the following section). Requests for files are made through the use of *hyperlinks,* (hypertext links). These hypertext links, embedded in documents using HyperText Mark-up Language (HTML), allow the user to jump to different pages in the existing document, to another document on the same server, or to another server anywhere in the world. The resulting page may contain text, graphics, moving images, or sound. For the most part the data available via the WWW will exist in free, unstructured form. However, as sites become more sophisticated, increasing processing power is being used to structure data and implement database solutions to meet business needs.

2.2 Gopher

Gopher is a searching and browsing tool developed at the University of Minnesota in North America. Gopher is also a distributed information system but, rather than using hyperlinks like the WWW, it uses a hierarchical menu structure to allow the user to navigate through *Gopherspace.* While this interface presents the user with a fairly structured navigation path, it also constrains the user to following these set paths because none of the documents at the end of these menu branches have any means for linking to other documents. Gopher was extremely popular in the early 1990s but is gradually being superseded by the WWW; many existing Gopher sites are gradually becoming Web sites to take advantage of the added functionality available. A Gopher site remains a very popular means for disseminating information and, where funds do not permit the migration from Gopher to Web, there has been an integration of the two technologies.

[4] The WWW is not the same as the Internet. The Internet is basically the cables and wires which facilitate the communication of protocols which services sit on top of and use. The WWW is a virtual space which uses the Internet as its communication medium.

2.3 File Transfer Protocol

The Internet *File Transfer Protocol* (FTP) is used to retrieve any of the hundreds of thousands of files stored on Internet ftp servers. Users of this service are usually granted 'anonymous' access to the machine on which the desired file is held. This means that the user does not have to have a specific account on all of the ftp servers that exist on the Internet but they will only have specific, limited, access to those machines where retrieval rights are granted. Ftp servers are still one of the most commonly used methods for disseminating shareware applications and documents over the Internet and have been widely integrated into WWW sites. When downloading executable programs from the Internet one must take care to check for viruses before running the software to avoid loss of data and damage to hardware and/or software; many software programs exist that will scan files for viruses before they are run.

The same protocol is also used to place files on a host server; a function performed in the majority of cases by the system administrator.

2.4 Usenet News

Usenet news presents all users of the Internet with a worldwide forum for discussion of virtually any topic. The best analogy for this environment is that of a bulletin board where users (members of a particular discussion group) can scan or reply to articles posted up previously. Users have the choice of some 16,000 Usenet newsgroups to which they may post comments via email on a topic they wish to discuss. Discussion groups are split into main headings, the *big seven* [Young & Levine, 1995] include *comp.* (computer-related), *misc.* (miscellaneous), *news.* (Usenet related), *rec.* (recreational), *sci.* (scientific), *soc.* (social issues), and *talk.* (controversial debates). A huge discussion group not mentioned in the big seven is the *alt.* (alternative) group. Users may subscribe to whatever discussion group they choose, although some sites may restrict access to some of the more colourful and controversial discussion groups, usually from the *alt.* postings, (e.g. alt.binaries.erotic.pictures!).

On deciding to participate within a newsgroup a user would be wise initially only to read messages posted to the group to get a feel of the general tone of the discussions. This cautious approach will ensure comments are not posted out of turn. Should remarks be made to the group which result in negative reactions *flaming* may ensue. *Flaming* takes the form of strongly worded protests to the group by several of its members complaining about the tone or content of the offending message. Flaming may also include sending multiple mail messages to the offenders email box resulting in an overloading of his/her system. Usenet news is worldwide and so care must be taken with postings.

Usenet news capable sites have Network News Transfer Protocol (NNTP) servers to *listen* on the network for new news postings and sort them into groups for the local site audience. Any new postings are passed from NNTP server to NNTP server until all the Usenet capable sites have them. Newsgroups, are in most cases, *unmoderated* but some *moderated* groups exist; however, it is more common to find them existing as *mailing lists* (see below).

2.5 Telnet

Telnet allows a terminal-based session to be started on the chosen machine; by logging onto a remote machine a user can work as if using a terminal connected directly to that machine. To access a machine by logging on will require, in the majority of cases, a username and password for that machine, as relatively few machines allow anonymous remote telnet access.

Along with ftp and email, telnet is one of the oldest Internet services but has diminished in popularity with the advent of newer and more sophisticated services like the WWW [Young & Levine, 1995].

2.6 Mailing Lists

Mailing lists are email address lists set up so as to create a forum where people having some common interest can take part in a dialogue on a particular topic. As in the case of Usenet news groups, these lists may be moderated or unmoderated and will usually state this on signing up. Moderated lists have the advantage that a certain amount of message filtering has been done by the moderator, which should, theoretically, cut out much of the *noise* (useless messages) that plague a number of mailing lists; however, messages may take longer to appear to other members of the group due to the limited amount of filtering possible in one day.

The concept of the forum is similar to that of Usenet news but mailing lists are more selective and expect prospective users to subscribe actively via email to the list manager or to the software application that handles the management of mailing lists (mail-servers or list-servers/listservs). Postings to mailing lists are sent via email to every person who has subscribed to the list. Mailing lists are an excellent source of information (and they are free) but before hastening to join all the lists imaginable, consider carefully the need to join any list and the time it would take to read the possibly vast numbers of email messages generated.

2.7 Internet Relay Chat

Internet Relay Chat (IRC) is a multi-user on-line chatting system commonly called the *CB of the Internet.* IRC allows groups of people to converse almost in real-time over the Internet by joining one of the many *channels* running on the IRC server. These channels are usually given self-explanatory names to give new users an idea of the conversations taking place within them. IRC has gained a reputation for being a time waster consisting mostly of inane conversations which draw in the unsuspecting who may all too easily forget about the time. This reputation has unfortunately detracted from some of its more spectacular successes, such as live coverage from Iraq during the Gulf War and the Russian coup against Boris Yeltsin in 1993 [Winder, 1994].

3. Searching the Internet for Information

The Internet is already large and is growing by the day. Much information contained on it is unstructured in form, which certainly does not ease the task of finding useful information. Vinton C. Cerf has claimed to the House of Representatives that the user population of the Internet could exceed 100 million by 1998; some conservative estimates claim one billion users by the year 2000 [Negroponte, 1996]. With each of these information retrievers being a potential information provider it is likely that information retrieval (IR) problems will grow beyond solution. If that happens, the aphorism, "using the Internet is like drinking from a fire hose," rings true [Ellesworth & Ellesworth, 1995].

However, all is not gloom and there is hope to be found in some tools for IR now available on the Internet.

3.1 Searching on WWW

"AltaVista gives you access to the largest Web index: 30 million pages found on 225,000 servers, and 3 million articles from 17,000 Usenet news groups. No wonder it is accessed over 10 million times daily!" [Digital

Equipment Corporation, 1996]. AltaVista[5] is only one of the WWW-based IR tools available free today on the Internet and may be overtaken in breadth of coverage and popularity by one of the many fierce competitors in this growing area.

WWW-based IR tools may be defined in two main ways; a *catalogue* of sources or an *index* of sources. Catalogue IR tools aim to classify any data received into subject areas and allow users a *top-down* view of information; by working through the levels of subjects one should reach the desired source. A notable success using this method is *Yahoo!*[6]. Rather than use statistical algorithms and sheer computing power to classify data, Yahoo! performs manual sorting of data received from automated *Web crawlers*[7] via a workforce of about 30 classifiers. This has advantages of a filtering process but classification is subjective; what may easily fit into a particular category for one person may be entirely different for another. Yahoo! is also faced with a diminishing percentage of classified Web pages as the Web grows faster than human classification methods can keep pace; instead, they aim to catalogue *the best of the Web.*

Web index sites also send out *robots*[8] or *spiders* to search WWW servers for filenames, *titles* of html files, *header* sections of html files which may contain keywords specifically put in by the author of the paper, and others perform WAIS-like operations and index the whole of the html page. Due to the amount of data involved with these methods, powerful computers are required to run statistical algorithms to attempt to classify information and to ensure requests for information from the index are completed in a reasonable time. Search robots are not all equally good; some have been known to damage sites seriously by requesting too much information too quickly and to have been generally 'impolite' to the host operating system. Consequently, steps have been taken to curb access to some sites by robots using a special file which tells the incoming robot it is not welcome. Unfortunately this method is not foolproof and some robots ignore these requests. Many IR services will allow manual additions to the indexes/catalogues by the html page author; a necessary service due to the increasing amounts of data

[5] http://www.alta-vista.com/
[6] http://www.yahoo.com/
[7] A Web crawler is a software application that will roam the Web gathering pages of data.
[8] http://info.webcrawler.com/mak/projects/robots/faq.html

now available on the Web and therefore the increasing time it takes to gather site information automatically.

One of the more unfortunate features of these IR tools is that the majority use a search *language/schema* specific to that particular search engine which therefore can lead to a steeper learning curve for one to use such tools to their maximum capabilities. Most search engines allow Boolean operators, ranking of results, and placing limits on the number of *hits* returned as a result of the search; several also allow the user to specify the closeness of the results returned to the keyword/s specified. Search strategies aside, there can be no doubt that these tools have *shrunk* the Internet to a manageable size for all its users and have only added to its popularity.

A comprehensive list of search sites has been compiled by the WWW server manager in Information Systems[9] at Kingston University.

3.2 Searching in Gopherspace

VERONICA *(Very Easy Rodent Oriented Net-wide Index to Computerised Archives*) allows many thousands of Gopher menu items to be indexed and searched from one place. There are several large Veronica servers throughout the world and notionally they should all replicate one another. Searching is performed by matching keywords or keyword strings entered by the user as search terms. Whereas VERONICA will return results from searching the whole of Gopherspace another Gopher IR tool called *Jughead* allows users to make VERONICA-like searches, but at a local level only; i.e. on one particular Gopher server, assuming the service has been implemented by the site administrator.

3.3 Archie

Archie is a scheme developed at the McGill School of Computer Science in Canada which has rapidly grown in worldwide popularity. It addresses problems associated with storing and serving large numbers of files on the Internet. Archie servers centralise indexing information on file-name data distributed over Internet ftp archive sites. This requires archived files to have something approaching a meaningful filename

[9] Search sites on the Internet (via Information Systems). http://infosys.kingston.ac.uk/ISSchool/search_page.html

if this service is to work to its full potential; nonetheless it provides a very useful and, at times, essential service.

There are now at least 20 Archie servers throughout the world; the consistency between the databases is maintained by each site having a copy of the main site listing information held on the main Archie server in Montreal, Canada [Danzig et al, no date available]. As a result of a search, the user will be presented with the machine name and the path to retrieve the specified file. Several Archie clients (for example, *Anarchie* for the Macintosh computer) allow the initiation of an ftp session to retrieve the desired file simply by means of double clicking the filename found as a result of a query.

3.4 Wide-Area Information Server[10]

The *Wide-Area Information Server* (WAIS) was developed by Thinking Machines in the US in conjunction with Dow Jones & Company Inc., Apple Computers and KPMG Peat Marwick. The WAIS project provides an electronic publishing, full text information retrieval architecture based on the Z39.50 library protocol standard. WAIS servers keep an inverted index file of all the words found in every document held on the server. The WAIS architecture supports distributed information servers for remote question-and-answering based on multimedia files and makes a backbone for decentralised information distribution. Searches are made using words and the documents that the search produces are returned as ASCII text. Problems occur when the results of a query return a document which may not be the most relevant to the user but with many hits on the search string; the hits may be taken out of context. Solutions to this problem lie partially in *relevance feedback* allowing a continued refined search based on the results of the original query. This method of information retrieval is not perfect and much work is being undertaken in this area.

A *directory of servers* is maintained by Thinking Machines and all new WAIS servers are required to register at this site to ensure that users of the system can locate all other WAIS servers in operation; this appears to work well in practice. The WAIS search engine is frequently integrated into WWW sites to allow users to make broad and narrow searches on the local server and has been found to be an ideal, (and essential), value added service.

[10] http://www.wais.com/

3.5 Searching in Usenet news

Deja News[11] based in Austin, Texas has been on-line since May 1995 and claims to be the first and largest Usenet news IR service for the daily 500 megabytes[12] of Usenet news traffic carried by the Internet. Deja News offers keyword and string searching capabilities, Boolean searches, nested searches, and temporal relevance of requested data sources can be obtained. Searches using this service must be performed via a WWW browser and as yet there is no means for performing searches using a dedicated Usenet news client. Several of the major Web IR tools (for example, AltaVista, Lycos[13], Infoseek etc.[14]) have started to provide Usenet news search services.

3.6 The Impact of the Browser

A common problem that existed no more than three years ago was lack of integration of tools; we had a network that would enable and organise communication between disparate computer systems but there was little available to organise the users desktop. If one wanted to access an ftp site one would use an ftp software application; similarly if one wanted to access Usenet news one would use a dedicated news reader etc. It is true that these software applications made the user's life easier at one level by opening up access to a vast range of services, but at the same time they made life more difficult by requiring each user to select a specific item of software to access services.

Fortunately the advent of the WWW led to the development of the WWW *browser*. This type of software application allows the user to interact with all of the services, (WWW, ftp, WAIS, Usenet, Telnet etc.), seamlessly from within one application. As mentioned above, this seamless integration of services at both the technological and application level is a key factor to the popularity of the Internet today. The client applications discussed, especially WWW browsers, have changed the balance of power on the Internet. Users no longer need a degree in Computer Science to make sense of what happens on the Internet; it has been opened up to the masses.

---

[11] http://www.dejanews.com/
[12] NetPartners. Usenet storage space calculator. http://www.netpart.com/
[13] http://www.lycos.com/
[14] http://www.infoseek.com/

4. Organisational Impact of the Internet

In the same way as IT has in many cases had largely intangible effects on productivity [Scott Morton, 1991], there is still no precise means for evaluating how far commercial participation in the Internet can actually affect the profitability of a business as a whole.

However, one of the key business areas where the effect of the Internet is more measurable is marketing. Firms are now becoming aware that the Internet provides a level of interactivity with which traditional types of marketing media cannot compete. This is being demonstrated constantly with the increasing use of the Internet for shopping, advertising and customer feedback.

Companies using the Internet face a marketing model quite different from that in which they were experienced, (i.e. one-way broadcast; newspapers, radio, TV etc.). Highly interactive, two-way marketing strategies are now possible allowing users to tailor the amount of information they wish to have about a company or product to suit their needs. At the same time, managements of many on-line-oriented companies are finding that they can use generalist staff as opposed to specialist staff in order to meet user requirements [Gahan C, 1994]. While some have adapted well and seized the opportunities offered by this new medium, (e.g. Levis Co.[15] and Benetton[16]), others made no headway, by failing to grasp the differences between traditional and on-line cultures and the need for strategic input. They failed in short by not affording the Internet the time, resources and thought necessary to exploit it as a major tool in their marketing campaigns.

Commercially the Internet is concerned with rapid provision of information-based services and rapid production of supporting software applications, (for example, Adobe plug-in software modules which add functionality to the WWW browser). A fresh model of distribution is thus required in order that a sustainable economy can be created: an information economy[17]. What is vital for companies venturing into this economy is the provision of information-based value to customers to achieve a comparative advantage in a time-based competitive environment [The Internet Group, 1995]. An example of this would be Just-In-Time (JIT) purchasing between trading partners. If companies

[15] http://www.eu.levi.com/

[16] http://www.benetton.com/

[17] A framework for the value which is added by systematically processing data and for organising its distribution. [Lindsay, 1993].

choose not to provide added value, potential customers can seek satisfaction elsewhere.

Evolution, enhancements and integration of the major Internet applications are happening daily and can be observed simply by logging on to the network. This metamorphosis of tools will ensure sustained interest and development of the infrastructure and services of the network.

There are few examples of direct transactions between machines, (i.e. transactions that initially involve some user input which are transmitted and processed without subsequent rekeying of data), over the Internet. There is a lack also of specially formulated standards for interchange of commercial data via the Internet. However, notable exceptions appear in many of the *electronic cash* initiatives and increasingly databases are being used to automate business processes. For serious business use, the Value Added Networks still retain an advantage, mainly in terms of standardisation and perceived security of data. There is a form of Internet EDI using the ANSI X12 standard, a cataloguing standard (Z39.50) used in the WAIS architecture and Multipurpose Internet Mail Extensions (MIME[18]). These protocols exemplify strategies for future data exchange and groups are being created to speed developments, (for example, the CommerceNet[19] initiative).

The members of the whole Internet community co-operate well and provide feedback that is essential in establishing effective standards. This is shown by the history of TCP/IP and of many other standards developed through dissemination of drafts and *Requests For Comments* (RFCs) over the Internet. The *Internet Engineering Task Force* (IETF)[20] is a not-for-profit organisation set up to aid development of protocols, tools and services on the Internet; it has been responsible for many successes in these areas. The IETF is currently working on the creation of the *Internet Mercantile Protocol* (IMP) to include adaptations of current

[18] MIME allows information other than standard text, (i.e. pictures, sound etc.), to be included in standard mail messages.
[19] http://www.commercenet.com/ is a collaborative initiative between major corporations to try to forecast and smooth the way forward for organisations wishing to use the Internet for any form of electronic data interchange.
[20] http://www.ietf.org/

protocols to automate business transactions using Privacy Enhanced Mail (PEM) [Ellesworth & Ellesworth, 1995].

5. Security

Critical to the success of any Internet business proposal involving use of the Internet will be the difficult issue of security. If there is no security then there will be no trust in the system and hence no business. Scare stories abound asserting that no responsible business should transfer critical work to the Internet because that would leave the data (and hence the business) open to damage. The most secure machine is one which is not connected to a network but a requirement to communicate with business partners dictates this issue.

For most organisations security concerns will fall into three main categories:

- People. There is much naivety on security aspects of the Internet. Organisations must train users to become aware of the security facilities that are provided and of the possible effects their actions may have on the business. Domains of trust must be established and areas of responsibility defined; one should be wary of too much empowerment and beware of user resistance. **"The weakest link in any system is the person"**
- Systems. There are now available Routers/Firewalls and Cryptography schemes of considerable security. This is too specialised an aspect to treat adequately here but further reading may be found in [Stallings, 1995]
- Processes. Management procedures for dealing with network security and the physical security of the machines themselves must be worked out and enforced. They should be kept simple and auditable.

There is no simple formula for solving all security concerns. The key lies not in the technology alone; that may ensure only a certain level of security, (indeed it could give a false sense of security). Where security is put and how it is used are at least as important: therefore identify carefully the *boundaries* of the security area.

The government is concerned about information as an asset which must be protected and the Department of Trade and Industry along with several major companies, (e.g. Unilever), is developing a Code of Practice for Information Security Management; this may be used as a reference document by those managers responsible for the safeguarding of information within a given organisation. The Code appears under British Standard *BS7799.*

6. Conclusions

Over many years the characteristics of the Internet have changed radically; such changes must continue if the diversity of current experimental uses is to become widely accepted as standard practice. Industrial culture has moved into the information age and is driven relentlessly by market forces. The immediacy and accessibility of information technology are facilitating the development of a global information economy, in which the major participants are likely to be small and medium sized businesses, plus small autonomous units that have arisen from the restructuring of large businesses [Naisbitt, 1994].

The capital outlay, required to exploit the Internet, is now smaller than might have been expected. The price, speed and capacity of available Internet connections provide comparable opportunities for businesses, irrespective of size [Cronin, 1994].

The change in business culture from *command-and-control* to flatter organizational networks [Drucker, 1988] and the evolution of the information-based organization have been no accident: each has been an enabler for the other. The Internet has provided further opportunities for the formation of partnerships[21] and coalitions on a global scale to meet economic goals or political missions. However, implications of a global community are causing concern at national levels throughout the world because individuals and organisations are now empowered to bypass institutions of nation states [Rzevski, 1994] and bring about change in the larger social system of relationships itself [Land, 1993]. Should this scenario arise no-one is clear what the outcomes would be. At the present time the Internet community is relatively self-policing but, as a community in its infancy, it lacks many complex social and financial restraints. Forays into this 'world' by outside forces such as governments, have in the past led to massive outcries within the community, (e.g. the 'Clipper'[22] computer chip controversy and the US Civil Liberties Law[23] recently signed by President Clinton). The Internet community is unlikely to be sustainable in the long term in the absence of some form of authority.

[21] For example the collaboration between Microsoft, Netscape, Visa and Mastercard to try to standardise a secure payments system based on credit transactions.
[22] http://www.hotwired.com/clipper/
[23] http://www.vtw.org/speech/

It must be recognised that many future developments in internet working will require significant resources. The Internet has so far been relatively cheap but is widely used by only a small part of the world's population. To bring digital services to a large number of homes will require a competitive network market place in order that the necessary components, such as servers and set-top boxes, become avaiable at consumer prices [Bubley & Bennett, 1995].

It is true that in an information-led and market-driven global economy, entry level costs are likely to be driven down to a level affordable by the vast majority of potential users, but it is also true that there is some way to go before the Internet becomes as ubiquitous as some in the media would argue. As it stands it is an enabler for research and worldwide communications and an experiment in global trading. It has been able so far to grow and adapt to its environment and should continue to do so up to the point at which it becomes subsumed by some larger faster and more effective infrastructure.

References

BUBLEY, D. and BENNETT, P., "Information Superhighways: The new information age," Financial Times Management Reports, 1995.

COMER, D.E., "The Internet Book," Prentice-Hall Inc., 1995.

COMER, D.E and STEVENS, D. L., "Internetworking with TCP/IP Volume III," Prentice-Hall Inc., 1993.

CRONIN, Mary J., "Doing Business on the Internet: How the Electronic Highway is transforming American companies," International Thomson Publishing Company, 1994.

DANZIG, Peter B. et al., "Internet resource discovery services," Computer Science Department, University of Southern California, no date available.

DIGITAL EQUIPMENT CORPORATION, "Welcome to AltaVista," http://www.alta-vista.com/, May 1996.

DRUCKER, P.F., "The Coming of the New Organisation," Harvard Business Review (January/February) from Cash et al. (1994) "Building the Information-Age Organisation: Structure, Control, and Information Technologies," The Irwin Case Book Series, 1988.

ELLESWORTH, J.H. and ELLESWORTH, M.V., "The Internet business book," John Wiley & Sons, Inc., 1995.

GAHAN, C., "Future Telecomms," BT, 1994.

KAHLE, B and MEDLAR, A., "An information system for corporate users; Wide-Area Information Servers," Thinking Machines Corporation, 1991.

LAND, F., "The Information Systems Domain," The London School of Economics, 1993.

LINDSAY, J., "The fundamentals of Information Systems Design," Kingston University, 1993.

NAISBITT, J., "Global Paradox," Nicholas Brealey Publishing Limited, 1994.

NEGROPONTE, N., "Caught browsing again," WIRED, May, 1996.

QUARTERMAN, J.S. & CARL-MITCHELL, S., "The Internet Connection," Addison Wesley Publishing Company, 1994.

RZEVSKI, G., "Building an Information Society in Europe," Kingston University, 1994.

SCOTT MORTON, M.S., "The Corporation of the 1990's - Information Technology and Organisational Transformation," Oxford University Press, 1991.

STALLINGS, W., "Network and internetwork security: principles and practice," Prentice-Hall, New York, IEEE Press, 1995.

SWATMAN, P.M.C., "Integrating Electronic Data Interchange into Existing Organisational Structure and Internal Application Systems: The Australian Experience," School of Computing, Curtin University of Technology, Australia, 1993.

THE INTERNET GROUP, "Steps for successful business entry into the Internet," http://www.tig.com/, 1995.

WINDER, D., "All you need to know about using the Net," Future Publishing Limited, 1994.

YOUNG, M.L and LEVINE, J.R., "Internet FAQs," IDG Books Worldwide, Inc., 1995.

Biography

Matthew Jeoffroy graduated from the School of Information Systems at Kingston University in 1993 and returned there in 1994 to work as a research student. His PhD work is centred on strategic business opportunities through use of electronic data interchange over the Internet. He has spoken at length on the Internet at a number of seminars for the British Computer Society and has been instrumental in promoting its use throughout Kingston University. He also manages the WWW server for the School of Information Systems.

An Architecture for a Business Data Warehouse

M.H. Derbyshire

Systems Design Group, ICL High Performance Systems, West Gorton, Manchester

Abstract

Enterprise re-structuring (with federalization, mergers and acquisitions) and technical right-sizing (with departmental servers running packaged applications) lead inevitably to operational data diversity. Corporate decision makers, however, need a single, consistent view of cross-organizational data.

A data warehouse provides just such a single, consistent view. At its simplest it is a single database where corporate data is integrated and stored.

By pulling together the disparate data into a central data set, away from the demands of the operational environment, a data warehouse aims to satisfy both the decision makers and the operational users. Decision makers can use powerful front end on-line analytic processing (OLAP) tools to interrogate this data, viewing it via a multidimensional business model (e.g. accounts by customer by time). Data miners can explore the data, using tools such as neural nets, to reveal the unknown and unexpected. Operators can use the data to provide, for example, a customer focused service.

Data warehouses, however, are custom built rather than bought off the shelf. This paper describes the technical challenges of such warehouses and offers an architectural framework for a choice in the building process.

1. Introduction

The provision of information for making decisions has long been a key business requirement and, in our increasingly competitive society, timely decisions based on accurate and comprehensive data become more and more crucial to business success. However, the disparate operational data of a business is seldom suitable for decision making, while the IT requirements of users in the operational and decision environments often conflict. This has led to a growing interest in data warehousing, as a means of integrating operational data to provide information for decision support, while separating the operational and decision environments to avoid conflict. Integrated data can also provide a significant business advantage back in the operational environment.

There is no simple agreed and all-encompassing definition of a data warehouse. At its simplest it is a place where corporate data is stored and made available to decision makers and knowledge workers for complex analysis:

- it is designed to deliver real competitive advantage by providing a single image of cross-organizational data
- it provides end-user driven analysis of critical information and more confidence in strategic decision making
- it can use key technologies with affordable capability, i.e. relational database systems and parallel processor, either SMP or MPP, UNIX platforms
- it is an accelerated Systems Integration project which delivers early business benefit, typically within 6 months, leading to higher returns from increased investment.

This paper describes a Business Warehouse Architecture (BWA) which encompasses and spans both operational and decision use of data and provides transformations between these two forms.

The paper is structured as follows:

- sections 2 and 3 explain what data warehousing is, and what the main trends are (both business and technical) which have led to its current popularity
- data warehousing is primarily about decision support, and section 4 outlines the main concepts in this area
- within an enterprise, various classes of people are involved in IT and section 5 describes warehousing from their perspectives
- sections 6 and 7 describe the architecture
- section 8 describes how parallel processors can provide the required capacity, performance and extensibility needed for a large warehouse.

2. What is a Business Warehouse?

A Business Warehouse Architecture is a framework for business information management, spanning the operational and decision environments. The operational data of a business is seldom suitable for decision making. Such data is only current and is often held in a variety of disparate data sets, which are structured to support rapid, on-line access. Decision support users, however, require a coherent view of current and historic data, structured according to the subjects of the business, and may perform time-consuming analyses. A Business Warehouse Architecture (BWA) encompasses and spans these two forms of data use. It provides an environment which manages an enterprise's information flow from the initial and often diverse operational data collection, through its integration and on to its use as higher-value information both for decision support and back in the operational environment.

A BWA clearly has much in common with the concept of a Data Warehouse, which deals with the integration and transformation of operational data into a (possibly virtual) central data set or warehouse. A BWA, however, is concerned with the whole environment, from the operational data use (typically involving On Line Transaction Processing (OLTP)) to the decision use with techniques such as On Line Analytical Processing (OLAP), [Codd et al, 1993], [Pendse & Creeth, 1995]. Data warehouses are custom-built to satisfy an individual enterprise's requirements, whereas the Business Warehouse Architecture provides a framework for a choice in the building process.

2.1 Concepts of Business Warehousing

Terminology and "buzz words" are prevalent in the data warehousing arena, where they may have a variety of overlapping meanings. Here we define major concepts and terms used in this paper. Decision support concepts are described in detail in section 4.

Decision support
An umbrella term covering all aspects of information use, whereby users (typically executives, analysts and managers) gain insight into the business by examination of data. Decision support subdivides into two categories: On Line Analytic Processing (OLAP) and Data Mining (DM), [Carnelley & Cappelli, 1995].

On Line Analytic Processing (OLAP)
OLAP involves deductive analysis of the data according to predefined or adhoc queries.

Integrated data set (IDS)
The conceptual data set, central to the BWA, which provides a coherent view of enterprise data.

Data mart
A subset of the central data warehouse within the IDS, tailored to the analysis requirements of a particular user group.

Data mining
Analysis of data populations seeking patterns, trends or relationships.

Meta-data
Data which records details about data. For example, the analysis meta-data of the BWA holds the syntactic and semantic descriptions of the business model and of the IDS.

3. Trends leading to Warehousing

The need for a coherent view across enterprise data for business and operational use is not, in itself, new. However, as described in this section, recent trends in business and in technology have combined to make the need more pressing and its resolution economically feasible.

3.1 Business Trends

The main business trend is, in a word, competition. In our increasingly competitive world, each enterprise seeks to increase its profits, margins and market share. This can translate into:

- shortening the business cycle from review to decision to action, then back to review, all of which demands accurate, precise and timely information. High quality decisions demand high quality information. For example, in retail an ideal is where each store holds exactly the required goods at any given time, minimising stock cost, with goods positioned together on shelves to maximise sales. This requires accurate analysis of buying patterns.
- providing better customer service. For example, in finance, ensuring that service staff have a customer-level view when responding to queries, rather than having only a series of account-level views.

A counter trend, as far as coherent information is concerned, is the pace of change in business organizations. The shape of a large enterprise is continually adapting to the needs of the business, from internal reorganizations, federalization of previously monolithic businesses, and acquisitions and mergers. An enterprise IT strategy must accommodate this.

3.2 Technical Trends

Although the terminology may be new, businesses have been doing OLAP for some time. However, they were constrained by the amount of data which could be economically integrated and analysed. The main technology trends which are removing these barriers are parallel processors, RDBMSs which exploit them, cheap on-line storage, and sophisticated analytical tools. Together these provide fast access to large quantities of data at affordable cost.

A trend which increases the need for a business warehouse is the rise in the diversity of the primary data; for example:

- the introduction of new data input devices (such as EPOS and ATM machines)
- the continuing de-centralization of IT, with "right-sizing," departmental servers and the use of packaged applications
- the trend to business organization changes identified above.

In any modern business, these are inevitable and positive changes and any architecture must allow for such growth and evolution. It will never be possible to "stop the world" and implement some unified model throughout an organization.

4. Decision Support

The prime business trend for warehouses is improved decision support. This section describes the main concepts behind the two distinct support categories of:

- on-line analytic processing (OLAP)
- data mining (DM).

4.1 OLAP

The term "OLAP" was introduced by E.F. Codd [Codd et al, 1993] to distinguish between the decision and operational environments; OLAP as contrasted with OLTP. OLAP covers all the mainstream decision support processes which analyse enterprise data using well established mathematical and statistical techniques; processes ranging from predefined reports for operational decisions (MIS) through semi-defined queries for tactical decisions (DSS) to unstructured queries customized for executive users and strategic decision making (EIS). Each of these overlapping decision levels includes and extends the earlier levels, and reflects a change in the number of users, from the medium number of MIS users through the small number of skilled analysts submitting DSS queries to the very small number of EIS users. In practice, the enterprise will also make queries that are neither on-line nor subject-based (e.g. lengthy adhoc queries), which, in general, can be subsumed under the OLAP requirements.

The OLAP Report [Pendse et al, 1995] defines the critical characteristics of OLAP as FASMI (Fast Analysis of Shared Multidimensional Information) in which the terms have the following meanings:

Fast:	5 to 20 second response time for realistic amounts of data
Analysis:	supports both predefined and adhoc queries
Shared:	supports many users while maintaining security
Multidimensional:	provides a business view of data
Information:	handles all volumes of data in a timely, integrated fashion.

The key characteristic is "multidimensional analysis," which is based on accessing information from a strictly business perspective. The dimensions of the enterprise are modelled in one or more business models (see Figure 1 and box) and the information can be analysed in terms of these models with access at all levels from the highest level of consolidation down to the lowest level of data. Business information to support OLAP takes the form of raw data plus consolidations a t varying levels of granularity. Note that a special form of database is not necessary (see Figure 1).

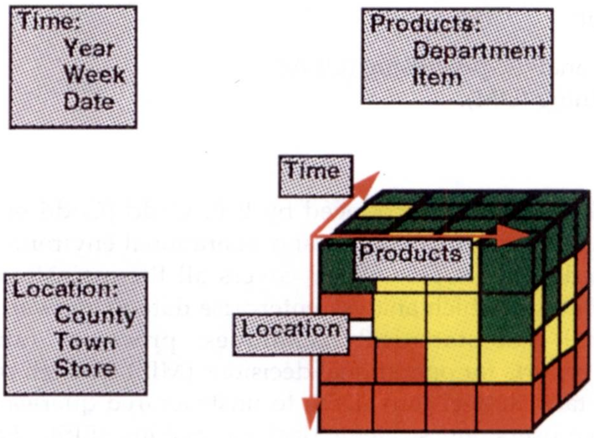


Figure 1 Retail Business Model

Business Model
An enterprise business model represents the enterprise in a subject-oriented way, with *subjects* corresponding to the perspectives, or analysis categories, required by the enterprise (for example, *time* is a subject in almost every business model). Each subject in turn normally has a hierarchical structure of members (for example, *date &time* can have a hierarchy of years, quarters, months, weeks, days, hours). The data items themselves are often held in a *variables* dimension (also known as facts, measures, etc.); dimension members may include primary data items such as volume, cost and price, and derived (calculated) members such as total price.

The *subjects* are the dimensions of the model. The model can be considered as a hypercube whose dimensionality is the number of subjects, the basic data thus conceptually being a multi-dimensional array or data set. The facts may be viewed at various levels of granularity, from the most detailed (the basic facts themselves) to the most refined (aggregations of the facts in all the hierarchical levels of each subject). Note that an enterprise may superimpose any number of models onto the underlying data, to represent the different perspectives of users (e.g. the marketing perspective, the logistics perspective).

Three *subjects* of almost any business are time, location and product (that is, any raw fact about the business has associated with it the dimensions of time, location and product - e.g. a retail sale is a sale on a certain *date* at a certain *store* of a certain *product*). As an example, consider a typical OLAP query:

> "show total sales in Q1 this year against Q1 last year for all sales in all stores in NW England"

This query:

- is multidimensional (involving all three dimensions of location (all stores in NW England), product (all sales) and time)
- requires time-variant data; i.e. current and historic (this year and last year)
- requires aggregation of the basic sales data; i.e. the underlying detail (location: across all stores in NW England; products: all sales; time: all dates in Q1).

Having displayed the results of this query, users could then "drill" down to a lower level of detail. If, for example, the total sales figure for one particular store in Q1 was significantly different from the expected value, then they could display the monthly figures for that store.

4.2 Data Mining

A second, and increasingly important, form of information is that resulting from sifting through data to discover implicit patterns, trends and relationships, using a range of techniques covered by the term Data Mining (DM).

Data mining processes exploit information in a way that extends beyond OLAP querying or reporting. These processes aim to complement human intelligence in the decision making process. The main use of DM is in discovering knowledge, that is revealing concealed patterns, trends and relationships inherent in the data but unknown to users. A current definition of a Data Mining system is one that:

- uncovers useful, previously unknown information
- gives an accurate portrayal of the contents of the database
- offers a user-friendly way of displaying the information

and DM systems can:

- discover information (patterns, trends, relationships) in corporate data
- search for answers without knowing what to look for.

The discovered knowledge should lead to a visualization of the data, a predictive model or a set of decision rules enabling a deeper understanding of the factors (e.g. customers, products, processes) that can improve a company's business. This knowledge can be verified against a larger data set and then exploited in some way.

At the discovery and verification stages, users of data mining will typically be a small number of specialist analysts who understand the enterprise data at the detailed level. The data they use will, most probably, be the pre-processed factual data (rather than aggregated data). The exploitation stage may well also involve these specialist users, or it may be embodied into some automated business process. At the present time, the techniques are usually employed against sampled or subsetted data, rather than against large volumes. This is a consequence of current products and newness of use and, in the future, we may expect to see larger and larger volumes mined. As with OLAP, the data being mined needs to be integrated but, unlike OLAP, there is no inherent requirement for a business view.

As an example, consider the benefits to retailers in discovering combinations of products which sell well together - affinity analysis. Mining tools can sift through market basket data and reveal such affinities. A classic example of this in action was the case of the beer and nappy (diaper) sales at the US store, Wal-Mart, who discovered that sales of both these items increased on a Friday evening, when fathers picked up the nappies on their way home. By placing premium beers next to the nappies, WalMart both increased beer sales overall and, since these sales were of premium brands, further increased profits.

5. Perspectives on Business Warehouse Architectures

This section describes the main ways in which a Business Warehouse Architecture supports the four OPEN*framework* [Brunt, 1993] perspectives of *enterprise management, application developers, users* and *service providers.*

5.1 The enterprise management's perspective

The enterprise management determines how the information systems will help the enterprise gain competitive advantage. Integration of operational data into a central information system, providing a single, consistent view of corporate data, increases competitive advantage by:

- improving *decision taking* at all levels
- improving *operational processes* by complementing existing product-based views with subject-based views
- providing a *framework for organizational change,* such that changes in the operational IT environment can be incorporated in the business view with minimal disruption and cost
- facilitating the *redesign of operational processes* (if required) by providing cleansed data.

5.2 The application developer's perspective

The requirements for enterprise data/information can be partitioned conceptually between the operational and decision environments, corresponding to data collection and use for day-to-day running of the business in the operational arm, and information use for decision making at operational, tactical and strategic levels. Application developers are challenged with satisfying the requirements of both environments, which differ in terms of data scope and integrity, data storage and data access.

5.2.1 Operational data

The operational environment is concerned with the day-to-day running of an organization, where the raw primary data is collected, updated and used. This is typically an on-line transaction processing (OLTP) environment, with data held in single dimensional, atomic records in a number of disparate data sets. Secondary integrated data is also used in the operational environment to complement existing OLTP operations (e.g. by providing a customer-focused view across disparate product views) and to exploit knowledge gained in the decision environment.

These operational environmental characteristics for primary data can be summarised [Inmon, 1993] as:

• current data:	that which reflects the current (and constantly changing) state of the enterprise
• transaction driven data:	that which is accessed within transactions, and stored to optimise such access (i.e. held in single-dimensional, atomic records for fast processing)
• volatile data:	that which has values that are changed frequently
• disparate data sets:	those which are determined by operational processes (applications).

5.2.2 Decision data

The role of a decision environment is to support decision making throughout the organization. The two categories of decision support are:

- OLAP (on-line analytic processing), the well-established MIS/EIS (management/executive information systems) techniques. This involves analytical processing of multi-dimensional (i.e. subject-oriented) views of consolidated data, with ability to "drill down" to finer and finer levels of detail
- DM (data mining), the emerging use of AI (artificial intelligence) tools (e.g. neural nets) to discover hidden patterns in data.

The data required for both is effectively a long series of snapshots where:

- each snapshot reflects the state of the enterprise at some defined moment in time; e.g. end of a trading day (contrasting with OLTP, where values are volatile)
- data includes current and historical snapshots (i.e. is time variant)
- data may be viewed in terms of the subjects of the business
- the data is clean, providing a single, consistent view of corporate data (i.e. the data is integrated)
- external data (e.g. weather, post codes, share indexes) may also be included.

5.3 The User's Perspective

A BWA addresses the requirements of two classes of user, operational and decision.

Operational users, above all, do not want their service performance to be impacted by decision users.

OLAP users, like operational users, require on-line access with predictable response times. However, their response time requirements are less critical and they will accept that some queries may take longer to execute. They may not need continuous seven day availability.

OLAP spans a range from the predefined queries of many DSS users to the unstructured queries of a few EIS users; these groups have different requirements in terms of availability and response times. DSS users will want predictable response times for the majority of their queries, whereas EIS users will want predictable response times for some queries, but may accept longer times for adhoc queries.

OLAP users may also be mobile, requiring to log-in over the public network; thus they may want to instigate a query, log-out and then reconnect for the result. They will want to use standard laptops rather than more powerful machines.

5.4 The Service Provider's Perspective

Provision of operational and decision environments can involve complex data integration. Staging and control of all this is a key concern of the service provider. A BWA, with its distinct stages for data movement supported by tools and meta-data, provides a means for control.

6. The Business Warehouse Architecture

6.1 The Architecture

The Business Warehouse Architecture is shown in Figure 2, where the differing needs of the two environments, operational and decision, are reconciled by the introduction of a conceptual IDS. This IDS presents the decision users with integrated, non-volatile, time-variant information. The IDS data derives from the primary (operational and external) data sources via a synthesizing process, which integrates this disparate primary data before migration to the IDS. The logical components are:

- a synthesis engine which determines how the IDS data is derived from primary data
- an IDS which presents the analysis engines with the data they require, and presents data directly back to operational applications and (where required) to decision applications
- a DM analysis engine which incorporates the DM tools
- an OLAP analysis engine which presents the decision users, via their chosen applications, with the multi-dimensional, subject-oriented data views that they require
- two sets of metadata: these contain the "data about the data," which underpin the two processes.

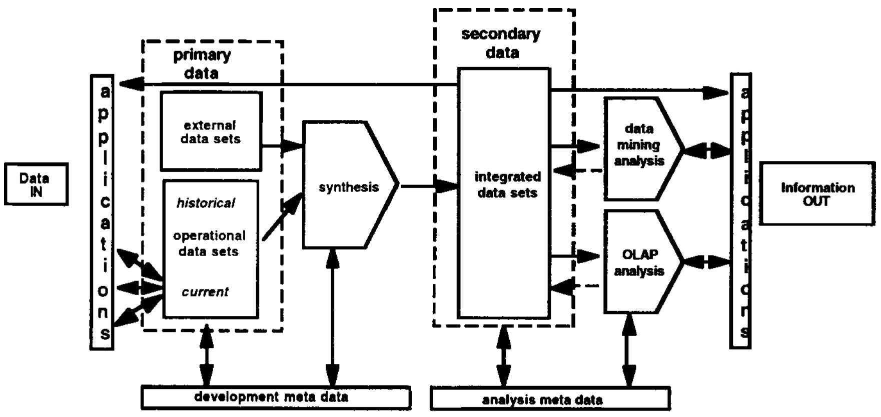


Figure 2 The Business Warehouse Architecture

To give some idea of the effort involved in building a data warehouse, an initial pilot instantiation could take from three to six months, with up to 80% of the effort going into the synthesis stage. Expanding the scope incrementally could then require a further three to six months for each iteration.

These logical components are now described in more detail.

6.2 OLAP Analysis

6.2.1 Analysis Engine

The analysis engine is the link between the users' applications and the data available from the IDS. Fundamental to its use is the underlying meta-data, required for the process of analysis, which provides a repository for data definitions, data descriptions and calculation formulae. The engine:

- embodies the enterprise business model(s)
- provides dynamic mappings between a business model and the data model which represents the underlying data available from the IDS
- provides customised views of the data
- provides an API for users' applications and supports standard data interchange formats, both to standard front end tools (e.g. spreadsheets) and to groupware distribution tools
- provides predictable response times
- incorporates query policing and scheduling capabilities to reject (or at least check) queries that appear to have long execution times and to allow prioritizing and deferred execution of acceptable queries.

To map between the analysis engine and the IDS, the IDS in turn must present, within its API and data model, data objects corresponding to business subjects. This mapping is set up within the analysis engine at design time. Typically, the engine will also hold user-configurable calculation formulae, templates, filters etc. for viewing selections of the data.

The analysis engine has an API which is business-oriented and supports the range of business information views and presentation capabilities required by OLAP applications; there are currently no standards for this. Front-end applications and application development capability may be provided in one or more ways:

- an engine-specific packaged front-end application, e.g. a multi-dimensional viewer
- data export to standard front-end tools such as spreadsheets (this must be considered as essential)
- an engine-specific application development environment (ADE)
- libraries to support the API for other ADEs (e.g. a Visual Basic DLL)

6.2.2 Meta-Data for Analysis

The capabilities of the analysis engine are underpinned by the analysis meta-data, comprising:

- the business model (i.e. the multi-dimensional structure)
- the data descriptions corresponding to the views of the business model used by applications
- mapping between these data descriptions and the IDS data model - consolidation paths and calculation formulae
- IDS-access optimizations, e.g. optimized queries, indexing.

Note that the meta-data does not itself contain any facts or aggregations of facts. However, it is dependent on the IDS wherever it uses the API of the IDS.

6.3 Data Mining Analysis

Fitting DM into the BWA presents some difficulties due to the relatively immature use of these techniques in the corporate environment, the great variety in the form taken by the rule/model output, and the lack of standards. For simplicity, we choose to represent DM architecture as a single engine mediating between the IDS and the analyst.

The *discovery* stage of DM (see section 4.2) falls firmly into the decision environment of the BWA, and involves on-line, interactive use of the DM engine. In contrast, *verification* and *exploitation* are essentially off-line (batch) type processes applying the discovered knowledge. These processes may either use the DM engine itself (for example, where the technique is a neural net), or may use the rules/model output of the engine in some way (for example the rules could be embedded in a new application program).

Thus the engine may run in one of two main modes: *discovery* or *exploitation*. For both modes, input to the engine is a dataset selected from the IDS, whereas output depends on the particular mode.

In discovery mode, the output comprises:

- the data structuring rules
- the knowledge discovered (rules, model etc.)

In exploitation mode, the output represents the application of the rules/model to the structured input data.

The input API is a standard data input method: the engine should support the API of the IDS. The form of the output is dependent on the DM tool employed: there are currently no standards in this area.

There is no known current use of meta-data in data mining.

6.4 The Integrated Data Set

Conceptually the Integrated Data Set (IDS) is a single database holding integrated enterprise data which it then provides to OLAP and DM applications. Where it also provides data back to the operational environment, this operational data should be held in a physically separate database to avoid decision queries adversely affecting operational response times. Current thinking is that, to preserve the single, consistent view of corporate data, users cannot update the IDS, they can only read it.

In addition to conforming to the requirements discussed in section 5.2, the IDS must be:

- responsive; the IDS must support the required query response times. It must not have an adverse impact on operational performance
- secure
- manageable.

Other areas which must be covered by the technology are:

- how the IDS is physically instantiated: e.g. by virtual (dynamic) mappings to one or more physical databases
- how the data is stored: degree of simplification/de-normalization, ranging from complete normalization for maximum flexibility regarding queries and future uses, to de-normalization for improved speed of access
- what data is stored:
    - summarised data (at varying levels) may be held either in the absence of detailed factual data (to reduce data volumes) or together with the detailed data (to speed access)
    - detailed current and near-current data should always be present. To reduce data volumes it may be sufficient to hold only summarised historic data, although the detailed historic data should be available from an archive if required.

These topics are described in more detail in section 7.

6.5 Primary Data

Primary data comprises:

- current operational data about the enterprise: typically thirty to ninety days of data
- historical operational data about the enterprise: decision users require five to ten years of information for trend analysis
- external data.

Key aspects of operational data (resulting from factors such as downsizing, distributed companies, acquisitions and mergers) include:

- diversity of data sets: data stores will often be held in physically heterogeneous and disparate data sets
- diversity of data representations: each separate data set could hold the same semantic information in a different syntactic way (for

example, "m," "M," 0, "male" to denote gender and inches or centimetres to denote linear measurements).

- duplication of data: the same semantic information could be held in more than one data set. This problem is increasing with increased use of packaged applications, for example, each package may hold its own version of customer address details
- change of data field use over time; a given data entity could represent different information from one time period to another (for example, an insurance policy with type code "HSR" could have widely different conditions when issued in 1995 compared with its cover in 1985)
- ownership; these data sets are owned by different departments/companies (implications for privacy etc.).

6.6 Synthesis

The synthesis process must extract, cleanse, transform, integrate and transfer the disparate primary data to achieve the integrated view presented by the IDS. It must therefore address all the aspects of the primary data outlined above. The main work in this area arises at the development stage:

- identification of sources of required data
- development of transformation/cleansing algorithms, applying business rules where appropriate
    - integration of data from different sources
    - elimination of duplicates
    - insertion of missing data
- mapping data on to IDS schema
- management of initial transfer to IDS (assuming a physical database will exist for the IDS)
- management of updates to the IDS
- management of requests to "reach-through" to source data via the IDS.

The scale of this task should not be underestimated: it is widely agreed that it requires 70% to 80% of the effort of implementing a BWA. Clearly the amount of work depends on the number and diversity of the primary data sources. Operationally, the tasks of periodic update and of "reach-through" requests consist of data extraction, transformation and integration.

For an enterprise scale environment, management and automation of the synthesis process is essential, and this is where the development meta-data comes in. In addition to details of the target IDS (i.e. its data model or schema) the meta-data captures for each data source:

- details of that source (platform, DBMS, schema etc.)
- details of data items to be extracted
- mapping of items onto IDS
- business rules for transforming data.

6.7 Meta-Data

The two types of meta- data, required for development and analysis, have already been discussed above in conjunction with the processes (synthesis and analysis) that they underpin. The relatively immature state of the technology in this area means that there are no standards for meta-data access and that the two types remain separate. A recently formed consortium of tools vendors (The Meta-Data Council) has been formed and is addressing this issue.

7. The Integrated Data Set

This section examines aspects of instantiating the IDS including:

- databases and data staging: how the IDS is instantiated as one or more physical database(s) in a one, two or three stage architecture
- database structure: how the database structure depends on the staging architecture
- database management systems: the DBMS technology commonly employed: relational and multi-dimensional.

7.1 Databases and Data Staging

A virtual IDS, with all data being accessed on-the-fly from the operational environment, is rejected as impractical for large or complex enterprise systems. Three configurations for physical IDS instantiations are identified:

- one stage: single IDS database (Figure 3)
- two stage: primary data is fed into a single central integrated database from which subsets are taken into second stage decision data marts and operational data sets (ODSs). The marts can each address the requirements of a specific business group, giving that group control over its own analysis data and allowing it to choose the most suitable analysis tool for its purposes. The ODSs can be used via OLTP to support business processes, and via batch

applications to exploit information from OLAP and DM analyses (Figure 4)

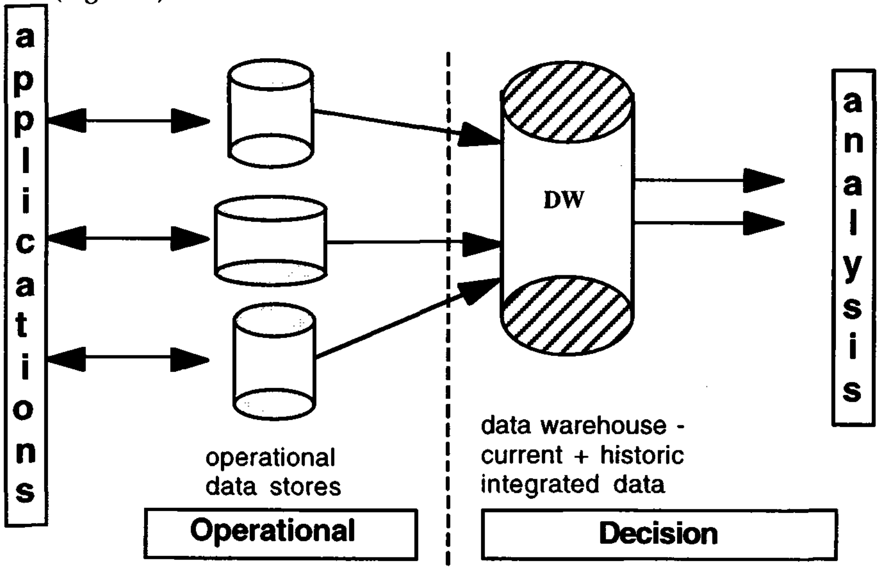


Figure 3 One Stage IDS Architecture

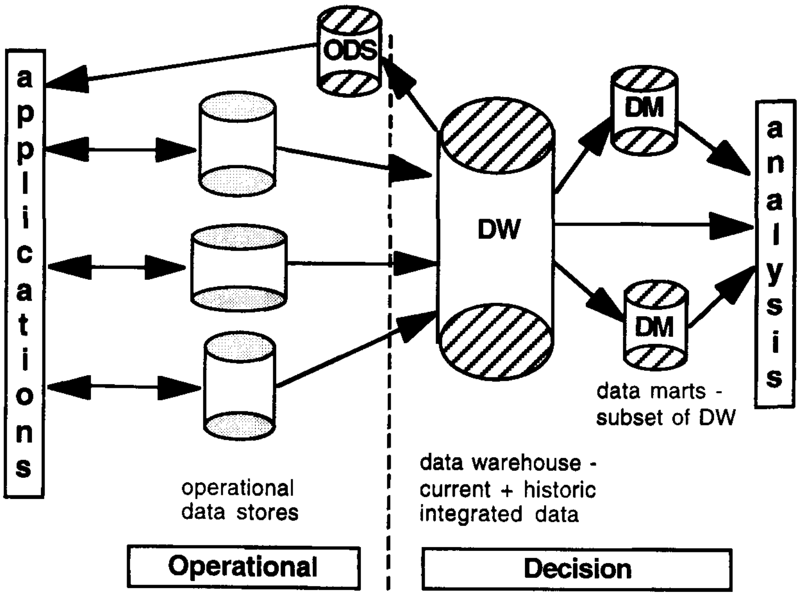

ODS
applications
DW
DM
DM
analysis
data marts - subset of DW
operational data stores
data warehouse - current + historic integrated data
Operational
Decision

Figure 4 Two Stage IDS Architecture

- three stage: a new, single operational data set (ODS) is introduced as a first stage before the central and data marts of the two stage configuration. This virtually up-to-date yet integrated ODS opens up new possibilities for application migration and Business Process Re-engineering within the operational environment (Figure 5).

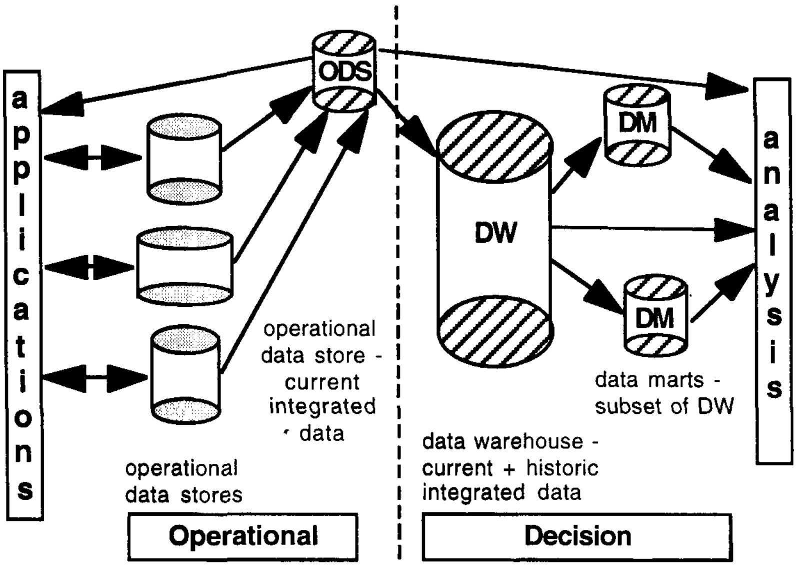

applications
ODS
operational data store - current integrated data
operational data stores
Operational
DW
DM
DM
data marts - subset of DW
data warehouse - current + historic integrated data
Decision
analysis

Figure 5 Three Stage IDS Architecture

An essential feature of all these schemes is a single central data warehouse. The alternative architecture in which the primary data feeds directly into a set of data marts must be considered strategically unsound, since the unified discipline of the central store is missing, and, without such a central store, integrity across the marts cannot be ensured, unless the synthesis process for all data marts is rigorously controlled via a single set of metadata. Where data marts are used, detailed data can remain in the central store with queries from marts reaching through to it as required. All data within the IDS stages adheres to the IDS data model, so no cleansing is required between stages.

Today, the choice is usually between a one or two stage configuration, and will depend on the individual requirements of the enterprise. A two stage IDS offers:

- potential for change: the central store can be thought of as a reception store for integrated data. It can be:

    - structured to retain flexibility for known and future uses
    - queried indirectly for detailed data i.e. reach-through from data marts
    - queried directly for ad-hoc queries and more time-consuming analyses
- updatability: although the central warehouse should maintain its single, consistent view of corporate data and not be updatable from any analysis tool, data marts do allow different users (or departments) to own, store and share extra information, e.g. user-defined aggregates of data, scenario (budgeting and planning) data
- operational use: the results of analyses may be exploited via new business processes; for example, a one-off targeted mailshot following analysis of customer data. By subsetting data into a second stage operational data store (ODS), a new application can be produced to implement this new (batch) process. Equally, integrated data may be of value directly in OLTP operations, giving a subject-based view (e.g. a customer database). In both cases, operational use is kept distinct from the decision environment, ensuring that both environments can be independently optimised.

7.2 Database Structure

There are four database cases to consider:

- central warehouse in multi-stage IDS: the database is not used directly for analysis (though analysis tools may reach through to it for detailed data)
- central warehouse in single stage IDS: the database is both the central warehouse and the analysis database
- decision data marts: databases are subsets of a central warehouse and are used only for analysis
- operational data sets: subset databases are used back in the operational environment.

For a central data warehouse in a multi-stage IDS, the structure of the central warehouse must be such as to retain greatest flexibility for current and future business requirements. The warehouse does not, in itself, need to present a business view of the data if all OLAP analysis is via the second stage data marts. An RDBMS must be the choice, with the data structure normalized as far as is possible taking into account storage requirements and the degree to which historic data detail can be sacrificed.

For a central warehouse in a single stage IDS, the structure must be tailored for analysis use whilst retaining flexibility for change. A data mart does not need to compromise in this way: it can be structured for optimum performance against its analysis tool. For OLAP the prime analysis requirement is, as we have seen, presentation of integrated, subject-based information, which in turn involves accessing the basic facts as if they were a multidimensional array, where each dimension represents a subject of the business (e.g. time, products, geography). An analysis database has to support this view and to deliver the data with predictable response times: its design is crucial to the success of a BWA.

Operational data sets will be structured for the process(es) which they are to support. Thus an ODS may be structured for OLTP to supplement existing operational processes or for batch work to exploit OLAP or data mining knowledge.

The challenges to the technologies are:

- raw data: current requirements for raw factual data in the central data warehouse are for up to 1 terabyte, but this is predicted to increase with the normal growth of primary data and if and when multimedia data is to be included. The volume of raw data in a data mart will normally be less, as only a subset of the data is held there
- aggregates: some pre-calculation of aggregates is essential to ensure reasonable response times. This not only increases the amount of data to be held, but also imposes complexity on the data update (synthesis) process. Holding aggregates for every dimension can represent a significant increase in size over the raw data, and will be impracticable for large databases
- indexes: for an RDBMS, access time for some types of query can be significantly improved by storing indexes to the data. The storage requirements for these can increase the size of the database by factors of up to 3, and will increase the computing time needed to complete the synthesis process
- data model: the data must be held in a way which supports fast access from business views, but does not unnecessarily restrict the views. Efficient storage with good handling of sparse data is important.

7.3 Database Management Systems

There are two contenders for the DBMS managing the IDS databases:

- RDBMS: this is the database of choice for the central database in a multi-stage IDS and for large data marts, where data volumes exceed 10 Gbytes. An RDBMS supports OLAP analysis by using a star or snowflake schema - this is known as ROLAP (Relational OLAP)

- MDDB: these specialised multidimensional databases are wrapped in with their OLAP tools. They are very efficient for OLAP, but lack the required flexibility and have difficulty delivering an acceptable performance on databases in excess of 10 Gbytes.

Further factors in favour of an open RDBMS rather than a proprietary MDDB include:

- manageability: the sheer size of the central data warehouse imposes new demands on management of the data, in terms of tools and strategies. For example:
    - data load/change data capture; e.g. amount of data involved, update scheduling, indexing/aggregating, updating versus rolling period archive/insert
    - archiving strategy; e.g. when/where, media for historical data
    - backup/restore strategy, standby requirements
    - mirroring/RAID requirements
    - security management.
- platform exploitation: as section 8 describes, parallel processors are ideally suited to large warehouse implementations. But parallel execution within a query demands a database manager which exploits the capabilities of the platform. Today, only a parallel RDBMS can deliver this.

8. Parallel Processing and the BWA

Parallel processors, and the database managers to exploit them, are key to a large warehouse installation. Only a parallel machine and an RDBMS can support the data volumes (10 Gbytes to terabytes), which a large enterprise will typically generate, and support the on-line analysis of that data with acceptable and predictable response times.

In any large warehouse, the ability to perform adhoc queries and on-the-fly calculations is essential (it is not possible to hold pre-calculated summaries at all data levels, as the storage demands rise exponentially). Even where data is subsetted into small data marts (less than 10 Gbytes), there will usually be a need to interrogate the central database for detailed information.

An RDBMS executing on a massively parallel processor (MPP), such as the ICL *GOLDRUSH Mega***SERVER**, can execute queries against the databases in realistic time-frames, by splitting the work across many processing elements [Watson & Catlow, 1995]. Furthermore, many such queries can be executed in parallel, ensuring that "the SQL from hell," resulting from one user's query, does not adversely affect other users.

As well as responding to the individual requests from users in a timely fashion, an MPP provides linear scalability as user populations and/or data volumes grow: and all analysts agree that data volumes will grow. Quoted rates of increase in data volumes are 50 Gbytes per year.

9. Conclusions

9.1 A Business Warehouse Architecture

Businesses have two potentially conflicting requirements:

- timely decision making: as businesses become more and more competitive the need for timely decision making based on a single, consistent view of corporate data becomes more imperative
- responsiveness to change: at the same time businesses need to remain responsive to organizational change, allowing for departmental flexibility within each business as well as corporate-level restructuring.

Such requirements can be satisfied by a business data warehouse where, at its simplest, corporate data is integrated and stored to provide a single image of cross-organizational data. This integrated data is available both to decision makers for complex analysis and to operational users for new business processes.

But a data warehouse is not an "off-the-shelf" product. Rather it is a custom-built solution for an individual business, and the business warehouse architecture presented in this paper offers a framework for a choice in the building process.

9.2 Business Warehouse Solutions

The architecture described in this paper has highlighted the various components of any business warehouse and their interrelationships. Designing and implementing such a warehouse requires business knowledge and consultancy, systems integration and technical skills, partnerships and platforms. ICL, through its group-wide ICL Business Warehouse Solutions initiative, encompasses these areas of expertise.

- ICL has invested in Business and IT Consultancy services, focused on clients' business issues such as strategy, direction setting, requirements analysis and specification, information and high level system design. ICL Retail Systems has already invested heavily in

Data Warehousing with the launch of Precision Retailing both in the US and the UK
- technical architecture, design and implementation services are available
- through the Business Warehouse Associates programme, ICL has relationships with a number of vendors with product offerings in specific areas related to data warehousing
- platform products include: ICL *GOLDRUSH Mega***SERVER** & Nile Series, the ICL search accelerator and Fujitsu/ICL servers.

References

BRUNT, R.F., "An Introduction to OPEN*framework,*" *ICL Technical Journal*, Vol 3, Issue 3, May, 1993.

CARNELLEY, P. and CAPPELLI, W., "Beyond the Data Warehouse," Ovum Ltd., 1995.

CODD, E.F., CODD, S.B. and SALLEY, C.T., "Providing OLAP to User-Analysts," Codd & Date Inc., 1993.

INMON, W.H., "Building The Data Warehouse," Wiley, 1993.

PENDSE, N. and CREETH, R., "The OLAP Report," Business Intelligence, 1995.

SPRAGUE, R.H. and CARLSON, E.D., "Building Effective Decision Support Systems," Prentice Hall, 1982.

WATSON, P. and CATLOW, G., "The Architecture of the ICL GOLDRUSH MegaSERVER," *Ingenuity - the ICL Technical Journal*, Vol. 10, Issue 2, November, 1995.

WATSON, P. and ROBINSON, E.H., "The Hardware Architecture of the ICL GOLDRUSH MegaSERVER," *Ingenuity - the ICL Technical Journal*, Vol. 10, Issue 2, November, 1995.

More detailed information on ICL Business Warehouse Solutions may be obtained from:
Graham Yule, ICL MAN05 (7272) 2725
+44 (0) 161 223 1301,
g.s.yule@man0504.wins.icl.co.uk

Biography

Margaret Derbyshire holds degrees in computer science from Manchester University, is a Member of the British Computer Society and is a Chartered Engineer.

Her early work was with ICI, programming a message switching system. She then joined Ferranti where she had technical responsibility for language products on the ARGUS range. From 1987 to 1989 she worked at Manchester University on the Flagship project, a joint project with ICL, which was a precursor of the ICL *GOLDRUSH* parallel processor. Joining ICL in 1989, initially she designed enhancements to the Series39 transaction processing system, TPMS, for open interworking. Currently she is the strategist for data warehousing within High Performance Systems Division.

Virtual Reality as an Aid to Data Visualization

Doug Urquhart

ICL Retail Systems, Southport, CT, USA

Abstract

Since the advent of Point of Sale systems in the early eighties, the Retail community has been faced with the challenge of making sense of the huge volumes of data produced by such systems. Even with powerful mainframes, retailers have had to deal with data on a Summary and Exception basis, often discarding useful data in the process.

The limiting factor has been not just the processing power of the mainframe, but its inability to present its output in an intuitive form.

Graphics packages have improved matters to some degree, but are still clumsy when dealing with very large quantities of data.

The author believes that Virtual Reality systems, which allow the user to change scale and viewpoint in real time, can provide a powerful and intuitive method for data visualization.

The experimental system described in this paper was shown to some members of the Retail community at the National Retail Federation Show in New York in January 1996.

1. Introduction

Virtual Reality (VR) has a lot in common with the laser, which was once described as a solution in search of a problem. Until recently, platform costs had limited its applications to high-cost, high realism systems, such as the kind of flight simulators used by airlines, and low-end, pre-rendered games for PC users. Pre-rendering, the process of rendering each frame separately and then building the result into the product, to be played back like the frames of a film, limits the user's viewpoints to those anticipated by the designers of the virtual world: realism in a PC game drops off sharply if you attempt to step off the path.

With the availability of low cost, high power platforms, based on Intel's Pentium chip, it is now possible to construct Virtual Reality Worlds, based entirely on the PC platform, without recourse to more powerful processors to perform pre-rendering of frames. The result is a true VR system, with rendering being performed in real time.

The affordability of VR systems has now made them feasible as part of a Retail System, just as the PC had offered inexpensive in-store computing power and the Server a more cost effective alternative to the Mainframe. VR has the potential to become a visualization system, front-ending the Head Office processor or, as an aid to the Store Manager, the In-Store System.

2. The Retail Environment

The IT functions used in running a retail store are fairly straightforward; the data collected during a sale is used for several well-understood processes, namely:

a)	Security:	Does the till balance?
b)	Performance Monitoring:	How much money are we making?
c)	Prediction:	How many widgets do we need to order, to avoid running out?
d)	Speculation:	How can we make more money?

Mastery of these processes leads to successful retailing.

2.1 The Corner Shop

At the lower end of the retail spectrum is the free-standing Corner Shop.

In-store equipment is limited to one or more cash registers (maybe electronic), which are capable of calculating totals and printing them on request. A pocket calculator can be a help for calculating tax and discounts and for adding up the totals from several registers. In this environment, data collection is minimal.

Security:

The most important security feature of a cash register is the bell that rings when the cash drawer is opened. However, the cash register also keeps account of the amount of money in the till (the more sophisticated systems may break this down into cash, cheques etc.). At the end of the day, the money is counted to see if the two figures balance.

Monitoring:

The totals from each cash register are added. This gives an indication of daily sales. These items will probably be written down in a daily ledger for the benefit of the auditors and may even be traced on a two dimensional graph.

Prediction:

At the end of the day, staff *walk the shelves*, looking for gaps, and ordering more stock to fill those gaps. Based on experience, other items may be ordered before they run out.

Speculation:

This process is highly intuitive and relies on Retailers' knowledge of their trading environment, experience of past successes and failures, business acumen, analysis of sales results etc., but mostly on luck. It also relies heavily on feedback from the other processes.

2.2 The Retail Chain

A modern Retail Chain may have several hundred (or even thousand) stores, each containing a highly sophisticated Data Collection and Store Management system. There may be anything from one to three hundred Data Collection Devices in each store, ranging from Point of Sale Terminals, Customer Service Terminals, Hand-Held Radio Frequency devices, Customer-Operated Kiosks, burglar alarms, temperature sensors etc.

Stores are connected to the Head Office, either directly or through regional offices, via extensive communications networks, either satellite or terrestrial or a mixture of both. The network is traditionally used to collect data from the stores and to send updates to local files: a price change would be a good example of this.

Every attempt is made to ensure that the data is brought to Head Office as quickly as possible, so that the Head Office mainframe can process it by the beginning of the next trading day. This processing traditionally results in a set of reports, which are given to Head Office staff every morning.

These reports are then used for the processes of Security, Performance Monitoring and Prediction and, in addition, provide the necessary feedback to facilitate Speculation.

Well, that is the theory!

2.2.1 Trade-offs

In practice, the overwhelming volume of data leads to a number of trade-offs:

Localization:

Sophisticated in-store systems can handle the routine aspects of security, monitoring and prediction:

- balancing the tills
- controlling prices
- reporting on sales performance
- automatically restocking.

Some central control is lost in this process.

Summarization:

Rather than transfer sales data at the item level, the in-store systems might summarize their findings, and send this summary information to Head Office.

For example: rather than reporting that a widget was sold for $17.50 on March 15 at 12.17 pm to Jane Doe, at register 15, which was being operated by John Smith, the system might report that on March 15:

- 57 widgets were sold
- Register 15 handled 2000 transactions
- John Smith sold $5,172.50 worth of merchandise
- Jane Doe spent $125.25 on her total purchases.

This process involves substantial data loss.

For example, it might be highly significant that 90% of all widgets are sold between 11.00 and 11.35, or that a typical transaction takes 2.5 minutes. It might also be significant that Jane Doe is John Smith's sister and widgets normally cost $100.00!

In summarizing, this information is lost.

Exception reporting:

In addition to summary data, reports are made on occurrences which are out of the ordinary, for example:

- 100 widgets were returned
- John Jones was short of $100.00 when his till was balanced

- A widget was sold at $17.50, when its usual price is $100.00

This process is useful but limits the reporting to exceptions which were defined in advance: new anomalies might go unnoticed.

Store and Ignore:

Data is collected at the detail level, but is merely stored in the mainframe, to be available on demand. For example, it might be processed at a later date to determine transaction performance, or it might be sold to a vendor (who will use it to determine the performance of his own products, or those of a competitor).

In many cases, the data is never analysed. In fact, retailers are prepared to go to quite considerable expense to collect information manually at the store, rather than analyse the data at Head Office. Sending a team of people with stopwatches, to derive transaction timings, for example, is somehow considered easier than processing the data, even though it contains timing information.

This behaviour is usually blamed on the inflexibility of the Head Office system!

2.2.2 Data Visualization

Printed Reports

The typical media for data visualization are printed reports. These tend to fall into two categories: lengthy and unread or heavily summarized.

Lengthy and Unread:

It is not unusual for a retailer to produce a report containing several thousand lines, for example, when reporting on the stock availability levels of items from a particular vendor. This report is usually filed, unread, to be thrown away when a new report is produced. It may occasionally be used as a reference source when the on-line system is down, but it is too unwieldy for normal use.

Heavily Summarized:

Since the overall data volume is too large, it must be summarized, hopefully concentrating on important data. Typical rules would be:

- The best
  e.g. Highest sales, highest margins, fastest movers

- The worst
  e.g. Lowest sales, lowest margins, slowest movers

- The exceptions
  e.g. Unbalanced drawers, unusual pricing, abnormal number of voids.

There are several problems with this approach:

- the rules must be decided in advance (and hence the level of detail)
- the middle ground is ignored (which may contain the most interesting information)
- having established what are considered to be important exceptions, the retailer very rarely considers others.

On-line Terminals

Most Retail systems allow the user to make detailed enquiries about a specific object in the system. This information is typically displayed on a screen, in text form.

In addition, there are a number of tools offering the ability to display simple graphs based on system data. These typically are very limited in the amount of data that they can process, and in the degree of detail that can be represented on the screen.

3. Hardware Limitations

Head Office Computer System:

- Often held to be the bottleneck in Retail systems.

- Traditionally a mainframe, or large mini, with all the attendant problems of high cost and inflexibility: these systems are being replaced by client/server networks, with significant improvements in cost, performance and flexibility.

- The use of powerful servers to perform *Data Mining* is beginning to gain acceptance and, if properly managed, this approach should go a long way towards clearing the processing bottleneck.

Communications Network:

- There are still many operational retail systems which rely on dial-up communications using batch-orientated protocols, but their number is diminishing.

- The trend is towards dedicated circuits, using a number of technologies, such as satellite, leased lines, data over voice, ISDN, together with more effective protocols such as X.25.

- The bandwidth of these new communication channels should be more than adequate for retail applications.

Data Collection Devices:

- These have seldom been a system limitation and, in fact, they tend to collect more data than can be processed. Their individual cost has dropped dramatically over the last ten years and, as a result, they are being seen in larger numbers and being used for novel applications.

- The net result is that there is even more data to process.

Data Output Devices:

- When the very first Retail systems were installed, in the early eighties, they used printed reports and on-line interrogation screens to display the results of their processing.

- Modern systems still do this.

- To be fair, affordable high resolution graphics workstations were not available until PC technology became powerful enough: say within the last two years.

- This capability is now ready for exploitation.

4. Requirements for a Visualization System

High resolution graphics terminals are now capable of displaying very detailed images but these are not very large. Any attempt to display, for example, a three-dimensional graph based on Retail data volumes, is likely to be either too detailed to display or too big to fit on the screen.

This gives a first requirement:

Scalable:

The system must allow the user to see an overview of the data, at any level of detail selected by the user, in real time.

The whole point of Data Visualization systems is to present data in such a way that its meaning (in addition to its value) is apparent to the user. The user should be able to look at the visualized image and immediately grasp its significance. This gives rise to a second requirement:

Intuitive:

The images used should be simple to understand and relate to real world objects, rather than mathematical concepts (although they may be based on the latter).

Since the user requires control over the visualization system, in order to change the viewpoint, and level of data (and perhaps the content of the data), we have the third requirement:

Interactive:

The user needs the ability to inspect the data from any angle, distance or orientation, as well as to be able to *drill down* to obtain more detail than the representation can handle and to change representations or data contents.

The system must respond in real time.

5. Virtual Reality

5.1 Definition

Virtual Reality (VR) is an interactive computer environment which can be freely explored, examined and experienced in real time.

Note that this definition rules out:

Computer games:

- no free exploration, since most are based on pre-rendered frames and predefined viewpoints.

Computer generated movies:

- no interaction (as yet).

Graphics packages:

- free exploration requires real time scalability, which tends to be limited in such packages.

Although Virtual Reality is a relatively new field, it has already acquired more than its fair share of jargon. Virtual Reality and Retail terms used in this paper are defined in the Glossary. Many of these terms are related to the special requirements of the field. For example, the need to control the user's viewpoint in three dimensions has led to the production of several special-purpose input devices, notably the Spacemouse™ (which was used in our experiments). There is no clear leader in this field, however, and the existing devices are expensive. No doubt the evolutionary pressures of the market will remedy this.

5.2 Platform Requirements

The power of modern PCs allows real-time rendering (for example, 10 frames/second) of simple virtual worlds. A standard desktop Pentium is perfectly capable of handling quite sophisticated worlds, with shading, distancing and fully mobile viewpoints. (Reflections, shadows and smoothing await more powerful processors for real-time rendering).

The cost is now low enough to come within the range of most Retailers.

5.3 Does it fit our requirements?

Scalable

VR systems allow the user to change viewpoint, and move freely within the Virtual World. Thus the user can move back, to obtain an overview, or move in for more detail.

In order to conserve processing power, they often contain a distancing feature, where detail is displayed only when the user is close to an object. This may have multiple levels, and is normally defined when the world is being built, to reflect the processing power of the target machine.

So, for example, the user may see a graph derived from monthly sales data. On moving closer, weekly data will appear, and so on. (Most retail systems time-stamp to the nearest second, but it is doubtful if this level of detail would be useful in practice).

Intuitive

This is very subjective. A representation which is intuitive to one user may be totally opaque to another.

However, VR systems do allow easy experimentation, so that one can envisage a *modus operandi* of starting with conventional representations, such as three dimensional graphs, and then working with end users to determine the most appropriate visualization for their environments.

Interactive

By their nature, VR systems are interactive. The degree of interaction can vary, however, depending largely on the cost of the system and its input devices.

In its simplest form, interaction would be limited to allowing users to vary their viewpoint: to fly across the world. This can be accomplished using a conventional mouse, but is easier if a special purpose device, such as a Spacemouse is used. Such devices allow translation and rotation in three dimensions.

The next level, is to add the conventional *point and click* of a regular mouse, allowing the user to inspect a particular object and, perhaps, to *drill down* to a more detailed level.

There is then the class of *virtual gloves* and similar devices, which allow users to pick up objects and move them around. The VR package chosen for our experiment did not support this class of device, so we were unable to put together a world allowing this type of input. There are interesting possibilities for users if they are given facilities to reach into the data and manipulate it. Further study is needed in this field.

It would appear that a VR system could meet our (somewhat arbitrary) requirements, but it was far from clear whether the Retail community would agree. In order to test our requirements, and to gauge the level of acceptance, it was felt that it would be useful to construct a VR world and demonstrate it to some retailers.

6. A Practical Experiment

We used a VR development system called Superscape VRT™ to develop our Virtual Worlds. This package was chosen for a number of reasons:

- it was designed specifically for the PC environment
- it was fairly low cost
- it offered an acceptable level of performance on relatively small machines
- the author had seen it in operation, and was impressed
- minimal coding was required to produce the effects for which we were looking.

The development system supported the following VR features:

- construction of 3D objects
- colouring
- lighting (up to eight light sources)
- dynamics: movement, rotation, animation, friction, restitution, gravity
- distancing
- sound
- standard mouse, for *Point and Click*
- multidimensional mouse (or equivalent) to control movement
- 3D goggles
- multiple viewpoints, with six degrees of freedom
- clip art library, to shorten the learning curve
- ability to generate VRML, for use on the World Wide Web.

The development and display platform was a simple desktop Pentium (75 MHz), with a sound card and 16Mb of memory. For the purpose of demonstration, a 35" monitor was used, which enabled demonstrations to be given to small groups rather than just to individuals.

The objective was to construct a Virtual World, to demonstrate the concept to the Retail community in an informal environment, and to get some feedback on whether the approach was acceptable.

In the event, three virtual worlds were constructed:

- a store environment, showing how data visualization *in situ* could be substituted for the intuitive process of *walking the store*
- a demonstration of how very simple data filtering and scalability could allow qualitative information to be gleaned from a large amount of data
- a free-form, three dimensional environment, showing scalable data visualization techniques.

Data in all cases was hand-generated, but the package does have the ability to import data from an external database.

6.1 The Virtual Store

Most retailers are unfamiliar with the details of Virtual Reality, but have the intuitive view that its main use is for simulating actual environments: designing buildings or equipment, either real or imaginary. Several of the retailers, with whom I was in contact, had kitchen design packages in their store, and they consider this, correctly, to be a form of Virtual Reality.

Rather than leap directly into data visualization, then, the first world begins as the simulation of an environment most retailers are eminently familiar with - a store.

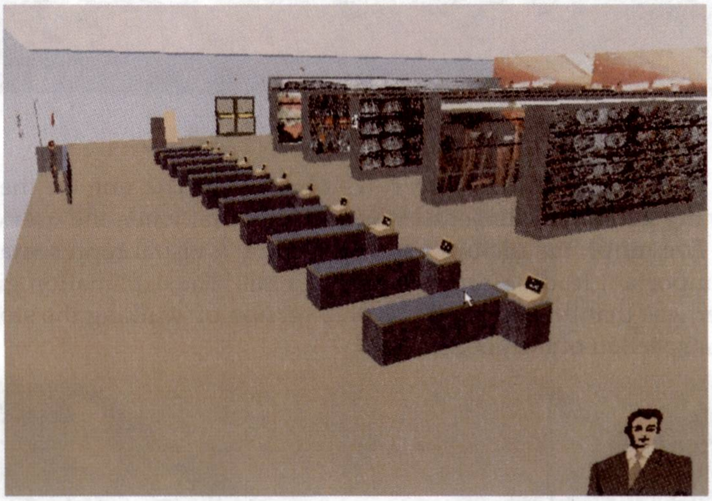


Figure 1 VR Store Interior

Figure 1 shows an interior shot of a hypothetical store. Several other viewpoints were available, illustrating among other things, that Virtual Reality could be used to decide on the locations of security cameras and advertising displays, and could be used to show interested parties what a proposed store might look like. We attempted to be as realistic as the system would allow, with self-opening doors, check-outs, shelves filled with items (whose images had been scanned from trade magazines) and we even managed to reproduce the annoying background music found in such places!

Once our audience was at ease with its surroundings, we presented the first Visualization.

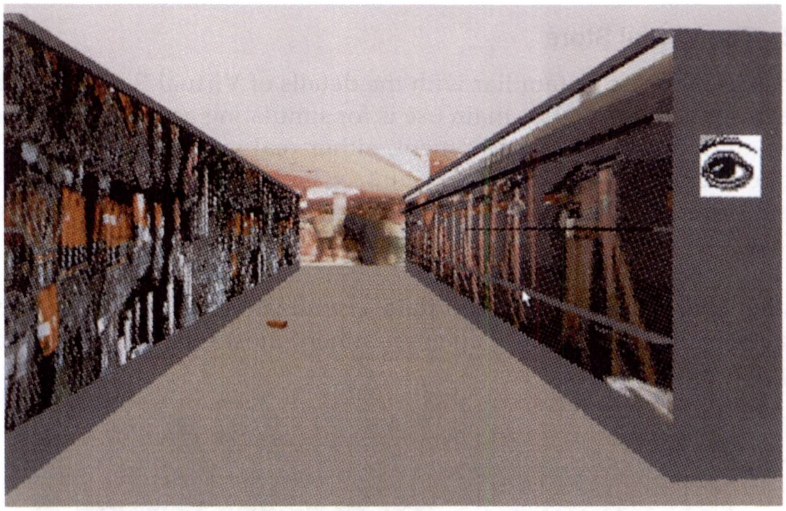


Figure 2 VR Store - goods on shelves

Figure 2 shows a view of two rows of shelves. The one on the right, containing garden implements, was set up so that when the user clicked on the Eye motif, the picture was replaced by a visual representation of some important trading property of the item. The explanation given to the user was that this was an electronic version of 'walking the store', i.e. visual inspection of the goods.

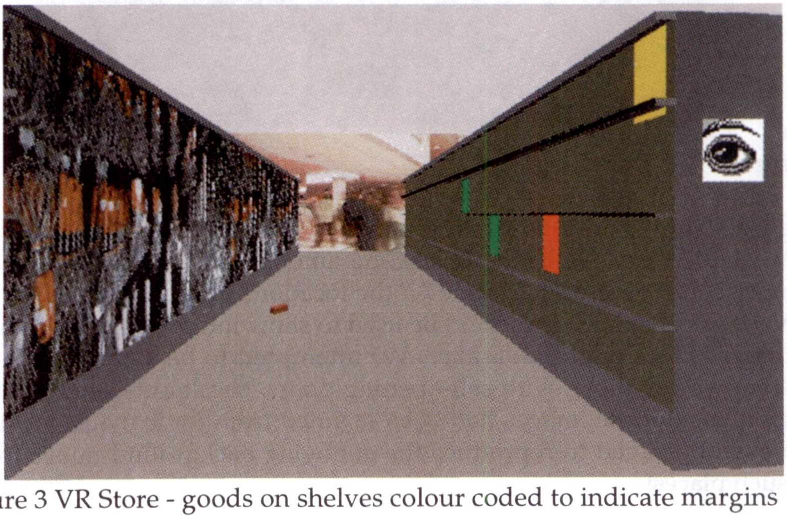


Figure 3 VR Store - goods on shelves colour coded to indicate margins

Figure 3 illustrates the first of these: Colour codes indicate the Gross Margin level of the items:

Light Green:	High Margin
Dark Green:	(The prevailing colour) Within acceptable limits
Yellow:	Low Margin
Red:	On sale below cost.

The user was able to click on a coloured square, to gain more information about the item (SKU, price, description etc).

Subsequent clicks on the Eye displayed different information:

Item velocity :	lighter colours indicating faster selling items
Stock levels:	lighter colours indicated 'overstocked' darker colours indicated 'time to reorder' black indicated 'out of stock'
Item location:	(in response to a Customer request) indicates the location of an item on the shelves.

Feedback from the retailers was very positive. The general feeling was that although this kind of information was readily available from more conventional packages, placing it *in situ* made it easier to understand.

It was time to move a bit further from home.

6.2 The Data Mine

Using the environment of a Data Mine (complete with dark tunnels lined with numbers, and the occasional rumble of distant blasting), we introduced the idea of deriving qualitative information from apparently disorganized data.

Figure 4 shows a 'Virtual Machine' capable of displaying 65,000 points of light on the far wall. Note that these are points of light on an object; they are not pixels on a screen. Thus, by varying our distance from the wall, we can vary the resolution of the data.

The display in Figure 4 is, in fact, a visualization of the results of performing a GMROI (Gross Margin Return on Investment) calculation on all 65,000 items in a product database. The intensity of each point relates to its Return on Investment (the brighter the better), and its colour relates to the vendor who supplied the product.

The result, as can be seen, is not encouraging - multiple areas of colour, with no obvious pattern.

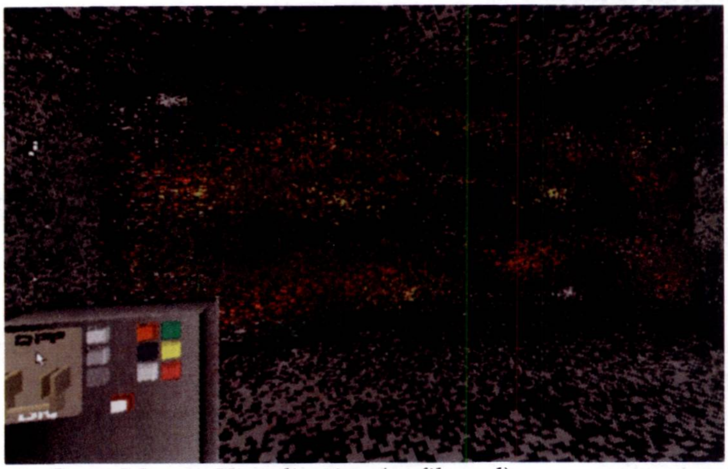


Figure 4 Gross Margin Visualization (unfiltered)

However, by introducing simple data filters, such as the one in Figure 5, which blanks out all but the highest intensity of dot, information can start to be gleaned, admittedly only of a qualitative nature.

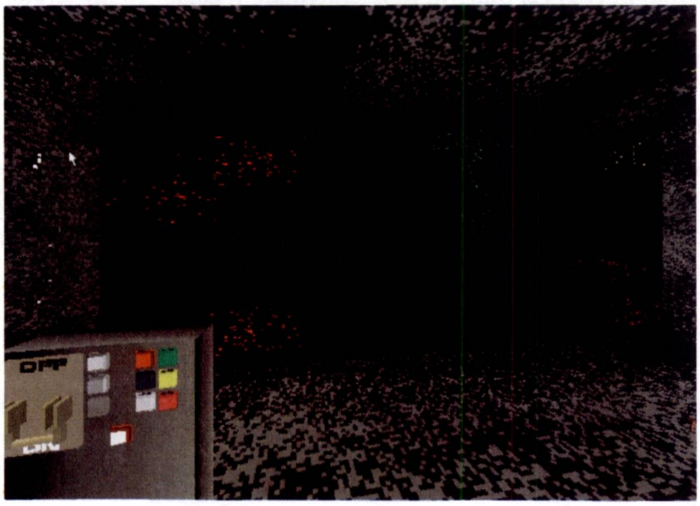


Figure 5 Gross Margin Visualization (filtered)

For example, the illustration shows that neither the white nor the purple vendors supply high-return items. In addition, the contribution of the yellow vendor is minor compared with the red and green.

The other data filters allowed the users to toggle the vendors themselves (permitting the contribution of a single vendor to be displayed). In this case, the overall intensity was significant.

Once a particular display had been selected, the user could then move closer to the wall, so that individual items could be resolved. A minimal degree of 'drill-down' was allowed in the demonstration - clicking on a point merely displaying the item number.

Response to this visualization was less enthusiastic, although this may have been due in part to the 35" monitor, which lacked the colour definition of a smaller screen and made it difficult to discern the various levels of intensity.

6.3 The Mountain Range

The remaining world contained three visualizations, laid out on an open plain. The user's viewpoint was 'flown' in three dimensions, giving the audience an aerial view of the constructs and adding a touch of realism. The effect was similar to a helicopter ride over mountains.

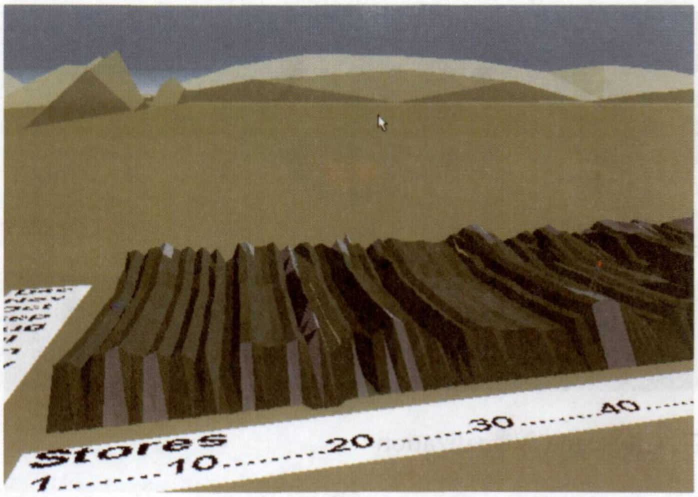


Figure 6 'Mountain Range' Visualization

Figure 6 is a three dimensional graph showing a year's sales figures from 100 stores, broken down by month. The resemblance to a mountain range was deliberately emphasized, by giving it a green colouration with particularly high values in light grey (snow capped peaks?).

The overall shape of the graph is a saddle, with most sales in December and January, reflecting a normal Retail year. Variations, however, can be seen: high performing stores appearing as ridges and occasional good days appearing as peaks. The audience had no difficulty in recognizing these features.

Since the viewpoint was highly manoeuvrable, the users were able to move back, for an overview, and zoom in to inspect points of interest. Some members of the audience commented on the fact that the graph not only showed the high and low points, but also the detail in between. They felt this was lacking in their present systems.

Future developments will include ground detail as the user descends. The sales data in the demonstration was resolved only to the monthly level: subsequent versions will show weekly, daily and even hourly resolutions, as the height decreases. It is technically feasible to display minute by minute data, but it is far from clear whether this would be useful.

6.4 The City

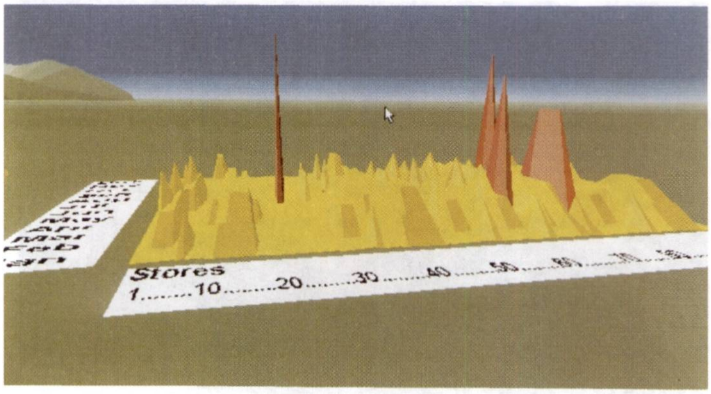


Figure 7 'Cityscape' Visualization

Figure 7 is another three dimensional graph, this time indicating out of balance conditions (Retail jargon calls these *overs* and *shorts*). These are occasions where the amount of cash actually taken in does not match the totals calculated by the system. *Shorts* are occasions where the amount of cash is too low. *(Overs* are the opposite and very rare). The height of the pyramid indicates the severity of the *short,* a ridge indicates a succession of *shorts.*

In any retail operation, there tends to be a background level of shorts, caused by human error and minor pilfering. This is shown on the graph as the clutter of smaller pyramids. More serious shorts, which indicate major breaches of security, are shown as tall pyramids, coloured red for emphasis.

By clicking on a red pyramid, the user can find out specific details, such as the store number, the date and the amount involved.

Again, it was possible to *fly the viewpoint* around the visualization to achieve the required level of detail.

Audience reaction was favourable. The comment was made that it would make an excellent *front end* to the ICL Loss Prevention package (Loss Prevention is the Retail euphemism for the detection of theft - a new ICL Loss Prevention package was on display at the show).

6.5 The Tree

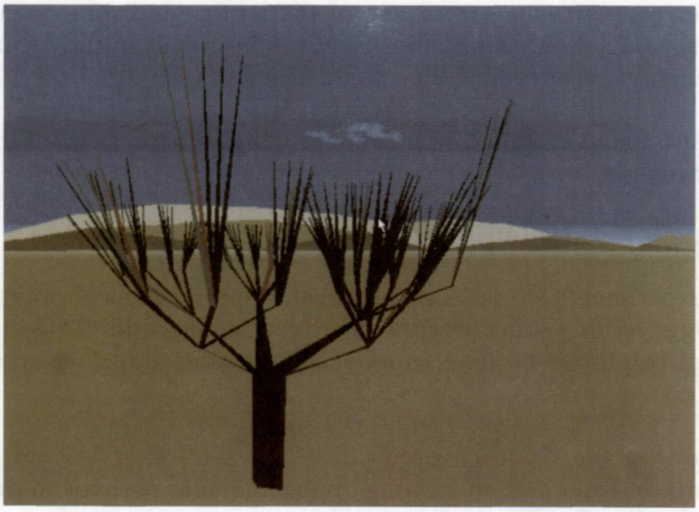


Figure 8 Tree Visualization

Figure 8 shows a somewhat unorthodox visualization but one with strong intuitive appeal. Each branch represents a store in a 100-store chain. Stores are grouped into regions and then districts.

The length of each branch is related to revenue, while the colour indicates profitability: light green is above average, green is average and brown is unprofitable. The pruning analogy is deliberate (and highly relevant in today's economy!).

The audience loved this one. A representative from a major retail chain remarked that a relatively unskilled individual (he used the term, "18 year old kid") could make major corporate decisions based on this visualization: a somewhat chilling prospect!

7. Feedback

The three Virtual Worlds were displayed at the National Retail Federation show in New York. This show is attended by a wide range of retailers, from owners of single stores to representatives from major chains like Wal-Mart. It also attracts a large number of Retail system vendors, including IBM, NCR and, of course, ICL, plus representatives from the major consultancy firms.

Reaction, in general, was positive. It was interesting to note that the most positive reactions came from one of the larger (and more successful) retailers, who not only found the visualizations highly intuitive but made some suggestions as to how they might be improved.

It was also found that there seemed to be a higher acceptance of the more realistic images.

8. Conclusions

Initial experiments would suggest that this approach is not only feasible but would receive some acceptance in the Retail world. As yesterday's video game players become senior executives, one could see acceptance growing.

Further work needs to be done, to investigate other Virtual Reality platforms, to front end an existing application and perhaps to pilot this system in a live Retail environment.

9. The Future

When speculating about the future of VR, one tends to concentrate more on its ability to simulate reality, than its ability to represent data, but there are exciting possibilities in both areas.

Reality Simulation:

- I want to buy a house which has not yet been built. There are five different designs.

  The developer sends me a CDROM containing five simulated houses. I can add furniture and decoration, and then move around inside the houses, until I choose the one I like.

- It is a bad day at the office. Ideally, I would like a holiday, but I do not have the time (or the money). Oh well, I'll just spend my lunchtime walking through the gardens at Kyoto. It is amazing how clear these new goggles are and the scrunch of gravel is a nice touch. I wonder when they'll be able to reproduce smells. I feel better already. I will use this service this evening to plan a real holiday...

- Shopping by wire? You do not need VR for that, unless you genuinely like walking the aisles of a supermarket. But what about clothes? I hate buying clothes. All you need is a personal avatar to send to the Virtual Tailors, and you can see how the clothes will look on you as well as find out what alterations need to be made (an avatar is an encoded representation of oneself. Current use of avatars is limited to visual representations, usually in online forums).

Data representation:

- I am moving to a new location, but don't know anything about the area. I would like somewhere with good schools, near the sea with low taxes and a low crime rate (oh yes! and affordable housing). The local authority send me a CDROM with a fully detailed map of the area. As I *fly around* it, I can select demographic information to display as areas of colour: the crime statistics, the catchment areas of schools, the tax rates. I can even zoom down to street level and see which houses are available for sale.

- I have a small portfolio of shares and I would like to add to it. I wonder which shares are doing well. I can connect to the stock market report, and see a screen which looks like a small city: every building is a stock, colour coded to indicate risk, height coded for performance. If I see an interesting stock, I click on it for details. I can then add it to my portfolio by picking it up...

- I am a retailer. I am currently plugged into my Head Office server, looking at all the data collected from my stores, for the past year. I am wearing 3D goggles and wearing VR gloves. All around me, the data forms a glittering cloud of chaos - no discernable pattern. In front of me is a virtual control panel, allowing me to change the meaning of the dimensions - cost versus revenue versus date or whatever. I let the server run through various combinations, looking for patterns. Suddenly, shapes coalesce out of the cloud. I reach in and turn the shapes, until they become clear. There is a correlation within the data - it may be the one which gives us an edge over the competition...

It is possible to speculate in this way for hours, with varying degrees of accuracy and imagination.

The important fact is that Virtual Reality has finally become generally available.

The next step is general use, which will drive down the cost of the technology, and lead to the production of better, more intuitive user interfaces. More importantly, it will lead to the development of exciting new applications which will exploit this technology in fundamentally new ways.

Glossary

Back Office: The domain of the Store Manager. Usually refers to a computer system within the store, which is used for local reporting, data concentration, and the gateway to the communications network.

Clip Art: (In the VR sense), predefined objects, often with complex animation, which can be incorporated into a virtual world, thus saving the developer from having to construct them himself.

Data Mining: The technique of using very powerful processors, clever algorithms and a great deal of luck to derive meaningful information from large masses of data.

Distancing: Reduction of detail dependent on the perceived distance of an object. Useful for reducing the processor load.

Dynamics: The ability to give a virtual object the apparent physical characteristics of a real one - elasticity, mass, inertia, friction.

Head Office: The nerve centre of a retail chain. The location in which the chain keeps its main computers, data centre and staff. Data collected from the stores ends up here.

Kiosk: A multimedia device, designed around a touch screen, intended for use by the customer (to order goods, obtain information, etc.).

Loss prevention: A retail euphemism for catching thieves. Retail is full of such terms; e.g. Business Continuation Procedures (meaning Disaster Recovery).

Multidimensional Mouse: An input device, based on computer mouse technology, allowing translation and rotation control in three dimensions.

Overs and Shorts: A report indicating discrepancies between the contents of the cash drawer and the amount calculated by the system.

Point of Sale Terminal: A sophisticated electronic Cash Register and Data Collection device.

POS: Point of Sale. The place where the customer actually parts with her money. There is usually a Point of Sale Terminal there.

Pre-Rendering: The practice of rendering a complex image, and then storing the result as a picture. This approach is used when rendering the image would take too long to allow real time interaction. A technique often used in PC games, capable of producing beautiful and realistic images, it severely limits the viewpoints of the user.

Rendering: The process of converting an image from its mathematical representation to a visible computer image. VR systems perform this in real time.

Shading: Rendering the colour density of an object in such a way as to give the appearance of the object being illuminated by a light source. Realism is improved as a result.

SKU: Stock Keeping Unit. An unique number used by a retailer to identify a specific product.

Till (UK): A POS terminal.

Till (US): The cash drawer.

Viewpoint: The current view of a Virtual World, as displayed on the computer screen. This viewpoint is varied using the control device.

VRML: Virtual Reality Markup Language. A text-based language used to define virtual worlds on the World Wide Web.(c/f HTML, Hypertext Markup Language - the ad hoc standard for defining Web sites).

VR World: A self-contained Virtual Reality environment.

Walking the Shelves (or Store): Making a visual inspection of the store, with a view to judging performance, and deciding which products require ordering.

Acknowledgements

Superscape and *VRT* are trademarks of Superscape Inc., of Palo Alto, California and Hook, Hampshire.
Spacemouse is a trademark of Logitec Inc.
Pentium is a trademark of Intel Inc.

Biography

Doug Urquhart is a Senior Industry Consultant, working for ICL Retail Systems, and based in Southport, Connecticut.

He is a graduate of the University of Glasgow and joined ICT in 1967, based in Reading. He spent his early years working on various design and development aspects of COBOL compilers on 1900, 2900 and later the System 25. In 1975, he joined the VME/K team, and spent some time providing on-site support for the European Space Agency, in Darmstadt, West Germany. In 1981, he joined the Retail group, and was responsible for producing the world's first COBOL-based Point of Sale Terminal.

In 1984, he moved to New England, where he has been ever since, as part of ICL Retail Systems. His involvement with the Retail Group has been wide-ranging, from Development through Marketing and Consultancy. In 1994, he won a Gold Excellence award for his part in the rollout of a major Retail Chain.

Re-engineering the Hardware of CAFS

Richard Illman

ICL High Performance Systems, Manchester, UK

Abstract

This paper describes a technique for upgrading chip designs from one chip technology to a more modern technology. This technique has been applied to the chips used within the ICL Content Addressable Filestore (CAFS). CAFS, for many years, has been an important item in the VME product line. Its functionality allows filestore to be searched for records matching user defined criteria. The search is done on the data stream as it comes off the disc and does not involve the main processor. This allows search operations to be done far faster than if done by the main processor.

1. History

The Content Addressable Filestore [Maller, 1979], [CAFS, 1985] has, for many years, been an important item in the VME product line. Its functionality allows filestore to be searched for records matching user defined criteria. The search is done on the data stream as it comes off the disc and does not involve the main processor. This allows search operations to be done far faster than if done by the main processor (search time decreases typically by a factor between 2 and 10), the workload on the processor is reduced, typically by greater than 90%, and the amount of data transferred is significantly reduced. CAFS is an important product differentiator for the Series 39/VME product and is extensively used by many of High Performance Systems' key customers.

The CAFS product has, in the past, been implemented in three different technologies. It was initially developed as CAFS 800 in the 1970's at Stevenage using modified disc drives. Around 1980/81 an MSI (medium scale integration) version was designed in Kidsgrove, and, in the late 1980s, a VLSI implementation was developed in West Gorton.

2. The Obsolescence Threat

The Very Large Scale Integration (VLSI) implementation of CAFS used a CMOS chip technology which allowed approximately 50,000 gates to be included on a single chip; these gates had a typical delay of approximately 800 picoseconds. During the summer of 1994 the chip manufacturer warned of approaching "last-time-buy" dates for chips built in this technology; as the technology became more dated it was no longer economic to manufacture these devices and they wished to switch production facilities to newer product lines.

High Performance Systems (HPS) was faced with a decision; do a "last-time-buy", redesign the chips into a new technology, or replace the hardware CAFS product with an alternative software based product. Each of these options appeared to be unacceptable.

If a "last-time-buy" of components were to be made the investment required would run into millions of pounds, even looking only three years into the future. Additionally, there were problems of how to estimate sales volumes over a period of several years and of trying to determine the future requirements for the product.

A redesign of the chips seemed unacceptable. The design of the previous chips had required a development effort of around twenty man years and this excluded such items as code development and validation. The staff with the relevant skills and knowledge were working on other key developments, and funding would be needed to cover the development effort and the "up front" charges made by the chip manufacturer for any new chip design.

To replace the hardware CAFS product with a software based product would give an unacceptable degradation and variability of performance across applications. This would be unattractive to the many end-users of CAFS. Overall CAFS is fitted, at customer request, to around 60% of disk controllers.

3. A New Approach

A small group started looking at possible ways of upgrading the chips from the existing 1.2 micron technology to a newer 0.8 micron technology. (The figures of 1.2 and 0.8 micron refer to the basic size of a single transistor in the chip, the smaller the size the larger the number of transistors which can be included on the chip, and the faster they will operate). Previously, a chip design had been converted from one technology to another, as part of the Esprit "Integrated Design and Production System" project, to prove the concept of vendor independent chip design. This had established some of the basic methods for converting a chip from one cell library to another. However, the upgrading of the CAFS chips was more demanding because areas of *full* *custom* design were to be included. In *full custom* design the chip is represented by a map of the physical position of the gates on the silicon. This makes conversion more difficult because more recent design techniques known as *sea-of-gates* use a logic diagram (known as a network) rather than a physical representation. Also, the completion of a real chip design would involve checking the performance (clock rate) and the functional operation of the chip.

This investigation identified a possible way of converting the chip design to a later technology and christened the process *transmogrification*. The OED defines this word as, "a change of form, especially in a magical or surprising manner."

4. Transmogrification

The original designs used a semi-custom design style, and the chip designer had a variety of basic building blocks available. These were:

- Paracells
- ROMs
- Programmable Logic Arrays (PLAs)
- Primitive gates.

A Paracell is a complex function which can be generated automatically, for use by the chip designer, in a range of bit sizes and functional variants. One example of a Paracell is a multiplexor which can be generated to select one of two, one of three, or one of four signals. Additionally the signal can be between 4 and 32 bits in data width. The chip designer can generate automatically as many different types of multiplexor as he requires as part of the overall chip design.

Paracells are generated using *composition* software, and each one generated has the following data associated with it:

- a *symbol* consisting of a *drawing shape* associated with timing/electrical data. The symbol is the shape of the paracell as it will be seen on the logic diagrams. The timing and electrical data are used by the computer aided design (CAD) software to check the performance of the chips

- a simulation model called a *behaviour*. This is a piece of software which represents the function performed

- the *geometrical data* for the physical layout of the chip. This is a *map* of the paracell showing the physical positions of the transistors, input/output connections etc.

Note that paracells, although they are built internally from gates, do not have a logic diagram captured within the CAD software to show this internal structure.

ROMs and PLAs are different structures for implementing any desired combinatorial function directly from some form of behavioural description without designing at the gate level. Like paracells they have symbols, behaviours and geometrical data, but no network.

Primitive gates such as NAND gates, flip-flops etc. are the simplest design elements and are provided in the vendor's cell library. They have a symbol, behaviour, and geometrical data.

The chip designer creates his design using a hierarchy of logic diagrams (or networks). The top level network contains symbols for a number of lower level blocks. These blocks will, in turn, each be broken down into sub-blocks with their own networks. At the bottom level in the hierarchy the networks will contain only symbols of paracells, ROMs, PLAs and primitive gates. This structure is illustrated in Figure 1.

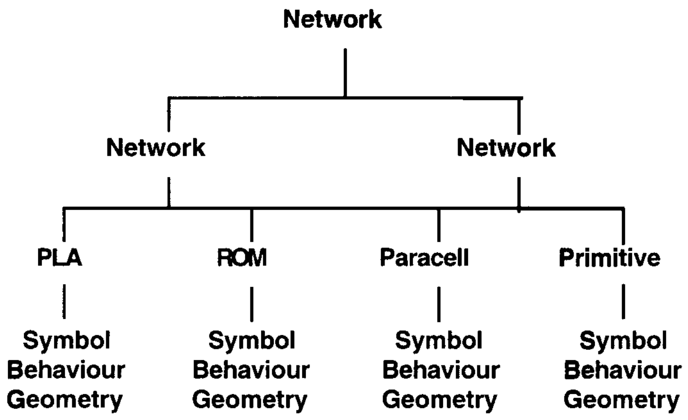

Network
Network
Network
PLA
ROM
Paracell
Primitive
Symbol Behaviour Geometry
Symbol Behaviour Geometry
Symbol Behaviour Geometry
Symbol Behaviour Geometry

Figure 1 Hierarchy of Logic Design Networks

The key problem was how to convert this semi-custom design style, with its emphasis on physical design data and few networks, to a *sea-of-gates* design style which has an entirely network based representation and no physical data (except in the vendor's layout tools).

For primitive gates the approach was to reuse the existing drawing shape and behaviour, but to add a level of indirection for the timing and electrical data so that this would be mapped to a primitive gate from the new 0.8 micron library data.

The technique for PLAs and ROMs was to use the existing symbols and behaviours, but to replace the physical design data with a network synthesised using the Synopsys software package. This software automatically generates a network to perform a specified function, the function being derived from the behaviour of the original ROM or PLA.

For Paracells composition software was brought back into use. The software for generating the drawing shape and the behaviour was simply reused. The physical data generation was replaced by network generation. By studying the behavioural models and revisiting the original design documentation it was possible to derive a suitable composition algorithm for the networks. Given that the design documentation was over five years old and the staff responsible were mostly no longer available, this was an exercise which seemed to owe more to archaeology than engineering!

5. Simulation

In addition to the problem of converting the design, it was also necessary to *simulate* the new design to ensure that the conversion had been done correctly. Since some manual intervention was required in the conversion process it could not be guaranteed to be correct. The CAFS chip set had originally been simulated using a complex "test harness" which applied validation tests to a simulation model of the complete multi-chip system. For several reasons it did not seem practical to reuse this method. The staff, who had initially developed and used the test harness, were committed to other projects; the system was slow and cumbersome to use, and the chip designers, who would understand how to interpret any failing tests, were also committed elsewhere.

However, the chips had originally been implemented incorporating Built-In Self-Test (BIST). This feature was intended for testing the chips in manufacturing and also for testing the chips and PCBs in the system. BIST allowed very high quality test coverage to be achieved with minimal external support. We realised that we could use this test data to check that the converted chip functionality exactly matched that of the original chip. Thus applying these tests and resolving differences by simply comparing the simulation models of the old and new chips allowed the chip to be verified without the need for the test harness or any understanding of the functionality of the chips.

The fact that the converted chips would be *test data compatible* with the old versions, meant that the system test and diagnostic support data would be unchanged, so reducing the firmware development required and simplifying the system validation.

6. Funding

The problem of how to fund the development still remained. A costing exercise was done for the new technology. The existing CAFS PCB required 11 chips in the 1.2 micron technology. Upgrading to the new chip technology, with a lower cost for each chip, would give significant cost reduction. So the development funding could be justified on the basis of cost saving alone.

7. An Initial Proposal

A trial of the conversion route was carried out on part of a chip design to establish the timescales for the project and a rough outline of a proposal was prepared and presented to a management review in July, 1994.

The consensus was that the project was very demanding in terms of technology, capability and timescale. The major areas of concern were:

a) handling the changes in timing between two different chip implementations would be difficult. There were two possible problems. Firstly, because most of the logic would run faster, so called "race hazards" would result (this is when some part of the logic operates so fast that it gets out of step with the rest of the chip and the correct sequence of operations no longer occurs). Secondly, the ROMs and PLAs were being replaced by gates which would run more slowly and these would affect the overall performance of the chips

b) any introduction of functional changes or bug fixes would lead to a much longer timescale

c) it might prove difficult to maintain the timescales from the pilot study on a full chip design. A slip in the projected dates would mean that the development could not be funded from cost reduction.

No decision was made as to the future of the project so, for the next two to three months, the work continued on an informal basis investigating further the feasibility of the full project.

8. Extended Pilot Study

The effort was now concentrated on the conversion and validation of two full chip designs (part of the Macrolan chip set rather than CAFS because, at the time, they were felt to be more urgent). These designs answered the criticisms raised at the previous review.

a) The timing of the new chip did not raise any major problems. *Race hazards* were not a major issue because of the improved design techniques available in the newer technology. The conversion of PLAs and ROMs to functionally equivalent networks gave a performance advantage, rather than a degradation as had been believed.

b) It was possible to introduce modifications to the logic design, so long as this was put into the design as an *additional* block of logic rather than a *replacement*. The old version had to be included to allow the existing BIST tests to run, and the modification was added in parallel so that it did not affect the operation of the BIST tests, but was available for functional use.

c) The conversion of the two chips also confirmed the synthetics for the task.

9. Other Benefits from the Technology Conversion

During this investigation three other major advantages of the conversion process came to light, which had not been recognised until then. Firstly a major performance increase, secondly a physical reimplementation, and finally a reduced system complexity.

9.1 Performance: Beating the Technology Trends

The original CAFS chips were designed to operate with a clock period of 95 ns (this is the time interval in which each step of the logic operation must take place). When upgrading from a 1.2 micron technology to a 0.8 micron technology the typical nominal gate delay decreases from 790 ps to 550 ps, a factor of 0.7. Therefore the converted chip might be expected to operate in a 66 ns clock cycle. However two additional factors also have an effect. Firstly, the converted chips were to be designed into *surface mount* packages rather than *pin grid* packages (surface mount packages, unlike pin grid packages, do not have leads which pass through the board, so connections can be made to both sides. This allows more chips to be fixed on the board, thereby saving space). The surface mount packages have a far higher thermal resistance so that, consequently, the chips operate at a higher temperature which would degrade the performance by a factor of 1.1, giving a clock beat of 73 ns. Also, the chips were to be converted from a semi-custom design style to a *sea-of-gates* technology, which would normally be expected to degrade performance by a factor of 10% to 20%. It was therefore initially projected that the clock beat would be between 80 ns and 87 ns for the converted chips giving only a marginal performance boost.

In practice, during the conversions, we found that we could reduce the clock beat for the chips to 50 ns, which was far better than the simple technology trend graphs would suggest, and achieve a near doubling in logic speed (See Figure 2).

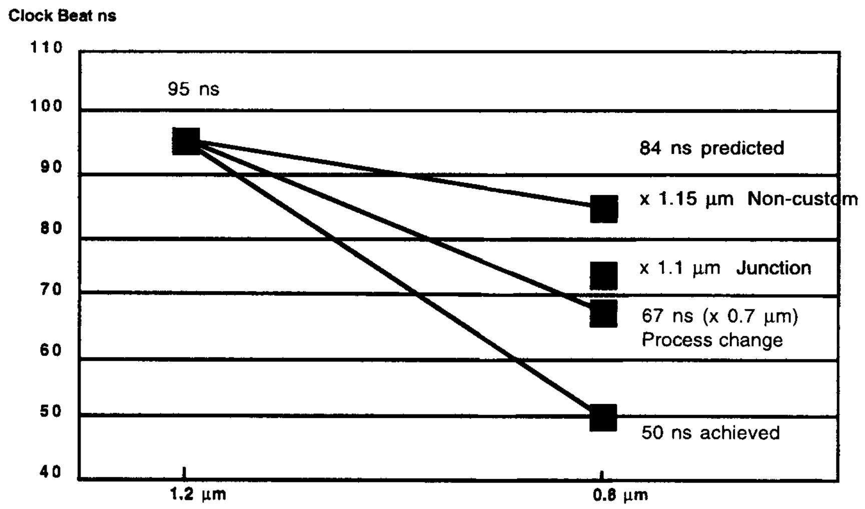


Figure 2 Chip Performance

This improvement in performance arose for the following reasons:

a) advances in the CAD timing tools made it easier to identify the slowest parts of the design which limit the overall performance of the chips. These parts of the design could be optimised to improve the overall system performance

b) the 0.8 micron cell library contained a wider variety of gate functions than had been available in the 1.2 micron cell library, so timing critical functions could be reimplemented with fewer gate delays

c) the performance of one of the chips, and so of the full system, was limited by a path through a RAM. This path involved only one bit of the 9 bit RAM. The performance limitation was overcome by having a register file (a high speed, but higher gate count equivalent of a RAM) in parallel with the RAM to replace the relevant bit. This reduced the memory access time from 13 ns to 5 ns and increased the chip performance by 15%

d) the group involved in the task included technology specialists with collectively a wide range of experience in exploiting the CAD tools, cell libraries and VLSI design techniques. Additionally, they were able to concentrate purely on the technology aspects, without having to be concerned with understanding the functionality.

9.2 Physical Reimplementation

A further advantage of conversion that became apparent was that we could update the packaging technology as well as the performance of the chip set. When the chips were originally designed *pin grid array* packages were used, as surface mount technology was not then sufficiently mature or reliable for such large pin count devices. The chip conversion would enable the number of VLSI devices to be halved, the area of the board occupied by each package would also be halved, and the use of surface mount technology would allow double sided PCBs to be used.

Consequently, future developments using the chip set could have lower PCB costs and would fit into smaller cabinets.

9.3 Reduced System Complexity

The MSI CAFS implementation had sixteen *key channels*. The *key channel* is the basic comparison unit which checks for data matching particular criteria when performing a search. By having multiple key channels it is possible to perform complex searches by combining the results from two or more key channels. When the VLSI CAFS was developed, the number of key channels in the hardware was increased to allow more complex search functions to be executed. Consequently the VLSI chip set had thirty two *key channels* in eight chips. However, due to a change in development priorities, the necessary software changes to support these hardware enhancements were not implemented. This left the new design with sixteen redundant *key channels*, which could not be removed without modifying the chip interfaces.

During the extended pilot study a possible modification to the chip design was identified, which would allow the redundant sixteen *key channels* to be removed, leaving four chips containing four *key channels* each. This modification, which would further reduce the physical size of the CAFS system, was of the additive type mentioned earlier and so would have no impact on the validation strategy.

Removing the redundant logic would also give larger cost savings. The chip count for the system would be reduced from ten to five as well as the cost per chip being lower.

10. A Second Proposal

At the end of October 1994, the proposal was again put to a management review. This proposed a plan for chip availability during April, 1995, with validation complete by the end of the third quarter of 1995, ready for the new board to be manufactured during the fourth quarter of 1995. The new proposal went beyond a simple obsolescence and cost reduction proposal, since it mapped out a future strategy for the CAFS product. The performance upgrade and physical repackaging were both seen to be vital for future products because of higher speed discs and smaller cabinets. The proposal also included a bug fix and the "mopping up" of logic contained in a few smaller chips on the CAFS PCB into the new chips. These were aspects which had not been part of the original proposal.

This second proposal was accepted and the funding was reserved for the chip development charges.

11. Implementation Results

Two chip designs were sent to the vendor in January and February, 1995. At the same time as the chip designs were being completed, the existing CAFS PCB was being retracked to accept the new chips.

While the chips and PCBs were being fabricated, firmware development was done to support the reduction from 32 to 16 key channels. Since the new chips were to be test data compatible with the old versions (with the exception of the additive logic), it was possible to model the new PCB using a hand-wired version of the old board. This allowed most of the firmware to be debugged and validated before the new PCB was available. Consequently the initial test phases of the PCB were faster than for a conventional product development.

The first assembled PCB was received on May 1st and within three days all the basic tests and 80% of the On Line Test Software (OLTS) tests had been run successfully. An error was identified in the chip pin mapping which resulted in certain pairs of signals crossed over. This was corrected by a PCB modification. After a handwired board was received the remaining OLTS tests passed and the board was handed over to Product Validation and Release (PV&R). During this period only one bug in the firmware was discovered and since then no further hardware or firmware faults have been found. The two week period between receiving the first PCB and handing over the validated PCB to PV&R compares favourably with the period of at least ten weeks normally expected for a product of this complexity.

Overall the conversion of the three paracell chips was achieved in forty four man weeks of work, giving a productivity figure of 9,400 gates per man-month. This compares very favourably with the figure of 520 gates per man-month for new chip designs and 3,200 gates per man-month for a previous manual technology upgrade (Figure 3).

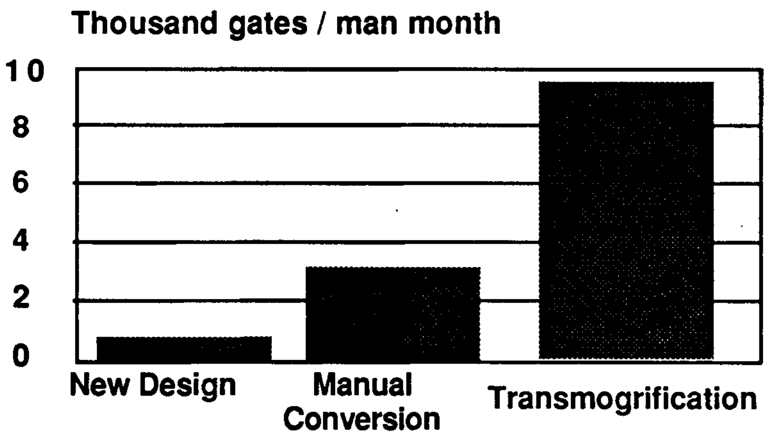


Figure 3 Design Productivity

12. CAFS: Future Options

The currently available chip set has solved the problem of the approaching obsolescence of the 1.2 micron technology and will give considerable cost savings. The new chip set also gives future development options for the CAFS product.

13. Other Developments

The CAFS chips were not the only chips designed using the 1.2 micron process. Two of the chips used for Macrolan were also affected by this problem of obsolescence. These have also been converted to a newer technology, although in this case other chips were also included to give a Unified Macrolan Chip. The total project included:

- MAC (Medium Access Controller)
- BCL (Buffer Control Logic)
- Serdes (Serializer-Deserializer): a chip conversion from ECL technology
- Data Recovery: this new function is to replace the current clock recovery modules
- MPSU (Macrolan Port Switch Unit): a replacement of an existing ECL chip set.

This chip development was not done entirely by the conversion route described above, because the conversion of ECL to CMOS requires extensive manual redesign to meet the required clock rate, and some new functionality was required for the Data Recovery and MPSU. However the MAC and BCL functions, which account for 85% of the logic, were converted as described above.

The resulting "Unified Macrolan Chip" will provide all the functionality required for the wide range of existing Macrolan products.

14. Conclusions

Back in the summer of 1994 we were faced with a major problem. The chip set required for one of our key products, used by some of our largest customers would shortly become unavailable. A "last-time-buy" was financially unacceptable and a redesign of the chips would be too time consuming and costly. By making use of specialist skills we have solved this problem, made available future development options for a key product, protected twenty years' worth of development effort and provided a route forward for other products.

Acknowledgements

This paper is based upon a talk given at the ICL Engineering Conference, which won the 1995 Outstanding Performance in Technology Award. The winning team were:

Terry Bird, Balvant Mistry, Danny Traynor, George Woad and the author.

References

MALLER, V.A.J., The Content Addressable File Store - CAFS, *ICL Tech, J.*, Vol 1(3), 1979.

CAFS, A set of 11 papers, *ICL Tech. J.*, Vol. 4(4), Nov. 1985.

Biography

Richard Illman works for the "Technology and Engineering Directorate" within ICL High Performance Systems. He joined ICL in 1982 after graduating from the University of Hull, and has worked on CAD software, "Built-In Self-Test", chip technologies and quality management.

He is a Senior Member of the IEEE, a Member of the IEE, and he currently serves on the Editorial Advisory Panel of the IEE Computing and Control Journal. He has published a number of papers and (with his co-author) won the "Best Paper" award from the International Test Conference in 1989.

An Innovative Solution for the Interconnection of Future Component Packaging

Michael Hendriksen, Steve Hodgkinson
Neil Taylor, Rob Trussell

D2D Ltd, Kidsgrove, Staffordshire

Abstract

The dramatic growth in complexity and content of integrated circuits (ICs) continues to make increasing demands on the performance, pin count and compactness of microelectronics. New developments in electronics packaging have lagged behind those in IC technology, resulting in a bottleneck preventing the full potential of IC technology being realised in final system performance. Solving this Interconnection Bottleneck in a cost effective way is one of the major challenges for the electronic industry.

A new technology (MCM-L/BGA) has been developed at D2D Ltd., which provides reliable fabrication of high performance and low cost Multi-Chip Modules (MCMs), to satisfy the emerging requirements for electronics packaging. The MCM-L technology is based on an advanced laminate interconnect structure capable of supporting naked silicon attachment. The Ball Grid Array (BGA) assembly technology completes the MCM-L/BGA package by providing the connection from the substrate and silicon chip level (MCM-L) to the next level of electronics package, PCB or motherboard.

This paper considers the reasons for solving the Interconnection Bottleneck and then describes the pertinent stages involved during the development of the MCM-L/BGA technology. The challenges, faced by the development team in each of these stages, are discussed and the significant strides made by the team in developing and enhancing the individual processes that define the technology are described.

1. Introduction

1.1 Electronics Packaging Evolution

Until recently, interconnection of discrete electronic devices on printed circuit boards (PCB) was achieved using technology that had not changed radically in over twenty years. Although circuit densities have certainly increased and track widths and spacings have decreased on printed circuit boards, the increased levels of integration and smaller geometries at the chip level have been supported by evolution of conventional circuit technology. This evolution has been satisfactory in so far as it has allowed the move from dual-in-line and through-hole leaded components to the early surface mount technology.

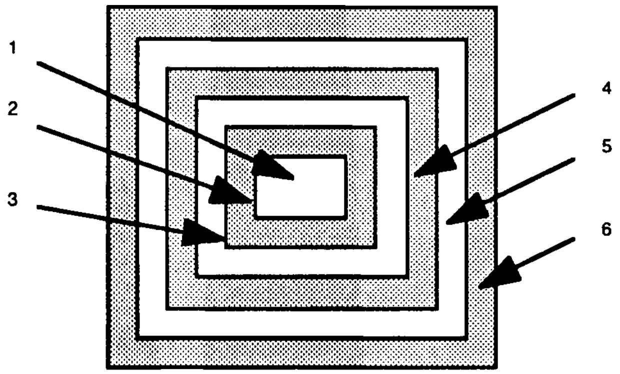
However, the demand for higher levels of integration has been relentless and this has continued to force the need for increased track routing (interconnection) densities at the PCB level. An empirically derived rule, Rent's rule, may be used to indicate the number of signals connected to a component as a function of the complexity of the component (See Appendix 1). The inexorable increase in signal count predicted by this rule has occured, and pin count is currently limited only by the interconnect (substrate with the interconnections) and component packaging technology available, thus creating the interconnection bottleneck. Until recently, surface mount component package design favoured the peripheral I/O (Input/Output leads emanating from the edges of the component) approach. However, array packaging (leads emanating from the area beneath the component) in the form of Ball Grid Array (BGA), and direct attachment of naked silicon, are now becoming increasingly popular as they offer higher output and improved on-board real estate efficiency.

This move from peripheral to array style surface mount component packaging presents the bare board fabricator with several challenges in providing sufficient interconnect density at the printed circuit board level. As the pitch and pin count on these packages decrease and increase respectively, the routing difficulties using conventional manufacturing methods become greatly exaggerated [Goosey et al, 1996].

1.2 Challenges of Electronics Interconnection and Packaging

The major driver, forcing a review of interconnect and component packaging technology, is the clock signal delay. The increase in clock frequency reduces the volume accessible within a clock period. As clock frequencies increase still further it becomes important to consider transmission line effects, and the PCB track related signal delay. Very short tracks have a resistance which is small compared to their characteristic impedance and, therefore, behave as transmission lines with a signal delay proportional to the track length. Very long tracks, on the other hand, have a resistance which is large compared to their characteristic impedance and, as a result, behave as a lumped RC element with a signal delay proportional to the square of the track length (Both track resistance, R, and capacitance, C, are proportional to track length).

Thus, the most leverage from a packaging standpoint can be obtained by moving the interconnection to the silicon chip level, thereby reducing signal track length to a minimum and subsequently enhancing signal speed from silicon chip to PCB. This effect on signal speed is reflected in Figure 1 illustrating packaging penalties incurred for six different component package and attachment techniques.



1. Bare chip & BGA	4. Flip TAB (89% larger)
2. Flip Chip & CSP (8% larger)	5. Conventional TAB (164% larger)
3. Wire Bond COB (39% larger)	6. 196 Lead QFP (>188% larger)

Figure 1 Device Area Requirements

The different component packaging formats are all based on the same silicon chip size with the resulting penalties for signal speed and available real estate. Signal speed for the 196 lead Quad Flat Pack (QFP) is dictated by the lead frame design of the component which results in a much longer line length when compared to that of the naked silicon chip attachment to the interconnect. Likewise for real estate usage, the QFP component requires an area, on the interconnect, of 188% larger than that of the naked silicon chip attachment technique.

Similarly, other component packaging and attachment techniques such as Tape Automated Bonding (TAB), Chip on Board (COB), small-outline Ball Grid Array (smBGA), and Chip Scale Packaging (CSP) require additional real estate area on the interconnect for attachment. A reduction in these packaging penalties would contribute to enhanced signal speeds and increased component packaging densities on the interconnect. Ideally, the ultimate component packaging and attachment technique is Direct Chip Attachment (DCA) of the naked silicon chip onto the interconnect.

1.3 Interconnect Density Requirements

The interconnect density requirement has been defined as the ability to provide interconnection to typical BGA and DCA packages. For example, in the case of a 32 x 32 array package the total number of I/O's is 1024. If one assumes that 25% of these are power/ground connections and that there are 126 I/O's on the external ring of the package, there will be 642 I/O's needing to be routed out of the component area through 124 possible escape channels. Therefore, the number of signal layers required will be as follows:

6 signal layers for 1 track between channels
3 signal layers for 2 tracks between channels
2 signal layers for 3 tracks between channels

Similarly for a 23 x 23 array (assuming 306 I/O's needing to be routed through 88 possible escape channels) the number of signal layers required will be:

4 signal layers for 1 track between channels
2 signal layers for 2 tracks between channels
2 signal layers for 3 tracks between channels

If one now considers the printed circuit board geometries for both typical and advanced technology to be as follows:

	Typical	Advanced
Track/space (mm)	0.15	0.1
Via diameter (mm)	0.35 (through-hole)	0.15 (blind via)
Pad diameter (mm)	0.6 (through-hole)	0.4 (blind via)

It is clear that the number of tracks per channel varies with ball pitch as follows:

Ball pitch (mm)	1.5	1.27	1.0	0.5	0.3
Comp. package type	BGA	BGA	BGA	smBGA	DCA
Tracks (typical)	2	1	0	0	0
Tracks (advanced)	5	3	2	0	0

The above analysis clearly indicates that a quantum leap in interconnect technology is necessary to support emerging component packages such as small-outline BGA (smBGA) and DCA. The smBGA component package uses the same grid array format as conventional BGA, but is much smaller in size. The smBGA package is the same size as the silicon chip that it is packaging and so is termed a Chip Scale Packaging (CSP) technology. While reducing track widths and spaces even further will provide some density enhancements, the limiting factor for most printed circuit board fabricators is the ability to produce smaller vias and associated pads.

In order to provide sufficient interconnect density, D2D have developed a Multi-Chip Module (MCM) technology based on an advanced laminate (L) interconnect fabrication, capable of accommodating naked silicon chip attachment. The laminate interconnect provides the interconnection density at the silicon chip level (MCM-L), whilst a Ball Grid Array (BGA) technology is used to provide connections from the MCM-L to the outside world, either PCB or motherboard.

The achievable geometries using the above technology are:

<u>MCM-L Interconnect</u>

Track/space (mm)	0.050
Via diameter (mm)	0.0625 (blind via)
Pad diameter (mm)	0.100

and this allows an increase in the maximum number of tracks as follows:

Ball pitch (mm)	1.5	1.27	1.0	0.5	0.3
Comp. package type	BGA	BGA	BGA	smBGA	DCA
Tracks per channel	12	10	8	3	1

Thus, the newly developed MCM-L interconnect technology is fully capable of supporting current component packaging formats and the emerging smBGA and DCA technologies.

2. Packaging Technologies

2.1 Multi-Chip Modules

MCMs have emerged as the new electronics packaging technology capable of achieving high silicon chip densities (many silicon chips together on interconnect), shorter interconnection paths (track lengths), and improved IC system performance. An MCM is a structure consisting of two or more naked silicon chips interconnected on a supporting interconnect (substrate) and packaged as an interconnected group. The substrate conductors are formed in multiple layer structures separated by dielectric material and connected by metallised *vias* (holes through multi-layer which are electroplated inside so as to interconnect tracks at different levels) [Doane et al, 1993].

Early hybrid microcircuits resembled MCM's but recent advances in materials and processing technology have resulted in alternative MCM technologies to compete in the current price-sensitive markets and offer a cost-effective solution. The alternative MCM substrate technologies include thin film (MCM-D), ceramic (MCM-C) and organic (MCM-L).

MCM-D uses IC technology to produce high-density thin-film interconnections. The 'D' refers to the deposited dielectric fabricated using IC techniques such as photolithography, sputtering and wet and dry etching. This technology offers the highest interconnect density of the MCM categories but at a premium.

MCM-C technology is an extension of thick-film hybrid and co-fired ceramic single chip package technologies. Structures use either high-temperature or low-temperature co-fired ceramic designs depending on application. These structures are produced using green (unfired) sheets of dielectric tape with conductor patterns formed using screen-printing techniques. All sheets are finally laminated and fired at high temperatures to form the MCM-C. Manufacturing costs are cheaper than MCM-D but interconnect density is poor due to the screen-printing techniques used for circuit fabrication.

Organic laminate substrates with high interconnect density capabilities are emerging using enhanced materials, while at the same time allowing manufacturing processes common to the PCB industry to be retained. For this reason, manufacturing costs are kept to a minimum making MCM-L technology an affordable option for emerging component packaging technologies such as DCA.

2.2 Ball Grid Array (BGA) Technology

Ball Grid Array silicon component packaging is a rapidly emerging alternative technology to Quad Flat Pack (QFP), Pin Grid Array (PGA), and Tape Carrier Package (TCP) component formats. It is a surface mount technology and can be assembled using conventional surface mount pick and place equipment.

The BGA design offers high I/O densities without the need for very fine lead pitches. The I/O is an array of solder contacts on a specific grid pitch located on the underside of the packaged silicon chip. In this way, all the I/O connections are not routed to the periphery of the package, but instead are located across the package surface area and, therefore, have more space yielding a coarser pitch compared with peripheral device formats. The relatively coarse pitch allows higher yields to be realised compared with QFPs used for a similar function. Electrical performance is also enhanced, as the shorter silicon chip to PCB line routes results in reduced inductive effects.

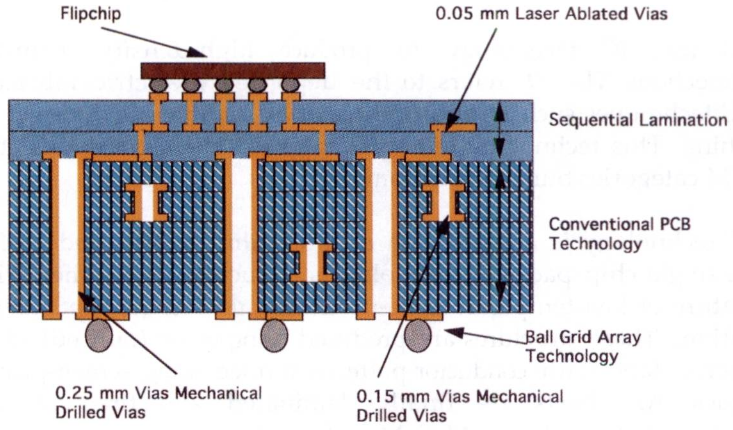

Flipchip
0.05 mm Laser Ablated Vias
Sequential Lamination
Conventional PCB Technology
Ball Grid Array Technology
0.25 mm Vias Mechanical Drilled Vias
0.15 mm Vias Mechanical Drilled Vias

Figure 2 Sequential Fabrication and Laser Ablated Vias

3. MCM-L/BGA Technology Development

Figure 2 illustrates the MCM-L/BGA technology developed by Manufacturing Technology group, D2D Ltd. The electronics package provides the I/O escape track routing (fan-out) from the silicon chip level to external PCB, by combining interconnect and assembly technologies into the one package. The laminate interconnect (substrate) is the heart of the package, providing the ability to support naked silicon chip attachment. The high density interconnect capability is achieved by using laser ablated vias and fine line circuitry technology. The interconnect is based on a sequential fabrication structure which can be built onto conventional PCB technology. The interconnect provides the interconnect density at the silicon chip level resulting in the attachment of silicon chip using peripheral wire bond techniques (flip chip technique is shown in diagram). Ball Grid Array (BGA) technology completes the package by providing the connection from the interconnect and chip level to the next level of the electronics package, PCB or motherboard.

3.1 Interconnect Technology

A dual approach was taken to increase track and via density per unit area, using smaller track/gap geometries and laser ablated vias in a sequential fabrication process.

3.1.1 Fine Line Technology

Increased track (circuit) density is achieved by the fabrication of track and gap geometries between 0.002" and 0.004".

Using standard chemical subtractive processes, similar to conventional copper circuit fabrication, this has been achieved, but also incorporating improved equipment design, and advanced material and chemical solutions. The new resist material type and resist lamination technique are both important for fabrication of fine line technology because of the physical and chemical processes during the imaging, developing and etching stages.

Copper circuits are prepared by the use of a *resist* material which protects selected areas of copper from being chemically etched away. These *resists* are complicated photosensitive materials which polymerise when exposed to ultraviolet light and hence become insoluble in an aqueous developing solution. The unpolymerised material is washed away during the developing stage to leave a copper pattern which is subsequently etched during the chemical acid etch stage. The polymerised *resist* material is resistant to attack by the chemical acid etchant and thus protects the underlying copper thereby forming the desired circuit pattern, after which the *resist* is finally stripped from the copper.

The desired circuit pattern starts as a mask (artwork) between the UltraViolet source and resist material. The pattern is defined with the circuit image in clear (which will finally be the copper features) against a black background. A copy of the artwork pattern is imaged onto the resist which is subsequently developed and etched to produce the desired copper pattern.

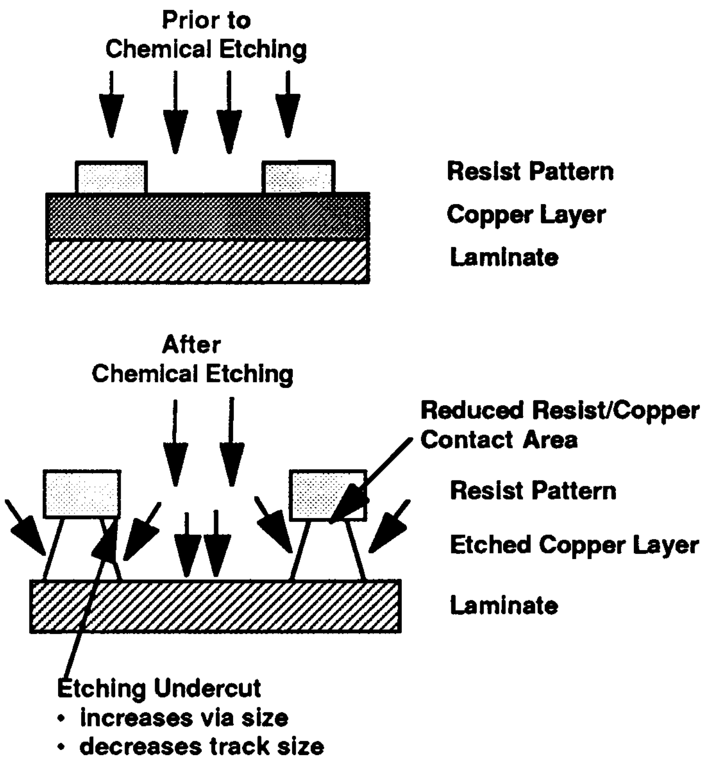


Figure 3 Chemical Etching Profiles

Figure 3 illustrates the nature of the chemical attack on the unprotected copper; note the *undercutting* of the resist. This phenomenon occurs regardless of resist or etchant, but its consequences are more critical the finer the geometries involved. As tracks become smaller, the contact area between resist and copper is reduced. Thus it is imperative to prevent resist coming away from the copper interface through lack of adhesion. Use of the new lamination technique overcomes this difficulty, by enhancing the initial adhesion between resist and copper surface.

Control of track widths is important for many reasons, including control of track impedance. The size of the developed image on the resist is critical, since *compensation factors* are included in the design of the artwork pattern to compensate for etching undercut (Figure 3). As gaps between tracks are reduced, scope for increasing the compensation factor is diminished as the chemicals must penetrate the gaps while developing and etching. Using the new resist, in conjunction with optimal chemical and imaging processes, tracks are generated with an improved profile, allowing smaller compensation factors to be used.

3.1.2 Dielectric Material Technology

Electronics continues to push the limits of material performance in the various materials used to manufacture printed circuit boards. For many years the traditional FR-4 (woven glass reinforced dielectric) has been widely used, but is starting to show limitations when applied to the leading edge packaging and interconnection technologies such as BGA and DCA. As these FR-4 materials tax the boundaries of performance and processability, the use of organic dielectric materials has shown an improvement in dielectric constant (permittivity), resulting in faster signal propagation, and reduced mismatch in the coefficients of thermal expansion (CTE), thereby improving the reliability of solder-joint s.

Major fabricators of advanced materials have recognised the needs of the electronics packaging industry and have developed a range of dielectric materials for future applications. The potential material identified was Thermount™, a non-woven dielectric reinforced with an *aramid* fibre. Table 1 illustrates the merits of Thermount compared with those of FR-4 material.

The most important characteristics of materials for MCM-L fabrication are the dielectric constant (Dk) and the coefficient of thermal expansion (CTE). To support high speed applications and high density packaging, lower Dk materials are imperative. as the transmission speed of signals along the circuit track is inversely proportional to Dk. As Dk is reduced to achieve a specific impedance, the interconnect can safely be made thinner. Consequently, crosstalk between adjacent signal lines is reduced permitting fabrication of closer lines, which, in turn, yields higher circuit density. Faster signal propagation, combined with higher circuit density, allows higher system operating speeds. Also, when considering MCM-L and DCA applications, the goal is to minimise the CTE mismatch between the substrate and the attached silicon chip. This becomes very important as chip clock speeds become faster (operate at higher temperatures) and chips are placed closer together on the supporting substrate.

Property	FR-4 Epoxy	Thermount Epoxy
In-Plane CTE (ppm/°C)	15 / 17	6 / 9
Dielectric Constant @ 1 MHz	4.7	3.9
Surface Smoothness (Å) (2-ply laminate)	4200	2200
Dimensional Stability (%)	±0.5	±0.3
Density (g/cc)	1.7	1.3
Laser Ablatable	No	Yes

Table 1 Thermount Characteristics

The *aramid* fibres in Thermount are randomly dispersed and orientated, which results in an extremely smooth surface as compared with woven glass reinforcement. A smoother surface allows the production of finer line copper circuits with higher imaging yields. Also, the fibres absorb ultraviolet radiation strongly, allowing *photoablative decomposition* or *laser ablation*, thus resulting in more highly defined structures. Trials of material laser ablation using an Excimer laser proved the suitability of Thermount for fabricating laser ablated vias of high quality and long term operational reliability.

3.1.3 Laser Ablated Blind Via Fabrication

Blind Vias

Conventional printed circuit board designs make use of mechanically drilled holes and vias through the circuit board to form electrical connections between internal layers or from internal layers to the external surface layer. However, these drilled vias have their problems when designing for high density circuit solutions to support emerging packaging technologies.

Currently, typical via pad size is 0.024" drilled with a 0.010" to 0.014" hole. To reduce the 'real estate' used by the via pad size and subsequently increase the circuit density for advanced PCB technology, smaller holes can be bored with drill sizes as small as 0.006". But as drills get smaller, the drilling process becomes more expensive. Small hole drilling can contribute as much as 30%-40% of the cost of a PCB. Also, if the PCB thickness does not decrease proportionally to the hole size, a higher hole aspect ratio results which can lead to reduced hole reliability after copper plating. The aspect ratio of a hole, for copper plating purposes, is the ratio of diameter to depth. Finally, the rapid rise in the cost of tungsten carbide, from which drill bits are made, adds another cost burden to small hole drilling.

As previously discussed, to support emerging packaging technologies, such as smBGA and DCA, an interconnect solution is required to achieve track width and spacing from 0.002" upwards and via and pad sizes from 0.0025" and 0.004" upwards respectively. To overcome the physical size limitations and associated costs with mechanical drilling, D2D Ltd have developed a small via generation technique in the form of non-drilled blind vias using laser ablation.

Figure 2 illustrates the D2D MCM-L package technology. The vias are formed using a reverse pillar build process, whereby an initial dielectric material layer is sequentially laminated on to the core structure, and the via (pillar) is formed from the top layer circuitry down to the circuitry layer below to complete the connection. Additional dielectric layers can be laminated depending on the I/O density of the flip chip package. Again, further vias are formed from the new top layer down to the underlying layer. Figure 2 shows two sequential dielectric layers with laser ablated blind vias fabricated to form an I/O fan-out capability from the flip chip package to MCM-L package.

Using the reverse pillar build process forms a via structure defined as a blind via. A blind via connects one layer to another and then terminates on that layer; it does not connect through the rest of the board and thus saves space within the layers of the board.

Blind Via Fabrication

The manufacturing process developed to fabricate the blind vias covers two stages; first define the position and size of via on the substrate using conventional printed circuit processing techniques; followed second by the laser ablation process, specifically developed for this purpose.

As with Fine Line technology, standard chemical subtractive processes are used, but incorporating enhanced *resist* material types and a unique optical alignment system to achieve the smaller size via and the required positional accuracy of its location within the circuit.

Conventional printed circuit manufacture uses a tooling pin alignment system for the alignment of circuit patterns to each layer and subsequent layer-to-layer registration. With typical via pad size of 0.024", a via size of 0.010" has some tolerance for the drilled via to locate within the circuit pad. However, with laser ablated blind via/pad geometries of 0.0025"/0.004", tolerances are extremely tight and virtually impossible to achieve using the conventional tooling pin system. However, incorporating a different system based on optical alignment such tolerances can be achieved. Referring to the sequential fabrication within Figure 2, the optical system aligns the via circuit pattern of the top layer to a set of fiducial markings on the underlying layer. In this way, each layer is sequentially registered reducing the effects of layer-to-layer mis-registeration and improving the tolerances of via/pad registration.

As discussed previously, dielectric layers are sequentially laminated and vias are formed from top layer to underlying layer. After sequential lamination, vias are formed in the copper layer to the desired diameter (via aperture) using print and chemical etch techniques. The copper layer, containing the via apertures, is used as a conformal mask for the laser ablation process. The dielectric material exposed by the via aperture is removed by Excimer laser ablation. The energy densities (fluences) used (<1Joule/cm^2) are such that selective removal of polymer dielectric occurs whilst ablation automatically ceases on exposure of the underlying copper capture pad, which exhibits much greater reflectivity and a higher threshold fluence for material removal (in the region of 3 to 4 Joules/cm^2 [Hussla et al, 1984]). Figure 4 illustrates blind via laser ablation.

Excimer Laser Ablation

Excimer lasers produce intense, ~10MW pulses of ultraviolet radiation in a range of wavelengths from 193 to 353 nanometres. Their rapid discharges (pulsewidths ~10ns) and high photon energies represent an ideal combination for processing a range of materials including fabrics [Brinkmann, 1990], ceramics [Brinkmann, 1992], metals and, particularly, organics [Dyer, 1992].

The absorption in most organic materials of an optical pulse from an Excimer laser is so strong [Miller et al, 1993] that, provided the photon flux is above a threshold value, there is a rapid disruption of molecular bonds and subsequent ejection of material, referred to as *photoablative decomposition* or *ablation.*

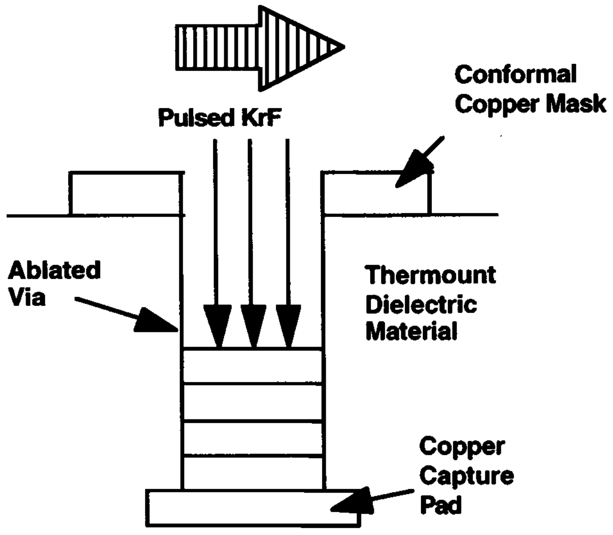


Figure 4 Laser Ablation Process

This bond-breaking process becomes more dominant over conventional thermal effects occuring at shorter wavelengths. The process can be so effective that a significant increase in the interaction volume results within the timespan of the laser pulse. This causes large, transient pressure gradients and an explosive removal [Dyer et al, 1992] of a surface layer of sub-micron-thickness in a direction predominantly perpendicular to the sample surface. The removal of the dielectric material is essentially defined by the beam cross-section, as indicated in Figure 4. Due to the efficient concentration of optical energy on the outer electronic structure of the ablated material, practically all the delivered energy is carried away by the ablation process leaving the unexposed material unmelted and free from the other thermally-induced degradation effects.

Prior to ablation, a tin/lead alloy is plated onto the copper conformal mask. The rapid ejection of material during ablation, which is carried out in an air pressure gradient caused by the shock front, results in a 'mushroom cloud' effect whereby ablated fragments return to the surface surrounding the interaction region [Dyer et al, 1995]. The tin/lead plating acts as a sacrificial layer by capturing the ablated fragments. Having no sacrificial layer would result in redeposited ablation fragments on the copper surface causing subsequent problems for fine line circuitry fabrication. The tin/lead is subsequently removed (along with redeposited material) in a chemical stripping process, leaving a clean, undamaged copper surface.

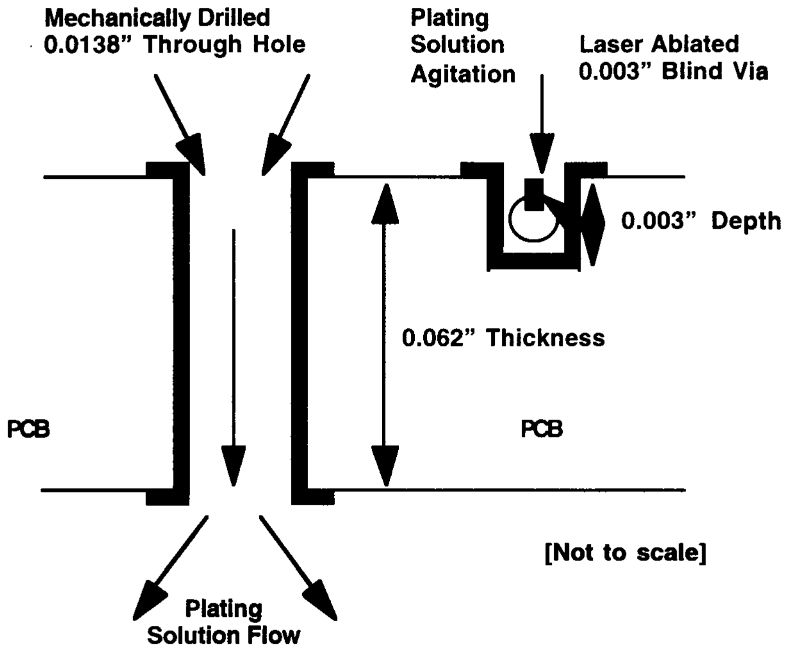

Mechanically Drilled 0.0138" Through Hole
Plating Solution Agitation
Laser Ablated 0.003" Blind Via
0.003" Depth
0.062" Thickness
PCB
PCB
[Not to scale]
Plating Solution Flow

Figure 5 Blind Via Metallisation

The fabrication of the via is completed by a metallisation process using electroplated copper. The ablated via is copper plated to form the connection from one layer to another. However, such via structures cause problems for plating due to their small diameter at the opening of the via, and the fact that the via is blind reduces the flow of plating solution in the via as shown in Figure 5. An electroplating process was developed, to metallise reliable blind vias successfully, incorporating the use of increased agitation of the substrate, while being plated, and an optimum 'throwing power' of the electrolytic cell to improve the consistency of the via plating. Agitation reduces air bubble entrapment in the blind via structure so assisting solution flow for initial electrodeposition. Optimum throwing power (current density of the cell) allows for correct plating thickness of the via structure.

3.2 Attachment of a Silicon Chip (Chip On Board)

The suitability of the MCM-L package for direct attachment of silicon was evaluated using specially designed test chips. These chips provide links between bond pads to test the thermal characteristics of the wire bonding, using heating resistors and sensing elements built into the chips themselves. The silicon chip attachment, wirebonding and encapsulation processes briefly detailed below were carried out by Rood Technologies of Alton, Hants.

The surface finish requirements of the bond pads on the interconnect, for aluminium wirebonding, were achieved using an in-house immersion nickel/gold system.

The silicon chip is first attached to the interconnect using either an epoxy adhesive, a conducting polyimide or a eutectic material. The chip attach material provides mechanical strength, can conduct heat away from the chip, and can relieve stresses caused over the life of the MCM-L package by differential thermal expansion between the silicon chip and the interconnect.

The design chosen for this project involved attaching two silicon test chips to the MCM-L package using a silver filled epoxy adhesive which enabled low temperature curing, avoiding possible interconnect damage.

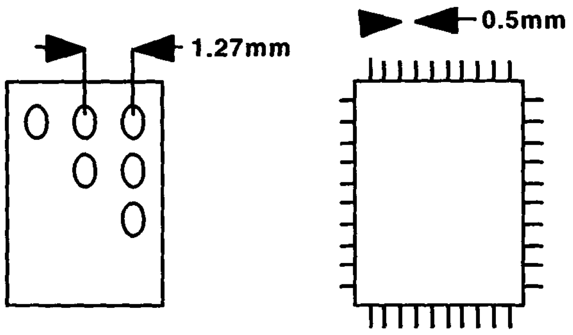
Wire bonding involves connecting bond pads on the silicon chip to pads on the interconnect using fine (25micron or 33micron) gold or aluminium wire. Common methods of wirebonding are thermosonic ball-to-wedge bonding normally using gold wire, or ultrasonic wedge to wedge bonding using aluminium wire.

The wire bonding process chosen used 33 micron aluminium wire attached using an automatic ultrasonic wedge to wedge bonder, which is the most common process used for Chip On Board (COB) applications. Ultrasonic bonding has the advantage of being a low temperature process suitable for organic substrates which have low glass transition temperatures. Wedge to wedge bonding was chosen because it is better suited to chips with a fine pitch and high pin count.

After the silicon chip has been attached, wirebonded and tested the chip surface and fine wire bonds need protection by some form of encapsulation. This is provided by a coat of black "Glob top" encapsulant applied over each die individually.

3.3 Ball Grid Array (BGA) Technology Development

The MCM-L package is attached to a PCB or motherboard using a Ball Gird Array (BGA) technique, which offers several advantages over conventional packages with peripheral leads and also accords with the development "roadmaps" of potential D2D customers, see Figure 6.



I/O	**256 BGA**	**256 QFP**
Pitch	**0.050"**	**0.020"**
Package Body Size	**27mm x 27mm**	**36mm x 36mm**

Figure 6 BGA Packaging Format

Advantages of the BGA format include, greater density of connections of an area array compared with peripheral (edge) connections, improved electrical and thermal performance and better manufacturing yields.

The improved electrical performance of the BGA package is due to shorter connection lengths giving lower inductance than peripheral leaded devices. The BGA package can also be designed with good thermal paths through the top of the package to a heatsink and through spare solder ball connections into the PCB or motherboard.

The BGA package provides more robust connections on a coarser pitch than peripheral leaded surface mount packages, which results in fewer problems during assembly. Assembly of the BGA package can be performed using existing surface mount placement and reflow equipment. Reflow involves passing the PCB or motherboard, with the BGA in position, through a hot gas convection oven in order to melt the solder and form the electrical connections between the BGA and the PCB or motherboard.

However, limitations of the BGA package include the requirement for X-Ray inspection and specialised rework equipment, and the reduced compliance provided by the shorter connections to allow for any thermal expansion mismatch between the BGA component and the PCB or motherboard.

3.3.1 Sphereing

Sphereing is a term used by D2D Ltd to describe the process of attaching solder balls (solder contacts) to the MCM-L package and so completing the fabrication of the MCM-L/BGA package. Solder paste is applied to the pads on the underside of the package to which the solder spheres are to be attached. The MCM-L packages are placed onto a modified printing machine with the underside of the package facing upwards. A template, with an array of holes just large enough to allow the spheres to pass through, is aligned above the MCM-L package and the solder spheres are sieved through the holes onto the solder paste deposits. These spheres are temporarily held in place on the package by the tackiness of the solder paste. Permanent attachment is achieved by passing the MCM- L package through a reflow process where the solder paste and spheres are melted and flown together to form the ball grid array connections on the MCM-L package.

3.3.2 Dispensing Solder Paste

Solder paste can be dispensed onto the MCM-L package using a number of different methods.

First method is using a full size brass stencil, mounted in a frame, with openings which match the footprint on the MCM-L package. This is used in conjunction with the modified printing machine. A squeegee is used to force paste through the openings onto the pads on the MCM-L package, providing the most consistent solder paste deposit in terms of volume and flatness. A flat topography is preferred for the sphereing process because it reduces the risk of shorts between adjacent sites on the ball grid array. This can be a high volume process with multiple packages processed simultaneously.

The single dot dispensing method uses a programmable fluid dispenser to apply a deposit of solder paste to each pad in turn. This method provides flexibility for small volume applications because it does not require specific tooling; for large volume applications, however, it is time consuming and does not provide the ideal paste deposit topography.

The final method is mini-stencilling which is similar to the first method but uses a much smaller stencil, similar size to the MCM-L package. This method gives similar results to the first method but is more of a manual operation, requiring less tooling than the first but being less suitable for high volume applications.

3.3.3 Assembly

The complete MCM-L package is finally attached to a PCB or motherboard using standard surface mount processes. Solder paste is stencilled on to the PCB or motherboard which is subsequently inspected for solder paste height, volume and accuracy. The MCM-L packages are placed on the PCB or motherboard together with other components and the board is passed through a convected hot gas reflow oven to form the solder joints.

X-Ray inspection is used to monitor the assembly process and will reveal shorts between adjacent joints, voids within the solder joint, misplacement, and poorly wetted joints. As stated previously, assembly yields are significantly better than with similar sized surface mount peripheral leaded devices.

4. Summary

This paper has introduced an innovative solution to resolving the interconnection bottleneck facing the electronics component packaging industry. An MCM-L packaging technology, incorporating BGA assembly technique, has been developed to provide a capability of supporting future silicon component design types for high performance applications.

This paper has described the challenges of the fabrication steps involved and how innovative processes and techniques have been developed to overcome limitations of the conventional manufacturing process for the fabrication of MCM-L packaging technology. All fabrication processes have been proven practical and economical for the introduction of such a component packaging solution. Although the technology has yet to be established as volume production, it is ready to be adopted by D2D's customers as potential solutions to their respective future technology roadmaps.

and form the electrical connections between the BGA and the PCB or motherboard.

However, limitations of the BGA package include the requirement for X-Ray inspection and specialised rework equipment, and the reduced compliance provided by the shorter connections to allow for any thermal expansion mismatch between the BGA component and the PCB or motherboard.

3.3.1 Sphereing

Sphereing is a term used by D2D Ltd to describe the process of attaching solder balls (solder contacts) to the MCM-L package and so completing the fabrication of the MCM-L/BGA package. Solder paste is applied to the pads on the underside of the package to which the solder spheres are to be attached. The MCM-L packages are placed onto a modified printing machine with the underside of the package facing upwards. A template, with an array of holes just large enough to allow the spheres to pass through, is aligned above the MCM-L package and the solder spheres are sieved through the holes onto the solder paste deposits. These spheres are temporarily held in place on the package by the tackiness of the solder paste. Permanent attachment is achieved by passing the MCM- L package through a reflow process where the solder paste and spheres are melted and flown together to form the ball grid array connections on the MCM-L package.

3.3.2 Dispensing Solder Paste

Solder paste can be dispensed onto the MCM-L package using a number of different methods.

First method is using a full size brass stencil, mounted in a frame, with openings which match the footprint on the MCM-L package. This is used in conjunction with the modified printing machine. A squeegee is used to force paste through the openings onto the pads on the MCM-L package, providing the most consistent solder paste deposit in terms of volume and flatness. A flat topography is preferred for the sphereing process because it reduces the risk of shorts between adjacent sites on the ball grid array. This can be a high volume process with multiple packages processed simultaneously.

The single dot dispensing method uses a programmable fluid dispenser to apply a deposit of solder paste to each pad in turn. This method provides flexibility for small volume applications because it does not require specific tooling; for large volume applications, however, it is time consuming and does not provide the ideal paste deposit topography.

The final method is mini-stencilling which is similar to the first method but uses a much smaller stencil, similar size to the MCM-L package. This method gives similar results to the first method but is more of a manual operation, requiring less tooling than the first but being less suitable for high volume applications.

3.3.3 Assembly

The complete MCM-L package is finally attached to a PCB or motherboard using standard surface mount processes. Solder paste is stencilled on to the PCB or motherboard which is subsequently inspected for solder paste height, volume and accuracy. The MCM-L packages are placed on the PCB or motherboard together with other components and the board is passed through a convected hot gas reflow oven to form the solder joints.

X-Ray inspection is used to monitor the assembly process and will reveal shorts between adjacent joints, voids within the solder joint, misplacement, and poorly wetted joints. As stated previously, assembly yields are significantly better than with similar sized surface mount peripheral leaded devices.

4. Summary

This paper has introduced an innovative solution to resolving the interconnection bottleneck facing the electronics component packaging industry. An MCM-L packaging technology, incorporating BGA assembly technique, has been developed to provide a capability of supporting future silicon component design types for high performance applications.

This paper has described the challenges of the fabrication steps involved and how innovative processes and techniques have been developed to overcome limitations of the conventional manufacturing process for the fabrication of MCM-L packaging technology. All fabrication processes have been proven practical and economical for the introduction of such a component packaging solution. Although the technology has yet to be established as volume production, it is ready to be adopted by D2D's customers as potential solutions to their respective future technology roadmaps.

During the phase of developing the technology, a number of patent applications have been granted.

Appendix

Rent's rule states $ckt=(N/K)^n$ where

N	is the number of signals
K	is a constant, which depends on the circuit type representing the commonality of signals. For high performance applications K=2.5
n	is a similar circuit type dependent constant, in the range 1.5 to 3.0. For high performance packages in the 10 to 100,000 circuit range n~1.79
ckt	is the average number of circuits supported by the N signals.

The two constants N, K depend on the type of application and the level of packaging, eg. chip, MCM, PCB [Tummala et al, 1989].

As an example, consider a system which contains a total of 2.5×10^5 gates which is to be built in a technology which allows 1×10^4 (ckt) gates on a 5mm square chip. Rent's rule suggests that each chip will require 430 signal pins (N).

Glossary

BGA	Ball Grid Array
COB	Chip on Board
CSP	Chip Scale Package
CTE	Coefficient of Thermal Expansion
DCA	Direct Chip Attach
IC	Integrated Circuit
I/O	Input/Output
MCM	Multi Chip Module
MCM-L	Multi Chip Module-Laminate
PCB	Printed Circuit Board
PGA	Pin Grid Array

QFP	Quad Fllat Package
smBGA	small-outline Ball Grid Array
TAB	Tape Automated Bonding
TCP	Tape Carrier Package

References

BRINKMANN, U. (ed). Lambda Highlights No.23. Lambda Physik, pp 4-5, 1990.

BRINKMANN, U. (ed). Lambda Highlights No.34. Lambda Physik, pp 2-5, 1992.

DOANE, D.A. and FRANZON, P.D. "Multichip Module and Alternatives–The Basics," Van Nostrand Reinhold, New York, p.160, 1993.

DYER, P.E. "Photochemical Processing of Electronic Materials," (ed). BOYD, I.W. and JACKMAN R.B., Academic Press, pp 359-385, 1992.

DYER, P.E., FARRAR, S.R. and KEY, P.H., Appl. Surf. Sci. 54, 255, 1992.

DYER, P.E., KEY, P.H. and SNELLING, H.V., Appl. Surf. Sci. 86, 18, 1995.

GOOSEY, M.T., HENDRIKSEN, M.W., MERRICKS, D. and SMITH, B., "Cost Effective Volume Production of Sequentially Built Substrates to Support Chip Scale Packages and Direct Chip Attach," ISHM European-MCM Conference Proceedings, 1996.

HUSSLA, I. and VISWANATHAN, R., Surf. Sci. 145, L488, 1984.

MILLER, J.C. and GEOHEGAN, D.B. (ed), Laser Ablation: Mechanisms and Applications-II, AIP Conference Proceedings 288, AIP, New York, 1993.

TUMMALA, R.R. and RYMASZEWSKI, E.J. (ed), Microelectronics Packaging Handbook, Van Nostrand Reinhold, New York, 1989.

Biographies

Michael Hendriksen

Michael Hendriksen joined ICL in 1988 as a sponsored student. He graduated from Loughborough University in 1992 with an honours degree in Electronics Manufacturing Engineering. Since then he has worked in the Manufacturing Technology group, involved with the development of advanced interconnect technologies and future component packaging solutions, such as Multi Chip Modules (MCM-L) and Chip Scale Packaging (CSP).

Steve Hodgkinson

Steve Hodgkinson joined ICL in 1987. He graduated from Staffordshire University in 1987 with an honours degree in Mechanical Engineering. Started work as a manufacturing engineer within Bare Board Manufacturing business. Currently working for the Manufacturing Technology group, he is involved with Ball Grid Array (BGA) technology, Multi Chip Modules (MCM-L) and optical registration projects. He became a Chartered Engineer in 1995.

Neil Taylor

Neil Taylor graduated from the University of Leicester in 1984 with an honours degree in Chemistry. He joined ICL in 1988 and started working in Bare Board Manufacture, involved with process support. He is currently working with the Manufacturing Technology group on future component packaging technologies, MCM-L and Chip Scale Packaging.

Rob Trussell

Rob Trussell graduated from Bradford University in 1984 with a degree in Electrical Engineering. He joined ICL and worked in Component Operations on digital, VLSI and memory devices. He is now working as a manufacturing development engineer on surface mount array packages (PBGA, CBGA, TBGA and Chip Scale Packaging CSP), including manufacturing process development, validation and electrical characterisation.

Development of Practical Verification Tools

D.J. King and R.D. Arthan

ICL, Winnersh, Berkshire

Abstract

Current best practice for high-assurance security and safety-critical software often requires the use of machine-checked rigorous mathematical techniques. Unfortunately, while there have been some notable successes, the development of software tools that adequately support such techniques has proved to be a difficult problem, albeit one that is slowly being solved. This paper describes some of ICL's contributions to this area of research and development in recent years. This work is based both on ICL's own experiences in building and using tools to support security applications and on work carried out by the Defence Research Agency into notations and methods for program verification.

1. Introduction

Writing in a recent volume of this journal, B.A. Wichmann asked the question "is quality assurance impossible?" [Wichmann, 1995]. The answer is probably "yes," in that we can never have certain knowledge of the correctness of an operational system under all possible conditions. Nonetheless, as Wichmann points out, there are techniques which can give genuine added confidence in the fitness-for-purpose of a system. One such technique, generally referred to as "formal methods," involves rigorous mathematical modelling to validate the critical aspects of a design or implementation. Machine-checked mathematical proof can then serve as a powerful adjunct to conventional checking and testing. This paper reports on recent work within ICL in developing practical tools for specifying and verifying formal mathematical models.

For the highest levels of assurance in security applications, UK and US government standards demand that security requirements be mathematically described and that mathematical modelling and proof be used to validate system architectures. For many years the High Assurance Team (HAT) at ICL Winnersh has been involved in developing and applying mathematical methods and tools for specification and verification of high assurance secure systems [Jones, 1992].

When we first began work in this area in the mid 1980s, the only tools available to support such work were academic research tools. We had some success in deploying one of these tools, namely Mike Gordon's Higher Order Logic (HOL) [Gordon, 1988] on several developments, including the UK's first product certified at UK Level 6. From the beginning, we found that we had to become tool-builders as well as tool-users in order to meet our customers' requirements. In particular, the customers generally made it mandatory that work be presented to them using the Z notation [Spivey, 1992]. This forced us to devote a great deal of time to customizing and extending the HOL tool to support Z-like notations.

In 1990, we began a three year research project, in collaboration with Program Validation Ltd. and the Universities of Kent and Cambridge. Our role in this project was to design and implement a completely re-engineered tool supporting specification and proof in HOL. In particular, we wanted a version of HOL that made it easier to provide support for other notations, such as Z. This project was very successful; by early 1993 we had a tool called ProofPower supporting both HOL and Z, which was sufficiently mature for us to make it available to other organisations and to give courses in its use. Since then, ProofPower has been used both for applications concerned with security and on case studies for other types of application.

Z was designed primarily as a language for specification, and therefore expressive power was of more interest to its designers than the requirement to undertake completely rigorous proof. The language, consequently, has many features that provide hard problems for the implementor of automated or semi-automated mechanical theorem-proving tools. We have continued to undertake research into these problems and to enhance the tool as opportunities have arisen.

Our main applications have been in high assurance secure systems. There is also considerable interest in Z and similar notations in the safety-critical community. Standards for safety-critical software [e.g., MOD, 1991] require the developer to provide an evaluator with convincing evidence that the software meets its specification. In safety-critical applications, e.g., avionics software, it has traditionally been an expensive and difficult task for both parties to prepare and maintain these so-called compliance arguments. The compliance argument for a system such as an engine controller might comprise scores of pages of narrative and mathematical material. Without tool support, it is difficult for the evaluator to relate the compliance argument to the program code and for the developer to keep it in step with the evolving requirements of the software.

As an outcome of approximately three years research into tools for software integrity, the Systems Integrity Research Team at the Defence Research Agency (DRA), Malvern, has designed a notation for presenting compliance arguments [Sennett, 1992, O'Halloran et al., 1994]. This notation permits programs written in the SPARK subset of Ada [Carré et al., 1992] to be specified and verified against Z specifications. Unlike most other such formalisms, DRA's approach gives the user considerable flexibility over the level of rigour to be applied. In 1994, DRA placed a contract with ICL to develop a tool supporting their notation. This tool was implemented on top of ProofPower and was delivered at the beginning of 1995. The notation and the tool are currently being assessed on a variety of case studies and being enhanced in the light of the experience gained.

The remainder of this paper is organised as follows:

Section 2 describes the ProofPower tool and shows it at work on a very simple security example using the Z notation.
Section 3 describes the Compliance Notation and gives a simple example of its use.
Section 4 makes some concluding remarks.

A detailed knowledge of Z is not required to understand sections 2 and 3, but some familiarity with the logical symbols generally used in discrete mathematics will be helpful in understanding all the technical details.

2. ProofPower

2.1 Overview

The heart of ProofPower is essentially a system for managing a database of logical theories and conducting proofs about the objects described by those theories. This part of the system follows the LCF paradigm [Gordon et al., 1979] and is implemented in the powerful functional programming language Standard ML [Milner et al., 1990]. Since ML is an interactive language, it serves not only as the implementation language but also as the command language for the system.

ProofPower follows the tradition of notations, such as Z, in supporting a document-based paradigm for developing and managing specifications and proofs. To write a specification is to produce a document in which the mathematics is interleaved with explanatory narrative. Editors, using appropriate fonts to provide the necessary mathematical symbols, are used to provide a near WYSIWYG appearance for this class of material.

Specifications and proofs are generally checked incrementally by ProofPower as they are being developed. When a fragment of the document being developed is to be checked, the user selects it with the mouse and then "executes" the fragment. This causes an appropriate ML command to be executed which will check the fragment and, if it is correct, cause the state of the ProofPower system to be updated appropriately; for example, when a logical definition is executed, the logical content of the definition is added to the theory database.

2.2 A Simple Example

We are now going to present, in the Z notation, the information-flow security properties of a very simple system. The treatment will follow in microcosm parts of the approach recommended by the UK regulatory authorities for high-assurance secure systems [CESG, 1991]. Jones gives further examples and a more detailed discussion of various approaches [Jones, 1992].

There are three parts to this example. Firstly, we identify the salient characteristics of the systems of interest and express a *security policy* by defining the subclass of systems which meet the desired information-flow constraints. Secondly, we describe a particular system which we will call an *architectural model.* Finally, we prove that the architectural model does indeed conform to the security policy.

2.2.1 Security Policy

We are going to consider systems whose state contains data at two sensitivity levels, unclassified data and classified data, and we will be concerned with security properties which restrict the data flows between the two levels. The internal structure of the data will not be relevant to our considerations. We can use, therefore, the following Z paragraphs to model the states of the systems of interest. The first paragraph introduces a type *DATA* and the second paragraph defines the set *STATE* to comprise pairs of *DATA* values labelled unclassified and classified.

z

$$\begin{array}{|l} \underline{STATE\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad} \\ \qquad classified : DATA; \\ \qquad unclassified : DATA \\ \hline \end{array}$$

We will model individual systems by their state-transition functions. The set of all systems of interest will then be modelled by the following Z paragraph which defines SYSTEM to comprise all functions from the set *STATE* to itself. Note that *SYSTEM* includes all possible transition functions, not just the secure ones.

z

$$\mid SYSTEM \mathrel{\widehat{=}} STATE \rightarrow STATE$$

Now we want to distinguish, within the set *SYSTEM*, those transition functions which satisfy the required constraints on data flows. Informally, let us assume that what is required is that no transition shall cause information to flow from the classified component to the unclassified component. Another way of saying this is that observation of the unclassified component in the state after a transition does not allow us to distinguish two before-states which have the same unclassified component. We can therefore capture our informal notion of security in the following definition of the set, *SECURE*, comprising transition functions, *trf*, which have the desired property:

z

$$\begin{array}{|ll} SECURE \mathrel{\widehat{=}} & \\ \{ & trf : SYSTEM \\ \mid & \forall st1,\ st2\colon STATE \\ \bullet & st1.unclassified = st2.unclassified \\ \Rightarrow & (trf\ st1).unclassified = (trf\ st2).unclassified\ \} \end{array}$$

2.2.2 Architectural Model

We are now going to produce a model of the architecture of a very simple system. This model will be an element of the set *SYSTEM* defined in the previous section. We will define the transition function of the model in terms of two auxiliary functions, one to compute the classified part of the state, and the other to compute the unclassified part. The actual computation carried out by these functions will be irrelevant to the security of the system. We introduce the two functions, *c_trf* and *u_trf* using the following Z paragraph which simply asserts that *c_trf* and *u_trf* are some functions with the specified signatures.

z

$$
\begin{array}{|l}
c\_trf : DATA \times DATA \rightarrow DATA; \\
u\_trf : DATA \rightarrow DATA
\end{array}
$$

We now use these functions to construct the architectural model of the system. The state transitions carried out by the system use *u_trf* and *c_trf* to compute their respective components of the state. The architectural decision embodied here is that while *c_trf* has both components of the state available to it, *u_trf* can only work with the unclassified component. This is a very simple model of the sort of separation which might be achieved in practice using hardware or software access control mechanisms of some sort.

z

$$
\begin{array}{|ll}
 & system : SYSTEM \\
\hline
\forall st : STATE \bullet & \\
 & (system\ st).classified = c\_trf\ (st.classified, st.unclassified) \\
\wedge & (system\ st).unclassified = u\_trf\ (st.unclassified)
\end{array}
$$

2.2.3 Proof of Security

In section 2.2.1, we have given a mathematical specification of information-flow security for a simple class of systems, and in section 2.2.2, we have given a mathematical construction of a member of this class which we believe to be secure. We can now test this belief by attempting to prove it mathematically. In this section we will show the steps of a ProofPower session in which this proof is conducted. We will suppress the Standard ML commands which control the proof, and just show the output of the interactive subsystem of ProofPower used to find proofs.

We begin by telling ProofPower the goal we want to prove, namely that the architectural model *system* is a member of the set *SECURE* of secure systems. ProofPower acknowledges this by echoing back our goal, together with a little bit of decoration designed to make later steps more readable.

ProofPower Output

```
|(* ?⊢ *) ⓩsystem ∈ SECURE⌝
```

Our first step is essentially to expand the definition of *SECURE*, resulting in the following goal. Note that ProofPower has replaced *SECURE* in the goal with its definition and then simplified away all the set notation.

ProofPower Output
```
|(* ?⊢ *) ⌜∀ st1, st2 : STATE
|            • st1.unclassified = st2.unclassified
|            ⇒ (system st1).unclassified = (system st2).unclassified⌝
```

We now need to attack the universal quantifier which has appeared in our goal. This is done with a standard logical transformation which results in a goal with three assumptions. The idea is that we may now take the facts labelled 1, 2 and 3 as given and deduce the conclusion from them.

ProofPower Output
```
|(*  3 *) ⌜st1 ∈ STATE⌝
|(*  2 *) ⌜st2 ∈ STATE⌝
|(*  1 *) ⌜st1.unclassified = st2.unclassified⌝
|
|(* ?⊢ *) ⌜(system st1).unclassified = (system st2).unclassified⌝
```

We must now make use of the definition of *system*. After rewriting the goal with that definition, we arrive at the following goal.

ProofPower Output
```
|(*  3 *) ⌜st1 ∈ STATE⌝
|(*  2 *) ⌜st2 ∈ STATE⌝
|(*  1 *) ⌜st1.unclassified = st2.unclassified⌝
|
|(* ?⊢ *) ⌜u_trf (st1.unclassified) = u_trf (st2.unclassified)⌝
```

It is now apparent that the equation in the assumptions implies the truth of the conclusion. Using the assumptions as rewrite rules should complete the proof, and ProofPower does indeed tell us we have finished. We then execute a command to save the newly proved theorem in the database, which responds with:

ProofPower Output

```
|Now 0 goals on the main goal stack
|val system_secure_thm = ⊢ system ∈ SECURE : THM
```

3. The Compliance Tool

3.1 The Compliance Notation

The target application of DRA's Compliance Notation [Sennett, 1992, O'Halloran et al., 1994] is the provision of convincing arguments that a piece of code complies with a detailed technical requirement. In typical safety-critical applications, some parts of the requirements are likely to be of a mathematical nature, albeit informally expressed. The compliance argument will often be prepared quite late in the development life-cycle, since much of it will depend on the fine details of the implementation. The argument will also need to be amenable to change as the software evolves. Since pragmatic constraints will prohibit full mathematical rigour at every stage of the argument, the notation must allow the developer to choose the level of formality at which to pitch the argument.

The Compliance Notation combines two key ideas:

- *Literate programming* [Knuth, 1984], whereby the design of a program is presented in a readable document in which the program text is embedded. In literate programming, the order of presentation of the program text is governed by the designer's preference for explaining the design rather than the programming language's rules for sequencing of program constructs

- *Refinement calculus* [Morgan, 1990], whereby the designer can use mathematical predicates as a kind of abstract program statement and can assert that one such program statement "refines" (i.e., correctly implements) another. At one extreme, a predicate expressing the state changes to be achieved by the program as a whole constitutes a rigorous mathematical specification of the program. At the other extreme, an actual programming language command can be viewed as a predicate. A sequence of commands which refine the specification of the program as a whole is a correct implementation of that specification.

The Compliance Notation exploits these ideas taking a subset of Ada, called SPARK [Carré et al., 1992] as the programming language, and using the specification language Z [Spivey, 1992] to express the mathematical predicates. These two languages are among the more important in military safety-critical applications. The ideas are applicable to most imperative programming languages, although, typically, a rigorous mathematical treatment would only be available for a "safe" subset of the language. The popularity of SPARK for safety-critical work suggests that using a small subset of a complex language is less of a problem than is often alleged. A discussion of the issues of programming language design is outside the scope of this document but the design philosophy of SPARK is described by Carré [Carré et al., 1992].

The literate script idea is realised by identifying places in the SPARK syntax where the user can use a place-holder instead of a programming language construct. This enables a compliance argument to be presented in a readable document in which the order of presentation is not constrained by the programming language syntax. The construct to fill in a place-holder can be given in the document at the point where it is relevant to the argument. For example, the internal details of an intermediate data structure need not be revealed until they are necessary. Where appropriate, the syntax for the place-holders permits the user to specify program behaviour mathematically by means of Z pre- and post-conditions.

Fragments of a program which are mathematically specified give rise to verification conditions. A verification condition (VC) is a mathematical proposition which, if true, guarantees the compliance of that program fragment with its specification. These VCs can be automatically generated from the Compliance Notation presentation of the program. The designer can then give either informal or rigorous justifications of the VCs to the evaluator as part of the compliance argument. If a very high assurance is required, the VCs are amenable to fully machine-checked proof. A simple example of the Compliance Notation is given later in this paper.

The notation allows the user to choose how much of the program is to be treated with full rigour. Fragments of program which are not critical can be entered without any mathematics. For example, the user can elect simply to provide the code for a subprogram body without having to specify it in any way. Where desired, the user can even circumvent the rather stringent syntactic restrictions imposed by SPARK and insert arbitrary Ada code, for example, to access library packages which do not conform to the SPARK subset. The notation makes it clear where these shortcuts have been taken, and the literate programming style allows clear documentation of what has been done and why.

The SPARK source of a program can be mechanically extracted from the document containing its compliance argument. Thus, in the interests of maintainability and traceability, the source document of the argument can serve as the master copy of the program source for future development purposes. If machine-checked proof techniques are used, then these can provide a very rigorous approach to impact analysis as software evolves.

3.2 The Compliance Tool

At the beginning of 1994, we began a project under contract to DRA to construct a tool to support the Compliance Notation. The basic requirement was for a tool to read a Compliance Notation script, perform syntax and type checking, and then output the SPARK program and the VCs as a Z document. Outputting the SPARK program is a relatively simple task, although not completely trivial because the program constructs in the script have to be assembled into the right order. The generation of VCs is a much more complex algorithm and DRA have devoted considerable effort to specifying all aspects of the tools that are critical to the soundness of the approach. This specification provided an objective and precise record of DRA's technical requirement.

The Compliance Tool, which was delivered at the beginning of 1995, is implemented on top of ProofPower. In addition to its support for specification and proof in Z that are described in section 2.2 of this paper, ProofPower provides some additional advantages for implementing this type of tool, including:

- Very high level implementation language. ProofPower is programmed using Standard ML, a powerful functional programming language, allowing complex algorithms for manipulating syntax to be coded easily and concisely.

- Library code for manipulating Z. The support for Z in ProofPower includes an extensive programming interface for manipulating Z syntax. The DRA was particularly concerned that the implementation could easily be traced back to the specifications. An experiment to code parts of its VC generation algorithm in ML showed that it was straightforward to write the code so that implementation details would not obscure the connection with the specifications.

- Document preparation. ProofPower supports presentation of specifications and proofs in good quality documents via an interface to LaTeX. It would be easy to extend this to handle the document preparation requirements for the new tool.

The development of the tool was very successful, both because of the good fit of ProofPower to the problem and because of the considerable efforts DRA had taken to specify their requirements for the tool.

3.3 An Example

To give a flavour of how the Compliance Notation is used, a simple example follows. The example is a very small one, involving only a few lines of Ada code, and is designed just to give the flavour of the Compliance Notation approach. Case studies have been carried out on life-size examples involving non-trivial algorithms and exploiting Ada's features for programming in the large.

Compliance Notation

```
procedure EG (X : in out INTEGER; Y : in out INTEGER)
is
⟨ space for local declarations ⟩                              (1)
begin
    Δ X, Y [ X = X₀*X₀ + Y₀*Y₀ ∧ Y = 2*X₀*Y₀]                 (2)
end EG;
```

Here, what we have is a regular Ada procedure declaration with certain parts replaced by place-holders to be filled in later. The first place-holder (labelled (1)) makes room for the declarations of any local variables which are required, and includes a comment to that effect. The second place-holder (labelled (2)) stands in for the statements forming the body of the procedure and includes a mathematical specification of what the body must achieve. This specification statement is introduced by a Δ followed by a list of the variables which the body may change and a mathematical predicate in square brackets specifying the desired state change. In this case, the predicate requires that the output values of X and Y should be equal to the indicated expressions involving their input values (written X_0 and Y_0).

Note that, apart from the decisions it records about the interface (e.g., the order of the parameters), there is no implementation bias in this specification of the procedure. Note also that the notation lets us use any mathematical assertions about the code as a specification statement, and so the tool can be used as a general means for reasoning about Ada programs. For example, one can have a specification statement which asserts that the computations to be carried out do not cause arithmetic overflow, or that the value assigned to X is not negative.

To implement the procedure, we must decide on an algorithm, and fill in the place-holders accordingly. By way of example, we are going to give a rather contrived solution (which might just be justified in practice by a desire to minimise the number of multiplications to be performed). First of all, we decide we need a local variable. The literate programming approach allows us to introduce local variables at the point where they are needed in the explanation of the algorithm. In this case, we need the local variable straight away, and we ask for it as follows:

Compliance Notation

```
| (1) ≡
|         A: INTEGER;
```

Now we use some special syntax provided by the Compliance Notation to carry out what is called a *refinement step*. In this case, the refinement step supplies Ada code to implement the algorithm specified mathematically at label (2) above.

Compliance Notation

```
| (2) ⊑
|         A := X + Y;
|         Y := X*Y;
|         Y := Y + Y;
|         X := A*A - Y;
```

Here we have implemented the procedure body in one refinement step. For a more complex algorithm, one can use further specification statements in the refinement step, and so present the salient features of the algorithm in a structured way. A more extended example may be found in the original paper on the Compliance Notation [Sennett, 1992].

At this point, it would be quite easy to make a mistake in the above coding. However, by analysing the statements in the refinement step against the specification statement being refined, the VC generator component of the tool allows us to check the coding by purely mathematical means. In this case, the only VC produced is the following conjecture (which is recorded in the Z document produced by processing the above script).

z

$$\begin{array}{l} ?\vdash\ \forall X : INTEGER;\ Y : INTEGER \bullet \\ \quad (X + Y) * (X + Y) - (X * Y + X * Y) = X * X + Y * Y \\ \land X * Y + X * Y = 2 * X * Y \end{array}$$

This conjecture is easy to check by algebra, either on paper, or, more easily, by using the automated proof facilities, provided with ProofPower, which include a decision procedure covering a useful range of arithmetic facts. This decision procedure will give an automatic proof of this conjecture, and, if it fails in a simple case like this one, then the most likely cause is an error in the coding.

To conclude the example, we can show the Ada code generated by processing our script:

SPARK Program

```
PROCEDURE EG (X, Y : IN OUT INTEGER)
IS
  A : INTEGER;
BEGIN
  A := X + Y;
  Y := X * Y;
  Y := Y + Y;
  X := A * A - Y;
END EG;
```

Note that this type of formal verification is essentially complementary to testing. Testing generally deals with the actual object code in a situation as close as possible to the real target environment and, while problems arising from interaction with the environment can be detected by testing, time constraints almost invariably prohibit complete test coverage. Formal verification, as discussed here, is concerned with the algorithm as expressed in the source language. While verification will not normally detect problems with the environment, it does offer complete coverage at the algorithmic level. In a sense, mathematical reasoning allows us to consider an infinite set of test cases in finite time.

We have presented a very small example. This of course raises the question of whether the methods and tools will scale up to real-world problems. Work on case studies, to date, has shown that both the method and the tool will handle significant examples. For instance, an Ada package taken from the safety-critical parts of the operator interface in an actual development has been specified and verified using the notation. The specification covers about 25 pages and includes a comprehensive narrative relating the mathematical treatment to the natural language statement of requirements. Processing the specification produces about 300 lines of Ada and 61 VCs, of which 28 of the VCs can be proved automatically. The proofs of the remaining VCs vary in complexity from relatively simple checks on the correctness of the control logic to more complex reasoning to analyse the structure of the code and its specification and then to verify the mathematical facts underlying the algorithms used. Carrying out the proofs took about 15 days including 5 days of familiarisation and analysis.

4. Conclusions

Research into methods and tools for high assurance security at ICL and at DRA for safety-critical applications has resulted in novel tools. These tools facilitate the use of formal methods for securing increased confidence in fitness-for-purpose. We have illustrated the use of the tools on "toy" examples taken from opposite ends of the development life-cycle. Indeed, requirements validation on the one hand and code verification on the other seem to be the niches where rigorous mathematical methods have much to offer.

Readers, familiar with the "European" tradition of formal methods, will note that both the examples we have presented differ in many respects from the use of notations such as Z and VDM as design and development methods as generally described in the literature. Too often, formal methods have been oversold as a panacea when, in fact, their main benefit would seem to be in particular niches where the details of the method have to be adapted to fit the special characteristics of the problem at hand.

In the case of the Compliance Notation, the Z notation and the SPARK subset of Ada are brought together to combine the expressive power of Z with the respected software engineering advantages of SPARK in the safety-critical domain. The capability of developing fully machine-checked proofs using ProofPower offers a very high degree of assurance, while the flexibility inherent in the Compliance Notation permits the user to choose an appropriate level of formality.

We would not claim that the tools and methods we have described are currently appropriate for mainstream, low-to-medium assurance software development, or, indeed, that they ever will be. It seems likely that the costs, in terms of human skills and elapsed time, of rigorous mathematical methods will rarely be justified in cases where cost and quality can simply be traded off against each other. Nonetheless, as software plays an ever-increasing role in safety-critical applications, we believe that such methods will become an important part of the developer's battery of validation and verification techniques.

Acknowledgements

We would like to thank Colin O'Halloran, Chris Sennett and Alf Smith of DRA for their painstaking technical work in designing and specifying the Compliance Notation, for their assistance and advice with the development of the tool and for providing comments during the preparation of this paper.

The ProofPower tool, on which the implementation was based, has benefited from contributions from all the other past and present members of the High Assurance Team; namely, Kevin Blackburn, Adrian Hammon, Andrew Hayward, Barry Homer, Roger Jones, Gill Prout, Geoff Scullard and Roger Stokes.

References

CARRÉ, B.A., JENNINGS, T.J., MACLENNAN, F.J., FARROW, P.F., GARNSWORTHY, P.F., "SPARK - The SPADE Ada Kernel," Program Validation Limited, 1992.

CESG. "Computer Security Manual F — A Formal Development Methodology for High Confidence Systems," GCHQ, 1991.

GORDON, M.J.C., MILNER, A.J, WADSWORTH, C.P. "Edinburgh LCF," LNCS, volume 78, Springer, 1979.

GORDON, M.J.C., "HOL: A Proof Generating System for Higher Order Logic," Proceedings of the 1989 Banff Conference on Hardware

Verification, G. Birtwistle and P.A. Subrahmanyam (Eds.), Springer, 1988.

JONES, R.B. "Methods and Tools for the Verification of Critical Properties," Proceedings of the 5th Refinement Workshop, Cliff B. Jones, Roger C. Shaw and Tim Denvir (Eds.), Springer, 1992.

KNUTH, D.E. "Literate Programming," Donald E. Knuth, The Computer Journal, 27(2), pp 97-111, 1984.

MILNER, R., TOFTE, M., HARPER R. "The definition of Standard ML," MIT Press, 1990.

MOD Interim Defence Standard 00-55. "The Procurement of Safety-critical Software in Defence Equipment," Ministry of Defence, April, 1991.

MORGAN, C.C. Programming from Specifications, Prentice-Hall, 1990.

O'HALLORAN, C.M., SENNETT, C.T., SMITH, A. "Specification of the Compliance Notation for SPARK and Z," DRA Technical Report DRA/CIS/CSE3/TR/94/27, 1994.

SENNETT, C.T. "Demonstrating the Compliance of Ada Programs with Z Specifications," Proceedings of the 5th Refinement Workshop, Cliff B. Jones, Roger C. Shaw and Tim Denvir (Eds.), Springer, 1992.

SPIVEY, J.M. "The Z Notation: A Reference Manual," Second Edition, Prentice-Hall, 1992.

WICHMANN, B.A. "Why is it difficult producing safety-critical software," Ingenuity: the ICL Technical Journal, Vol. 10, Issue 1, May, 1995.

Information about DRA's research into Software Integrity including the specification of the Compliance Notation [O'Halloran et al., 1994] may be found on the World Wide Web at URL:

http://daedalus.dra.hmg.gb/hewitt/swi/swi.html

Technical and marketing information on ProofPower and the Compliance Tool may be found on the World Wide Web at URL:

http://www.to.icl.fi/ICLE/ProofPower/index.html

Biographies

David King

David King has a Ph.D. in Computer Science and joined ICL in 1988. He has been involved in the design of secure applications and in the development and application of ProofPower. He led the team that implemented the Compliance Tool and has subsequently contributed to work on enhancing it. David recently left ICL to join Hoskyns as a security consultant.

Rob Arthan

Rob Arthan has a Ph.D. in Pure Mathematics and has worked at ICL since 1985, first on compiler technology for ICL's Distributed Array Processor and, since 1989, on using formal methods in security applications and on developing tools for formal specification and proof. He led the team that designed and implemented ProofPower and was heavily involved in the development of the Compliance Tool.

Coupling ORACLE™ with ECLiPSe

S.K. Das[1] and J. Dicker[2]

[1]IC-Parc , William Penney Laboratory, Imperial College, London
[2]ICL, Research and Advanced Technology, Bracknell, Berks

Abstract

We describe the design and implementation of an interface between a Prolog system, ECLiPSe, and the relational database management system, Oracle. ECLiPSe is used to solve sophisticated planning and scheduling problems and sometimes needs to access data stored in a database system to do this. A typical ECLiPSe application would be job shop scheduling and a legacy system might be used to store the orders that have to be scheduled by the system. The interface, called CORE, enables the writer of the ECLiPSe application to query the database for both the whole set of answers and tuple-at-a-time. We also describe the use of the interface in an application demonstrator for generating lecture timetables at Imperial College.

1. Introduction

The aim of this paper is to explain the requirement for and design of an interface between the Prolog system, ECLiPSe, and the relational database, Oracle. ECLiPSe is used to solve sophisticated planning and scheduling problems and sometimes needs to access data stored in a database system such as Oracle. The interface to be presented here enables the developer of the ECLiPSe application to query the database both on a tuple-at-a-time basis and on a retrieved set basis. This work was carried out on the DTI/EPSRC funded research and development project CHRONOS and was targeted towards writing application demonstrators for substantial problems. One of the demonstrators used in the project was an application from Imperial College itself which is described below in detail.

1.1 The Teaching Space Utilization Problem

The ECLiPSe language is typically used in applications which solve planning and scheduling problems. In the CHRONOS project our focus was on applying it to resource allocation problems, for instance allocating classrooms within lecture timetables. Our task was to write an application that did lecture and tutorial timetabling for departments at Imperial College. Each department has its own lecture rooms which can seat a certain number of students and have various available facilities, such as overhead projectors, videos or computers. Lecturers request a classroom with a certain capacity and facilities and they may also have preferences as to the time of the class. The application program will attempt to timetable the lectures and tutorials. The program takes into account factors such as making sure that the lecture room is large enough for the number of students and tries to take account of an individual lecturer's preferences, for example, the lecturer may particularly dislike giving lectures before 10 am.

The data about classrooms, lecturers and students are stored on a central Oracle database and finished timetables are also stored on an Oracle database so that they can be queried. The program that does the timetabling is written in ECLiPSe (and its extension for handling constraints) but needs to access the database for the information it requires to do the timetabling and to store the finished timetable.

The need to read and write to an external legacy database is not unique to this application. In many cases the application will be built on top of an existing database.

For a general planning or scheduling application, the data in a database provides a description of the resources required by the planning system during the process of generating a plan. Traditionally, the relevant data from existing databases are copied either into memory or to a flat file so that they can be used by the local environment, in our application an ECLiPSe program, to construct a plan. When a plan is constructed, the effects are reflected on the existing database through an update. This has several disadvantages:

- While the plan is being constructed, the status of the existing database might change and, therefore, the local copy of the relevant data is no longer correct.

- The relevant data in the existing database, possibly a large volume, can be locked before copying to prevent access by other users. This approach is clearly not practical as it may prevent other users from accessing the database for a long time.

- Updates to the existing database will not be instantaneous.

In view of the above, we decided to create a link from the planning component implemented in Prolog (specifically ECLiPSe) to the existing Oracle database to enable us to read and update interactively. The link also provided a manageable persistent store for Prolog. This document describes the architecture and implementation detail of the system.

1.2 Organization of the paper

The organization of the rest of the document is as follows. The next section provides some background necessary to understand the remainder of the paper. The loose-coupling architecture which we have adopted for linking is presented in Section 3 and the implementation details in Section 4. Two different kinds of query are possible from the Prolog system; whole set query and tuple-at-a-time query. The implementation details of these two approaches are described in Sections 4.2 and 4.3 respectively and the C data structure needed to communicate with Oracle for this implementation is given in Section 4.1. The process of reverse engineering extracts the description of the database schema at the Prolog level. The implementation detail of this process is described in Section 6. Section 7 provides some possible directions for further extensions to the system.

2. Background

This section provides some background on Prolog and relational databases and also describes various ways of interfacing the two.

2.1 Prolog

Prolog is a logic programming language which allows users to specify both application knowledge (the program) and declarative queries. The problem of how to solve a query using the knowledge is left to the interpreter. Prolog retrieves answers one by one using the backtracking mechanism. A pure Prolog program consists of facts and rules. For example, the following program represents the knowledge about parent-ancestor relationship in a family:

```
parent(robert, janique).
parent(robert, daniel).
parent(janique, sebastien).
ancestor(X, Y):- parent(X, Y).
ancestor(X, Y):- parent(X, Z), ancestor(Z, Y).
```

The first three are facts and the rest are rules. The symbols **ancestor** and **parent** are *predicates* or *relations*, **robert, janique, madeleine** and

sebastien are *atoms* and **X** and **Y** are *variables*. A rule of the form **parent(X, Y)** states that **X** is a parent of **Y**. The relation **ancestor** is recursively defined by the two rules. The first rule states that **X** is an ancestor of **Y** if **X** is a parent of **Y** and the second rule states that **X** is an ancestor of **Y** if **X** is a parent of **Z** and **Z** is an ancestor of **Y**. An example of a query or goal to the above program to find out who are the descendants of **robert** is encoded as follows:

```
?- ancestor(robert, Y).
```

The Prolog interpreter will *instantiate* the variable **Y** to the first answer as follows:

```
Y = janique   More? (;)
```

Upon receiving ';' from the user, the interpreter backtracks and the next answer is retrieved. This process can be continued until no more answers are found. Queries involving sets of answers are common in a typical database environment. These kinds of set queries can be implemented in Prolog by using its **setof** or **findall** predicate. For example, the following query retrieves all descendants of **robert** and returns the answers as a list:

```
?- findall(Y, ancestor(robert, Y), L).

L = [janique, daniel, sebastien]

yes.
```

ECLiPSe is a Prolog based system whose aim is to serve as a platform for integrating various logic programming extensions. ECLiPSe is produced by the European Computer Industry Research Centre (ECRC). The logic programming extensions which ECLiPSe provides include constraint libraries which support the solving of complex scheduling problems, e.g., job shop scheduling or timetabling.

2.2 Relational Databases

A *relational database* [Codd, 1970] consists of a collection of *relations* or *tables* with a description of them which is called a *schema*. The rows of the tables are called *tuples* and the columns are called *attributes*. Each attribute has an associated *domain* of values. An example of an **EMPLOYEE** relation is shown in Table 1. In this table, the attributes are Id, Name and Designation and there are five tuples in the table.

Integrity constraints [Das, 1992] are properties of a database that the data must satisfy. An example of an integrity constraint on the **EMPLOYEE** table could be that the identification field is the primary key. Another example of an integrity constraint would be to restrict the values that the designation of an **EMPLOYEE** could take to the set of programmer, software engineer and project leader. The schema for the **EMPLOYEE** table, together with these constraints, could be created in an Oracle environment using its query language SQL as follows:

Id	Name	Designation
1	john	programmer
2	susan	software_eng
3	tom	programmer
4	brenda	project_leader
5	harry	programmer

Table 1: The EMPLOYEE table.

```
CREATE TABLE employee
    (id            INTEGER      CONSTRAINT pk_emp PRIMARY KEY,
     name          CHAR(16),
     designation   CHAR(16)     CONSTRAINT ck_desig CHECK (desig IN
             ('programmer', 'software_eng', 'project_leader')));
```

A *view* [Codd, 1970] can be thought of as a mask overlaying one or more tables such that the columns in the view are found in one or more underlying tables or are constructed using the columns of underlying tables. An example of a view of the above table which extracts employees who are programmers is shown in Table 2.

Id	Name
1	john
3	tom
5	harry

Table 2 A view of the Employee table.

The view in Table 2 can be created in Oracle using SQL as follows:

```
CREATE VIEW qualified_cc AS
          SELECT   id, name
          FROM     employee
          WHERE    designation = 'programmer';
```

2.3 Interfacing Prolog and Relational Databases

Coupling [Berghel, 1985], [Bocca, 1986], [Brodie & Jarke, 1989], [Ceri et al, 1986], [Gardarin & Valduriez, 1989], [Ioannides et al, 1988] is the interfacing of a Prolog system, such as ECLiPSe [ECRC, 1995], with a conventional database management system, such as Oracle [Oracle, 1994]. There are two different types of coupling possible: *loose coupling* and *tight coupling* (Figure 1). In the loose coupling approach, Oracle is invoked from ECLiPSe through special built-in predicates using the syntax of SQL. This approach was used in a loosely coupled interface between ICL's DecisionPower Prolog and Ingres called SEDUCE. Users of the planning system on ECLiPSe will have a model of the existing database as a relational database model which has been obtained by a process of reverse engineering. In the tight-coupling approach, database relations can be used as normal predefined predicates from ECLiPSe by building a rule interpreter on top of Oracle.

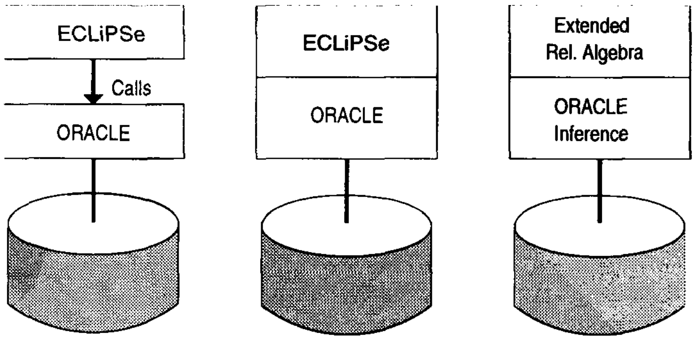


Figure 1 (From left to right) loose coupling, tight coupling and integration

Integration [Gozzi et al, 1990] differs from coupling in that it produces a single system which provides a number of the facilities of conventional database systems, plus efficient management of rules and a recursive query processor. In other words, an integrated architecture is specially designed to use a logic programming system as a deductive database system.

The above definition of coupling can be extended to individual levels of a system as follows:

- *Logical level coupling* is at the top level of the system and brings together a logic programming language and a relational data manipulation language [Naqvi & Tsur, 1989] to enhance the expressive power of data manipulation languages [Waugh et al, 1990], [Chang & Walker, 1986]

- *Function level coupling* brings together the logical inference mechanisms of a logic programming system and database management functions of a relational database management system so as to retrieve or deduce information efficiently

- *Physical level coupling* deals with the physical organization of tuples of relational database management systems and the rules of deductive database management systems.

A particular level (either logical, functional or physical) is *loosely coupled* if the two components co-exist independently; otherwise it is *tightly coupled* [Tsur, 1988].

2.4 Oracle Pre-compiler and Prolog External Procedures

An Oracle *pre-compiler* is a programming tool that allows a user to embed SQL statements in a high-level host program. Six Oracle pre-compilers are available and a C pre-compiler (Pro*C) is one of them [Oracle, 1993]. Suitably compiled C-libraries are callable from ECLiPSe.

Since a Prolog pre-compiler is not provided, we are forced to adopt C as an intermediate language for communication between Oracle and ECLiPSe. Each member of a library is treated as an external procedure which imitates the action of a Prolog procedure. The flow-chart in Figure 2 shows how a C program with embedded SQL statements can be converted to an external procedure to ECLiPSe.

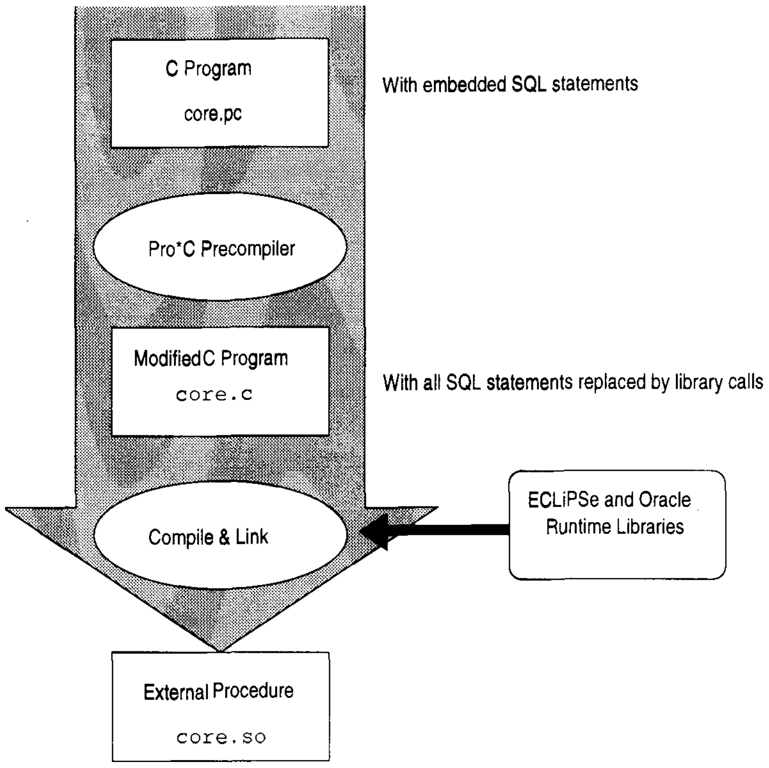

C Program
core.pc
With embedded SQL statements
Pro*C Precompiler
Modified C Program
core.c
With all SQL statements replaced by library calls
Compile & Link
ECLiPSe and Oracle Runtime Libraries
External Procedure
core.so

Figure 2 Embedded SQL program development

3. Architecture and Communication

The architecture of the Teaching Space Utilization (TSU) application is shown in Figure 3. The user of the application has a graphical user interface (GUI) with the planner (the application program in ECLiPSe). The planner requires data from the Oracle database which it gets using the coupling interface.

The functional level coupling within our architecture is loose; i.e., the inference mechanism of ECLiPSe is separate from the query evaluation mechanism of Oracle and ECLiPSe clauses exist independently of the tuples of Oracle. At the logical level the coupling is both loose and tight. Oracle is not invoked through special built-in predicates using the syntax of SQL. Planning users, e.g., the ECLiPSe user who writes the TSU application, do not have the model of the existing database as a relational database model. Database relations can be used as normal pre-defined predicates from ECLiPSe.

Because the planning user does not have a relational database model of the existing database he needs to obtain the schema of the database in ECLiPSe predicates. These predicates are obtained by reverse engineering the schema of the relational database. The planning user can then model his application using the ECLiPSe schema of the database. The reverse engineering process is explained in more detail in Section 5.

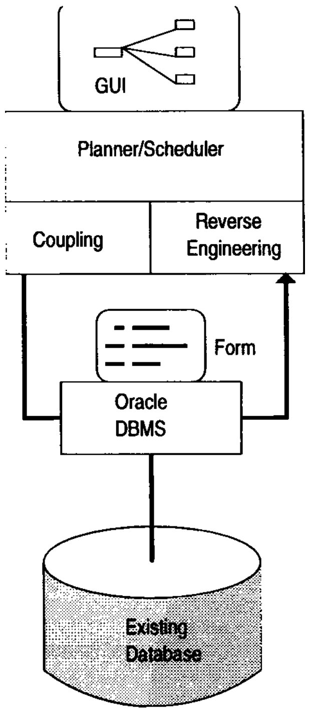


Figure 3 Environment of data retrieved from a planner

4. Implementation

The overall idea is that the database relations are accessed through a program written in C with embedded SQL statements. This C program is compiled as a set of external procedures to imitate the behaviour of some Prolog predicate corresponding to some database relations. The communication structure is shown in Figure 4.

This invocation of a predicate imitating a database relation results in a call to the corresponding external procedure. Therefore, there is a correspondence between the arguments of the external procedure and the arguments of the Prolog predicate. Each prolog argument represents two C arguments, one for the value and the other for its type. The external procedure which implements the set query evaluation is given below:

```
int p_oracle_set_query(sval, stag, fval, ftag, rval, rtag)
    value sval, fval, rval;
    type  stag, ftag, rtag;
{
  ... }
```

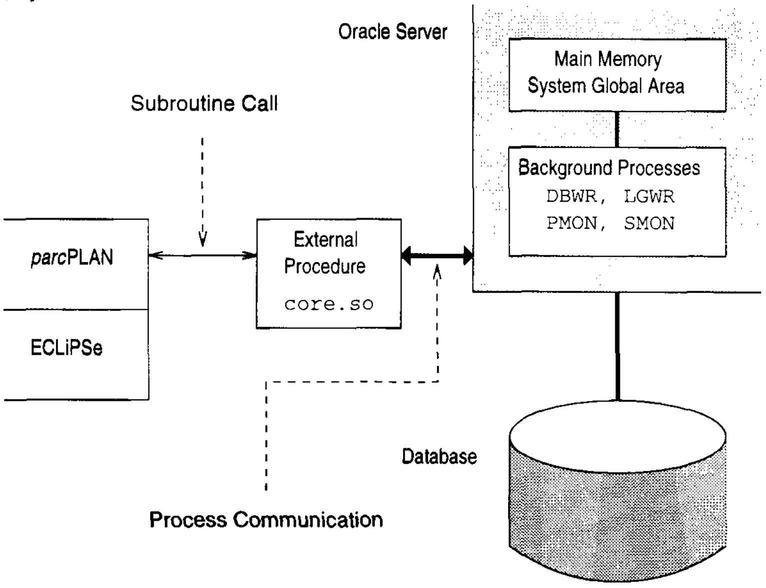


Figure 4 Loose Coupling between ECLiPSe and Oracle.

The first argument represents the SQL query string. The second argument binds to the place holders in the query if it has not already been parsed with the query itself. The last argument will be instantiated in the result. The following is an example of calling from Prolog:

```
set_query(employee(Id, Name, programmer))
```

This Prolog query is translated into an SQL query using a Prolog compiler for translating Prolog database goals into SQL database queries. The implementation of the translation software is described by Draxler [Draxler, 1991]. The Prolog query is translated into SQL, as follows:

```
oracle_set_query("SELECT   id, name
                  FROM     employee
                  WHERE    designation = 'programmer'", [], L).
```

The Prolog predicate **oracle_set_query** corresponds to the external C procedure **p_oracle_set_query** as described above. This correspondence is specified in ECLiPSe as follows:

```
external(oracle_set_query/3, "p_oracle_set_query").
```

The predicate **oracle_set_query** is satisfied only once by instantiating the result with L. The input arguments of **oracle_set_query** are checked for their validity through a set of ECLiPSe predicates using the tags.

Before introducing the detail of the procedure **p_oracle_set_query**, we introduce a C data structure used by this procedure to communicate information between Oracle and C programs.

4.1 Select and bind descriptors

Select-list items are the columns or expressions following the keyword **SELECT** in a query. For example, the following dynamic query contains two select-list items **id** and **name**:

```
SELECT   id, name
FROM     employee
WHERE    designation = 'programmer'
```

Place-holders are dummy bind variables that hold places in an SQL statement for actual bind variables. For example, the following dynamic SQL statement contains one place-holder **:d**:

```
SELECT   id, name
FROM     employee
WHERE    designation = :d
```

All the information Oracle needs about select-list items or place-holders for bind variables, except their values, is stored in a program data structure called the SQL Descriptor Area (SQLDA). The structure provided by Oracle is given in Appendix A.

Descriptions of select-list items are stored in a *select descriptor* and descriptions of place-holders for bind variables are stored in a *bind descriptor*. These two are referenced by pointers as follows:

```
SQLDA* bind_dp;
SQLDA* select_dp;
```

4.2 Set query

The function **p_oracle_set_query** was described at the beginning of section 4. The way it works is that it parses an SQL statement and names it. The function declares a cursor, gives it an identifier and associates it with a **SELECT** statement. The descriptors of place-holders are put into bind descriptors and the active set of the query is evaluated using the specified bind descriptors. If the statement is not a query the processing is complete. The routine **process_set_select_list** processes the selected list and if the statement is not a query this function returns immediately. If the statement is a query, **process_set_select_list** converts the selected list into a Prolog list and returns. The structure of **p_oracle_set_query** and **process_set_select_list** are given in Appendices B.1 & B.2 respectively.

4.3 Tuple query

The procedure **p_oracle_tuple_query** has an extra argument, when compared to **p_oracle_set_query**, representing the cursor identification of the query. This is needed because several tuple queries of this type can be active at any one time and therefore each one needs a unique cursor for identification. The procedure **p_oracle_tuple_query** is different because it opens a cursor every time it is called and closes it before returning to the calling procedure. The structure of **p_oracle_tuple_query** is given in Appendix C.1 and the tuple equivalent of **process_set_select_list, process_tuple_select_list,** is given in Appendix C.2.

5. Reverse engineering

In the context of this discussion, *reverse engineering* is the process of extracting the schema information of the database to be accessed. The schema information is represented through the following two predicates:

```
relation(<predicate>, <arity>, <relation>).
attribute(<position>, <relation>, <name>, <type>).
```

where **<relation>** is the actual Oracle relation name and **<predicate>** is its corresponding Prolog name and **<arity>** is their arity. An argument of a relation is designated by its position, name and type. For example, the *reverse engineering* process provides the following information for an Oracle relation **EMPLOYEE** with attributes **ID, NAME** and **DESIGNATION**:

```
relation(employee, 3, 'EMPLOYEE').
attribute(1, 'EMPLOYEE', 'ID', integer).
attribute(2, 'EMPLOYEE', 'NAME', string).
attribute(3, 'EMPLOYEE', 'DESIGNATION', string).
```

There follows a brief explanation as to how *reverse engineering* is implemented in CORE. Oracle provides a set of data dictionary tables and views to the user. One such table **ALL_TAB_COLUMNS** contains columns of all tables, views and clusters accessible to the user. In particular, it contains the description of each database table accessible to the user. A view called **USER_TAB_COLUMNS** has been created on the relation **ALL_TAB_COLUMNS** by selecting its second, third, fourth and ninth columns for each user in its own area. These columns represent table name, column name, data type and sequence number of the column and correspond, respectively, to relation, attribute, type and position as described above. The attribute information for **USER_TAB_COLUMNS** is itself coded into the ECLiPSe program and tuples under this relation are then retrieved by querying directly.

6. Conclusions

We have written a loosely coupled interface between Oracle and ECLiPSe which we have used successfully to serve data in the TSU application. It serves the data both set-at-a-time and tuple-at-a-time. The TSU planner runs on ECLiPSe with data accessed through a real-time link with an Oracle database. For example, we are able to ask room names as **rmname(X,Y)** and receive the answers tuple-at-a-time using **tuple_query** predicates as follows:

```
[eclipse 5]: tuple_query(rmname(X, Y)).

X = 1
Y = "Room 308 - Computing Lecture Theatre      "     More? (;)

X = 2
Y = "Clore Lecture Theatre                     "     More? (;)

...
yes.
```

The variable **X** is instantiated to the identification of a room and **Y** to its full name. We can also retrieve all the answers as a list of tuples, that is, a list of lists using a **set_query** predicate as follows:

```
[eclipse 6]: set_query(rmname(X, Y),L).

L = [[1, "Room 308 - Computing Lecture Theatre      "], ...]
yes.
```

The time taken by the above query is 0.02 cpu seconds to retrieve 200 tuples. This test was carried out within an ECLiPSe environment on a DRS6000 machine.

The logic programming environment for developing practical applications has been very popular. Our coupling system will help the developers in such environment to deal with interactive read and update operations on legacy databases. The system can also provide a manageable persistent store for Prolog.

7. Future Work

We have designed and implemented a loose-coupling architecture between ECLiPSe and Oracle but there is some more work that we would like to do. One of the major drawbacks in the loose-coupling architecture is that the Prolog goal representing a query needs to be recompiled every time it is called. To illustrate this, consider the following fragment of a Prolog procedure:

```
... :-
    ...
    p(Id)
    ...
    tuple_query(employee(Id, Name, Desig))
    ... .
```

If **p(Id)** succeeds *n* times upon backtracking then the tuple query has to be compiled to its equivalent SQL query string *n* times. This overhead can be reduced by making use of template query strings. There are eight possible forms of goal related to the table **employee**:

employee(X, Y, Z)
employee(a, Y, Z)
employee(X, b, Z)
employee(X, Y, c)
employee(a, b, Z)

employee(a, Y, c)
employee(X, b, c)
employee(a, b, c)

where **a, b** and **c** represent constant values of appropriate types. Now the template query string corresponding to a goal, say **employee(a,Y,c),** has the following form:

```
SELECT name
FROM   employee
WHERE  id = :u AND desig = :v.
```

where **u** and **v** are called place-holding variables. Whenever a call **employee(a,Y,c)** is encountered, for some constants **a** and **c**, these constants will be extracted and the relevant stored template will be retrieved. The call will then be translated to the following call:

```
oracle_tuple_query("SELECT name
                    FROM   employee
                    WHERE  id = :u AND desig = :v", [a,c], [Y]),
```

The only variable **Y** will be instantiated to a new value every time the query succeeds on backtracking. If the template had not been stored then the above form of **oracle_tuple_query** would have been as follows:

```
oracle_tuple_query("SELECT name
                    FROM   employee
                    WHERE  id = a AND desig = b", [], [Y]),
```

by the direct translation of **employee(a,Y,c)**. Within the procedure **p_oracle_tuple_query,** the statement which parses the query will have to be executed only once for each template query string.

The reverse engineering process brings up only the table and view definitions to the Prolog level. The definitions of integrity constraints are not translated in the current implementation as the transformation process from Prolog goals to SQL queries does not make use of the information related to integrity constraints. It would be desirable, however, to convey this information to the planning users to avoid them imposing the same constraints at the Prolog level.

Acknowledgements

The work presented here was carried out in the CHRONOS project. The authors would like to thank their colleagues Ho-Jin Choi, Amin El-

Kholy, Chris Harrison, Vassilios Liatsos, Barry Richards and Mark Wallace for many helpful discussions on this work. The project was supported under the ISIP initiative of EPSRC.

References

BERGHEL, H.L., "Simplified integration of Prolog with RDBMS," DATA BASE, 16, pp 3-12, 1985.

BRODIE, M.L. and JARKE, M., "On integrating logic programming and databases," Expert Database Systems, edited by L. Kerschberg, pages 191--205, 1989.

BOCCA, J. "EDUCE: A marriage of convenience: Prolog and relational DBMS," Proceedings of the Symposium on Logic Programming, pp 36-45, September, 1986.

CERI, S., GOTTLOB, G. and WIEDERHOLD, G., "Interfacing relational databases and Prolog efficiently," Proceedings of the First International Conference on Expert Database Systems, Benjamin-Cummings, 1986.

CODD, E.F., "A relational model for large shared data banks," Communications of the ACM, 13, pp 377-387, 1970.

CHANG, C.L. & WALKER, A., "PROSQL: a prolog programming interface with SQL/DS," Proceedings of the 1st International Conference on Expert Database Systems, pp 233-246, April, 1986.

DAS, S.K., "Deductive Databases and Logic Programming," Addison-Wesley, 1992.

DRAXLER, C. "Accessing Relational and Higher Databases through Database Set Predicates from Logic Programming Languages,"
PhD thesis, University of Zurich, 1991.

ECRC, "ECLiPSe 3.5 User Manual," European Computer-Industry Research Centre, Munich, 1995.

GOZZI, F., LUGLI, M. and CERI, S., "An overview of PRIMO: a portable interface to Prolog and relational databases," Information Systems, 15, 1990.

GARDARIN, G. and VALDURIEZ, P., "Relational Databases and Knowledge Bases," Addison-Wesley, Reading, MA, 1989.

IOANNIDES, Y., CHEN, J., FRIEDMAN, M. & TSANGARIS, T., "BERMUDA - an architectural perspective on interfacing Prolog to a database machine," Proceedings of the Second International Conference on Expert Database Systems, Benjamin-Cummings, 1988.

NAQVI, S. and TSUR, S., "A Logical Language for Data and Knowledge Bases," Computer Science Press, New York, 1989.

ORACLE, "Programmer's Guide to the Oracle Pro*C Precompiler Release 2.0," Oracle Corporation, Redwood City, California, 1993.

ORACLE, "Oracle 7 Server for UNIX (Administrator's Reference Guide) Release 7.1," Oracle Corporation, Redwood City, California,1994.

TSUR, S., "LDL - a technology for the realization of tightly coupled expert database systems," IEEE Expert, 1988.

WAUGH, K.G., WILLIAMS, M.H., KONG, Q., SALVINI, S. and CHEN, G., "Designing SQUIRREL: an extended SQL for a deductive database system," The Computer Journal, 33, 1990.

Appendix A: SQL Description Area

```
/* Global variables for oracle communication areas:
   struct SQLDA {
      long    N; - Descriptor size in number of entries
      char  **V; - Pointer to Array of addresses of main variables
      long   *L; - Ptr to Arr of lengths of buffers
      short  *T; - Ptr to Arr of types of buffers
      short**I;  - Ptr to Arr of addresses of indicator vars
      long    F; - Number of variables found by DESCRIBE
      char  **S; - Ptr to Arr of variable name pointers
      short *M;  - Ptr to Arr of max lengths of var. names
      short *C;  - Ptr to Arr of current lengths of var. names
      char  **X; - Ptr to Arr of ind. var. name pointers
      short  *Y; - Ptr to Arr of max lengths of ind. var. names
      short *Z;  - Ptr to Arr of cur lengths of ind. var. names
}; */
```

Appendix B: Set Query

B.1 Structure of p_oracle_set_query

```
int p_oracle_set_query(sval, stag, fval, ftag, rval, rtag)
     value sval, fval, rval;
     type  stag, ftag, rtag;
{
  /* Declarations */
  ...
  /*  must be used before any Request_Unify or Return_Unify macro
     */
  Prepare_Requests;

  /*  Return with an instantiation fault if argument is a
      variable; otherwise, with a type error if it is not a
      string/list */
  Check_String(stag);
  Check_List(ftag);

  /*  Return with an instantiation fault if argument is not a
      variable */
  Check_Ref(rtag);

  /*  Return with an instantiation fault if argument is not a
      list or a free variable */
  Check_Output_List(rtag);

  /*  Allocate memory for the select and bind descriptors. */
  ...
  ...

  /*  If there is an error then call the error routine. */
  EXEC SQL WHENEVER SQLERROR DO sql_error(0);
  ...
  EXEC SQL PREPARE S FROM :sql_statement[0];
  ...
  EXEC SQL DECLARE C_SET CURSOR FOR S;
  ...
  EXEC SQL WHENEVER SQLERROR DO sql_error(0);
  ...
  EXEC SQL DESCRIBE BIND VARIABLES FOR S INTO bind_dp;

  EXEC SQL OPEN C_SET USING DESCRIPTOR bdp;

  /* Processes the selected */
  result = process_set_select_list();
  ...
  EXEC SQL CLOSE C_SET;

  /* Commit any pending changes and disconnect from Oracle. */
  EXEC SQL COMMIT;
  ...

  /* Return the result list. Return_Unify_List unifies a general
      Prolog term with a list and Return_Unify_Nil unifies with a
      null list. */
  if(IsNil(result->tag))
      Return_Unify_Nil(rval, rtag)
  else
      Return_Unify_List(rval, rtag, result);
}
```

B.2 Structure of process_set_select_list

```
pword* process_set_select_list()
{
  ...

  /*  If the SQL statement is a SELECT, describe the select-list
      items.    The DESCRIBE function returns their names,
      datatypes, lengths (including precision and scale), and
      NULL/NOT NULL statuses. */
  EXEC SQL DESCRIBE SELECT LIST FOR S INTO select_dp;

  ...
  EXEC SQL WHENEVER NOT FOUND GOTO end_select_loop;
  ...
  for (;;)
  {

      EXEC SQL FETCH C_SET USING DESCRIPTOR select_dp;

      /* Convert the fetched tuple to a Prolog list */
      ...
  }
end_select_loop:
  ...
}
```

Appendix C: Tuple Query

C.1 Structure of p_oracle_tuple_query

```
int p_oracle_tuple_query(sval, stag, fval, ftag, rval, rtag,
     cval, ctag)
     value sval, fval, rval, cval;
     type  stag, ftag, rtag, ctag;
{
  /*  Declaration and check the types of arguments */
  ...

  /*  Extract the serial number of the query which is cursor id.
     */
  cursor = cval.nint;

  /*  Call the function to fetch a tuple. */
  result = process_tuple_select_list(fval, sval, cursor);

  /*  If no more tuples to be retrieved. */
  if (end_of_result[cursor] == 0)
  {
     ...

     /* Cuts all following alternatives. */
     Cut_External;

     /* Return from procedure as failure. */
     Fail;
  }
  else {
     if (IsNil(result->tag))
       Return_Unify_Nil(rval, rtag)
     else
       Return_Unify_List(rval, rtag, result);
       }
}
```

C.2 Structure of process_tuple_select_list

```
pword* process_tuple_select_list(fval, sval, cursor)
     value fval, sval;
     int cursor;
{
  /*  Declarations */
  ...
  ...
     switch(cursor) {
        case 1: EXEC SQL PREPARE S1 FROM :sql_statement[1]; break;
        case 2: EXEC SQL PREPARE S2 FROM :sql_statement[2]; break;
        ...
        }
        ...
        switch(cursor) {
        case 1: EXEC SQL DECLARE C1 CURSOR FOR S1; break;
        ...
        }
        ...
        EXEC SQL WHENEVER SQLERROR DO sql_error(cursor);
        ...
        switch(cursor) {
        case 1: EXEC SQL DESCRIBE BIND VARIABLES FOR S1 INTO
        bind_dp; break;
        ...
        }
        ...

        switch(cursor) {
        case 1: EXEC SQL OPEN C1 USING DESCRIPTOR bdp; break;
        ...
        }
        ...
        switch(cursor) {
        case 1: EXEC SQL DESCRIBE SELECT LIST FOR S1 INTO
        select_dp; break;
        ...
        }
    }
    ...
    EXEC SQL WHENEVER NOT FOUND GOTO end_select_loop;

    switch(cursor) {
    case 1: EXEC SQL FETCH C1 USING DESCRIPTOR select_dp; break;
    ...
    }

  end_select_loop:

    /*  If there are no more tuples to be processed. */
    if (end_of_result[cursor] == 0) {

        EXEC SQL WHENEVER SQLERROR CONTINUE;

        switch(cursor) {
        case 1: EXEC SQL CLOSE C1; break;
```

```
      ...
      }

      EXEC SQL COMMIT;

      EXEC SQL WHENEVER SQLERROR DO sql_error(cursor);
  }
  return result;
}
```

Integrating the Object Database System ODB-II with Object Request Brokers

Cormac McKenna

ICL ITCentre, South County Business Park,
Leopardstown, Dublin 18, IRELAND

Abstract

ODB-II is an object oriented database system, developed by Fujitsu and targeted at the multimedia marketplace, which supports an SQL-like query language. This paper looks at two techniques for providing remote clients access to ODB-II object bases using a CORBA-compliant Object Request Broker such as ICL's DAIS or Fujitsu's ObjectDirector.

1. Introduction

ODB-II [Kay & Izumida, 1995] is an object oriented database system that is targeted at the multimedia marketplace. ODB-II has been developed by Fujitsu, and is available on a number of platforms including SunOS, Solaris, DRS/NX/SPARC and HP-UX. In keeping with the object oriented model, ODB-II stores data as instances of classes (in an *object base*), where a class defines the attributes of instances of the class and methods (or operations) which are used to manipulate instances of the class. ODB-II provides an SQL-like retrieval language (ODQL), which can be used to query and update an object base. In their paper, Ishikawa et al [Ishikawa, 1993] state:

> "Our object manipulation language ... has the following features. It supports associative access to objects without explicit joins; it enables methods to be invoked in any part of an associative query... The language integrates individual access to objects for a general purpose programming language and set oriented access to objects for a database... Uniform treatment of systems and user-defined methods allow for user customisation of the language without modifying the language processor."

ODQL may be invoked via an interpreter (ODQLIE), or embedded in C or C++ programs.

Object Request Brokers (ORBs) provide support for distributed computing, in which components of the application are implemented as objects. These objects communicate over the network by invoking each other's methods remotely. The Object Management Group has defined a standard for object request brokers known as CORBA [OMG, 1995], and a number of implementations of this standard are available commercially, including DAIS from ICL [DAIS], and ObjectDirector from Fujitsu. The CORBA standard defines an Interface Definition Language, IDL, in which the interface to any object can be specified; if the IDL interface of an object is known, the methods defined by that object can be invoked by any CORBA client program anywhere on the network.

The purpose of this paper is to consider how to integrate an object database such as ODB-II with an object request broker such as DAIS or ObjectDirector. Potentially such integration can combine the rich facilities of an object database for handling large collections of persistent objects with the rich facilities of an object request broker for distributing those objects transparently around the network.

In principle, there are two ways in which such integration can be achieved and both are useful. The first is to make database objects visible to the request broker as CORBA objects; the second is to make CORBA objects visible to the database as database objects. In this paper we describe how we have tackled the first of these, by giving the objects in an ODB-II database an IDL interface and allowing them to be accessed by any CORBA-compliant client application.

The immediate focus for this work was to investigate whether an ORB can replace the proprietary Remote ODQL facility in ODB-II. At present, PC clients can access an object base using the proprietary RODQL server; the fact that it is a proprietary mechanism and that it supports only PC clients are the principal weaknesses of RODQL. With this in mind we look to ORBs as a mechanism for providing the necessary infrastructure to support remote clients.

This work is being undertaken at the IT*Centre* in partnership with Fujitsu. The current prototype is using ObjectDirector, but the ideas have also been tried with other ORBs including ICL's DAIS and Iona Technology's ORBIX.

2. Objects in ODB-II

An ODB-II class defines the state (or *attributes*) and operations (or *methods*) used to manipulate objects of that class; logically, a class is an abstraction of some real world objects of interest to an application. Class hierarchies may be defined, with all user defined classes inheriting from the system class *Composite*. No attribute or method hiding (such as private or protected members in C++) is provided in the current version. The type of an *attribute* can be either a system supplied class (such as *String, Integer*) or a user provided class (such as *Employee*): the latter case represents a relationship between one object and another.

Objects in ODB-II can be of two types: a *class* object contains attributes and methods which apply to the class as a whole (like static class members in C++), while *instance* objects contain attributes and methods for manipulating instances of the class. The ODB-II class definition identifies both the attributes and methods that apply at instance level, and the attributes and methods used at class level. There may also be methods that apply to sets of instances (or indeed, although not discussed further in this paper, to sets of classes). For example, the following is an ODQL definition for the class of people working in the IT*Centre*:

```
defineClass Person
  super: Composite
{
instance:
  String     name;
  Integer    employeeNumber;
  String     emailAddres          default:"postmaster@itc.icl.ie";
  Real       GetSpendingLimit();

class:
  Integer    nextEmployeeNumber                          default: 0;
  Person     Find(String n);        /* Find a person using name */

set:
  Boolean    CoLocated();          /* Are all the Persons in the*/
};                                        /* set at the one site?*/
```

In this class definition, there are three groups of attributes and methods. *Instance level* attributes are those attributes which every instance contains; each instance of the *Person* class will contain a separate *name* and *employeeNumber*. Similarly, instance-level methods (like *GetSpendingLimit()*) are those methods which can be invoked on individual instances. *Class level* attributes specify metadata pertaining to the class as a whole; similarly class-level methods are invoked on the class as a whole. *Class level* attributes and methods are normally used while creating instances (for example *nextEmployeeNumber* in the above), or to process the extent of the class (like *Find()*). A third group of methods, *set level* methods (such as *CoLocated()*), can be used to manipulate arbitrary collections of instances.

Methods are defined in C or C++ and are compiled into dynamic libraries.

When a class definition has been built and compiled, a class object is created, and an ODQL application may then create and manipulate instances or sets of instances of that class:

```
Person p;                          /* A class instance variable */
Person class pc;                   /* A class variable */
Person set ps;                     /* A set variable */

/*
 * Create a new instance of 'Person' class.
 */
p = Person.new(name:= "Cormac McKenna",
emailAddress:= "cormac@itc.icl.ie");

/*
 * Allocate the next personnel number for 'Person'
 */
pc = Person.getClass();            /*Get class object for'Person'*/
p.employeeNumber  =  pc.nextEmployeeNumber;
pc.nextEmployeeNumber   =  pc.nextEmployeeNumber + 1;

/*
 * Find all instances of 'Person' with grade 'iss18'
 */
ps = P from Person P where P.grade == "iss18";
if (ps.CoLocated())    /*Are all these people in the one spot? */
ps.print();
```

The set of classes contained in an object base is known as a *class family*. Every class and instance object is identified by a unique triple, known as the *object identifier*:

```
typedef struct ODB_OID{
     short cfno;            /* Class family identifier */
     short cno;             /* Class identifier */
     long ino;              /* Class instance identifier */
} ODB_OID;
```

The object identifier remains constant for the life of the object. An application can both directly access an object using its object identifier and extract the object identifier from a given object.

Access to object bases is controlled by an Object Database Adaptor (ODA), which acts as an *ODB-II Server*. An ODA may exclusively control a number of object bases but these have to be located on the same node. An ODB-II application has to connect to exactly one ODA.

Two approaches were examined for this ODA; these are known as the *fine grained object database adaptor* and the ODQL server.

3. A Fine Grained Object Database Adaptor

The concept of an object database adaptor is introduced in an Appendix to the Object Database Standard [ODMG-93, 1994]. As well as pointing out the different roles of an object database and an object request broker, this standard proposes a way in which the two can interwork: namely, that a subset of the objects in the object database should be registered with the ORB. These accessible objects can then be used by ORB client applications to navigate to other objects that are not registered with the ORB directly. Our first attempt to integrate ODB-II with an ORB uses this approach: specifically, the ODB-II objects that are registered are the class objects, and these can be used to navigate to the instance objects by means of an ODQL query.

The fine-grained approach represents each ODB-II class and each ODB-II instance (which are manipulated by client applications) as an ORB object. For each ODB-II class, two ORB interfaces are defined, one for the class-level attributes and methods, the other for the instance-level attributes and methods. Set level methods are not currently supported. This relationship between the various elements of the application is illustrated in Figure 1.

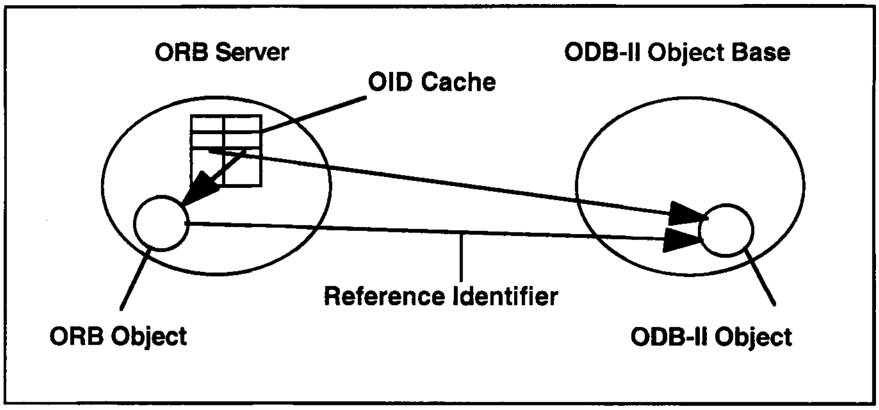


Figure 1 Fine Grained Server Architecture

For a given object base a number of IDL interface definitions are provided, which characterise the class and instance objects provided by the classes in the object base. In the above example, IDL interfaces encapsulating *Person* class and instance objects are defined:

```
interface Person
{
     attribute string          name;
     attribute short           employeeNumber;
     attribute string          emailAddress;
     float                     GetSpendingLimit();
};

interface PersonClass
{
     attribute short           nextEmployeeNumber;

     Person                    Find(in string f) raises
                               (RecordNotFound);
};

interface OdbiiServer
{
     PersonClass               GetPersonClass();
                                             /*Get class object*/
};
```

The interface, *OdbiiServer*, is used to enable clients to generate ORB object references to the class objects which are of interest to them.

Typically, the *OdbiiServer* reference itself is returned to the client via the trader interface.

On the server-side, the functions which implement this interface have to:

- translate the *this* (i.e. the ORB object reference which invokes the method) argument into the corresponding ODB-II object
- convert any arguments into their corresponding ODB-II type or object
- invoke the ODB-II operation (if necessary)
- translate the returned value from the ODB-II operation.

The conversion of simple types (such as *String, Integer,* etc.) is straightforward, and will be ignored here. The translation of incoming ORB references to ODB-II objects is implemented by storing the ODB-II object identifier as reference data in the ORB reference identifier. Translation from ODB-II object identifier to ORB object reference is facilitated by the *OID cache.* This is a table of object identifiers and their associated ORB object reference. A new ORB object reference is created only when the specified ODB-II object identifier is not found in the cache.

This approach allows a client to access and manipulate objects from an object base using their (be it class or instance) methods and attributes. For example:

```
PersonClass         pcobj;
Person              pobj;
ClassName           obj;

pcobj = ClassName_GetPersonClass(obj, &ev);
pobj = PersonClass_Find(pcobj, &ev,"David Miller");
```

Sets of objects can be created and manipulated as unbounded sequences of object references or basic types.

This method exploits the symmetry between IDL and ODQL attributes and methods, and ORB object references and ODB-II object identifiers. Part of the brief for the project is to generate automatically the server-side implementation functions from the ODB-II class definitions.

The main difficulty with this approach is developing the infrastructure required to access objects from the database using *queries,* that is, ODQL statements which select the instances satisfying some boolean predicate. The problem is that queries cannot be passed very naturally to IDL methods. The possibilities are

a) to define simple *canned queries* such as *Find()* in our example, where the form of the query is known statically, or
b) to pass the query condition as a character string at run-time, which has the drawback that the IDL interface is being used merely as a carrier for some higher-level protocol known to both the client and the server.

Obviously while such statements have to be executed by ODB-II, clients need to formulate the statements using ORB object references. For example, suppose *pset* is a set of persons, and a client wants to get all the people called *Dave* from the object. In ODQL this is expressed as:

```
pset = X from Person X where X.name == "Dave";
```

To translate this into C becomes very messy, and ultimately requires supplying functions which specify the ODQL statement to be executed as a string argument:

```
pset = SelectFrom(pcobj, &ev, "X.name == \"Dave\"");
```

The second approach we introduce attempts to solve this problem.

4. An ODQL Server

In our second approach we still make the ODB-II database accessible via ORB interfaces, but we no longer attempt to represent ODB-II objects as ORB objects. Instead we provide a server whose role is to execute ODQL statements. In order to allow the client to receive the results of the ODQL statements, we make the ODQL variables visible as ORB objects, providing functions to read and write them.

The embedded form of ODQL includes an *execute* statement which allows arbitrary ODQL statements to be executed by an application. The server application holds a session on the ODB-II object base and statements are executed in this session by the server on behalf of clients. The embedded ODQL language has a mechanism by which values can be transferred from the ODQL variables back into the C application. A client can set the variables using ODQL statements and can also get the current values of the variables. A single server is supplied implementing the following IDL:

```
interface Var                     /* represents an ODQL variable */
{
     void SetValue(in string odqlCmd)
                                           /* Value set from here */
     raises OdbiiError;
};

interface IntVar: Var
         /* Integer ODQL variable, a subclass of ODQL variable */
{
     readonly attribute long value;
};

/*
 * And so on for the other types ....
 */

interface ObjectVar: Var
/* You can't do very much with Object variables, except use them
 * to execute other methods, etc ...
 */
{
};

interface SetVar: Var
/* Sets are processed using a variable; the interface allows a
 * user initially to set this variable to the first element of
 * the set and move on to the other elements of the set.
 */
{
     void Open(in Var v)          /* Initialises set scan with v* /
          raises OdbiiError;
     void Next(in Var v)             /* Moves v to next element */
          raises OdbiiError;
     boolean Eof()                               /* End of set? */
          raises OdbiiError;
     void Close()
          raises OdbiiError;
};

interface OdbiiServer
{
/* This interface allows clients to create variables;
 * usually supplied via a trader.
 */
     IntVar CreateIntVar()
          raises OdbiiError;
/*
 * And so on for the other types ....
 */

      ObjectVar CreateObjectVar(in string className)
           raises OdbiiError;
};
```

A function:

```
char *Obj2Var(Var v);
```

is supplied which translates the ORB object reference into its associated ODB-II variable name. This enables clients to create ODQL statements constructed from ORB references. For example:

```
OdbiiServer           os;
ObjectVar             o1, o2;
SetVar                s;
StringVar             n;
char                  statement[BUFSIZ],
                      *address;


/*
 * Create object references ......
 */
n  = OdbiiCreateStringVar(os);
o1 = OdbiiCreateObjectVar(os, "Person");
o2 = OdbiiCreateObjectVar(os, "Person");
s  = OdbiiCreateSetVar(os, "Person");

/*
 * Lets find Dave using the Find.  In ODQLIE this
 * would be:
 *    var = Person.Find("Dave")
 */
ObjectVarSetValue(o1, "Person.Find(\"Dave\"");

/*
 * Any other Daves?  In ODQLIE this is:
 *    setVar = X from Person X where X.name = var.name
 */
sprintf(statement, "X from Person X where X.name =
      %s.name",
      Obj2Var(o1));
SetVarSetValue(s, statement);
```

```
SetVarOpen(s, o2);
while (!SetVarEof(s))
{
/*
 * Do something with o2 ....
 */
     sprintf(statement, "%s.address", Obj2Var(o2));
     StringVarSetValue(n, statement);
     SetVarNext(s, o2);
}
SetVarClose(s);
```

The advantages of this mechanism over the fine grained interface are that:

- a single ORB server provides the interface to all object bases
- arbitrary ODQL statements can be executed and their results processed
- the developer need concern him or herself only with ODQL to manipulate the ODB-II object base.

The main disadvantage is that the ODQL language, which is specific to ODB-II, is more visible to the client.

5. Future Directions

In all the situations considered above, a client connects to a single ODB-II server. If a client connects to more than one ODB-II server, the possibility of setting references from one object to others in object bases controlled by different ODB-II servers arises. Consider augmenting the ODB-II *Person* class with an additional instance attribute:

```
Person boss;
```

and a client owning references to person objects *p1* and *p2* from different servers executes the following statements:

```
PersonClass_set_boss(p1, &ev, p2);
```

How is the ODB-II object corresponding to *p1* updated to include the reference to that referred to by *p2*? One approach to handling this situation is to maintain proxy classes in object bases which use the fine grained interface to make invocations on the remote ODB-II objects. This situation is shown in Figure 2 and is an area that we propose to investigate further.

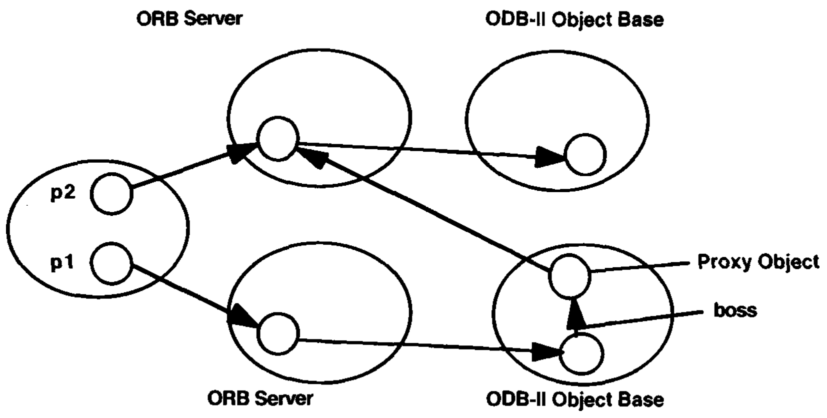


Figure 2 Connecting to Multiple Servers

References

KAY, M.H. & IZUMIDA, Y., "Object Databases and their role in Multimedia Information Systems," Ingenuity, Vol 10, 1, May, 1995.

ISHIKAWA, H. et al., "The Model, Language and Implementation of an Object-Oriented Multimedia Knowledge Base Management System," ACM Transactions on Database Systems, 18, 1, March, 1993.

OMG, "The Common Object Request Broker: Architecture and Specification," Object Management Group, Revision 2.0, July, 1995.

DAIS, "An Introduction to DAIS," ICL Publication 56367.

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

Acknowledgements

The essential material in this paper was first presented at OOIS'95 in Dublin City University, Ireland and was published in *Proceedings of the Industry Day*, ISSN 0791-573X, edited by John Cardiff and John Murphy. In this new paper, many thanks are due to Michael Kay and David Miller for their comments and suggestions.

Biography

Cormac McKenna graduated from University College Dublin with a B.Sc in 1979 and an M.Sc. in 1981. Prior to joining the ICL IT*Centre* in 1986, he worked for Trinity College Dublin, Telecomputing and Prime Computers. Within the *IT*Centre he has worked on various projects including GRAPHICSPOWER and ODB-II.

SAMSON and the Management of SESAME

Barry A Byrne

High Performance Systems, ICL, Winnersh, UK

Abstract

The RACE II project SAMSON (R2058) developed an architecture and example implementations for the management of security services in open networks, where the management station was, in general, situated remotely from the managed resources. The project aimed to demonstrate three key points: one, that security services and policies are manageable using standard management protocols; two, that remote management can be at least as secure as direct management; three, that a consistent interface can be provided to the administrator no matter what service is being managed.

There are currently two candidate management protocols: CMIP from ISO and SNMP from the Internet Engineering Task Force. The user interface is based on X-Windows and Motif.

This paper concentrates on the contributions from ICL, first looking briefly at work done in the early part of the project using the ISO protocol CMIP and then looking in more detail at work done on the remote management of SESAME, in which, to demonstrate the inclusiveness of the SAMSON approach, the internet management protocol SNMPv2 was used (version 2 of SNMP). This paper describes in particular the remote management of SESAME Public Key services. The project has yielded a set of reusable models or descriptions of many types of security service and a general architecture for the construction of management applications.

1. Introduction

Increasingly, organisations are having to face the management of distributed systems that are large, or even global, in scale. Such systems are typically multivendor and heterogeneous (the system contains a number of different operating systems or other security relevant software). These considerations give rise to the following requirements for management: it should be centralised and, for efficiency, it should be based on open systems to support the mixture of architectures and provide a reasonably uniform presentation, or user interface, to reduce the complexity for the administrator. In addition, the potential security vulnerabilities introduced by remote management make it important to provide security for the management operations themselves. So, for example, it is necessary for the system to ensure that only authorized people perform security management and that the management messages passing over the network are not tampered with in any way.

A number of functional areas are recognised as part of system management [ISO7498-4,1989]. These are:

- fault management
- accounting management
- configuration management
- performance management
- security management, which is further defined as:
  - the creation, deletion and control of security services and mechanisms
  - the distribution of security relevant information and the reporting of security relevant events.

The management of security services, security policies and security attributes have received little attention and, in 1992, the European Union under its RACE II programme, agreed to support the investigation, prototype development and demonstration of the management of security and the security of management. The project set up to undertake these tasks was known as SAMSON (Security and Management Services in Open Networks). The following extract from the SAMSON press release shows the project's view of its requirements:

> "In recent years, security has been recognized as a basic issue for distributed systems and networks to prevent loss or unauthorized use of information. In addition to the provision of effective security services, it is also necessary to develop powerful security management facilities. This is especially important as inappropriate settings of security relevant parameters may lead to vulnerabilities and can jeopardise the integrity of the whole network. Therefore it is vital to provide ways of efficiently supporting security managers in performing their tasks."

An early view of SAMSON is given by Zatti [Zatti, 1992] and the importance of SAMSON to the management of telecommunications networks is discussed by Endersz [Endersz, 1992].

2. The SAMSON Consortium

The membership of the SAMSON consortium comprised four companies, Groupe Bull (France), IBM (Zurich and Heidelberg), ICL and Siemens AG (Germany), together with five research institutions, GMD (Germany), Telia (Sweden), Telis (France), CSELT and SIRTI (both Italy). Siemens was the prime contractor and Telis was an associate partner of Groupe Bull. CSELT and SIRTI left the consortium before the end of the project.

The consortium achieved its objectives, managing a wide variety of security services over open protocols (CMIP, and later SNMP), providing a consistent user interface based on X-Windows and Motif.

CMIP (Common Management Information Protocol) is the ISO management protocol and SNMP (Simple Network Management Protocol) is the management protocol defined by the Internet Engineering Task Force.

3. The Management Architecture

Systems Management is based on modelling the resources to be managed according to a well defined description language and manipulating the resources in terms of a small set of operations on these models. In modelling a real world resource, only those properties relevant to management need to be considered: in SAMSON, for example, generally only the security relevant properties were modelled.

In the ISO management scheme, CMIS/CMIP (commonly known simply as CMIP), modelling follows the principles of object-oriented design. The *managed object classes* of the model are classes in the object oriented sense and have the inheritance property. A managed object class can be derived from another and inherit its properties. Managed Object Classes are defined in a language called Guidelines for the Definition of Managed Objects (GDMO) [ISO10165-4,1992], a macro language based on ASN.1 [ISO8824, 1990], which enables a class to be defined in terms of the class it is derived from, its attributes and the operations permitted on those attributes, its behaviour and the notifications that it can send to the managing application.

An object instance typically represents a real world object and is a member of a managed object class. It is typically in a naming relationship with other object instances and, at any one time, its attributes have definite values.

The methods of object oriented design are represented by the operations of the protocol. The primitive operations of CMIP permit the creation and deletion of object instances and the retrieval and modification of sets of attribute values. There are also operations to support arbitrary actions on an object, to cancel a value retrieval and to send a notification.

A naming relationship between object classes based on specified attributes can be defined and this sets up a containment relation among objects. A powerful feature of this protocol is the ability to scope operations to a defined set of instances within a containment tree (such as those at a given level in the hierarchy) and to filter object instances on their attribute values using patterns (for instance, all instances whose names contain a specified substring).

Object classes, attributes and notifications are each given a globally unique identifier, known as a registration number and expressed in the syntax of an object identifier. An example of such a SAMSON object identifier, using the project conventions, is 1.3.9999.1.1.2058.3.3.2, which identifies the *x509userEntry* managed object class, that is used to model authentication in the X.500 directory. In this object identifier, 1 represents ISO; 3, identified_organisation; 9999, the CEC; 1, research programs; 1, RACE; 2058, is the reference number for SAMSON; 3 stands for authentication; the next 3, managed object class and the last digit specifies the particular class, *x509userEntry*. (Because the SAMSON project was working on a prototype only, its object identifiers were not, in fact, registered with any registration authority.)

These features of CMIS/CMIP give rise to three trees, the inheritance tree (of managed object classes), the containment tree (of object instances) and the registration tree (of object identifiers).

The following examples (Figures 1 and 2) show the definitions for access control of X.501 access control, DCE access control lists and POSIX permission bits.

The CMIP protocol itself operates at the application level of the ISO seven layer stack and requires the full stack to be present. The complexity of the object class definitions and this stack requirement leads to heavyweight management systems that may not be suitable for all areas of application (for instance, the remote management of X-terminals or laser printers).

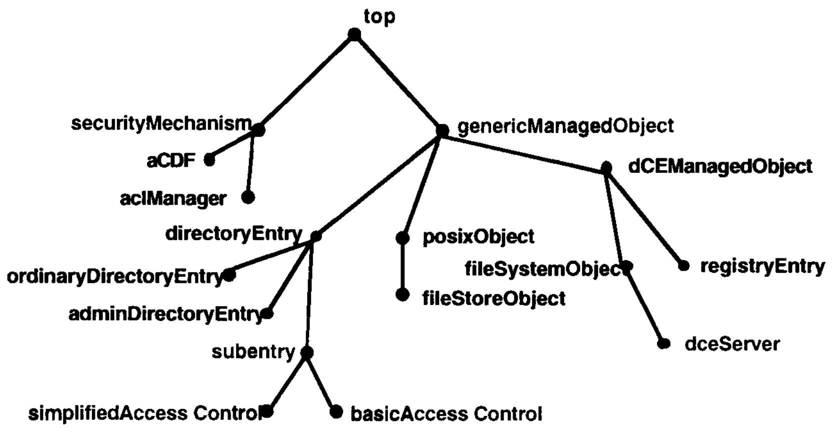


Figure 1 Example Class Inheritance Tree

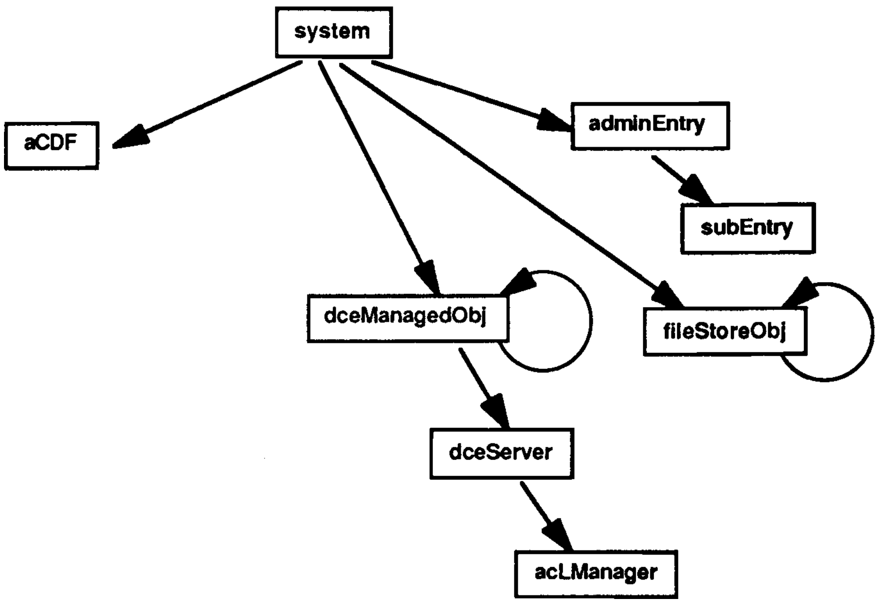


Figure 2 Example schema for containment trees

4. SAMSON ARCHITECTURE

The SAMSON management architecture is illustrated in Figure 3.

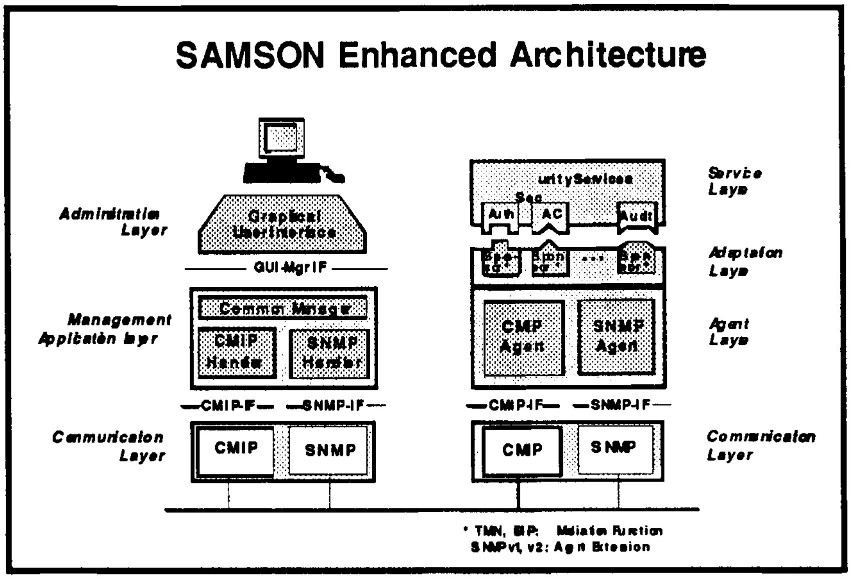


Figure 3 SAMSON Enhanced Architecture

This figure shows the architecture in its developed form in which it encompasses both the ISO and Internet management protocols. The figure shows the conceptual unification of the CMIP (in the left hand columns) and the SNMP (in the right hand columns) in the design of the management and agent applications. (SNMP is discussed in section 8 of this paper.)

Management commands are initiated at the Graphical User Interface (GUI), are processed through the lower layers of the management application and then transmitted over the communications link to the agent, where they travel upward to the adaptation layer and are there converted to native operations on the managed resources. Native operations might be UNIX® system calls, in the case of permission bit management, or SESAME administration commands, in the case of SESAME Public Key management (see section 5 later).

® UNIX is a registered trademark in the US and other countries, licensed exclusively by the X/Open Company Ltd.

The Manager side of the new SAMSON Architecture consists of the Administration Layer, the Management Layer and the Communication Layer.

The Administration Layer is the interface with the human and is realized in the form of a Graphical User Interface, based on X-Windows and Motif in the prototype, but the architecture does not preclude one that is based on a PC.

The Management Application Layer consists of the Common Manager and protocol specific parts. The Common Manager contains general, protocol independent functions of management applications. The ICL team defined a Management of Security Services API (MSS-API) between the GUI and the Common Manager. Each function in this interface is a binding of an abstract description of a management transaction. It is this interface that provides GUI independence to the other components of this model.

The protocol specific parts within the Management Application Layer are concerned with the proper formatting of information for data transfer. Protocol primitives are accessed via protocol specific interfaces. In the case of CMIP, the X/Open XMP [XOPEN, 1994] was adopted by the SAMSON project. Since XMP does not support SNMPv2, the interfaces provided by SNMP implementation had to be adopted directly. However, at ICL (as well as at some other SAMSON partners), the SMLI (SAMSON Management Library Interface) was introduced as an interface with restricted functionality compared with XMP, but capable of driving either CMIP or SNMP through a unified syntax.

The Communication Layer in SAMSON can be realized as CMIP over an OSI stack or SNMP over UDP/IP. For SESAME management, the Carnegie Mellon University (CMU) public domain SNMPv2 library was used.

The Agent Side

The Agent side consists of the Service Layer, the Adaptation Layer, the Agent Layer and the Communication Layer.

The Service Layer contains the different services that SAMSON manages. Examples are X.509 and DCE Authentication, Access Control Services, such as X.501, access control information and DCE access control lists, access control mechanisms and the Key Management services of a variety of architectures, etc. Within the prototype, some of these services could be managed over either CMIP or SNMP. SESAME Public Key Management (PKM) was the only service managed with SNMPv2. (If more time had been available, the same techniques would have been extended to one or more other areas of SESAME management.)

The Adaptation Layer contains the proxy modules between the services and the agents. These modules are called sponsors in ISO (and SAMSON) terminology. The sponsors transform management information, represented in CMIP or SNMP, to security service specific data and operations. In SNMP based management environments, the term "Agent Extension" is used to represent the sponsor concept. The different realisations of this concept lead to the use of different interfaces between agent and sponsor.

The Agent Layer is realized as a protocol specific agent. An agent forwards notifications from the security services to the manager, in the form of either CMIP event reports or SNMP traps, and dispatches management requests to the respective sponsors.

In general, a service would be managed by only one agent. This avoids the potential complexities of concurrent updates etc. It is possible for an agent to serve as a *proxy*, itself issuing management commands to other agents, particularly where a change of protocol occurs. This technique was used in the management of the X.500 directory, where the agent received CMIP protocol and then communicated with the Directory in DSP, the Directory System Protocol.

5. SESAME

In the second stage of the SAMSON project, the partners investigated other approaches to security management. For instance, Directory authentication management, previously implemented over CMIP, was reworked using SNMP. In addition, some approaches to the security of management were prototyped, including the use of X.509 certificates for management authentication. One proposal was that techniques from the sister RACE project SESAME would be used to provide security of management, while work would proceed simultaneously on developing a prototype for the management of SESAME itself. In the event, only the latter proposal was adopted.

Figure 4 shows the general architecture of SESAME Version 2.

SESAME (RACE Project R2051) is based on concepts that can be found in the ECMA documents [ECMA TR/46, 1988] and [ECMA-138, 1989] and provides a single logon capability for heterogeneous distributed systems. As is usual when security is required in networks, integrity protection and confidentiality are based on cryptography. Users are authenticated using a modified Kerberos V mechanism [SCHILLER,1994; McMAHON,1994] and distributed authorisation is mediated through the user's PAC (Privilege Attribute Certificate) and a system of delegation.

SESAME uses both symmetric and asymmetric cryptography. In symmetric cryptography, the same key is used both to encrypt and decrypt messages. Security depends on keeping the key secret. In asymmetric, or public key, cryptography, there are two keys so that a message encrypted with either one of the keys can only be decrypted with the other. The owner of keys keeps one of the pair, known as the private key, but can make the other one (the public key) available. A message sent to the key owner can be encrypted with the public key and then only the owner can decrypt it. A message sent by the owner can have a cryptographic check-value, encrypted with the owner's private key, attached to it and then anyone who has the public key can verify that the message came from that owner. This technique serves at the same time to guarantee the message against undetected modification, since the check-value is almost certain to fail in a modified message.

It is important for a holder of the public key to have a guarantee of its ownership. This is provided by a Certification Authority (CA) which supplies public key certificates, each holding a public key and the identity of its owner. The certificate is signed using the private key of the CA. Thus each public key user has only to obtain the public key of the certification authority by some external, secure method.

Certificates typically have a limited lifetime and SESAME can also support the revocation of certificates. A key that is thought to be compromised can be added to a revocation list. Then when SESAME checks a key, it first searches the revocation list. If the key is listed there, it is regarded as invalid.

More resources are required for the computations required in public key cryptography and the architecture of SESAME makes efficient trade-offs between the two schemes. More detail on cryptography can be found in Press [Press, 1989].

Figure 4 shows three machines: the User Server, where the user is located, the Security Server and the Application Server, where the target application resides.

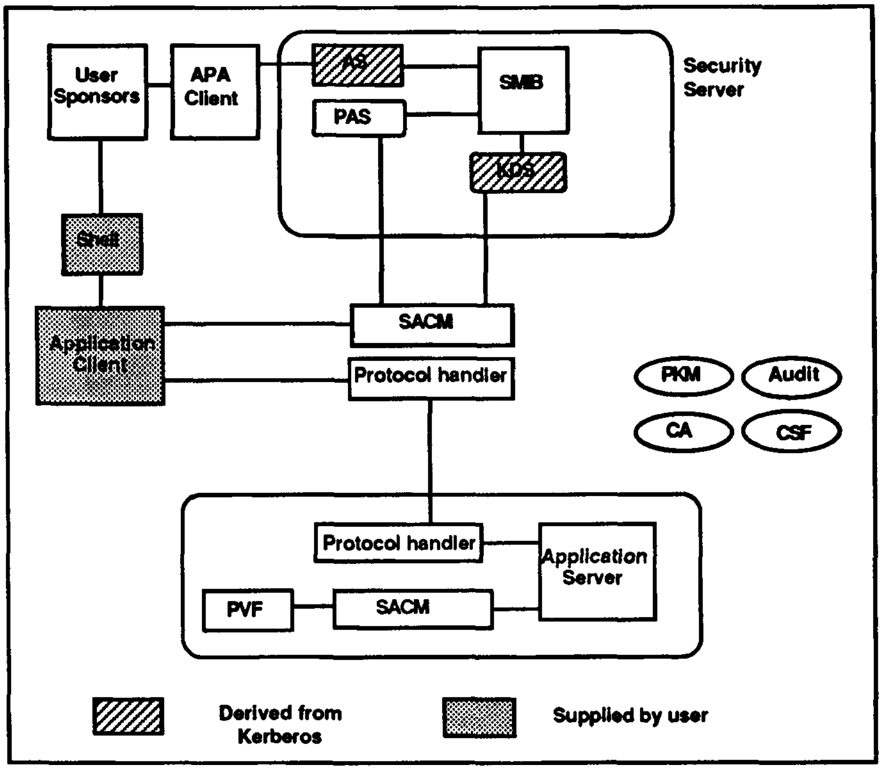


Figure 4 Server Machines

Acronyms used in the figure:

APA	Authentication and Privilege Attribute
AS	Authentication Server
CA	Certification Authority
CSF	Cryptographic Support Facility
KDS	Key Distribution Service
PAC	Privilege Attribute Certificate
PAS	Privilege Attribute Server
PKM	Public Key Management
PVF	PAC Validation Facility
SACM	Secure Association Context Manager
SMIB	Security Management Information Base

In SESAME, users are authenticated to the system by the Authentication Server, using an enhanced form of the Kerberos authentication system [Schiller, 1994; McMahon,1994]. In SESAME, the user interfaces with the user sponsor. In the usual log-on scheme (using a password), the user sponsor obtains a PAS ticket for the user from the authentication server. When the user wishes to use a remote application (application server), the local application client passes the PAS ticket via the SACM to the PAS, which returns a Privilege Attribute Certificate (PAC) for the user and the basic key for communication with the remote application. The PAC contains the security attributes of the user required for authorization (such as the role adopted) and for accountability (to be used by the audit system). The PAC is signed, as protection against undetected modification, using the private key of the PAS.

The data returned from the PAS is transferred by an ordinary protocol, such as TCP/IP, to the application server machine. The application server passes the data received, via its SACM, to the PAC Validation Facility (PVF) which verifies, from the integrity protection, that the PAC is genuine and a secure association is set up over which the user can communicate with the remote application. More finely grained access control to other resources is a matter for application design.

SESAME supports delegation from application server to application server and is able to support interworking between SESAME domains. In SESAME V2, public key technology is used by the PAS and by the Key Distribution Service in support of inter-working between SESAME domains.

Cryptographic operations, such as encrypting and decrypting and the generation and verification of signatures are supported by the CSF. Public keys are administered through the PKM and it is the management of these functions that was demonstrated in SAMSON.

6. SESAME Management

SESAME Version 2 provides tools outside the SESAME framework for the administration of a number of security services including the management of SESAME Public Key services.

PKM resolves into a number of sub-topics: keys, lists and revocations.

Keys:

- Generate asymmetric key pair from specified seed for specified subject
- Request CA certify specified key for specified subject
- Action request
- View key list
- Delete list entry
- Export all entries from list to specified file
- Import all entries from specified file to list

Revocations:

- Start using revocation list
- View revocation list
- Add entry to list
- Delete entry from list
- Generate (binary) revocation list
- Stop using revocation list

The SAMSON project implemented management of these services over the SNMPv2 management protocol. This involved modelling keys, keylists and revocations, extending the SNMP agent to process these objects and building a MOTIF based management application in the SAMSON style.

The project was able to demonstrate the security features available (at that time, see below) in SNMPv2 and to use the SNMPv2 Trap operation to demonstrate an alarm being raised, unsolicited by the manager, at the management station when a management relevant event occurred on the agent system.

7. SESAME GUI

Figure 5 shows an example GUI screen, the one that displays the list of keys to the administrator who can select a key from the list and perform further operations.

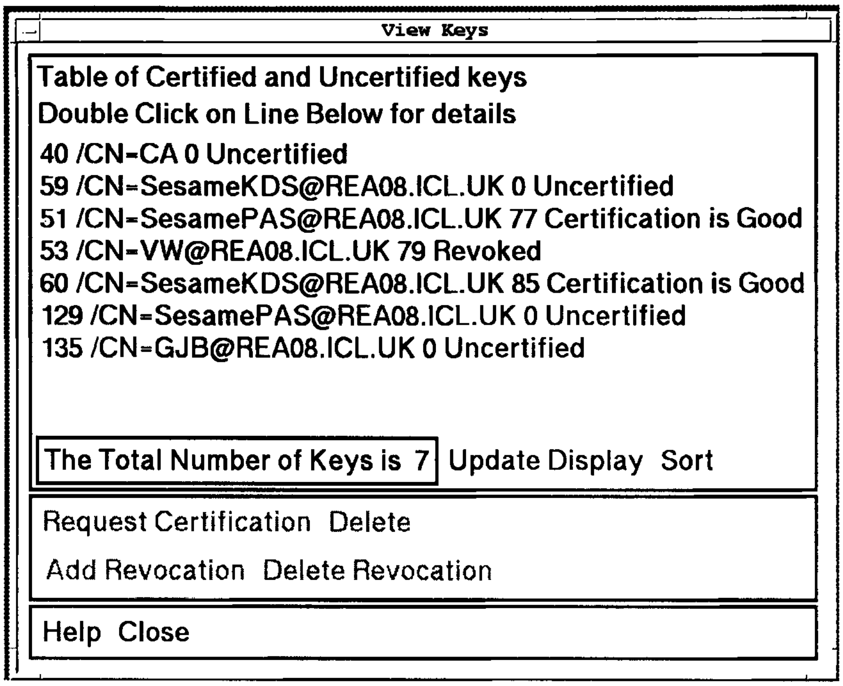

View Keys

Table of Certified and Uncertified keys
Double Click on Line Below for details

```
40 /CN=CA 0 Uncertified
59 /CN=SesameKDS@REA08.ICL.UK 0 Uncertified
51 /CN=SesamePAS@REA08.ICL.UK 77 Certification is Good
53 /CN=VW@REA08.ICL.UK 79 Revoked
60 /CN=SesameKDS@REA08.ICL.UK 85 Certification is Good
129 /CN=SesamePAS@REA08.ICL.UK 0 Uncertified
135 /CN=GJB@REA08.ICL.UK 0 Uncertified
```

The Total Number of Keys is 7

Figure 5 SESAME Graphical User Interface

The list above shows the key pairs of the certification authority itself, the PAS , the KDS and two users VW and GJB. It adds nothing for the certification authority to certify itself and its key pair is left uncertified. (In some systems, there would be a hierarchy of certification authorities, but the root CA would have to be trusted in some way outside the system.)

The certificate on the key pair of user VW has been revoked. From this display, the administrator can perform various operations, such as requesting the CA to certify a specified key pair, or adding or removing a certificate from the revocation list.

8. The SNMPv2 Management Protocol

It would have been possible to have managed SESAME Public Key Services using CMIP, but it was decided instead to extend the capabilities of project SAMSON to reflect the widespread market interest in the alternative management protocol, SNMP.

The Internet Engineering Task Group's Simple Network Management Protocol (SNMP) was designed, as its name suggests, to place the minimum demands on the system on which it operates. It does not even depend on the Transmission Control Protocol of the TCP/IP protocol suite but is designed to run over UDP (the User Datagram Protocol) which itself runs on IP, the Internet Protocol.

SNMP version 2 was introduced as a proposed standard in 1993 to provide extra functionality and security services. The objects of SNMPv2 are defined according to the object-type macro template, also based in ASN.1, but they do not have the inheritance properties of CMIS managed object classes. In SNMP, there is only one tree, the registration tree, which is used to name not only object classes, but effectively names object instances as well.

internet	{ iso org(3) dod(6) 1 }
private	{ internet 4 }
sesameMIB	{ private 2 }
sesameObjects	{ sesameMIB 1 }
sesamePkmObjects	{ sesameObjects 7 }
PkmKeyGenTable	{ sesamePkmObjects 10}
PkmKeyGenStatus	{PkmKeyGenTable 1 1}
PkmKeyGenIndex	{PkmKeyGenTable 1 2}
PkmKeyGenOwner	{PkmKeyGenTable 1 3}
PkmKeyGenSeed	{PkmKeyGenTable 1 4}

Figure 6 Partial SNMPv2 registration tree for SESAME management

SNMPv2 does support tables of managed objects, where the columns represent properties of the object and the rows, referenced by an index, can be used to represent an instance of the object. A cell (a property of an instance) is named by the registration number of the table, the number of the column and the index of the row. The collection of object definitions (together with other administrative definitions) is known as the Management Information Base (MIB).

The SAMSON project developed tools to transform MIB definitions directly into source codedata structures for the various components of the management system.

There are fewer operations in SNMPv2 than in CMIP. These operations support the modification of object values and provide three formats for the retrieval of object values relative to a point in the registration tree (usually a row in a table). There are also operations to send notifications from agent to manager and from manager to manager. There is no support for filtering.

The simplicity of the protocol makes it possible to run it in lightweight applications, such as network bridges and routers.

Despite the shortcomings of SNMPv2, it was found adequate to model the administration of Public Key services in SESAME. The SAMSON team worked from the point of view of thinking informally in terms of managed object classes and representing these as SNMP tables with the columns (apart from one or two specialized for SNMP uses) representing attributes and the rows representing object instances.

9. SNMPv2 Security

An advantage of SNMP version 2 (as of 1995) is that it supports a number of security services, making it possible to satisfy SAMSON requirements for the security of management.

SNMPv2 provides mechanisms for:

- data origin authentication
- message integrity protection
- replay protection
- protection against unauthorised disclosure.

Data origin authentication is the assurance that the message came from the party claiming to have sent it, integrity protection is the assurance that any unauthorised modification of the message can be detected and replay protection is assurance against the threat that a valid message can be delayed or sent again to cause undesirable effects.

Data origin authentication and message integrity are supported by the same SNMPv2 mechanism, namely that a specified portion of SNMPv2 protocol message is concatenated with a secret value known also by the intended recipient. A hash digest is computed over the concatenation and this is sent, together with the message but without the secret, to the recipient. The recipient can re-perform the calculation, using the known secret value and then check the results. MD5 [Rivest,1992] is the preferred digest algorithm.

Replay protection is provided by time-stamping the messages. The sender and the recipient each maintain two clocks with their own and the other party's view of the time and two time-stamps are supplied to the message when data origin authentication is configured.

The preferred algorithm for protection against unauthorized disclosure is the DES [NBS, 1977], with each party knowing the other party's secret key.

The context of a message determines the MIB view, the part of the MIB visible for the operation. In the case of SESAME it would be possible to define different views of the MIB reflecting the modelling of the security and application servers. The operations take place, not strictly between manager and agent, but between parties. The receiving party determines whether authentication is used and the sending party determines whether encryption is used.

Authorization to perform the SNMPv2 operations is based on a triple

```
    (source party,  destination party,  context)
```

and this leads to considerable configuration complexity when many managers and agents are defined.

10. The Subsequent History of SESAME

Version 3 of SESAME appeared in 1995 after the start of SAMSON's SESAME work item. SESAME version 4, in which the dependence on Kerberos has been removed, was released into the public domain in early 1996.

However, the SAMSON work on PKM management remains valid in these new versions.

11. The Subsequent History of SNMPv2

The IETG SNMPv2 working group decided in 1995 to replace the complex security and administrative models of the 1993 SNMPv2 proposals by simpler schemes, but was unable to agree on an alternative. The new proposals capture the improvements made on the basic protocols but re-instate the security features of version 1 of SNMP. This means that there is effectively no security. The IETG is scheduled to re-examine the security area later in 1996.

It is hoped that the new version of SNMP will have security services that can support SAMSON's requirements for security of management and that these will be more closely aligned with the services provided by SESAME itself.

12. SESAME Management Alarms

The SAMSON project was able to use the SNMPv2 trap operation to implement an alarm in the SESAME management system. The agent starts a watchdog process that detects unauthorised activity on the managed system. The watchdog examines activity on the system at regular intervals and transmits the details to the agent when anything questionable is detected. On receiving this data, the agent formats it as a trap message and transmits it to the management application. The manager displays a message at the GUI when the first trap arrives and is able to hold any subsequent traps in a queue, discarding any that arrive that are the same as a trap already in the queue. Each distinct trap received updates the count of pending alarms. The administrator can view the pending alarms and remove them from the queue as they are dealt with.

It is in keeping with the SNMP philosophy that the management application does this filtering, although in the SESAME environment the managed system and the managing systems would almost certainly be on platforms of comparable power.

Note that the SNMPv2 trap is an unconfirmed operation; that is, the agent is not informed that the trap has arrived safely at the management station. This fact limits the usefulness of SNMPv2 for supporting a secure distributed audit capability, since the sender would have no way of knowing whether audit records have been lost.

13. Conclusions

The SESAME consortium has demonstrated that it is possible to manage security services and security policies over a management architecture that can be realized in terms of the ISO/OSI CMIP protocol and the Internet SNMP (and SNMPv2) protocols. This work has unified the management of these different services in at least two ways: it has demonstrated that they can be modelled in common ways, using for instance the inheritance properties of GDMO, and it has unified the presentation of management information in the SAMSON GUI.

The SAMSON project has also shown that the administration of SESAME security services can be unified, centralized and simplified within the SAMSON management framework. Immediate benefits include the definition of the SESAME MIB, the macros that model the SAMSON management operations on the PKM and the tools that derive C code directly from these definitions. The project has also defined an API to represent the management operations so that an alternative GUI, such as Microsoft Windows, can be used and an API that hides, to a significant extent, the differences between the two management protocols.

Acknowledgements

The author acknowledges the work of the SAMSON consortium in systematizing the concepts presented in this paper. For the management of SESAME, the author expresses his thanks to Vincent West, who worked principally on the SNMP modelling, Terry O'Brien, who worked mainly on the management protocols and to Gerald Barnes, who worked mainly on the user levels of the management application.

References

ECMA TR/46, "Security on Open Systems; A Security Framework," July, 1988

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

ENDERSZ, G., "Information Security in a Telecom Perspective," Proc. of Telecom Europe92, Budapest, October, 1992

ISO7498-4, "Information Technology Open Systems Interconnection Basic Reference Model Part 4 Management Framework," 1989

ISO8824, "Information Technology Open Systems Interconnection Specification of Abstract Syntax Notation One," (ASN.1)1990

ISO10165-4, "Information Technology Open Systems Interconnection Management Information Part 4; Guidelines for the Definition of Managed Objects ," 1992

McMAHON, P., "SESAME V2 Public Key and Authorization Extensions to Kerberos," ISOC Symposium, 1994

NBS, "Data Encryption Standard," FIPS PUB 46, National Bureau of Standards, 1977

PARKER, T.A. & PINKAS, D., "SESAME V2 - Overview," Ingenuity, 1, July, 1994

PRESS, J., "An introduction to public key systems and digital signatures," ICL Technical Journal, June, 1989

RIVEST, R., "The MD5 Message Digest Algorithm," rfc1321, 1992

SCHILLER, J.I., "Secure Distributed Computing," Scientific American, Nov., 1994

STALLINGS, W., "SNMP, SNMPv2 and CMIP: the practical guide to network management standards," Addison Wesley, 1993

XOPEN, "X/Open CAE Specification: Systems Management; Management Protocols," API (XMP), March, 1994

ZATTI,S., "Security Management in Distributed Systems," OSITOP Conference, Paris, September, 1992

Further details of SAMSON may be found at the Web site:
http://saturn.darmstadt.gmd.de/TKT/security/samson/samson.html
and of SESAME at:
http://www.esat.kuleuven.ac.be/cosic/sesame.html

Biography

Barry Byrne has an Honours degree in Mathematics from the University of Durham and is a Chartered Engineer (CEng) and a Member of the BCS.

He has worked at ICL since 1968, specialising in language systems and later in security. He was SAMSON work item leader in access control and SESAME management. He is currently component controller for SESAME exploitation within the Access Manager Engineering group of HPS.

Appendix

Example SESAME MIB Macros

```
sesamePkmKeyGenTable          OBJECT-TYPE
          SYNTAX          SEQUENCE OF SesamePkmKeyGenEntry
          MAX-ACCESS      not-accessible
          STATUS          current
          DESCRIPTION
          "The table of key pair generations. Rows may  be  created  but
          not modified or deleted.  Viewing is not supported."
                          ::= { sesamePkmObjects 10 }

sesamePkmKeyGenEntry          OBJECT-TYPE
          SYNTAX          SesamePkmKeyGenEntry
          MAX-ACCESS      not-accessible
          STATUS          current
          DESCRIPTION
                   "An entry (conceptual row) in the key generation table"
          INDEX           { sesamePkmKeyGenIndex }
          ::= { sesamePkmKeyGenTable 1 }
SesamePkmKeyGenEntry ::=
          SEQUENCE {
              sesamePkmKeyGenStatus       RowStatus,
              sesamePkmKeyGenIndex        INTEGER,
              sesamePkmKeyGenOwner        SesameX500Name,
              sesamePkmKeyGenSeed         DisplayString
              }
sesamePkmKeyGenStatus         OBJECT-TYPE
          SYNTAX          RowStatus
          MAX-ACCESS      read-create
          STATUS          current
          DESCRIPTION
                   "Used for creating rows"
          ::= { sesamePkmKeyGenEntry 1 }
sesamePkmKeyGenIndex          OBJECT-TYPE
          SYNTAX          INTEGER (0..2147483647)
          MAX-ACCESS      not-accessible --  as
          INDEX
          STATUS          current
          DESCRIPTION
                   "The table index (a random number)"
          ::= { sesamePkmKeyGenEntry 2 }
```

```
sesamePkmKeyGenOwner          OBJECT-TYPE
        SYNTAX               SesameX500Name
        MAX-ACCESS           read-create
        STATUS               current
        DESCRIPTION
                "The X.500 name of the key owner"
        ::= { sesamePkmKeyGenEntry 3 }

sesamePkmKeyGenSeed           OBJECT-TYPE
        SYNTAX               DisplayString (SIZE (0..64))
        MAX-ACCESS           read-create
        STATUS               current
        DESCRIPTION
                "The seed used for SESAME random number generation.
                The characters permitted by the agent are
                alphanumerics and underscores. The length is limited
                to 64 characters to reduce the size of PDUs"
        ::= { sesamePkmKeyGenEntry 4 }
```

Book Review

"User Driven Innovation - The World's first Business Computer," by David Caminer, John Aris, Peter Hermon, Frank Land and others. McGraw Hill London and New York, 1996 ISBN 0-07-709236-8, pp 401, hardback, retail price £35.00.

Most readers of this journal will have heard the story of LEO, how J Lyons and Company came to build the first computer designed for, and put into service to do routine, business jobs. Before this important work, few will have had any detailed knowledge about how those jobs were planned and programmed, since, aside from papers on individual applications, little had been published about the programming procedures that had to be devised from the ground up. David Caminer and his twelve co-authors, many of whom were actively involved as team leaders reporting to him, have now magnificently filled the gap. Though each has written about the contribution he made in his own style, they all stressed the care with which plans were debated and coding minutely checked as a team effort. Telling the story of how the foundations of data processing were laid from 1949 onwards has evidently been a labour of love.

About the first third of the book by Caminer is a readable chronicle of the whole period from 1947, when T R Thompson and O W Standingford of Lyons office management visited the USA, and heard from Professor Goldstein at Harvard about plans for stored progam machines. The story of how the project came about is more fully told than previously; it is remarkable not only for the vision of Lyons Office Management but the backing the project got from the Board. The story continues up to 1966/68 when the LEO series, that had by then included LEO Marks I, II and III, was superseded by English Electric System 4. There follow, in part 2, four chapters under the general heading, "Pioneers," by Leo Fantl, Frank Land, John Gosden and Peter Hermon describing how early applications for both business and other purposes (such as weather forecasting, actuarial work and guided missiles) were successfully planned and tested; not all the earliest jobs were for Lyons.

The supreme intellectual effort needed to fit sophisticated applications into the tiny store of LEO I (it held 2048 instructions or the binary equivalents of 2048 5-decimal or 1024 10-decimal numbers) is well brought out. The lack of both an operating system for LEO I and II and the software tools, taken for granted today, was not consciously felt; these facilities, of course, had not been conceived at that stage. John Gosden's chapter, however, looks forward to their provision with LEO III.

In part 3 are accounts by those actively involved of six widely differing and original applications mounted on LEO I, II or III. They include the pioneering job on LEO I for Lyons Teashops, which had a two hour turn round between the telephoned order and passing aggregated totals to bakery or kitchen along with van loading and delivery schedules to the despatch. The telephone billing application for the GPO may have been the first business application of time-sharing, for which LEO III incorporated such special features as store protection and interrupts.

Part 4 includes an evaluation of the LEO approach to system design by John Aris, emphasising the many respects in which LEO thinking was well ahead of its time and foreshadowed present day practices. The invaluable experience gained by all the authors of the book is reflected in their successful later careers outside LEO.

There is a foreword by Professor Richard Nolan of Harvard University and a preface by David Caminer. Though this reviewer was personally involved with the engineering side of the project, he considers the title of the book to be very appropriate; the main impetus for hardware innovation really did come from the software side, who were then the users. The book is very well indexed; there are copious references to contemporary publications and other documents which survive with 3 Appendices, including substantial extracts from the report by T R Thompson and O W Standingford on their 1947 US visit. Each author writes in his own style and, occasionally, is not afraid to criticise his colleagues, amongst them Caminer himself. It is to his credit that these remarks are allowed to stand; they make the book more credible and intriguing. It is usual to talk about the increasing pressures under which people work today but they can hardly be greater than they were for the developers of applications for the LEO series in the fifties and sixties.

This is a work of scholarship but eminently readable nevertheless. It will be seen as a major contribution to the history of business computing; it is strongly recommended for anyone already working in or studying to enter the field of IT.

J.M.M. Pinkerton

Previous Issues

Vol.10 Iss.2 - November 1995

The Architecture of the ICL GOLDRUSH MegaSERVER
The Hardware Architecture of the ICL GOLDRUSH MegaSERVER
CAL in Higher Education - Potential and Pitfalls
The UK Technology Foresight Programme
Making the Internet Safe for Business
Developing Financial Services Kiosks
High Availability Manager
The Virgin Global Challenger
Design of the Format for EDI Messages Using Object-Oriented Techniques
New Aspects of Research on Displays

Vol.10 Iss.1 - May 1995

Object databases and their role in multimedia information systems
The ICL Multimedia Desktop Programme
Multimedia Information used in Learning Organisations
The Software Paradigm
Single Sign-on Systems
Why is it difficult producing safety-critical software?
Experiences using the Ingres Search Accelerator for a Large Property Management Database System
RAID
Improving Configuration Management for Complex Open Systems

Vol.9 Iss.2 - November 1994

Establishing Co-operation in Federated Systems
An ANSA Analysis of Open Dependable Distributed Computing
An Open Architecture for Real-Time Processing
Updating the Secure Office System
POSIX Security Framework
SQL Gateways for Client-Server Systems
Asynchronous transfer mode - ATM
The ICL search accelerator™, SCAFS™: functionality and benefits
Open Teleservice - A Framework for Service in the 90s
LEO, A personal memoire

Vol.9 Iss.1 - May 1994

Client-server architecture
How ICL Corporate Systems support Client-server: an Architectural Overview
Exploiting Client-server Computing to meet the needs of Retail Banking Organisations
A practical example of Client-server Integration
From a Frog to a Handsome Prince: Enhancing existing character based mainframe

applications
Legacy systems in client-server networks: A gateway employing scripted terminal emulation
The Management of Client-server Systems
Dialogue Manager: Integrating disparate services in client-server environments
Distributed Printing in a Heterogeneous World
Systems Management: an example of a successful Client-server Architecture
PARIS - ICL's Problem & Resolution Information System

Vol.8 Iss.4 - November 1993

Toward the 4th Generation Office: A Study in Office Systems Evolution
IPCS - Integrated Product Configuring Service
CGS - The ICL Configurer Graphics Service
Location Transparency in Heterogeneous Networks
Future Office Interconnection Architectures for LAN and Wide Area Access
Parallel Lisp and the Text Translation System METAL on the European Declarative System
Detecting Latent Sector Faults in SCSI Disks

Vol.8 Iss.3 - May 1993

An Introduction to OPEN*framework*
The Evolution of the OPEN*framework* Systems Architecture
Creating Potential-for-Change
OPEN*framework* in Action at DEVETIR
Strategic Information Systems planning: A Process to Integrate IT and Business Systems
Describing Systems in the OPEN*framework* Integration Knowledge Base
Multimedia and Standards for Open Information
VME-X: Making VME Open
A New Approach to Cryptographic Facility Design
CHISLE: An Engineer's Tool for Hardware System Design
Distributed Detection of Deadlock

Vol.8 Iss.2 - November 1992

Open Networks - The Key to Global Success
Infrastructure of Corporate Networks in the Nineties
Broadband Networking
FDDI - The High Speed Network of the Nineties
The Evolution of Wireless Networks
Communications Technology for the Retail Environment
RIBA - A Support Environment for Distributed Processing
Information Technology: Support for Law Enforcement Investigations and Intelligence
Standard for Keyboard Layouts - The Origins and Scope of ISO/TEC 9995
ESS - A Solid State Disc System for ICL System for ICL Series 39 Mainframes

Vol.8 Iss.1 - May 1992

Defining CASE Requirements
ICL's ICASE Products
The Engineering Database

CASE Data Integration: The Emerging International Standards
Building Maintainable Knowledge Based Systems
The Architecture of an Open Dictionary
The Use of a Persistent Language in the Implementation of a Process Support System
ALF: A Third Generation Environment for Systems Engineering
MASP/DL: The ALF Language for Process Modelling
The ALF User Interface Management System
A New Notation for Dataflow Specifications

Vol.7 Iss.4 - November 1991

Systems Management: A Challenge for the Nineties - Why now?
The Evolution within ICL of an Architecture for Systems Management
Manageability of a Distributed System
Distribution Management - ICL's Open Approach
Experience of Managing Data Flows in Distributed Computing in Retail Businesses
Generation of Configurations - a Collaborative Venture
Operations Management
OSMC: The Operations Control Manager
The Network Management Domain
An Overview of the Raleigh Object-Oriented Database System
Making a Secure Office System
Architectures of Knowledge Base Machines
The Origins of PERICLES - A common on-line Interface

Vol.7 Iss.3 - May 1991

Introduction to the technical characteristics of ISDN
ISDN in France: Numéris and its market
The Telecoms Scene in Spain
Future Applications of ISDN to Information Technology
A Geographical Information System for Managing the Assets of a Water Company
Using Constraint Logic Programming Techniques in Container Port Planning
Locator - An Application of Knowledge Engineering to ICL's Customer Service
Designing the HCI for a Graphical Knowledge Tree Editor: A Case Study in User-Centred Design
X/OPEN - From Strength to Strength
Architectures of Database Machines
Computer Simulation for the Efficient Development of Silicon Technologies
The use of Ward and Mellor Structured Methodology for the Design of a Complex Real Time System

Vol.7 Iss.2 - November 1990

The SX Node Architecture
SX Design Process
Physical Design Concepts of the SX Mainframe
The Development of Marketing to Design: The Incorporation of Human Factors into Specification and Design
Advances in the Processing and Management of Multimedia Information

An Overview of Multiworks
RICHE-Rèseau d'Information et de Communication Hospitalier Europèen (Healthcare Information and Communication Network for Europe)
E.S.F - A European Programme for Evolutionary Introduction of Software Factories
A Spreadsheet with Visible Logic
Intelligent Help - The Results of the EUROHELP Project
How to use Colour in Displays - Coding, Cognition and Comprehension
Eye Movements for A Bidirectional Human Interface
Government IT Infrastructure for the Nineties (GIN): An Introduction to the Programme

Vol.7 Iss.1 - May 1990

Architecture of the DRS6000 (UNICORN) Hardware
DRS6000 (UNICORN) software: an overview
Electromechanical Design of DRS6000 (UNICORN)
The User-System Interface - a challenge for application users and application developers?
The emergence of the separable user interface
SMIS - A Knowledge-Based Interface to Marketing Data
A Conversational Interface to a Constraint-Satisfaction System
SODA: The ICL interface for ODA document access
Human - Human co-operation and the design of co-operative mechanisms
Regulatory Requirements for Security - User Access Control
Standards for secure interfaces to distributed applications
How to Use Colour in Displays - 1. Physiology Physics & Perception

Vol.6 Iss.4 - November 1989

Time to Market in new product development
Time to Market in manufacturing
The VME High Security Option
Security aspects of the fundamental association model
An introduction to public key systems and digital signatures
Security classes and access rights in a distributed system
Building a marketeer's workbench: an expert system applied to the marketing planning process
The Knowledge Crunching Machine at ECRC: a joint R&D project of a high speed Prolog system
Aspects of protection on the Flagship machine: binding, context and environment
ICL Company Research and Development Part 3: The New Range and other developments

Vol.6 Iss.3 - May 1989

Tools, Methods and Theories: a personal view of progress towards Systems Engineering
Systems Integration
An architectural framework for systems
Twenty Years with Support Environments
An Introduction to the IPSE 2.5 Project
The case for CASE
The UK Inland Revenue operational systems
La solution ICL chez Carrefour a Orleans

A Formally-Specified In-Store System for the Retail Sector towards a Geographic Information System
...towards a Geographic Information System
Ingres Physical Design Adviser: a prototype system for advising on the physical design of an Ingres relational database
KANT - a Knowledge Analysis Tool
Pure Logic Language
The 'Design to Product' Alvey Demonstrator

Vol.6 Iss.2 - November 1988

Flexible Manufacturing at ICL's Ashton plant
Knowledge based systems in computer based manufacturing
Open systems architecture for CIM
MAES - An expert system applied to the planning of material supply in computer manufacturing
JIT and IT
Computer Aided Process Planning (CAPP): Experience at Dowty Fuel Systems
Use of integrated electronic mail within databases to control processes
Value engineering - a tool for product cost reduction
ASP: Artwork specifications in Prolog
Elastomer technology for probing high-density printed circuit boards
The effects of back-driving surface mounted digital integrated circuits
Reliability of surface-mounted component soldered joints produced by vapour phase, infrared soldering techniques
Materials evaluation
On the human side of technology

Vol.6 Iss.1 - May 1988

ICL Series 39 support process
The ICL systems support centre organisation
ICL Services Product Centre
Knowledge engineering as an aid to the system service desks
Logic analysers for system problem solving
Repair - past and future
OSI migration
A Network to Support Application Software Development
Universal Communications Cabling: A Building Utility
Collecting and generalising knowledge descriptions from task analysis data
The architecture of an automated Quality Management System
ICL Company Research and Development Part 2: Mergers and Mainframes, 1959-1968

Vol.5 Iss.4 - November 1987

Open Distributed Processing
The Advanced Network Systems Architecture project
Community management for the ICL networked production line
The X/OPEN Group and the Common Applications Environment
Security in distributed information systems: needs, problems and solutions

Cryptographic file storage
Standards and office information
Introducing ODA
The Technical and Office Protocols - TOP
X400 - international information distribution
A general purpose natural language interface: design and application as a database front end
DAP-Ada: Ada facilities for SIMD architectures
Quick language implementation

Vol.5 Iss.3 - May 1987

What is Fifth Generation? - the scope of the ICL programme
The Alvey DHSS Large Demonstrator Project
PARAMEDICL: a computer-aided medical diagnosis system for parallel architectures
S39XC - a configurer for Series 39 mainframe systems
The application of knowledge-based systems to computer capacity management
On knowledge bases at ECRC
Logic languages and relational databases: the design and implementation of Educe
The semantic aspects of MMI
Language overview
PISA - a Persistent Information Space Architecture
Software development using functional programming languages
Dactl: a computational model and compiler target language based on graph reduction
Designing system software for parallel declarative systems
Flagship computational models and machine architecture
Flagship hardware and implementation
GRIP: a parallel graph-reduction machine

Vol.5 Iss.2 - November 1986

The Management into the 1990s Research Programme
Managing strategic ideas: the role of the computer
A study of interactive computing at top management levels
A management support environment
Managing change and gaining corporate commitment
An approach to information technology planning
Preparing and organising for IPSE
Global Language for Distributed Data Integration
The design of distributed secure logical machines
Mathematical logic in the large practical world
The ICL DRS300 management graphics system
Performance of OSLAN local area network
Experience with programming parallel signal-processing algorithms in Fortran 8X

Vol.5 Iss.1 - May 1986

ICL company research and development, 1904-1959
Innovation in computational architecture and design
REMIT: a natural language paraphraser for relational query expressions

Natural language database enquiry
The *me too* method of software design
Formal specification - a simple example
The effects of inspections on software quality and productivity
Recent developments in image data compression for digital facsimile
Message structure as a determinant of message processing system structure

Vol.4 Iss.4 - November 1985

History of the ICL content-addressable file store, (CAFS)
History of the CAFS relational software
The CAFS system today and tomorrow
Development of the CAFS-ISP controller product for Series 29 and 39 systems
CAFS-ISP: issues for the applications designer
Using secondary indexes for large CAFS databases
Creating an end-user CAFS service
Textmaster - a document retrieval system using CAFS-ISP
CAFS and text: the view from academia
Secrets of the sky: the IRAS data at Queen Mary College
CAFS file-correlation unit

Vol.4 Iss.3 - May 1985

Overview of the ICL Series 39 Level 30 system
VME nodal architecture: a model for the realisation of a distributed system concept
Processing node of the ICL Series 39 Level 30 system
Input/output controller and local area networks of the ICL Series 39 Level 30 system
The store of the ICL Series 39 Level 30 system
The high-speed peripheral controller for the Series 39 system
Development of 8000-gate CMOS gate arrays for the ICL Level 30 system
Development route for the C8K 8000-gate CMOS array
Design automation tools used in the development of the ICL Series 39 Level 30 system
Design and manufacture of the cabinet for the ICL Series 39 Level 30 system
Manufacturing the level 30 system I Mercury: an advanced production line
Manufacturing the Level 30 system II Merlin: an advanced printed circuit board manufacturing system
Manufacturing the Level 30 system III The test system

Vol.4 Iss.2 - November 1984

Modelling a multi-processor designed for telecommunication systems control
Tracking of LSI chips and printed circuit boards using the ICL Distributed Array Processor
Sorting on DAP
User functions for the generation and distribution of encipherment keys
Analysis of software failure data(1): adaptation of the Littlewood stochastic reliability

growth model for coarse data
Towards a formal specification of the ICL Data Dictionary

Vol.4 Iss.1 - May 1984

The ICL University Research Council
The Atlas 10 computer
Towards better specifications
Solution of the global element equations on the ICL DAP
Quality model of system design and integration
Software cost models
Program history records: a system of software data collection and analysis

Vol.3 Iss.4 - November 1983

Expert system in heavy industry: an application of ICLX in a British Steel Corporation works
Dragon: the development of an expert sizing system
The logic language PROLOG-M in database technology and intelligent knowledge-based systems
QPROC: a natural language database enquiry system implemented in PROLOG
Modelling software support

Vol.3 Iss.3 - May 1983

IPA networking architecture
IPA data interchange and networking facilities
The IPA telecommunications function
IPA community management
MACROLAN: a high-performance network
Specification in CSP language of the ECMA-72 Class 4 transport protocol
Evolution of switched telecommunication networks
DAP in action

Vol.3 Iss.2 - November 1982

The advance of Information Technology
Computing for the needs of development in the smallholder sector
The PERQ workstation and the distributed computing environment
Some techniques for handling encipherment keys
The use of COBOL for scientific data processing
Recognition of hand-written characters using the DAP
Hardware design faults: a classification and some measurements

Vol.3 Iss.1 - May 1982

Software of the ICL System 25
Security in a large general-purpose operating system: ICL's approach in VME/2900
Systems evolution dynamics of VME/B
Software aspects of the Exeter Community Health Services Computer Project
Associative data management system
Evaluating manufacturing testing strategies

Vol.2 Iss.4 - November 1981

Architecture of the ICL System 25
Designing for the X25 telecommunications standard
Viewdata and the ICL Bulletin System
Development philosophy and fundamental processing concepts of the ICL Rapid Application Development System RADS
A moving-mesh plasma equilibrium problem on the ICL Distributed Array Processor

Vol.2 Iss.3 - May 1981

A dynamic database for econometric modelling
Personnel on CAFS: a case study
Giving the computer a voice
Data integrity and the implications for back-up
Applications of the ICL Distributed Array Processor to econometric computations
A high-level logic design system
Measures of programming complexity

Vol.2 Iss.2 - November 1980

The ICL Information Processing Architecture, IPA
VME/B: a model for the realisation of a total system concept
Birds, Bs and CRTs
Solution of elliptic partial differential equations on the ICL Distributed Array Processor
Data routing and transpositions in processor arrays
A Bayesian approach to test modelling

Vol.2 Iss.1 - May 1980

Security and privacy of data held in computers
CADES - software engineering in practice
ME29 Initial Program Load: an exercise in defensive programming
Project Little - an experimental ultra-reliable system
Flow of instructions through a pipelined processor
Towards an 'expert' diagnostic system
Using Open System Interconnection standards

Vol.1 Iss.3 - November 1979

Meteosat 1: Europe's first meteorological satellite
An analysis of checkpointing
Statistical and related systems
Structured programming techniques in interrupt-driven routines
The content addressable file store - CAFS
Computing in the humanities
The data dictionary system in analysis and design

Vol.1 Iss.2 - May 1979

Computers in support of agriculture in developing countries
Software and algorithms for the Distributed Array Processor

Hardware monitoring on the 2900 range
Network models of system performance
Advanced technology in printing: the laser printer
The new frontier: three essays on job control

Vol.1 Iss.1 - November 1978

The origins of the 2900 series
Sizing computer systems and workloads
Wind of Change
Standards for open-network operation
Distributed computing in business data processing
A general model for integrity control

To order back issues

Contact

Sheila Cox

Research and Advanced Technology,

ICL, Lovelace Road, Bracknell, Berks., RG12 8SN

Telephone +44 (0)1344 472900

Fax +44 (0)1344 472700

Email: S.D.Cox.bra0114@oasis.icl.co.uk

or

The Editor, V.A.J. Maller

Telephone +44 (0)1438 833514

Email: V.A.J.Maller@ste0418.wins.icl.co.uk

ICL Systems Journal

Guidance for Authors

1. Content

The ICL Systems 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.

The Journal 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 1_/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 Journal's 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 the ICL Systems Journal 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.

ICL Systems Journal May 1996