CONTENTS

SPECIAL ISSUE ON RECENT ADVANCES IN COMPUTER AUDITING

42-47 Trends in the Use of EDP Audit Techniques
    R.C. REEVE

48-53 Modelling as an Audit Technique
    B.J. GARNER and J. PINNIS

54-59 An Interactive Package for Teaching EDP Audit
    W.N. HOLMES and P.M. STANLEY

60-62 The EDP Auditors Association in Australia
    G.S. BREYDON

TECHNICAL PAPER

63-70 Experiments with a Britton-Lee Data Base Machine on CSIRONET
    H.G. MacKENZIE, D.M. RYAN and J.L. SMITH

SPECIAL FEATURES

41 Guest Editorial
62 Call for Papers for Next Special Issue on Expert Systems
71-72 Letter to Editor and Reply
73-78 Book Reviews
79-80 Information for Authors
TANDEM COMPUTERS' powerful new NonStop TXP™ system significantly improves price/performance ratios in existing on-line applications, and can also be configured to handle hundreds of transactions per second with absolute data integrity. Based on a 32-bit multiple processor architecture, it delivers two to three times more processing power than the NonStop II™ system, reduces the cost per transaction by up to 50 percent, and maintains full software and hardware compatibility with NonStop II systems.

3 Bowen Crescent, Melbourne, Vic. 3004
Telephone: (03) 267 1577

22 Atchison Street, St. Leonards, N.S.W. 2065
Telephone: (02) 438 4566

148 Adelaide Terrace, Perth, W.A. 6000
Telephone: (09) 325 2242

S.G.I.O. Building, Cnr. Turbot & Albert Sts., Brisbane, Qld. 4000
Telephone: (07) 229 3766

3rd Floor, 151 South Terrace, Adelaide, S.A. 5000
Telephone: (08) 212 1110

148 Adelaide Terrace, Perth, W.A. 6000
Telephone: (09) 325 2242

20th Floor, William City Centre, Boulcott Street, Wellington, N.Z.
Telephone: (04) 72 3286

NCATC/14
FIRST TERMINAL BEGINS NEW FAMILY RANGE FOR WEBSTER

Australian minicomputer manufacturer, Webster Computer Corporation Pty Ltd, has released its first video terminal.

Webster Computer Corporation’s Managing Director, Mr David Webster has named the new detachable keyboard terminal the ST97, and says that it is the first of a coming family of terminal-based products, to be designed and produced by his company.

As with the Spectrum minicomputer range, the ST97 is compatible with Digital Equipment Corporation’s PDP-11 systems. It features full compatibility with DEC’s VT100 terminal with the three exceptions of smooth scroll, 132 column operations and double-sized characters.

Its enhancements include anti-glare screen, green phosphor, printer port, menu-driven set-up mode, low profile ergonomic keyboard with layout identical to the VT100, and rate limited escape sequences.

It is the second new product to be launched by the Webster Computer Corporation for 1984, in what Mr Webster foresees as a year of intensive product development for his company.

Last week Webster’s announced the extension of its marketing into add-on memory boards for DEC and DEC-compatible Qbus systems. These memory boards, known as the SMSV11 range come in sizes of 256 Kilobyte, 512Kb and 1 Megabyte and are currently being promoted on the Australian and US markets.

NBS DEFINES PRACTICAL WAYS OF MAINTAINING SOFTWARE

A recent study by the NBS Institute for Computer Sciences and Technology found that many of the problems associated with software maintenance stem from inadequate control and review of software maintenance activities by management. Guidance on Software Maintenance (SP 500-106) suggests practical ways to solve some of these problems. Tools and techniques that may be used to improve the control of software maintenance activities and the productivity of a software maintenance organisation are discussed along with programming techniques that help make maintenance easier. The report emphasises the need for strong control by managers. Available for $2.50 prepaid from the Superintendent of Documents, US Government Printing Office, Washington, DC 20402. Order by stock number 003-003-02535-6.

NBS STUDIES DESIGN OF THE AUTOMATED OFFICE

All too often, offices have automated their activities through piecemeal acquisition of new devices and systems. This "hardware driven" approach is often done without adequate planning, and results in systems which do not meet users’ needs. In particular, physical design issues are neglected, resulting in problems with the visual, thermal and acoustic environment in many offices. In a recent pro-
IBM ANNOUNCES NEW HIGH SPEED IMPACT PRINTER

IBM Australia has announced its fastest impact line printer, the IBM 4248, which has three different speeds for varying job requirements, viz. 3600, 3000 and 2200 lines per minute selectable under operator or program control. At 2200 lines per minute, it can print optical recognition characters.

A number of innovative features are standard on the 4248 including:
- 6 to 8 lines per inch (2.54 cm) variable within a page under program control.
- Side by side printing of duplicate pages using a 4248 unique command.
- Hammer flight time adjustment performed electronically by operator initiated micro-program to ensure optimum print quality is maintained.
- Microprocessor control of skipping speed to maintain optimum paper stacking, even at 3600 lines per minute.
- Microprocessor monitoring of printer operation to detect and diagnose potential problems to minimise service requirements.
- Twelve character alphanumeric LED display for diagnostic and test messages. Under program control, operational messages may also be displayed.
- Automatic verification that the print band matches the program requirements and auto buffer load of the matching character set.

The basic 4248 has 132 print positions, but an optional 168 print positions is available for plant or field installation.

TANDY INTRODUCES OS-9 AND BASIC-09 FOR TRS-80 64K EXTENDED COLOUR COMPUTER

Tandy Australia Limited introduces two new programming languages for the TRS-80 64K Extended Colour Computer. The OS-9 Disk Operating System is a real-time operating system that accesses the entire memory of the new 64K Extended Colour Computer. Basic-09 is an enhanced version of standard BASIC written for the 6809 microprocessor.

OS-9’s editor/assembler allows development of assembly language applications. The diskette consists of three programs — text editor, assembler and debugger.

The text editor offers the ability to edit assembly programs easily. It includes commands to edit single characters, groups of characters, complete lines or groups of lines. A macro definition facility allows the creation of new edit commands to perform commonly used complex editing functions.

The assembler converts assembly programs to 6809 microprocessor machine language for execution. The assembler will also produce a formatted program listing and alphabetised symbol table listing. It can also convert compiler-produced programs.

The debugger is designed to facilitate testing of machine language programs. It includes commands to examine, dump, change and test memory; examine change or initialise registers; insert or remove program breakpoints; execute programs; run OS-9 commands; and evaluate and convert arithmetic expressions in or to binary, hexadecimal or decimal number systems.

OS-9 is offered with documentation, reference manual and 5¼" diskette.

The new BASIC-09 includes advanced features derived from PASCAL for structured programming. It uses an interactive compiler that produces compact object programs for high speed execution.

BASIC-09 offers four major modes, including instructions for saving, loading and listing programs. In the edit mode, programs are entered or modified — with simultaneous error checking — and compiled. Programs are run in the execution mode, and the debug mode includes commands which allow high-level symbolic testing and debugging of programs.

HEWLETT-PACKARD SIGN ON LOCAL MANUFACTURER

Hewlett-Packard Australia Limited announced that it has signed a contract for local manufacturer Data Cable Pty Ltd to supply a range of interconnecting, high technology data cable assemblies.

Dr David Booker, Managing Director of Hewlett-Packard, said the cables, used to connect all HP computers to the full range of HP equipment including instrumentation, terminals, printers, analytical and medical equipment, would be distributed to HP customers throughout the Pacific Basin.

"Our data cable assemblies, previously manufactured in the USA, are expected to generate significant exports for Australian industry during the next twelve months."

The purchase of high quality cables from Data Cable demonstrates that Hewlett-Packard is serious about the commitment to Government offset requirements. The contract with Data Cable is just one of a number of other activities we are currently evaluating," said Dr Booker.

ACCESS FLOORS AND THE AUTOMATED OFFICE

Geoffrey Boswell, Vice President, International Operations, Tate Architectural Products, the leading manufacturers of access floors in North America, sees access floors expanding from the computer room, where they had their beginning, to the total office environment with many advantages for the building owner and the office planner.

"In meetings with building owners' representatives and their consultants during a recent visit to Sydney, Melbourne and Brisbane, he related recent developments in the United Kingdom and USA where infinite access floors are being used as the answer to the problem of handling the ever-increasing proliferation of office information systems and related services including power lines, telephone extensions, air conditioning, VDU and computer connections, all of which have become common place in today's automated office.

New standards for access floors used in the office are

(Continued on page ii at back)
Guest Editorial

Special Issue on Recent Advances in Computer Auditing

Computer audit is a relatively new area of specialisation that emerged to meet the challenge posed to the audit profession of disappearing audit trails in computerised information (accounting) systems. The requirement for increasing sophistication in the design of control techniques and audit trails as business information systems expanded in scope and complexity led to the recognition that new skills and EDP knowledge requirements were necessary to assure the integrity of commercial information systems.

The theoretical and practical basis of systems based auditing is not widely appreciated or understood by programmers and analysts with the result that system designs frequently lack effective internal controls and audit trails. The justification for this special issue of the Journal is consequently to provide appropriate insights into the role and contribution of the EDP auditor by reference to current practice and research directions.

A survey of EDP audit techniques currently used in external auditing of computer-based accounting systems in Australia is reported by Mr. Reeve in the first paper. As audit practice requirements are changing substantially, this snapshot of current practice is particularly valuable.

Recent results are reported by the Deakin Centre for EDP Audit. Research on the use of modelling as an audit tool demonstrates the exciting capability now possible for strengthening the power and specificity of system-based auditing.

The educational problem of transferring appropriate skills to auditors and computing practitioners is considerable. The paper by Holmes and Stanley, which describes an interactive microcomputer-based package for teaching EDP audit, is, therefore, a welcome addition to this issue.

Breydon's paper provides a valuable overview of the development and role of the EDP Auditors Association (EDPAA) in Australia over the last decade and draws particular attention to the Certified Information Systems Auditor (CISA) programme sponsored by EDPAA for practising auditors.

Many auditors expressed enthusiasm for and interest in the concept of special editions of the Journal on EDP Audit. The many verbal offers of future contributions were particularly encouraging and augur well for the journal as an important medium for the exchange of field experience, ideas and research findings by EDP auditors during the current explosive growth of this exciting new discipline.

The need to expose current practices and research to peer review is now well accepted, and present and future contributors deserve our commendation for making the time and effort available to convert verbal intentions into quality papers.

B.J. Garner,
Guest Editor
Trends in the Use of EDP Audit Techniques

R. C. Reeve*

This paper presents the results of a survey undertaken to establish the current use of EDP audit techniques in external auditing in Australia, of computer based accounting systems and expected trends in such use over the next two years. A distinction is made between large and small computer systems. In particular the relationship between current use of particular EDP audit techniques and expected trends in such use is investigated.

Keywords and phrases: EDP audit techniques, Computer-assisted audit techniques.
CR categories: J.1, K.6, K.7.

INTRODUCTION

The use of computers for the processing and storage of accounting information has grown at an increasing rate since they were first used for this purpose in Australia approximately 25 years ago. Indeed the stage has now been reached where, according to Humphries (1982), "Right now, there are very few accounting systems which do not use some form of EDP. In three years time, there will be none."

Although the computerisation of accounting records does not change the objective of the external audit task it has necessitated significant changes in audit techniques and methods. These result from the fact that with a computerised accounting system the methods by which the information is processed and stored are different to those in a manual or mechanical system. For instance many of the accounting data files (ledgers and journals) and many of the elements in the audit trail will be available only in machine readable form. Furthermore the introduction of computers will affect the procedures employed by the client firm to achieve internal control as, for example, the separation of accounting related functions among the client’s staff, for control purposes, becomes more difficult.

With computerised accounting systems it is also necessary for the auditor to devote more time to audit planning. This is necessary because of the relative inflexibility of the client’s data processing schedules and by the need to ensure availability, for audit work, of the client’s data processing personnel and the auditor’s computer audit staff.

Although the advent of computer based accounting systems has necessitated changes in audit methods this must not be viewed entirely negatively. The very fact that much of a client's accounting records are available in machine readable form allows the auditor to use the computer itself to assist him in carrying out his audit work. Sardinas, Burch and Asebrook (1981) make the point that "Many routine audit procedures can be performed electronically if auditors possess the ability to use the computer's tremendous speed and power. For example, random selection and printing of

audit confirmations, scanning of files for items of interest (slow-moving inventory, overdue accounts, related party transactions, to name a few), ratio and trend analysis, comparisons of budgeted, standard and prior years' data with that of the current year, are more effectively and efficiently performed electronically."

Over the past ten years a considerable literature has developed wherein the computer assisted audit techniques (CAATs hereafter) that have been developed, and the audit situations in which they may prove useful, are described. These include, for example, publications by the Canadian Institute of Chartered Accountants (1975), Cash, Bailey and Whinston (1977), the Institute of Internal Auditors (1977), the American Institute of Certified Public Accountants (1979), Mathieson (1980), Tobison and Davis (1981), Chew (1982), the International Federation of Accountants (1983) and the Australian Accounting Research Foundation (1983).

The majority of these works are of a prescriptive nature. Only two (Institute of Internal Auditors, 1977, and Tobison and Davis, 1981) contain information regarding the actual use of CAATs in audit practice and both of these are based on North American data.

The purpose of this paper is to describe certain results arising from a survey of Australian external auditors carried out by the author during 1983. In particular the survey was directed towards:

(a) establishing the actual use of specific CAATs by Australian Chartered Accountant firms in their audit work,
(b) establishing the expected trends in such use over the next two years, and
(c) offering some possible explanations for the current actual and expected future use of particular CAATs.

SURVEY DETAILS

In April 1983 a questionnaire was mailed to 195 offices of Chartered Accountant firms in Australia. These included all of the capital city offices of the 'Big Eight' public accounting firms and a random selection of other offices of each of these firms. This group totalled 103. The remaining questionnaires were mailed to a random selection of other public accounting firms. By July 1983, 66 usable responses had been received giving an overall response rate

Copyright © 1984, Australian Computer Society Inc.

General permission to republish, but not for profit, all or part of this material is granted, provided that ACJ’s copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Australian Computer Society.


**TABLE 1. Actual Use of CAATs**

<table>
<thead>
<tr>
<th>Type</th>
<th>Index Rank</th>
<th>Index Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Large Systems</td>
<td>On Small Systems</td>
<td></td>
</tr>
<tr>
<td>Generalised Audit Software Packages (GASP)</td>
<td>D</td>
<td>255</td>
</tr>
<tr>
<td>Specialised Audit Programs (SAP) written by auditor</td>
<td>D</td>
<td>145</td>
</tr>
<tr>
<td>Utility Program/s used as an alternative to GASP</td>
<td>D</td>
<td>143</td>
</tr>
<tr>
<td>Utility Program/s used as an adjunct to GASP</td>
<td>D</td>
<td>139</td>
</tr>
<tr>
<td>Parallel Simulation</td>
<td>P</td>
<td>118</td>
</tr>
<tr>
<td>Review of Program Logic</td>
<td>P</td>
<td>82</td>
</tr>
<tr>
<td>Test Data</td>
<td>P</td>
<td>61</td>
</tr>
<tr>
<td>Program Tracing (Manual)</td>
<td>P</td>
<td>48</td>
</tr>
<tr>
<td>Use of Job Accounting Data etc.</td>
<td>P</td>
<td>45</td>
</tr>
<tr>
<td>Integrated Test Facility</td>
<td>P</td>
<td>43</td>
</tr>
<tr>
<td>Timesharing programs</td>
<td>D</td>
<td>43</td>
</tr>
<tr>
<td>SAP is an existing or modified client program</td>
<td>D</td>
<td>39</td>
</tr>
<tr>
<td>Program Comparison of Source Program</td>
<td>P</td>
<td>32</td>
</tr>
<tr>
<td>SAP written by clients programmer</td>
<td>D</td>
<td>30</td>
</tr>
<tr>
<td>SAP written by outside programmer</td>
<td>D</td>
<td>25</td>
</tr>
<tr>
<td>Program Comparison of Object Program (Using Comparison Software)</td>
<td>P</td>
<td>23</td>
</tr>
<tr>
<td>Program Tracing (Using Tracing Software)</td>
<td>P</td>
<td>18</td>
</tr>
<tr>
<td>Embedded Audit Modules activated continually</td>
<td>P</td>
<td>16</td>
</tr>
<tr>
<td>Embedded Audit Modules activated periodically</td>
<td>P</td>
<td>14</td>
</tr>
<tr>
<td>Program Comparison of Object Program (Manual)</td>
<td>P</td>
<td>11</td>
</tr>
</tbody>
</table>

The questionnaire contained, *inter alia*, questions relating to the 20 CAATs mentioned in the authoritative publication “Computer Assisted Audit Techniques” (American Institute of Certified Public Accountants, 1979). Specifically, respondents were asked to indicate in respect of each of the 20 techniques:

(a) The extent of their current use of that CAAT (very widely used, widely used, regularly but not widely used, occasionally used, not used)

(b) The expected trend in the use of that CAAT over the next two years (significant increase in use, some increase in use, about the same as now, some decline in use, significant decline in use) in respect of both large and small systems. Responses to this section by respondents using one or more CAATs but not currently using a particular CAAT, have been included in the analysis. It is considered that responses received in respect of a CAAT of which the respondent is not a current user indicates sufficient familiarity with that technique to be relevant to this study.

Of the 66 respondents, 44 were currently using one or more of the CAATs listed. The results which follow are based on an analysis of those 44 completed questionnaires. A brief description of each of the 20 CAATs considered appears in Appendix 1.

**SURVEY RESULTS**

Table 1 summarises the extent to which each of the CAATs is currently being used in auditing both large and small systems. Table 2 summarises the expected trends in such use in respect of both large and small systems.

The ranking of the techniques in Tables 1 and 2 is based on indexes developed to indicate aggregate actual use and also aggregate expected trend in use. The actual use indexes were calculated by weighting the percentage of respondents advising that a technique was “very widely used” by 4, “widely used” by 3, “regularly but not widely used” by 2, “occasionally used” by 1 and “not used” by 0 and summing the results. The expected index was also calculated by weighting the percentage of respondents advising that a technique was “very widely used” by 4, “widely used” by 3, “regularly but not widely used” by 2, “occasionally used” by 1 and “not used” by 0 and summing the results. The expected trend in use indexes were calculated by weighting the percentage of respondents advising that their expectation for the trend in use of a particular phase is “significant increase” by 2, “some increase” by 1, “about the same as now” by 0, “some decline” by −1 and “significant decline” by −2 and summing the results. It should be noted that no attempt has been made to vary the weight of a response in any other way, for instance by the size of the responding firm

Furthermore the techniques have been classified as either data oriented techniques (Type D) or processing oriented techniques (Type P). This classification is derived from the phase in the audit process in which the technique is most likely to be used. This process can be divided into four phases:

(a) Initial review of the system of internal accounting control.

(b) Testing to ensure processing complies with the established system of internal control. It is in this phase that the processing oriented (Type P) tests would be used.

(c) Testing to confirm the validity of the data used in and produced by the accounting system. It is in this phase that the data oriented (Type D) tests would be used.

(d) Final analytical review of the financial statements. Although CAATs can be, and are, used in both the initial review and final review phases they are more frequently used for the data oriented and processing oriented testing mentioned above.

Inspection and analysis of the data contained in
TABLE 2. Expected Trends in Use of CAATs

<table>
<thead>
<tr>
<th>Type</th>
<th>On Large Systems</th>
<th>On Small Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generalised Audit Software Packages (GASP)</td>
<td>D 116 1</td>
<td>85 2</td>
</tr>
<tr>
<td>Specialised Audit Programs (SAP) written by auditor</td>
<td>D 38 4</td>
<td>46 3</td>
</tr>
<tr>
<td>Utility Program/s used as an alternative to GASP</td>
<td>D 69 2</td>
<td>86 1</td>
</tr>
<tr>
<td>Utility Program/s used as an adjunct to GASP</td>
<td>D 48 3</td>
<td>40 4</td>
</tr>
<tr>
<td>Parallel Simulation</td>
<td>P 22 9-10</td>
<td>22 6-7</td>
</tr>
<tr>
<td>Review of Program Logic</td>
<td>P 7 15</td>
<td>12 9-11</td>
</tr>
<tr>
<td>Test Data</td>
<td>P 27 5-6</td>
<td>31 5</td>
</tr>
<tr>
<td>Program Tracing (Manual)</td>
<td>P 0 18</td>
<td>5 16-19</td>
</tr>
<tr>
<td>Use of Job Accounting Data etc.</td>
<td>P 24 7-8</td>
<td>12 9-11</td>
</tr>
<tr>
<td>Integrated Test Facility</td>
<td>P 22 9-10</td>
<td>22 6-7</td>
</tr>
<tr>
<td>Timesharing programs</td>
<td>D 10 13-14</td>
<td>15 8</td>
</tr>
<tr>
<td>SAP is an existing or modified client program</td>
<td>D -5 20</td>
<td>5 16-19</td>
</tr>
<tr>
<td>Program Comparison of Source Program</td>
<td>P 15 12</td>
<td>12 9-11</td>
</tr>
<tr>
<td>SAP written by clients programmer</td>
<td>D -2 19</td>
<td>-2 20</td>
</tr>
<tr>
<td>SAP written by outside programmer</td>
<td>D 2 17</td>
<td>10 12-14</td>
</tr>
<tr>
<td>Program Comparison of Object Program (Using Comparison Software)</td>
<td>P 20 11</td>
<td>10 12-14</td>
</tr>
<tr>
<td>Program Tracing (Using Tracing Software)</td>
<td>P 10 13-14</td>
<td>10 12-14</td>
</tr>
<tr>
<td>Embedded Audit Modules activated continually</td>
<td>P 24 7-8</td>
<td>5 16-19</td>
</tr>
<tr>
<td>Embedded Audit Modules activated periodically</td>
<td>P 27 5-6</td>
<td>7 15</td>
</tr>
<tr>
<td>Program Comparison of Object Program (Manual)</td>
<td>P 5 16</td>
<td>5 16-19</td>
</tr>
</tbody>
</table>

NOTE: In this table the techniques are listed in order of their current use on large systems.

Tables 1 and 2 allows several general conclusions to be drawn.

Firstly, the techniques which are widely used on large systems are strongly correlated with those that are widely used on small systems, and vice versa. The correlation coefficient between the indexes of use for large and for small systems was calculated to be .899 which is significant at the .01 level.

Secondly, CAATs are currently more widely used on large systems, as reflected by an index value total of 1330, than on small systems, as reflected by an index value of 954.

Thirdly, the same seven techniques lead the actual use rankings in respect of both large and small systems. Although only amounting to 35% (7 out of 20) of the techniques under consideration, this group accounts for 70% (943 out of 1330) of the aggregate index of use for large systems and for 72% (689 out of 954) for small systems.

Fourthly, the four CAATs most widely used in both large and small systems are data oriented techniques (Type D) rather than processing oriented techniques (Type P). Furthermore these four most widely used techniques, are the techniques for which the greatest increase in use is expected over the next two years.

Fifthly, the techniques for which the greatest increase in use is expected on large systems are strongly correlated with those for which the greatest increase in use is expected on small systems, and vice versa. The correlation coefficient between the indexes of expected trend in use for large and for small systems was calculated to be .911 which is significant at the .01 level.

Sixthly, the expected increase in the use of CAATs on large systems, index value 479, was slightly higher than the expected increase of use on small systems, index value 438.

Lastly, there is a strong indication that, in general, the CAATs which are in wide current use will be more widely used in future, and vice versa. This tendency exists for both large and small systems. Correlating the index of use with the index of expected trend in use gave a correlation coefficient of .868 for large systems and of .869 for small systems, both of which are significant at the .01 level.

Following on from this last point Table 3 contains aggregate data reflecting current actual use and expected trend in use relationships for all of the CAATs taken together for large and for small systems.

Further analysing these aggregate figures suggests a strong positive relationship exists between current actual use and expected future use. Inspecting the figures for the Wide/Very Wide use category it can be seen that for every 10 responses indicating that future use will be about the same as current use there are a greater number of responses (14 for large systems and 32 for small systems) which
### TABLE 4. Relationship Between Extent of Actual Use and Expected Trend in Use on Large Systems

<table>
<thead>
<tr>
<th>Type</th>
<th>Spearman’s Coeff.</th>
<th>Signif.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generalised Audit Software Packages (GASP)</td>
<td>D .22 .08</td>
<td></td>
</tr>
<tr>
<td>Specialised Audit Programs (SAP) written by auditor</td>
<td>D -.01 .49</td>
<td></td>
</tr>
<tr>
<td>Utility Program/s used as an alternative to GASP</td>
<td>D .50 .01</td>
<td></td>
</tr>
<tr>
<td>Utility Program/s used as an adjunct to GASP</td>
<td>D .08 .30</td>
<td></td>
</tr>
<tr>
<td>Parallel Simulation</td>
<td>P .31 .03</td>
<td></td>
</tr>
<tr>
<td>Review of Program Logic</td>
<td>P -.05 .38</td>
<td></td>
</tr>
<tr>
<td>Test Data</td>
<td>P .01 .49</td>
<td></td>
</tr>
<tr>
<td>Program Tracing (Manual)</td>
<td>P -.21 .10</td>
<td></td>
</tr>
<tr>
<td>Use of Job Accounting Data etc.</td>
<td>P .34 .01</td>
<td></td>
</tr>
<tr>
<td>Integrated Test Facility</td>
<td>P .16 .16</td>
<td></td>
</tr>
<tr>
<td>Timesharing programs</td>
<td>D .15 .17</td>
<td></td>
</tr>
<tr>
<td>SAP is an existing or modified client program</td>
<td>D -.08 .32</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Source Program</td>
<td>P -.24 .07</td>
<td></td>
</tr>
<tr>
<td>SAP written by clients programmer</td>
<td>D -.24 .07</td>
<td></td>
</tr>
<tr>
<td>SAP written by outside programmer</td>
<td>D -.33 .02</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Object Program (Using Comparison Software)</td>
<td>P .06 .36</td>
<td></td>
</tr>
<tr>
<td>Program Tracing (Using Tracing Software)</td>
<td>P -.16 .16</td>
<td></td>
</tr>
<tr>
<td>Embedded Audit Modules activated continually</td>
<td>P .39 .01</td>
<td></td>
</tr>
<tr>
<td>Embedded Audit Modules activated periodically</td>
<td>P .24 .06</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Object Program (Manual)</td>
<td>P -.05 .38</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** In this table the techniques are listed in order of extent of current use on large systems.

Indicate an increase in use in the future. In the Occasional/Regular Use category for every 10 responses indicating that future use will be about the same as current use there are a smaller number of responses (six each for both large and small systems) which indicate an increase in use in the future. For the Not Used category there is an even smaller number of responses (two for large systems and one for small systems) indicating increased future use for every 10 responses which indicate no change on current use — which in this category indicates no current use.

This is not to suggest that the level of current use alone is causing the expected increase in future use. The decision to continue to use, or to increase the use of, a particular technique will depend on its feasibility in a particular situation. Feasibility has technological, operational and economic facets. The continuing development in computer technology will make certain techniques technologically feasible and make others technologically obsolete. Increasing knowledge of and competence in the computer technology area by both auditor and client staff will make these techniques operationally feasible. It is in this area that experience with a technique through current use could lead to its greater use in the future. Technological and operational feasibility, whilst necessary, are not of themselves sufficient conditions for the decision to use a particular CAAT in a particular audit situation. This will only take place if the use of the technique is also economically feasible — that it is the most cost-effective method of achieving a particular audit objective in a particular audit situation.

Continuing, Tables 4 and 5 contain data for each of

### TABLE 5. Relationship Between Extent of Actual Use and Expected Trend in Use on Small Systems

<table>
<thead>
<tr>
<th>Type</th>
<th>Spearman’s Coeff.</th>
<th>Signif.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialised Audit Programs (SAP) written by auditor</td>
<td>D .06 .36</td>
<td></td>
</tr>
<tr>
<td>Utility Program/s used as an alternative to GASP</td>
<td>D .63 .01</td>
<td></td>
</tr>
<tr>
<td>Generalised Audit Software Packages (GASP)</td>
<td>D .55 .01</td>
<td></td>
</tr>
<tr>
<td>Utility Program/s used as an adjunct to GASP</td>
<td>D .39 .01</td>
<td></td>
</tr>
<tr>
<td>Parallel Simulation</td>
<td>P .46 .01</td>
<td></td>
</tr>
<tr>
<td>Review of Program Logic</td>
<td>P .01 .46</td>
<td></td>
</tr>
<tr>
<td>Test Data</td>
<td>P .24 .07</td>
<td></td>
</tr>
<tr>
<td>Timesharing programs</td>
<td>D .16 .16</td>
<td></td>
</tr>
<tr>
<td>SAP is an existing or modified client program</td>
<td>D .12 .24</td>
<td></td>
</tr>
<tr>
<td>Program Tracing (Manual)</td>
<td>P -.13 .21</td>
<td></td>
</tr>
<tr>
<td>SAP written by outside programmer</td>
<td>D -.32 .02</td>
<td></td>
</tr>
<tr>
<td>SAP written by clients programmer</td>
<td>D -.18 .13</td>
<td></td>
</tr>
<tr>
<td>Integrated Test Facility</td>
<td>P .31 .02</td>
<td></td>
</tr>
<tr>
<td>Program Tracing (Using Tracing Software)</td>
<td>P -.12 .22</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Source Program</td>
<td>P -.18 .13</td>
<td></td>
</tr>
<tr>
<td>Use of Job Accounting Data etc.</td>
<td>P .68 .01</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Object Program (Manual)</td>
<td>P -.04 .41</td>
<td></td>
</tr>
<tr>
<td>Program Comparison of Object Program (Using Comparison Software)</td>
<td>P -.11 .25</td>
<td></td>
</tr>
<tr>
<td>Embedded Audit Modules activated periodically</td>
<td>P .36 .01</td>
<td></td>
</tr>
<tr>
<td>Embedded Audit Modules activated continually</td>
<td>P .46 .01</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** In this table the techniques are listed in order of extent of current use on small systems.
the individual CAATs regarding the relationship between extent of current use and expected trend in use on large and small systems respectively. For each technique a value has been calculated for the Spearman rank correlation coefficient of the relationship between each respondent's advice regarding current use, on the 1 to 5 ranking scale, and expected trend in use also on the 1 to 5 ranking scale.

The values of the correlation coefficients are then used to classify the techniques into three groups. Firstly there are those which have a positive coefficient of sufficient magnitude to be significant at the .10 level, secondly there are those which have a positive or negative correlation coefficient which is not of sufficient magnitude to be significant at the .10 level and thirdly there are those which have a negative coefficient of sufficient magnitude to be significant at the .10 level.

Techniques in the first group have a statistically significant positive relationship between current and future use. In the case of large systems the techniques falling into this group are: Generalised Audit Software Packages, Utility Program/s used as an alternative to GASP, Parallel Simulation and Use of Job Accounting Data etc., which are all fairly widely used at present, and Embedded Audit Modules activated either continually or periodically, which are not widely used at present. In the case of small systems the techniques falling into this group are: Utility programs used either as an alternative or an adjunct to GASP; Generalised Audit Software Packages; Parallel Simulation, and Test Data, which are all fairly widely used at present; and Integrated Test Facility, Use of Job Accounting Data etc., and Embedded Audit Modules activated either continually or periodically, which are all not very widely used at present.

The interpretation of the correlation coefficient value for the third group (significant negative coefficient) is that this indicates techniques in which a decline relative to current use is expected. In the case of large systems the techniques falling into this group are: Utility programs user either as an alternative or an adjunct to GASP; Generalised Audit Software Packages; Parallel Simulation, and Test Data, which are all fairly widely used at present; and Integrated Test Facility, Use of Job Accounting Data etc., and Embedded Audit Modules activated either continually or periodically, which are all not very widely used at present.

The balance of the CAATs fall into the second group, that is those where the correlation coefficient, whether negative or positive, was not large enough to be significant at the .10 level. This indicates that no significant relationship exists between their level of current use and the expected trend in their use.

CONCLUDING COMMENTS
It is hoped that the results of the survey described above will provide useful information for both practitioners and educators in the field of EDP auditing, especially those who are relative newcomers to this area. Whilst the widespread use of Generalised Audit Software Packages and similar data oriented techniques is expected to continue the emergence of newer and more complex processing oriented techniques should be noted. The proper use of these techniques, such as Parallel Simulation, Integrated Test Facility, Embedded Audit Modules, and Use of Job Accounting Data, requires a far greater knowledge of computers and data processing than exists in the majority of accounting graduates entering audit practice today or, for that matter, in the majority of audit practitioners. This situation is likely to continue into the future. Developments in computer technology will lead to organisational control and information processing/storage changes within client firms. These changes will present new problems to the auditor for which new audit methods will have to be developed. These new audit methods will be likely to require a greater understanding of computer technology. To some extent this computer expertise will be obtained by including non-accountant computer specialists on the audit team. However the present position of the auditing profession in Australia on this last matter is that "...when the EDP system is an integral part of the accounting system and related internal controls, the auditor should not delegate to an EDP expert his responsibility for forming important audit conclusions" (Australian Accounting Research Foundation, 1983). Thus the auditor will have to continuously improve his knowledge of computer technology and computer audit techniques so that he is able to carry out considerable auditing work on computerised accounting systems without assistance from EDP experts. He must, however, be able to recognise when expert assistance is required and have the knowledge to be able to direct, supervise and review the work of such persons. Both the auditing profession and the tertiary education institutions will have to make a concerted effort to develop strategies to meet this challenge.

The author would like to acknowledge and thank those practitioners who gave of their time to complete the questionnaire. Thanks are also due to those delegates to the EDP Auditors Association Conference in Melbourne in 1983 who provided valuable feedback to the author on an earlier paper in this area. Also to those colleagues and referees who assisted the author in various ways in the preparation of this paper. Naturally all are absolved from any responsibility for the result.

APPENDIX 1
BRIEF DESCRIPTIONS OF THE CAATS CONSIDERED

Generalised Audit Software Packages (GASP)
A computer program or series of programs designed to perform certain data processing functions of interest to auditors on the client's computerised records.

Test Data
A set of transactions processed by the auditor to test the programmed controls and procedural operations of the client's computerised application.

Integrated Test Facility (ITF)
The establishment of a "dummy" entity in the client's computerised records through which data can be processed at the direction of the auditor.

Program Tracing
A technique in which an auditor follows the processing steps performed by a computer program for a given transaction.

Program Tracing (Manual approach).
Inspection of source code listing.
Program Tracing (Using Tracing Software).
Trends in The Use of EDP Audit Techniques

Tracing software displays a list of the program steps executed for each transaction processed by a program.

Review of Program Logic
Reviewing the documentation, including the source program if necessary, to develop a sufficient understanding of the subject program or application.

Program Comparison
Generally this technique is used to compare a copy of a program that is under the auditor's control to the version of the program currently used in processing. Program Comparison of Source Program, Program Comparison of Object Program using manual methods, Program Comparison of Object Program using comparison software.

Utility Programs
Most computer manufacturers provide software called "utility programs" to perform common data processing functions such as sorting of files and printing of files. Utility Programs used as an adjunct to a generalised audit software package. Utility Programs used as an alternative to a generalised audit software package.

Specialised Audit Programs (SAP)
Computer programs written by or for the auditor to perform particular audit tasks in specific situations. SAP written by auditor, SAP written by outside programmer, SAP written by client's programmer, SAP is an existing or modified client program.

Timesharing Programs
Many timesharing vendors have libraries containing programs that can be helpful to auditors e.g. analytical review, statistical sampling.

Parallel Simulation
The auditor develops a program to perform the same key functions as the client's application being tested. The same data is processed through both systems and the results are compared.

Embedded Audit Modules (EAM)
Sections of program code designed to perform audit functions, which are incorporated into the clients application program/s. EAM activated periodically, EAM activated continuously.

Use of Job Accounting Data etc.
Many computer manufacturers provide software to generate utilisation reports of the resources used by the computer system. This information can be of use to the auditor.

REFERENCES

BIOPGRAPHICAL NOTE
Robert Reeve qualified as a Chartered Accountant in South Africa in 1965 and remained in public practice until 1972. From 1973 to 1975 he worked as Financial and Administrative Manager of a clothing manufacturer. He then joined the Department of Accountancy at the University of Natal in Durban where he lectured in Auditing and Management Accounting. In 1981 he emigrated to Australia to take up an appointment at the University of New England where he lectures in the Information Systems and Auditing areas. He holds a Bachelors degree in Accounting and a Masters degree in Information Systems.
Modelling as an Audit Technique

B. J. Garner* and J. Pinnis†

This paper explores the creation and use of audit models as powerful new tools for the EDP auditor. A paradigm for model representation is offered as an extension of current methodology for systems-based auditing, and three types of audit model are defined. Research results are reported on the application of detective audit models to testing for compliance with general (environmental) controls.

Keywords and phrases: Audit model, general controls, preventive audit.

THE CONCEPT OF AUDIT MODELS

The word model has wide currency, typically in financial planning (e.g. use of 'spreadsheets' and analytical review). Statements on Auditing Standards often include mathematical models. The AICPA standard (1981) on audit sampling, for example, provides a statistical model for analysing ultimate risk.

The term audit model as used in this paper is intended to extend the generic class of behavioural model to the domain of the auditor, including where appropriate, mathematical descriptions of the behaviour under investigation such as the behaviour of internal control systems. There is thus a logical connection through their common audit practice requirements of audit models and systems-based auditing.

THE NEED FOR NEW TOOLS

Attention has recently been directed (Garner, 1983) to the audit implications of future technology. Concern is justified that as commercial information architectures become more complex, the credibility of the audit opinion may suffer, and auditor productivity will be adversely affected, unless audit practice requirements for the new environments are reviewed and appropriate tools are developed. In particular, we note that system-based auditing — the cornerstone of modern audit practice — is, in itself, no longer a sufficient basis for cost-effective audits in the light of modern developments in information technology, e.g.

— The increasing sophistication and operational scope of new application systems.
— User-driven systems, including fourth generation languages and decision support in unstructured environments.
— Electronic messaging systems including the automated office.

Field work is predicted by Mar (1983) to decline substantially as a percentage of the total audit time, reflecting the greater emphasis on planning, selection of internal controls and system monitoring requirements.

The corollary to this prediction is that audit methodologies based on systems-based auditing require strengthening. Expanded capabilities are seen to be required for example:

(i) Methodologies that possess inbuilt cost-effectiveness criteria;
(ii) More powerful experimental tools; and
(iii) Greater degree of sensitivity in the probes used to monitor the status of internal control. Analytical review, for example, is increasingly perceived as a powerful (sensitive) probe of compliance with commercial objectives.

This paper communicates recent research on the use of models as a way of strengthening systems-based auditing. The role and structure of models as a generic class of new audit tools are now examined in terms of appropriate methodology and audit practice requirements.

BASIC CONSTRUCTS OF A MODEL

The basic constructs of a model within information systems are now well defined, and consist of:

(i) Objects — Entities which are essentially the things or components that we wish to describe.
(ii) Attributes — Objects are endowed with certain characteristics known as attributes or properties. The attributes essentially define the object.
(iii) Relationships/Rules — In constructing or using a model a set of relationships (or rules) are required to specify how objects may interact or relate to each other.
(iv) Integrity Constraint — Rules that govern the manipulation of the model are usually limited in some way to preserve the integrity of the relationships under study. Integrity constraints essentially define the purposes for which the model may be used.

Two examples of the use of these constructs are now briefly referenced:

Data Models

The relational data model is a particularly good example, and comprises:

— Data structures — objects, attributes, relations.
— Set of operators — join, etc.
— Integrity constraints — i.e. semantic constraints.

Model for Specification of Office Communications

A model recently proposed by Konsynski, Bracker and Bracker (1982) for the specification of office commu-
communications has found particular application in office automation:
- Objects — office functional and communications components.
- Attributes — properties of objects with clarifying values.
- Relations — logical object interconnections; relations are needed to support the information flow and object interconnections.
- Integrity Constraints — Consistency rules for allowable relations between objects.

It should be noted in passing that interest in system performance measurement by computer engineers and software programmers has required detailed computer operation models since the sixties, and the increasing interest in software metrics will facilitate auditor access to the operating measures they require. We also note the growing urgency of the need to model internal control systems as a fundamental prerequisite to increasing the credibility of audit opinions.

INTRODUCTION TO AUDIT MODELS
Auditors are concerned with the behaviour of information systems and with the effectiveness of management controls in such systems. Models are representations that provide insight into the behaviour of a system; and the objective of modelling is a better understanding and prediction of system behaviour under diverse conditions. Audit models, which we define as abstractions of the control structures within information systems environments may, therefore, serve as a medium for experimentation and analysis.

While auditors will typically not be familiar with the more general paradigms proposed for modelling complex information systems, there is evidence from the techniques currently in use that modelling as an EDP audit tool is not a new concept. Parallel Simulation, for example, which involves the development by the auditor of a computer program to simulate the logic of an important application program, such as payroll, is essentially a model of the application process, albeit with limited objectives. Financial Modelling, in the form of spreadsheet software, is often used by auditors to predict the anticipated financial ratios of performance of an organisation. The value of these techniques for independent verification of the validity of material items is well established, but there is little apparent recognition of the scope and power of modelling past these essentially simplistic applications.

With systems based audit, for example, the auditor achieves an understanding of the overall design of a system, its data flow, processing, functions and controls. From the perspective gained of the overall system, the auditor identifies the major controls/processes that require review for reasonable assurance on the quality of internal control rather than attempting the impossible task of tracking all transactions.

There is an implicit similarity in this approach to that required to formulate the model of a particular application or environment, and a key feature of this paper is to relate the auditor's work process to the requirements of modelling. This is illustrated in Figure 1.

CREATION OF AN AUDIT MODEL
For all practical purposes* the reader can now assume that modelling, as intended in this paper, is essentially an audit tool that provides additional input into the decision making process. The usefulness of this additional input depends on a number of factors, such as:
(i) Selection of a class of model pertinent to the particular problem under investigation.
(ii) The 'transfer error' between reality and the model. The model may, for example, be a poor substitute for reality due to incomplete knowledge of the system structure.
(iii) Quality of inputs from other sources, thereby making comparison of findings rather difficult.
(iv) Quality of the analysis of the experimental results.

In terms of audit modelling practice, however, the adoption of a standard procedure in creating the model will facilitate subsequent reliance on the experimental findings: The recommended procedure is summarised in Figure 2.

BENEFITS OF AUDIT MODELLING
The practical utility to auditors of modelling infor-

* Statements on Auditing Standards often include mathematical models. The AICPA standard (1981) on Audit Sampling, for example, provides a statistical model for analyzing ultimate risk. Such standards in no way limit the generality of modelling per se. Practical constraints may nevertheless limit the usefulness of certain types of model, as indeed, often happens in the application of statistical sampling.
Modelling as an Audit Technique

STEP CONSIDERATIONS

1. Identification of the Scope and Functional Objectives of the Model:
   - Definition of the extent and purpose of the model in the light of audit objectives.

2. Identification of the type of model and the elements that comprise it.
   - What type of model is required and how should the composite elements, relations and permissible rules/operations be selected.

3. Definition of the Properties (i.e., attributes) of the elements constituting the Model:
   - Values and states to be assigned to the elements. Mathematical models will include algebraic relations.

4. Formulation of the experimental conditions applicable to the audit model, i.e., Choice of Modelling/Analytical technique:
   - Sources of data required by the Model and the availability of software tools.

5. Validation of the Model:
   - Measures of proof/effectiveness.

Figure 2. Standard Procedure for Model Creation.

Information systems environments may be briefly summarised as follows:
(i) Provides a basis for predicting the efficacy of individual control techniques, or of a group of controls, in specific environments. Note in comparison the imprecise nature of other methods (Liebeck and Rittenberg, 1981) of obtaining such information.
(ii) The general applicability of modelling promises useful results in a wide range of environments (i.e., potentially technology independent!)
(iii) Offers the benefits of incorporating prior knowledge (standards) in the modelling process leading to exception reporting.
(iv) Quantitative audit measures may be incorporated in the model, (e.g., by using statistical sampling!)
(v) Modelling makes maximum use of available software tools for improving auditor effectiveness and productivity.
(vi) Audit models are portable at the conceptual level. It follows that improved audit effectiveness, and subsequently, improved management control, may be expected from audit models that address three major areas of concern; namely:
   - Unwarranted reliance on the system of internal control.
   - Auditability of the control techniques.
   - Quality of substantive testing.

Three distinct classes of audit model directed to the concerns cited above are identified below, and for the purposes of demonstrating the practical utility of audit models some recent results are reported of detailed investigations into Detective Audit Models. The three classes of audit model are:
   - Preventive Audit Models.
   - Discovery Audit Models.
   - Detective Audit Models.

PREVENTIVE AUDIT MODELS

The concept that audit objectives should be incorporated in the design of system controls is apparently new (Richardson, 1983) and is not covered in the ICA or ASA statements on Auditing Standards and Practices. Clearly, auditors may have great difficulty in getting the information they need unless they are involved in the development of the system controls (Holley and Cash, 1981).

Preventive (preventative) audit can be defined as the application of practices and techniques for building into systems sufficient auditability to ensure that audit objectives are met efficiently. The auditor's participation is seen to be required during new system development, design and implementation to establish a control model, within which specific aspects of control can be evaluated and an audit trail readily created. A preventive audit model is consequently a model of internal control with the following characteristics:
(i) Permits error conditions to be simulated (e.g. wrong data types) with a view to studying the adequacy of internal control, and the quality of the audit trail.
(ii) Provides a framework (structure) for the integration of control elements and an overall assessment of their effectiveness/auditability, e.g. integration of application controls, and general controls.
(iii) A strategic assessment of the structure of internal control will be possible using preventive audit models.

DISCOVERY AUDIT MODELS

Discovery models for post-audit situations have been available for some years as a special kind of attribute sampling model when the auditor believes that the occurrence of control violations is close to zero. Our research indicates, however, that probably the most fertile application of statistical discovery models is through the extension of concurrent audit techniques. These powerful new tools are characterised by:
(i) Multiple (concurrent) attribute sampling on a statistical basis.
(ii) 'In-flight' analysis using the discovery model (e.g. for fraud detection!)
(iii) Development of predictive indicators which the model may change dynamically as a consequence of changes in transaction mix.

Discovery audit models are thus seen to be very powerful analytical tools for the identification of trends or for monitoring unusual patterns of transaction activity. Immediate application is possible in such areas as:
   - Financial Ratios.
   - Sales Trends.
   - Bad Debt Control.

DETECTIVE AUDIT MODELS

The function of a detective model is the analysis of past activity to review its compliance with required procedures and standards. Utilising the suggested procedure for audit model creation proposed in Figure 2, this chapter describes the creation of such a model as applied to the general controls (environment) audit of a data processing installation.

1. The first function of the modelling process is to determine the scope and purpose of the model. In this example, the activities of a computer operations department with regard to an individual application job are to be considered.

Traditionally the auditor reviews the operations department with a variety of questionnaires, examination of documented standards, interview of personnel, practical experience and observation. On this basis an opinion is expressed on the adequacy of controls and procedures existing in this department,
2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships, Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.

2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships. Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.

2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships. Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.

2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships. Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.

2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships. Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.

2. The second stage of devising the model is to determine its form and composite elements. In this example it was to compose a description of the application job, its component programs and files, their resource requirements and relationships. Parameters to describe this would be collated from defined/reported operational procedures, followed by a comparison with what was considered the 'actual' activity of the operating system, as described on the operating system log.

A simulation modelling process was then initiated between these two descriptions of the application. This required definition of a time-frame (say, the first three months of 1984) and an initial set of values for each parameter. Then the computer program that conducts the process alters the state of the model in response to the detail on the operating system log and its own heuristic logic.

Figure 3 illustrates the application job structure that was utilised by this model, which was adapted from the structure of the Standard Audit File (Garner and Pinnis, 1981). In brief, it shows that a model must be composed of at least one program, and each program has custody of a major portion of the company's information resource, and which provides the run-time environment for all application systems.

A weakness in control in the operations department potentially compromises all application systems, and that current audit procedures to review the department are generally manually based and transitory in nature is unsatisfactory. This strongly supports the need for more sophisticated, technology based tools in this area.

If a model were developed of an application system, describing its elements, their parameters, and its run-time schedule; and if this model was used to test the actual running of that application system, the auditor has substantively tested the activities of the operations department and independently verified the processing of the application system.
control language specifications, the operations schedule, and practical knowledge of the application. It provides the initial set of values for the model, and the information for specifying the relationships that exist.

Secondly, data that describes the actual processing activities of the operations department is required for comparison with the current version of the model. The source for this data is the operating system log, which is first reduced to a manageable format. This reduction process involves extracting the detail relevant to the running of the application from the operating system log, and collating it in the Standard Audit File (SAF) format, which corresponds to that of the model (see Section 2 and Figure 3).

We have identified the structure of the model, the relationships that exist between its various elements, and the source of the modelling data; an overview of the processing logic of the modelling program is given in Figure 5.

Having defined the model, and located the appropriate data sources the conduct of the actual modelling process must be conducted, and conclusions drawn.

This heuristic procedure involves the auditor running the modelling program, which analyses compliance of the model as currently defined in regard to the detailed description of actual processing activity held on the SAF. Discrepancies between these two descriptions are reviewed by the auditor, the model parameters altered if appropriate, and the modelling program re-run. This process is depicted in Figure 6, and is continued until the auditor is satisfied that an accurate model has been developed. Discrepancies identified in the final iteration reflect variations in procedure of legitimate concern and warranting further investigation.

CONCLUSIONS

The usefulness of this detective modelling process to a particular audit has been validated in an operating envir-
Modelling as an Audit Technique

The information produced is detailed, substantive, and provides proof of the activities of the operations department. Typically, the auditor was able to investigate matters such as:

- Security analysis: confirming exactly what use has been made of particular programs and files.
- Schedule analysis: evaluating whether application jobs have been run correctly, and to schedule.
- Termination analysis: analysing all invalid terminations of programs to assess compliance with software modification and re-run standards, or the presence of problem programs.
- User analysis: to establish the exact activities of a particular user.
- Documentation analysis: to determine the correct format of jobs run, and ensure that this corresponds with that found in the documentation.

The compliance modelling process described has been validated utilising IBM’s System Management Facility (SMF) data. Matters of audit significance detected by this test included:

- Use of test files in production.
- Use of non-standard filenames.
- Jobs not run to schedule.
- Unreported runs of a production job.
- Inefficient use of files by vendor software.

Concurrent with detecting these deviations in internal control standards, confirmation of compliance with standards was also observed.

Although the model described was based on batch processing, the potential of a compliance model in database/online environments has been briefly investigated, and identified as plausible. We are very optimistic concerning the potential of audit models as new tools for the auditor and their value in strengthening systems based auditing. The development and use of preventive and discovery models will be reported in depth in later papers.

BIBLIOGRAPHY


BIOGRAPHICAL NOTE

Professor B.J. Garner was appointed in July 1978 to the Foundation Chair in Computing at Deakin University from the position of manager, corporate data processing for the Broken Hill Proprietary Company Ltd. He has extensive experience of business information systems and directs a major research program in EDP Audit. His research is supported financially by Arthur Young and Company and by RAN DATA Pty Ltd.

Mr John Pinnis is an EDP Project Leader in the office of the Victorian Auditor General. His duties include the planning and supervision of EDP audit projects throughout the Victorian Public Sector, and the research and development of EDP audit techniques. Mr Pinnis is a member of the Australian Computer Society, on the board of directors of the Melbourne Chapter of the EDPA, and holds degrees in economics and computer science.
An Interactive Package for Teaching EDP Audit

W. N. Holmes* and P. M. Stanley†

This paper briefly discusses Generalised Audit Packages and EDP audit education. A modest portable interactive EDP audit package was developed for basic EDP audit training based on microcomputers. This package is first described from the user's point of view and then some of the details and problems of implementation are reviewed.

Keywords and phrases: EDP audit, audit education, interactive package.
CR categories: D.1, J.1, K.6.

INTRODUCTION

In the development of EDP hardware and software over the past twenty years, there has been little regard for the problem of auditability. As a consequence, the task of the auditor has been made more difficult, and many in the auditing profession have not built up their skills to cope with the complex tasks of auditing on-line and real-time systems.

In the early years of EDP systems, circa 1963-1970, auditors clung to the belief that computer systems could be audited by the "Audit Around" approach. This was effective on batch systems, but is now recognised as out of date and inadequate to handle the complex on-line systems now in use. The use of the "Audit Through" approach is now almost universally accepted.

At this time there is a serious and pressing need for the adequate education of auditors in modern EDP audit techniques. The major techniques of use to auditors in auditing computing systems are computer assisted. In essence, if an auditor is to achieve the best possible view of a computer system which has generated vast amounts of data and files then he should use computer assistance in the task.

The task of the auditor is to arrive at an opinion of the system. Obviously he cannot himself examine every transaction processed or every line of code in the programs, so such an opinion can never be, in the absolute sense, correct. Nonetheless the computer offers help to examine more data and check more computations and results, and so if used appropriately it will increase the audit evidence gathered, and lead to a more informed opinion as to the status of the system.

Clearly the computer cannot be used in every nook and cranny of the system — there is a time to use a bulldozer and a time to use a spanner — and the auditor must know how and when to harness the power of the computer to the audit task.

The first problem is therefore educating the auditor in the use of the computer to help in the task of auditing EDP systems. The interactive package resulting from the authors' work described in this paper is an educational aid, written to explore how a simple package might help with solving this first problem.

AUDIT TECHNIQUES

Audit techniques cover a wide variety of methods and tools. In general, they fall into three main categories:
1. Data Oriented.
2. System Oriented.
3. Problem Oriented.

General education in EDP audit must start with data oriented programs, which take three forms. These are:
2. Generalised Computer Programs.
3. Subroutines Embedded into User Programs.

The interest of this paper is towards the first of these forms — Generalised Computer Audit Packages — in particular towards such packages in classroom use.

Outside the classroom, Generalised Computer Audit Packages are a widespread, possibly the commonest, form of EDP assistance in an audit. However, use of such packages is not the only technique, nor could it be, in the total EDP audit problem.

GENERALISED COMPUTER AUDIT PACKAGES

There is a considerable number of Generalised Computer Audit Packages available on the market at the present time. They come in a variety of makes and models. The majority are written for IBM or IBM-compatible machines, and are generally designed for mainframe computers. Some are oriented to other machines specifically, and some will operate on a variety of machines.

In general, they divide into two main classes, viz:
- Macro Languages, such as EASYTRIEVE,
- Code Generators, such as CARS and STRATA.

They perform the same major functions such as:
- create, edit, accept
- extract, select, sample
- re-organise, sort, merge, scan, compare, summarise
- process, calculate
- report, write confirmation letters

The work described in the following sections was aimed at producing a generalised audit package for educational use, and to be used interactively on microcomputers. The resulting package has been called APC.
While there are many capable packages available for use on mainframe computers, there is little help available for microcomputer based accounting systems. Such a capability would provide a relatively cheap training base for audit firms and educational establishments.

BACKGROUND TO THE PACKAGE

The work on APC was sparked off by a number of influences. These were:

1. The creation of ACL (Audit Command Language) by Professor Hart Will (1981), University of British Columbia, Canada, as a universal, general purpose, audit language.

2. The development of TREAT by Associate Professor Miklos Vasarhelyi (1979), Columbia University, New York, in the APL language, as a portable version of STRATA, a Touche Ross Generalised Audit Package, for use on mainframe computers.

3. The use of EASYTRIEVE, a Generalised Audit Package marketed by Pansophic Software, by one of the authors (PMS) during study leave in 1981.

4. The loan of two 5100 Portable Computers by IBM Australia for development work. These machines embody an APL interpreter (Friedl 1983) and produce a video signal which can be fed to a television monitor to allow students in a lecture to see software in use.

The work sought to produce APC as a prototype Generalised Audit Package for educational use based on microcomputers. Vasarhelyi's work had shown that APL could be used to produce a portable audit package, but a more modest audit package was needed for use with more modest computers.

At first the EASYTRIEVE approach was favoured because of its free-form COBOL-like structure. After due consideration, it was decided to adopt a less structured approach based on an English vocabulary, and with a syntax like that of English. Such an approach was relatively easy to implement with the function oriented interpretive system used for APL work.

The approach allowed the creation of simple and commonly understood commands which could be confidently used by auditors and students with limited computing background. While this may appear counter productive in one sense, it will help in the transitional problem of getting a substantial number of accountants and auditors in public practice up to the necessary level of competence to audit microcomputer systems.

The longer term approach is to train adequately all accountants and auditors in EDP and EDP Audit as a substantial part of their professional training. This will take at least five years. In the meantime an approach like that described here could get EDP Audit training off to a good start.

DESIGNING THE PACKAGE

The following description of the features of the package APC will show how the major functions of typical Generalised Audit Packages were provided. The approach was to exploit the interactive capabilities of the APL system to build a package which would be easy for students to use, but which would have enough capability to allow challenging assignments to be set. The following main objectives were deemed appropriate:

1. To use the package, the auditor or student should be able to specify actions in a natural way, using language which is ordinary to him.

2. To apply the package to the data, the auditor or student should be able to nominate data in an obvious way, and the data should be arranged in an obvious way.

3. The actions from which the auditor or student can choose, roughly corresponding to the major audit package functions outlined above, should include the ability to:
   - select from the data stored,
   - inspect the data selected,
   - manipulate the data selected or stored, and
   - from time to time apply the package to different sets of data.

4. When the auditor or student errs in using the package, he should simply be able to restate the action he wants carried out.

USE OF NATURAL EXPRESSIONS

It is sensible and practical to allow or to provide English words to be used in a reasonably natural way as part of an interface to an audit package.

In APC the user can specify an action by putting in an expression such as

```
SELECT CREDIT GT 100
```

in which the name SELECT is provided by the package, CREDIT is chosen by the user, GT is a conventional abbreviation provided by the package, and 100 is a mathematical notation chosen by the user. Rows of data for which CREDIT is greater than 100 will be selected for use by other subsequent actions — for instance, by PRINT to display the rows of data to the user.

Thus, to display a certain four columns of selected data, the user could put in the action specification

```
PRINT CUST,NAME,CREDIT,CLBAL
```

in which PRINT is provided by the package, the remaining names would have been chosen earlier by the user or instructor, and the commas are provided by the APL notation on which APC is based.

The same action may also be specified by the expression

```
PRINT CUST AND NAME AND CREDIT AND CLBAL
```

in which the user has chosen to substitute the package-provided word AND for the notational commas of the preceding example. At least some natural variations are therefore handled, including multiple interpolated blanks.

The points to be noticed here are, firstly, that the words and conventional notation that the user has at his disposal are usefully combined in a way that closely resembles English imperative sentences, and, secondly, that the user is supplied a vocabulary which he can extend and from which he can compose potentially new and useful expressions.

OBVIOUS ARRANGEMENT OF DATA

There are two aspects to the handling of data in an interactive general purpose package.

One aspect is how the data appear to the user — how they are displayed, and consequently how the user thinks about them. In this package the data appear as rows of related items where like categories of data are lined up in columns. This is a most natural way to have data laid out — the way stock market figures and sporting results are presented in newspapers.
The other aspect is how the columns of date are named—how they are referred to by the user.

In APC commonplace names are provided for, or chosen by, the user. Names of data, such as CUST, NAME, and OPBAL are normally chosen by the user or instructor.

Further, functions such as PRINT which display data for inspection use the commonplace name as a column heading, as shown in the following example of the result of keying in

```
PRINT CUST, NAME, OPBAL, DEBIT, CREDIT, CLBAL
```
to APC, where a small selection of data is displayed.

<table>
<thead>
<tr>
<th>CUST</th>
<th>NAME</th>
<th>OPBAL</th>
<th>DEBIT</th>
<th>CREDIT</th>
<th>CLBAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>9 P. Papadopoulos</td>
<td>88</td>
<td>0</td>
<td>0</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>12 John Smith</td>
<td>423</td>
<td>199</td>
<td>299</td>
<td>323</td>
<td></td>
</tr>
<tr>
<td>16 Jones Hardware</td>
<td>23</td>
<td>1044</td>
<td>0</td>
<td>1067</td>
<td></td>
</tr>
<tr>
<td>22 Chung Wah Cafe</td>
<td>0</td>
<td>322</td>
<td>322</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

The package also provides the name ALL so that, in the example above, if all the columns available are to be displayed, then the user need only key in

```
PRINT ALL
```

Along similar lines, provision has been made for the user to give a name to groups of names. The user can specify, for example

```
'NAME' IS OPBAL, DEBIT, CREDIT, CLBAL
```

and these must be used with a column name for at least one of their arguments, though a constant such as 0 or 999 may be used for the other argument instead of a column name.

Other kinds of selection functions are provided. For instance, specifying

```
SELECT ANY 10
```

will cause ten rows of data to be pseudorandomly selected (auditors should be made familiar with the traps laid by use of pseudorandom numbers), while specifying

```
SELECT ANY 0.2
```

will cause one fifth of the available rows of data to be pseudorandomly selected.

For an auditor, specific comparative selections are not as useful as categorical selections. Such selections can be done in APC by action specifications like

```
SELECT CUST BETWEEN 1000 AND 2000
```

where CUST is the name of functions provided as part of APC.

Student auditors need to be taught to build up a flow or sequence of investigation. Having investigated one selection, an auditor will sometimes want to examine those data which were not among those selected. To do this he specifies

```
SELECT OTHERS
```
to get the complementary selection.

It is important in an educational package, and probably in a working auditors' package, to provide staged selection whereby the field of view can be widened or narrowed as the user finds out more about his data.

Thus, the package user will want to go from one selection to some other related selection. This capability is provided in APC by the functions WIDEN and NARROW used like SELECT as shown in the examples

```
WIDEN 0 EQ OPBAL
```

which will add any unselected instances of zero opening balance to the data already selected, and

```
NARROW CLBAL NE ZERO
```

which will remove from the present selection any instance of a zero closing balance. Any selection predicate which can be used with SELECT can be used with NARROW and WIDEN.

Audit packages should keep the user informed about the immediate effect of any action. SELECT, NARROW and WIDEN report to the user as they finish how many rows of data end up in the selection. WIDEN also reports the overlap—how many rows in the original selection also satisfy the WIDENing criterion.

**INSPECTION OF DATA**

Given that certain rows of data have been selected, the user will then want to find out something about that selection.

The most obvious thing to do is to have a look at the selected data, and in APC the verb PRINT is used as illustrated above to do this.

Sometimes there will be too much data selected to scan conveniently. In this case some form of analysis or summary would be useful. APC provides TOTAL and AVG, to be used in the style

```
TOTAL DEBIT, CREDIT
AVG ALL
```

with the expected results and where non-numeric columns are automatically excluded.

There are two adjunct capabilities provided, which are of frequent use. Data may be more meaningful when they are viewed in particular sequence. Users of APC may sort the data by a specification such as

```
SORT OPBAL, NAME
```

which will cause the data first to be sequenced by NAME, then by OPBAL.

The user may wish to take a quick sidelong glance at some of the data, without relinquishing his present selection—a tentative investigation. To do this, SHOW, with any selection predicate which may be used with SELECT, WIDEN, or NARROW, causes a new temporary selection to be made, that selection to be displayed as though with a PRINT ALL, then the prior selection restored.

**MANIPULATION OF DATA**

For educational use, APC simply provides for someone, the user or an instructor, to enter directly the data to be used by the package. Primarily this is done by the function SET, to be used in the style

```
SET 'CHECK'
```

which would cause the data name CHECK to be added to the names already in use (and to the names in ALL), which would ask whether the column is to hold numeric or character data, and which would ask the user to key in the values to go under that name. If no values are put in, the column is filled out with zeroes or blanks. The function WIPE is provided in case the effect of SET needs to be reversed.

Again for educational use, data already put in may need to be extended. The function AUGMENT is provided for this purpose. Similarly, the function CHANGE is provided to allow the user to alter individual values among the data in use.

The user will often need to derive new values arithmetically from old. The function RECALC is used to instigate this process, and the functions INTO MINUS OVER PCT PLUS and TIMES are provided (syntactically similar to LT EQ and so on) to allow appropriate calculations to be specified.

To check that OPBAL plus DEBIT is equal to CLBAL plus CREDIT, the student could

\[ \text{SET 'CHECK'} \]
\[ \text{RECALC CHECK} \]
\[ (\text{CLBAL PLUS CREDIT}) \text{MINUS (OPBAL PLUS DEBIT)} \]
\[ \text{SELECT CHECK NE ZERO} \]

and SELECT would report how many rows were then selected. PRINT could be used to look at selected rows.

**ERROR INSENSITIVITY**

The user of APC expresses the actions he needs carried out by combining words which are either the names of functions provided in the package, or the names of data being manipulated by the package.

Therefore, assuming that almost all coding errors within the functions have been eliminated through usage, most errors made by the user in specifying the action he wants carried out will be picked up by the APL interpretative system's syntax checker at the level of the user's action specification.

In this case an error message will be issued with a caret pointing to that part of the user expression in error. This is a property of the interpretative system.

In most ordinary APL usage, errors occur within user-defined functions, and the APL system suspends interpretation at the very point of error to allow corrections to be made and interpretation to be resumed. This naive APL user cannot cope with this suspension — too much knowledge of the system is needed to allow such correction and resumption.

In the case of APC, because the error is typically detected at the outermost level of interpretation, interpretation is not suspended, and the user is straightforwardly able to get going again — just by keying in another action specification.

**IMPLEMENTATING THE PACKAGE**

Behind the simplicity of the design described above lurk many details of implementation which are not relevant to the student user of APC, but which explain how the package was put together and how it might be extended.

**USE OF NATURAL EXPRESSIONS**

The expressions which the user can compose resemble English imperative sentences because the underlying APL syntax resembles English syntax.

In the examples above, SELECT and PRINT can be used as imperative verbs because, as provided in the auditing package, SELECT and PRINT are the names of monadic (single argument) functions whose scope of application is everything to their right in the action specifications in which they appear, just as in English the predicate of an imperative verb is everything following it in the sentence.

Similarly, GT and AND can be used as conjunctions because, as provided in the auditing package, they are the names of dyadic (double argument) functions following the APL convention of having one argument to the left of the function name, and the other to the right.

The names like CUST and CREDIT, which the user chooses, can be used like nouns because as names of data they stand alone, without argument. They indirectly name the data on which the user wants to work, and can appropriately be used within predicates for SELECT or PRINT.

**OBVIOUS ARRANGEMENT OF DATA**

There is a clear benefit from providing for the APC user to choose and specify commonplace names as shown in the example

\[ \text{PRINT CUST,NAME,CREDIT,CLBAL} \]

but use of such names cannot directly refer to the data to be used as these commonplace names will be used with somewhat different significances from time to time.

When used with PRINT, a name should lead APC only to data presently selected, but when used with functions such as GT in for example data selection, a name should lead the package to all data, not just that already selected.

In APC names such as CUST NAME CREDIT and CLBAL therefore stand for the character string which is the actual name under which each corresponding column or category of data is kept by the APL system. In evaluation, functions such as PRINT and GT can select rows of data or not as appropriate, since, when invoked by the user, they are supplied the name of the data, not the data themselves.

The APL notation provides a primitive function to evaluate such a reference, and this primitive function is used within the defined functions of APC.

In the package as implemented, the name under which any data is actually kept is the name chosen by the user with a single underlined letter \( V \) prefixed to it. For instance, in the example used just above, VCUST names the vector of customer numbers which are the actual data which APC gets indirectly from the commonplace name CUST. CUST does not directly name the vector of customer numbers. For most package users this effectively hides the actual data name, in the example VCUST, as underlined letters are not easy to enter from their keyboard.

The value of the commonplace name, the one which is not hidden from the user, in the example CUST, is then a column matrix holding the character string which is the hidden actual data name.

The function SET, which is the only function through which the user can bring new data names into use, makes sure that all the column matrices are kept to the same length, padding with blanks if necessary, so that APL catenation (\( , \)) can be used to group names, as in the example PRINT NAME,MONEY explained above. SET also maintains a variable called ALL as a matrix holding in...
its columns all the hidden data names in use, so that the user can simply specify
PRINT ALL
to have all columns of selected data shown to him.

**SELECTION OF DATA**

The user of the APC will normally change his selection of data rows from time to time. A vector of indexes to the current selection is kept hidden from the ordinary user under the name S so that functions such as PRINT, which must select data rows, can use the current selection as a global parameter.

The function SELECT will replace the current selection vector S with new indexes, while the functions NARROW and WIDEN will modify S. Incidentally, the function names NARROW and WIDEN were purposefully chosen rather than the conventional computing terms AND and OR because AND and OR, however familiar they are to the programmer or logician, are not familiar at least in their logical meaning to the typical audit student.

The comparative functions such as GT and EQ, and the discriminating functions such as ANY LOW and NEAR, produce a boolean vector to allow SELECT, NARROW and WIDEN to replace or modify S appropriately. If new comparative or discriminating functions are needed, these are fairly simple to write using the existing functions as models, and are easy to add to the package.

**INSPECTION OF DATA**

Two kinds of inspection function are provided by APC. On the one hand, PRINT and SHOW display selected rows of data to the user. On the other hand, TOTAL and AVG derive new data from the selected data, and present the derived data to the user.

There would be a need to derive data more complex than simple totals or averages. Such derivation would typically use some statistical algorithm or other. These algorithms would not normally be appropriate to basic EDP audit education, but for advanced training complex statistical derivations could easily be added to APC using TOTAL and AVG as a model.

Students have trouble distinguishing between the capabilities provided by PRINT and by SHOW. PRINT takes names of columns as predicate, and displays rows selected by S, while SHOW takes a selection specification as predicate and displays all columns in use and rows temporarily selected as specified by the predicate. The distinction is difficult even to describe.

The confusion seems to arise from the similarity in functions and from the similarity of the ordinary meanings of the names PRINT and SHOW. PRINT and SHOW could be merged into one function, with the action to be carried out determined by a test of the kind of predicate used. Such a merger would not be difficult for a coder with modest skills in APL.

The function SORT is intended to be used as a prelude to PRINT. As implemented, it acts on all rows of data, and so resets the selection effectively to ALL. Sometimes the user expects his prior selection to remain undisturbed, which suggests that either only the user should be able to reset his selection, or that two varieties of SORT should be provided, one which resets selection and one which doesn't.

**MANIPULATION OF DATA**

APC was developed without any interface to external data. On the one hand this kept the package and its use simple, and on the other hand this required the test and instructional data to be loaded fairly laboriously before use.

For advanced educational use, and for any case study work, some facility for bringing selected data in from external files is needed.

While APL systems are notationally extremely uniform, their interfaces to the host operating system vary, often greatly, from host system to host system. Therefore, APL facilities for importing data for APC use will need to be specific to the host system. However, it is easy in APL systems to separate completely the data importing component from APC, and to use the APL systems to copy imported data into an APC workspace from the importing APL workspace.

If such an approach is taken, writing an APC data import facility would be fairly easy for a skilled APL programmer though difficult for a novice.

**ERROR INSENSITIVITY**

Probably the weakest aspect of APC as it stands is in its handling of errors, or more properly in its leaving the user unaware of the error handling to the APL system. Although recovering from an error is usually very simple for the panic-free user, the message produced when the error is committed is typically useless for explaining to the APC user in his terms what has caused the error. For instance, if the user in RECALCulating values forgets to use the name PLUS and quite naturally uses the symbol + instead, he is confronted with the laconic message DOMAIN ERROR.

The best way to overcome this problem is to write a "front-end" to APC through which the user passes his action specifications to APC proper.

A simple front-end, able to pick up most errors, would only take a day or two for a skilled APL programmer to write. More effort would allow errors such as using + for PLUS and using lower case letters instead of upper case to be corrected. However, as the front-end becomes more complex, producing the documentation and maintaining both it and the code would soon become a major task.

Development of a front-end is greatly aided by such features of APL systems as the "latent expression" which allows the front-end to use automatically indexed, and "system" functions which, for instance, determine the class of a name — whether it is a proper name and if so, whether it names a function or ordinary data.

**CONCLUSIONS**

The creation of a prototype EDP audit package for classroom use was not a new or novel idea. However the criteria which had to be observed, that is, microcomputer orientation, general application, economical on resources, easy to use — had not been evident in other packages available.

The general experience gained in its classroom exposure has been that more functions are needed to take students deeper into more specialised areas of EDP audit. Nevertheless, as a first introduction it has conveyed EDP audit concepts well to students.

**REFERENCES AND BIBLIOGRAPHY**


**BIOPGRAPHICAL NOTES**

Neville Holmes is Senior Systems Engineer, Corporate and Scientific Programs, IBM Australia Limited, and Honorary Associate, School of Computing Sciences, NSW Institute of Technology. After graduating from Melbourne University in engineering, and spending some time in the Patents Office, he joined IBM in Melbourne in 1959, and was a foundation member of the Victorian Computer Society. After working in the automotive and general manufacturing industries, he spent some years overseas, and in Canberra at IBM's Systems Development Institute. The work described in this paper was carried out while he was on secondment to the NSW Institute of Technology as Visiting Principal Lecturer and Head of the Department of Computing Sciences.

Philip Stanley is Senior Lecturer, Department of Information Systems, NSW Institute of Technology. Philip Stanley has been involved in EDP audit training since 1978 when he introduced the elective subject into the Computing Sciences degree course at the Institute. He also introduced the Computing Sciences submajor courses into the Bachelor of Business degree. He is a qualified accountant and was a director of the EDP Auditors Association. His special area of research is computer fraud and he lectures regularly to law enforcement officers on prevention and detection of computer fraud.

Short Communication:

The EDP Auditors Association in Australia

G. S. Breydon*

This paper provides an overview of the development and role of the EDP Auditors Association (EDPAA) in Australia over the last decade. It outlines the development and content of the Certified Information Systems Auditor (CISA) programme sponsored by the EDPAA and other professional development activities. The relationship of the EDPAA to other professional bodies in the audit, accounting and computing fields is also described including actions to co-ordinate and develop audit standards, education and academic research in EDP audit.

Keywords and phrases: Role and development of EDPAA in Australia; Certified Information Systems Auditor programme; auditing standards; privacy standards.


INTRODUCTION

One of the most significant developments in computer auditing in Australasia over the past decade has been a growing interest and awareness of the subject in the business community. The growth of the EDP Auditors Association (EDPAA) is one indicator of this increased awareness.

INTERNATIONAL BACKGROUND

The EDPAA was incorporated as a non-profit corporation in Los Angeles, California, USA in 1969. Since then it has experienced rapid expansion and growth as evidenced by a current international membership in excess of 6000 (Table 1). In 1972, principally for US taxation reasons, the association established a separate body, the EDP Auditors Foundation for Education and Research (EDPAF) to manage its research and educational programme.

AUSTRALIA

During the 1970's several small special interest groups and discussion groups concerned with audit and control of EDP systems operated within the Australian Computer Society and the Institute of Chartered Accountants in Australia. In 1976 the first Australian Chapter of the EDPAA was formed in Sydney and in 1979 that group, together with the Institute of Internal Auditors, organised the first annual Australian EDP Audit Conference. In 1980 a chapter was established in Melbourne, followed by a third chapter in Brisbane. Perth and Adelaide chapters were established during 1982 and groups of members also meet in Canberra and the Hunter Valley. The EDPAA has also formed chapters in Hong Kong, Singapore, Wellington and Auckland which, together with the Australian groups, comprise EDPAA Region 8.

Membership is unrestricted and comprises a mixture of persons from computing and accounting disciplines (Table 2). However, an analysis of Australian chapter boards reveals that over 60% of office bearers work for Chartered Accounting firms. Chiefly with funds generated from the annual conferences, which since EDPAC81 in Melbourne have alternated between the two major Australian cities, the local chapters have been able to sponsor a number of professional development activities in addition to the regular monthly technical meetings. These have included bringing to Australia P.J. Corum, Dr. Jerry Fitzgerald,

TABLE 1. EDPAA (International) Membership (July 1983)

<table>
<thead>
<tr>
<th>Region</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region 1-7 (USA)</td>
<td>5,067</td>
</tr>
<tr>
<td>Region 5 (Latin American)</td>
<td>36</td>
</tr>
<tr>
<td>Region 6 (Canada)</td>
<td>457</td>
</tr>
<tr>
<td>Region 8 (Australia/SE Asia) (note 1)</td>
<td>671</td>
</tr>
<tr>
<td>Region 9 (Europe)</td>
<td>281</td>
</tr>
<tr>
<td>Other</td>
<td>30</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>6,542</td>
</tr>
</tbody>
</table>

Notes:
1. Region 8 statistics exclude new members arising from a national series of PD workshops in June. These, together with further new members joining the second half of 1983 have resulted in a substantial subsequent increase for Region 8.
2. Discrepancy in Melbourne numbers between Tables 1 (200) and 2 (271) is due to timing difference. Table 1 analysis was extracted prior to international registration of all 1983 renewals.

SOURCE: EDPAA (1983b)
TABLE 2. EDPAA (Melbourne) Membership Profile (March 1983)

<table>
<thead>
<tr>
<th>By Industry</th>
<th>EDP</th>
<th>General</th>
<th>Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Accounting</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Government</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Education</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Financial, etc</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manufacturing, mining, etc.</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not known</td>
<td>94</td>
<td></td>
<td>271</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>By Activity</th>
<th>EDP</th>
<th>General</th>
<th>Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit — General</td>
<td>55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Audit — EDP</td>
<td>92</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other EDP</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other activities</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not known</td>
<td>56</td>
<td></td>
<td>271</td>
</tr>
</tbody>
</table>

Further Analysis of Audit Activity

<table>
<thead>
<tr>
<th>EDP</th>
<th>General</th>
<th>Totals</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>49</td>
<td>25</td>
</tr>
<tr>
<td>Internal</td>
<td>36</td>
<td>23</td>
</tr>
<tr>
<td>Government</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Totals</td>
<td>92</td>
<td>55</td>
</tr>
</tbody>
</table>

SOURCE: EDPAA (1983a)

Keagle Davies and David Fairbairn. In addition to local chapter newsletters members received the quarterly EDP Auditor Journal, a bi-monthly international newsletter, "the EDP Auditor Update", and "Control Objectives", a bound volume providing basic guidelines to assist in auditing or reviewing systems data processing functions. The Melbourne chapter provides annual prizes for EDP Audit Articles published in Australia and for essays by tertiary students. It has also offered funds to publish research monographs but has experienced difficulty in obtaining appropriate applications for this scheme.

THE CISA PROGRAMME

In June 1978 the EDPAF officially announced the inception of a certification programme for information systems auditors. The objectives of the programme were stated to be:

- The development and maintenance of a testing instrument which could be used to evaluate an individual's competence in conducting Information Systems Audits.
- To provide a mechanism for motivating and monitoring information systems.
- To aid top management in developing a sound Information Systems Audit Function by providing criteria for personnel selection and development.
- Initially certification was awarded under the Professional Experience Provision (PEP). A minimum of 5 years work experience in the field of auditing, data processing and Information Systems Auditing was needed to qualify for certification under PEP (EDPAF, 1978). The EDPAF terminated PEP on 30 June 1979 and announced all future certifications would only be awarded through the successful completion of a comprehensive examination and completion of a qualifying period of experience in EDP auditing.

The examination is developed, maintained and administered by Educational Testing Services of Princeton, New Jersey, based upon a survey and profile of Job Dimensions involved in EDP Auditing (EDPAF and ETS, 1981). The annual examination was administered in Australia for the first time in April 1982 with a pass rate of 65%. However, the administration of the exam internationally has not been without some difficulties and arrangements are now made for Spanish and Hebrew translations to aid presentation in some test centres.

The Certified Information System Auditor programme includes a compulsory continuing education requirement of 120 contact hours each three years to maintain the qualification (EDPAF 1983). There are currently approximately 4600 CISA's worldwide.

CO-ORDINATED COMPUTER EDUCATION

Several individuals active in the Australian chapters of the EDPAA have also been involved in various EDP related sub-committees of other bodies such as the Australian Society of Accountants and the Institute of Chartered Accountants. These organisations, together with the Institute of Internal Auditors and ACS are all involved to varying degrees in the provision of professional development courses in EDP concepts, control and audit. Each body tries to present a comprehensive range of courses but small internal markets for the advanced topics and clashes of dates have limited the success of such courses.

During 1982 the EDPAA initiated the formation of Co-ordinated Computer Education (CCE) committees in Victoria and NSW (Watts, 1983). Establishment of CCE committees in other states is in progress. Membership of the committees varies slightly but the EDPAA and ICA are represented in every case. The committees have begun some work in identifying existing courses, providing or refusing CCE endorsement for individual seminars, co-ordinating presentation dates and encouraging market growth through exchange of mailing services between member bodies. More ambitious projects, such as sponsoring courses to fill gaps in the current joint programme are also under consideration.

AUDITING AND PRIVACY STANDARDS

Being a relatively small organisation the EDPAA has not done a lot of direct work internationally to establish its own professional standards in the EDP audit and control sphere, other than setting the CISA code of ethics and publication of "Control Objectives". It has, however been very successful in the production and publishing of guidelines of appropriate practice, (for example, Perry, 1980).

In Australia the EDPAA has provided formal submissions to the Law Reform Commission on its privacy discussion paper (Hayes, R. et al., 1980), the specialisation sub-committee of the ICA and the Auditing standards board of the Australian Accounting Research Foundation concerning the exposure draft on Auditing in an EDP Environment (Pound, 1983). The ICA's specialisation sub-committee was established because of a concern by ICA members in specialist practice that the Institute was not adequately meeting their particular needs which was demonstrated by the devotion of their energies to specialist bodies such as the EDPAA.

GENERAL

Although the EDPAA has undertaken some work usually performed by a professional association the majority of its activities reflect its basic nature as a special interest group working through the existing professional bodies to influence and develop standards education and research.
in the audit and control of computerised systems. It has had some success in promoting awareness of EDP audit issues and requirements in Australia as evidenced by its membership growth, seminars and conferences. It has been a significant influence on professional development activities but its role in academic research in Australia is only just developing. Future activities of the EDPAA in Australia are likely to be influenced by the limitations of voluntary labour and the recommendations within the ICA to establish a specialist chapter in the area of Information Systems and EDP Technology which may divert resources from the EDPAA.

REFERENCES
EDPAF (1978): Certified Data Processing Auditor, A certification Program Sponsored by the EDP Auditors Foundation, EDP Auditors Foundation, Hanover Park, Ill, USA.
EDPAF (1983): The Continuing Education Program of the EDP Auditors Foundation, EDP Auditors Foundation, Carol Stream, Ill, USA.


BIOGRAPHICAL NOTE
Graeme Breydon has a BCom from the University of Melbourne. He is an Associate of the Institute of Chartered Accountants in Australia, and a member of the Australian Computer Society. He joined Coopers & Lybrand in 1972 and has held a variety of positions in Australian and US offices of the firm; currently that of National Computer Audit Research Manager. Graeme, who is a CISA was a director and chapter secretary of the EDPAA Melbourne chapter, a former editor of its newsletter and was associated with the organisation of the 1981 and 1983 national EDP audit conferences. He currently edits the EDPAA’s national magazine, “The Password”.

CALL FOR PAPERS
SPECIAL ISSUE ON EXPERT SYSTEMS

The Australia Computer Journal will publish a special issue on “Expert Systems” in February 1985. Research papers, tutorial articles and industry case studies on all aspects of the subject will be welcome, and both full papers and short communications will be considered.

Prospective authors should write as soon as possible to:
Dr J.R. Quinlan,
School of Computing Sciences,
New South Wales Institute of Technology,
PO Box 123, Broadway, NSW 2007
to notify him of their intention to submit material for the issue and provide a brief summary of their intended contribution.

In order to allow adequate time for refereeing and editorial review, complete manuscripts will be required by 28 September, 1984 at the latest.

Papers should be prepared in accordance with the guidelines published in the November 1982 issue of the Journal. Authors are requested to pay particular regard to the Journal’s preferred style for references.

Experiments with a Britton-Lee Data base machine on CSIRONET

H. G. Mackenzie,* D. M. Ryan* and J. L. Smith*

A small number of special purpose computers for database management are now becoming available. These are being offered as an alternative to mainframe software systems, operating as "back-end" machines. One of these, the Britton-Lee Intelligent Database Machine has been integrated into CSIRONET. Some of the experiments and experience gained are described here, and our impressions for the future stated.

Keywords and phrases: database machine, CSIRONET, database management.

CR categories: C.1, H.2.

INTRODUCTION

The focus of this paper is the Britton-Lee intelligent database machine (called the IDM), (see Britton-Lee, 1981). It is one of the first commercially available "back-end" computers dedicated to the functions of database management, and it was reported by Rauzino (1983) that 125 IDMs were installed and operating. One of these is installed at the CSIRO Division of Computing Research in Canberra and is accessible through CSIRONET, the CSIRO computer network which is regularly reported in CSIRONET News.

The IDM is placed physically and logically between one or more "front-end" (host) computers to which user access is available, and one or more disc storage devices. The host computers may be mainframes, minis, or any form of microprocessor down to personal computers. There are three major interfaces involved in this distributed computing arrangement, as depicted in Figure 1. The top interface is the one which impacts the user and this may take on a variety of forms, implemented in host software, and based on the relational database management system (dbms) of the IDM. The nature of the other two interfaces is completely specified in the IDM design. The host-IDM interface has a multi layer protocol with the highest level based on relational commands and data. The IDM-disc interface is compatible with the Control Data 9760 Storage Module Drive (see Control Data Corporation, 1981), and this interface accommodates a sizeable number of disc drives available from well known manufacturers.

Thus the major requirement for an organisation to exploit an IDM is the host software modules which provide the top two interfaces of Figure 1. Britton-Lee provide a limited variety of such software mainly for Digital Equipment Corporation computers (VAX and PDP-11) running VMS or UNIX operating systems; subsequently in this paper we will describe some portable developments of this kind in the CSIRONET environment. First we describe in some detail the functionality of the IDM and thereby indicate the requirements which must be met in host software.

PROFILE OF IDM DBMS

The IDM provides a host independent, multi-user, multi-database, relational dbms. The following sections describe this system according to those major areas of functionality expected of a modern mature dbms.

Data Model — Database Design and Definition

The only data model offered by the IDM is an imperfect form of the relational model (see 'Integrity' section below), enhanced with objects of an internal data model. The model must therefore be made to serve the functions of conceptual, external and internal schemas. As such it has shortcomings in all three roles, but it is state-of-the-art for commercial relational dbms. If other external data models were required, they would have to be implemented in host modules.

A few simple commands support the entire database definition process. Database implementation is reasonably flexible so that evolution of design throughout the life of a database can be relied upon.

Therefore the major mandatory requirement of the host is to provide the lexical, parsing and semantics phases for an implementation of the definition language. (This translation service must be provided by the host for every

---

Copyright ©1984, Australian Computer Society Inc. General permission to republish, but not for profit, all or part of this material is granted, provided that ACJ's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Australian Computer Society.

*CSIRO Division of Computing Research, GPO Box 1800, Canberra, ACT 2601.
†This paper is a revised version of a paper delivered at the Symposium on Data Base Management Systems, Sydney, November 15-17, 1983, organised by James Conran Pty Ltd. Manuscript received October, 1983; revised February, 1984.

function below where a user language is involved.) At the relational level this is a simple, well understood technique, with no effort required in code generation because the IDM’s encoded input form is still relational. Formatting of responses and results returned by the IDM must also be provided by the host for every function where user output is involved.

Data Dictionary/Directory

A basic dictionary/director is created by the IDM according to a predefined relational design, for every database. It records major dependencies which are used by the IDM to verify the legality of subsequent design modifications, and it is used to monitor and control the entire database operation according to accepted schema concepts. The most attractive feature is that database administrators and managers can manipulate this entire body of information with the same relational query/update language as is provided to users. Thus the data dictionary/directory can be extended with ease to cater to more sophisticated requirements.

Data Entry, Screens and Validation

No facilities are provided by the IDM in this area. However the stored command facility described below is well suited to adapting host data entry systems to the IDM.

Integrity

Apart from enforcing unique index keys, the IDM does not support the integrity rules of the relational model.

Host Languages

The translation and formatting functions mentioned above must be provided by the host system for any programming language in which the relational database facilities are to be embedded. The IDM provides a special interface command so that relational statements may be parameterised and blocked together at compile time, and the translated form is stored in the IDM for subsequent invocation at program execution time. Host support must also include mapping of the relational data into the external model of the particular programming language, and data flow control. The IDM supports all the common data formats used in host languages, but it does not attempt to interpret non-BCD floating point values. Its internal formats are ASCII and two’s complement arithmetic.

Interactive Query/Update

The bread and butter capability of the IDM lies in the powerful query/update language, normally expected of a relational system. (The same language statements would be used when embedded in a host language.) This level of language ensures a minimum of interchange between the host and the IDM to satisfy a user request. Higher level models and application specific interfaces may be supported in the host system. In these cases the stored command facility of the IDM provides a powerful implementation aid for interfaces which need to be particularly user-friendly.

The interactive stored command parallels the tool provided for the embedded host language interface. Once defined, as a sequence of relational statements (including transaction locking if required), a stored command can be invoked at any time with constants as parameter values. This means that for repeated transactions the traffic between the host and the IDM can be almost entirely restricted to the relevant data, and thus low bandwidth connections can be quite effective. The limits of the facility lie in the inability to create temporary relations during execution, only constant parameters, lack of any structured language features, and no nesting of commands.

Analysis and Report Generation

The IDM provides a few basic statistical functions in the relational language and a group-by facility which permits these functions to be repeated for sub groups in a retrieval. Apart from this the functions of report writing, data analysis and display must be provided in the host.

Commit, Logging and Recovery

The IDM provides a general transaction delineation facility with the option to commit or roll back. Transactions may be nested. An audit trail type of transaction log is maintained for every relation. Checkpoints are automatically made at specified intervals. Audit trails are retained only for those relations specified, otherwise they are destroyed at each checkpoint. Relations are automatically brought to a consistent state after a crash using the transaction logs if they are intact. The same log can be used to produce a report for an auditor by use of the relational language. Facilities are provided to dump transaction logs and databases to the host, or to another IDM database, or to an IDM tape drive. A load command then provides the basis for large scale recovery procedures.

Maintenance Utilities

In addition to physical dumping and loading of databases, bulk movement of formatted data in and out of relations is supported. This provides for the initial loading of databases from the host or from an IDM tape drive.

Security

Each database has its own independent security control, with ultimate security dependent on access by the database administrator of the IDM "system database". This access can be restricted through one host (e.g. a microprocessor located with the IDM and subject to lock and key security). The database administrator of the system database can retain the right to create all databases and nominate the database administrator of each database by specifying a host computer identifier and a host computer user identifier. Two hosts can simultaneously identify themselves to the IMD with the same identifier, and during non-overlapping connection sessions different host computers could use the same identifier. The database administrator is the only user who can delegate or withdraw access privileges. There are mechanisms to make the granularity of access permissions as fine as one desires, and to define groups of users and delegate to a group.

Concurrency Control

The IDM employs a straightforward two phase locking strategy for write locks, as well as read locks. Readers are deferred to permit a writer who has waited a certain time to proceed. The locking granularity is a 2K byte page. Deadlocks are detected and a transaction backed out with appropriate response to the host. Host co-operation is required to terminate transactions which are unduly delayed on the host side.
Experiments With a Database Machine

Performance
The access mechanism used to enhance retrieval performance is the B-tree index. These may be created or deleted dynamically, but secondary indexes will be dependent on a primary clustered index if it is used. The latter index corresponds to the key used to sort tuples on disk pages. Other placement controls are available to the database manager. A record of resource utilisation and other performance parameters is maintained by the IDM on its global operations.

Accounting
The IDM returns one accounting metric (the amount of time the IDM database processor was dedicated to the request) along with the results of each request.

Communication
There is a special protocol for host communication with the IDM, but the IDM does not support any recognised network protocol. Thus network communications must be managed by a host computer.

IDM HARDWARE
The encapsulation of the IDM is shown in Figure 2, being of comparable size (0.7m x 0.4m x 0.4m) to microprocessor systems. It consists of several different types of processors, as shown in Figure 3, on a common bus. The control processor which executes the dbms modules is Z8000 based, as is the serial I/O processor which supports eight R232 connections, each configurable at speeds up to 19.2K baud. The parallel I/O processor supports an IEEE 488 high speed channel. One disc controller can connect up to four disc drives and the total database address space is 32 gigabytes. The database accelerator represents the most advanced step towards special hardware for database manipulation in the architecture. This processor is optional but, if present, critical modules in the dbms which are mainly concerned with data selection criteria, are executed by this 10 Mip processor. Britton-Lee claim throughput performance improvement in the range 2 to 10 through use of an accelerator. The bus permits a maximum configuration of sixteen processor and memory modules.

IDM COMMUNICATION PROTOCOL
Any language which is chosen to be implemented in a front-end host must be translated to a relational language called IDL (intelligent data language, see Britton Lee, 1983) before transmission to the IDM. IDL is very similar to the language QUEL defined by Stonebraker (1976). The transmission format is a byte string representation of an IDL parse tree, with all language tokens represented by byte
encoded values. Byte strings (IDL statements and data) are transmitted between host and IDM using a packet message protocol.

Packets are transmitted following a request from the host, sent in a previous special communication packet. Communication packets usually contain a request to write or read a specified number of bytes. The IDM can exercise flow control by refusing a request. A data packet (containing either code or data) must follow a successful request. Each packet transmission must be acknowledged by the receiver. Packets are normally subject to error detection, with optional error correction, and negative acknowledgements are used to cause retransmission of the most recent packet in case of errors. The maximum size of a data packet is 2K bytes.

The host may choose any one of three different classes of read/write protocol for an exchange with the IDM.

1. Wait until the request can be granted by the IDM.
2. Not wait, but resubmit the request later.
3. Not wait, and ask the IDM to interrupt the host when the request can be granted.

The first two classes have the advantage that the host always knows the length and direction of the next packet to move between the two computers. In the third a packet of known length may arrive asynchronously from the IDM.

Because a data packet exchange must follow a successful communication packet exchange, the host must exercise this protocol from a single control process, or else employ a locking mechanism for IDM access.

**CSIRONET IDM INTEGRATION**

In this section the major developments undertaken in order to test the IDM potential in CSIRONET are described. The major constraint which was imposed on our experiments was that the host computers and the network communication protocols were determined by the existing CSIRONET environment. Apart from the purchase of one additional disc drive no additional hardware has been involved. Except for running cables to the various host computers involved, the installation and running up of the IDM was no more involved than the installation of a microprocessor. For convenience of connections we elected to mount the IDM in a rack along with other network equipment in one of our computer halls.

The IDM configuration installed is a minimal one, consisting of a database processor, one serial I/O processor, one parallel I/O processor, 0.5 Mbyte of memory, timing and bus control, and one disc controller. All host connections to date have been made through the low speed serial I/O processor using a maximum speed of 9600 baud.

The software developments to be described here largely concern an interactive dbms facility available through three different types of hosts, one mainframe, one minicomputer and one microprocessor, with the microprocessor software implementation aimed to be readily portable. The user language is an extension of IDL, defined for the purpose of the exercise. All implementations conform to a uniform multi-layer design. They use a common implementation of the parser and other modules. The main host-IDM connections with which our work has been concerned to date are shown in Figure 4. All software was written in PASCAL and compiled under the compilers available on CSIRONET for the respective machines.

The major protocol layers for a host-IDM connection
are shown in Figure 5. These are described in terms of the ISO OSI architecture layers (see Tanenbaum, 1981). This is a compressed view, and IDM packets would be transmitted by a session services protocol layer if a remote host system was connected by CSIRONET to the IDM (see comments on the user of CNIO below). A more detailed description of the top layers is as follows.

Application Layer
- Level 5 — Knows about relational tuples and IDL (parser and formatter).
- Level 4 — Knows about IDL encodings and tokens (translator).
- Level 3 — Knows about cursors (streams to an IDM database), and issues reads and writes of IDM messages.

Presentation Layer
- Level 2 — Knows IDM packet communication protocol.

Data Link
- Level 1 — Knows how to send and receive bytes.

Cyber 835 NOS Interface
This interface has been designed for a Control Data Corporation Cyber 835 host computer running the Network Operating System (NOS) Version 2 (see Control Data Corporation, 1983a). The implementation modules (UCP and IDF) are shown in Figure 6 along with the major system modules employed (NAM and IAF).

The application layer protocol is implemented one process per user, with no use of shared code permitted by NOS. These processes may support either a self contained language or a host language application and they may be run either interactively or in batch mode. Each process may have many cursors open at the same time. This corresponds to having many databases, possibly on different IDM's, open, or to having the same database open many times.

The presentation layer protocol is implemented in a specially installed module (Control Data Corporation, 1983b) which we called IDF. This module receives requests using the standard NOS subsystem request method. Bulk data handling utilities work by IDF transferring a file assignment to itself from a requesting UCP. IDF then spools the file to/from the relevant IDM, in such a way that normal read/write requests from interactive and batch users always have priority over spooling operations.

Lower level protocols are supported by a suite of system software broadly called Network Access Method (NAM) (Control Data Corporation, 1983c). The NAM transparent transmission mode was used because all byte values may possibly occur as IDM tokens. The host knowing the length of the next input message was put to advantage, in that preceding a write to the IDM the length of the subsequent read was set. This eliminated the real time overhead which would be incurred in using time outs.

PDP11 RSX11M Interface
This interface is implemented on a Digital Equipment Corporation PDP11/34 computer running the RSX11M operating system (Digital Equipment Corporation, 1979). It is dependent on a high level CSIRONET communication protocol called Connected Input Output (CNIO) (Paine, 1980). The implementation modules are shown in Figure 7.
The application and presentation layers of the IDM protocol are implemented in a multi-user task called the IDM process. It only supports interactive use. CNIO provides session services for CSIRONET terminals connected to the IDM process. The IDM process also uses CNIO session services to invoke the serial I/O to the IDM. The juxtaposition of the IDM process with respect to CNIO means that this implementation could be moved to a remote CSIRONET host (Smith and Paine, 1982).

This design was motivated somewhat by expediency in this the first implementation undertaken on CSIRONET, and it has some limitations. For example, in order to allow a range of IDM application systems to exist on this host it would be necessary to implement a single presentation layer process which could communicate with several application level processes. However the current role of this type of host system on CSIRONET is such that further developments are not envisaged.

A disadvantage arising from the use of CNIO for terminal sessions is that PASCAL I/O cannot be used. Hence special routines had to be written for formatted output.

**WICAT Microprocessor Interface**

This interface (see Figure 8) is implemented on WICAT 200 and WICAT 150 systems running Multi-User Control System (MCS) (WICAT Systems Inc., 1983). While the interface has a lot in common with the previous two, the older hardware and operating systems and the network protocol employed in the latter mean that those implementations cannot be seen as truly portable. This implementation has portability as a goal, and the software has already been ported to an IBM PC. The system is also seen as the root system for the standard components, such as the parser, which run on the other implementations. In addition the WICAT systems, having a modern operating system and being ASCII hosts, offer a much needed development facility still quite rare on CSIRONET. Of course it is this commonality in microprocessor systems which contributes more than anything to the possibility of a portable IDM facility.

The basic strategy is to use a standard set of PASCAL facilities. In addition a few commonly available operating system facilities are required. These involve opening and closing files with special characteristics, and associated I/O operations with both timeouts and unstructured data. The modules involving specialised needs have to be isolated into a small section of code which can be reimplemented for each different host processor that is targeted.

There is one IDM process per terminal user, all using shared code. The presentation level enforces a locking protocol for each IDM interchange by opening the serial port with exclusive access rights and closing the port at the end of the interchange.

**PERFORMANCE**

To date the major part of our effort has been concerned with integrating the IDM into CSIRONET. Our use of the IDM has been largely for demonstration, which is available through CSIRONET at all major locations in Australia and through a gateway to TYMNET in the US. Only one database containing real data of any size has been used for tests. (It is a simple matter to create artificial databases of any size.) We have yet to run a designed set of performance tests. Before doing so it would be desirable to upgrade the configuration to accommodate a reasonable number of concurrent users. This implies expanded memory and a later revision of the serial I/O processor. In this paper we can only quote a few independent measurements without substantial context.

A small database consisting of one relation and approximately 7 Mbyte of real data in 52000 tuples was loaded. The relation consisted of 25 attributes, all but one of which were variable length character strings. The load took place from the Cyber 835 using the IDM bulk load utility. The serial I/O channel employed operated at 9600 baud or about 1 Kbyte/sec. Packet sizes of 2 Kbyte were used and an effective transfer rate of 0.5 Kbyte/sec was achieved under NOS/NAM (see Figure 6). Investigation showed that the reduction from maximum channel transfer rate was due to the overhead in NAM calls implementing the IDM protocol. As NAM and IDM both limit packet sizes to 2 Kbytes no improvement is possible through increased packet size. Therefore, unless tuning of NAM is possible, faster IDM channels would have little effect on the net data transfer rate in the current interface design. One solution involving additional hardware would be to use an intelligent controller to implement the IDM protocol for the connection, thereby reducing the NAM calls to one per packet transfer. Observed NAM response times would then permit a theoretical bandwidth of 4 Kbyte/sec. On the IDM side the performance limit was database processor time for which the processing rate for loading data into the relation was 6 Kbyte/CPU-sec. Only about ten per cent of this time was spent in functions which could have been performed by an accelerator (these measurements are provided by the IDM). This processing rate is quite dependent on the complexity of relational tuples, an example binary relation tested was processed at 15 Kbyte/CPU-sec.

Physical dumping and loading of databases requires less IDM processing. When data was moved between IDM and host the database processor was used at a rate of 28 Kbyte/CPU-sec. When the data transfer was between disc files under the control of the IDM the usage rate was 55 Kbyte/CPU-sec. In the later case about one third of the time was spent in functions that could have been performed by an accelerator. In this test the database was dumped from one area of a disc to another area on the same disc and the actual data transfer rate was 15 Kbyte/sec. When the areas were on different discs (thereby avoiding a lot of disc seeks) the rate was 38Kbyte/sec.

Creation of a clustered index on a relation causes the tuples to be sorted into disc pages in index key order. The 52000 tuple relation described above was already in sort order when this command was tested. The elapsed time for index creation was 21 minutes, without any other significant use being made of the IDM. The database processor utilisation was 720 CPU-sec of which one third could have been performed in an accelerator. There were 22000 disc reads and 15000 disc writers, so that this operation might well have benefited from more IDM memory. A non-clustered (no sorting required on the data tuples) index was created on the same relation in an elapsed time of 15 minutes, with approximately the same database processor utilisation of which about 40% of the time was spent in accelerator functions.

Random retrievals based on an indexed key received good interactive response, with a database processor utilisation of about 0.3 seconds. A join to a similar indexed relation containing 19000 tuples required 0.5 secs database processor time for a unique key value, and this also gave...
good interactive response. When this join was specified as the qualification on a function which counted those 19000 tuples for which there was a key match in the larger relation, the database processor time was 275 seconds and the elapsed time approximately 360 seconds. In this case there was more than 50% of the time was spent in accelerator functions.

We have not identified those relational operations which would achieve the most gain from an accelerator. One operation which revealed a possible gain of six times the speed involved the join of a particular relation with itself when no index existed on the join attributes. However the use of a merge algorithm might have achieved better gains (see Upham, 1983).

The most important performance question concerns the overall reliability of the IDM. We have experienced two hardware failures in over twelve months of installed operation, one in the power supply unit and one in the disc controller. The IDM senses the AC power level and does not begin a disc write unless the supply is within tolerance, thus ensuring that once a disc write has begun it will have the necessary power to complete. When AC power is restored after a failure there is an automatic restart. Restart involves running the recovery program against all databases and any incomplete transactions are backed out. While we have had quite limited database activity, there have been hundreds of restarts for various reasons, and only once have we failed to recover a database in this way.

While our tests have by no means been exhaustive, the IDM system reliability we have experienced suggests that it may well be reliable enough for many database requirements. The most common cause of failure has been a lack of robustness in the IDM software. When our own host software systems have not conformed to IDM communication protocol (most commonly by generating incorrect IDL code) the IDM has frequently failed and required an operator initiated restart.

DISCUSSION

In this section we will discuss the potential of the IDM, drawing on our somewhat limited experience. Rauzino (1983) sees the IDM as a leading contender in the US amongst the small number of current commercial offerings which are aiming to use distributed computing and special processors in database management. The IDM design is very conservative in terms of its departure into new special purpose hardware. The design is conventional and uses well established LSI technology. The exception is the database accelerator which uses very high speed logic for executing critical software modules, notably those associated with database search and selection. While we understand that some tens of installations have database accelerators, we can offer no first hand evidence on their performance or reliability.

The IDM claims to offer a relational database management system and it meets this claim better than most so called relational software products. However the system is not “fully relational” as defined by Codd (1979). In particular it does not support the relational structure from a semantic integrity viewpoint, nor does it recognise or support most of the basic insert-update-delete rules pertaining to integrity. Furthermore there are no explicit null values. These are functions which should be implemented in a database machine.

The success of any database machine will be critically dependent on its communication capabilities. The designers of the IDM have addressed this point seriously, but unfortunately with a good deal of independence, in that they have proclaimed their own IDM communication protocol. In the future one would hope to see the IDM support standard protocols such as X.25 and ethernet. (We understand the latter product has been announced.) Until then a network interface processor is required to make the IDM a network resource.

The concept of a database computer offering shared databases, at the level of the relational model, to a collection of non-homogeneous computers might well be ranked with other such panaceas. But this one has been demonstrated in our CSIRONET IDM experiment, with diverse hosts. The major limitations to be recognised are:

(i) some dbms functions must still be performed in the host
(ii) the lack of suitable hardware interfaces to the high speed parallel channel for some hosts
(iii) the overhead involved in using general purpose access methods and communication protocols for each database transaction
(iv) the moderate amount of processor power in an IDM configuration.

The critical dbms functions which must be made up in at least one host are those pertaining to data entry and screen formatting for on-line database transactions, and the report writer functions.

These current performance characteristics give strong pointers to the type of environment which the IDM is likely to first impact. Hardware is readily available to use the IEEE 488 Channel with DEC and Hewlett-Packard hosts and other microprocessor manufacturers have announced support for this type of I/O channel. (Britton-Lee have selected DEC as one target for their already announced software developments.) The prospect for a cheap local network incorporating a central database is one particularly worth mentioning. Applications of relational database in this context are undoubtedly abundant. One of our experiments involved the implementation of a mail system. Using a design based on a few relations and stored commands, a surprisingly effective system was produced in about one man day! It is clear that with some complementary host software a powerful local network of microprocessors rang­down to personal computers can be established using RS232 technology and an IDM. Such a network would be able to span a half a building and provide a suitably powerful central database resource.

A useful feature of the IDL relational language is the pattern matching and string searching facilities which can be used on character strings. Unfortunately the maximum length for character string attribute values is 255 characters, and without text inversion one could not consider full text retrieval applications. Nevertheless the facility is an interesting move towards a hybrid capability which could be further developed, particularly with more special purpose hardware.

Some experts argue that special purpose database hardware is not cost effective and they do not foresee any move away from general purpose computers. This outlook will undoubtedly be favoured in the short term by the strategies of the major hardware and software manufacturers. However the evolutionary path of computing systems over a few decades suggests to us that the move to distributed special processors is gathering momentum.
Database management is a prime candidate because of the problems with implementing database systems with traditional operating systems (see Tanenbaum and Mullender, 1982), the demands for better performance and the potential to answer these with parallel hardware.

Database machines have been the subject of research for at least ten years. A number of efforts have concentrated on parallel hardware (for example the RAP associative processor, Ozkarahan et al, 1975). Such experiments were radical departures in processor design and it is not surprising that they have yet to make an impact on the technology. However the advent of readily available LSI components and the use of firmware have put the special purpose processor and the general purpose processor on a more equal footing.

To suggest that the buyer of a dbms should now have to fathom the technicalities of a database computer would be naive. It is clear that to be accepted in the market place this new form of distributed computing will have to be fully engineered to complement the front-end host and communication systems. In our view this will occur, and established dbms systems will survive only in name. It will also be possible to remove barriers which currently partition information systems, an obvious move being to integrate the management of full text with formatted data. Undoubtedly better distributed database technology will also arise at the same time. The development of high level interfaces to database computers sets the scene for rapid advances in these types of processor, without any further impact on the front-end host systems.

The IDM is a conservative move in database machines which has been proven to some degree. To survive in this highly competitive industry it must evolve further. It begs the involvement of OEMs who are prepared to take a risk. This should not exclude Australian groups interested in developing and marketing new technology. The obvious opportunity is in producing portable host software to complement the IDM capabilities.

ACKNOWLEDGEMENT

We wish to acknowledge the contribution of Michael Murray to the implementation of the parser, and John Ophel for porting software to the Cyber 835.

REFERENCES


Control Data Corporation (1983b): NOS Version 2 Main System Maintenance, Minneapolis, USA.


CSIRONET News: Newsletter of the CSIRO Division of Computing Research, published bimonthly, GPO Box 1800, Canberra, ACT 2601, Australia.


BIOGRAPHICAL NOTES

Hugh Mackenzie has been with the CSIRO Division of Computing Research since 1968. During this time he has worked on the design and implementation of database management software. Currently he is working on expert systems, and the application of formal logic to database access. He received a B.Sc. from the University of Sydney in 1966 and an M.Sc. in Computer Science from the ANU in 1979.

Doug Ryan graduated in Computing Science from the University of Adelaide in 1968. He was employed at the Weapons Research Establishment, SA and the University of Adelaide Computing Centre before joining CSIRO in late 1970. His work in CSIRO has varied from consulting on computer user projects to the development and maintenance of both operating systems and database systems.

John L. Smith is leader of the Information Systems Section of the Division of Computing Research, CSIRO, Canberra. He received the M.Eng.Sc. degree in electrical engineering from the University of Sydney in 1961 and the PhD degree from the University of Michigan in 1967. He has worked as an assistant research engineer at the Cooley Laboratory of University of Michigan between 1963 and 1967, and as an information scientist at the General Electric Research and Development Centre Schenectady, NY until 1969, when he joined the CSIRO Division of Computing Research.
COMMENTS ON THE PAPER BY
THOM AND THORNE

The paper of J.A. Thom and P.G. Thorne (1983) on Privacy Legislation and the Right of Access gives a distorted view of the aims and capabilities of Privacy Legislation. The earlier work did not pretend that a single rule could be stated that would cover all requirements. The principles enunciated in the OECD documents and repeated in NSW Privacy Committee and ALRC reports should be treated as an indivisible set of rules. Taking the right of access principle in isolation can lead to erroneous conclusions that some aspects are not covered.

In particular, discussions in the late sixties and early seventies, for example Niblett (1971), clearly identified concern about deducible facts (such as the salary of the person on extension 1234), being able to get around allowed access rules, and about cross-correlation of databases. The writer, in a private communication to the ALRC, also raised the point that the indivisibility of the Crown should be limited not to allow such cross-matching of Governmental files collected for some other purpose. Such correlation of data is contrary to the principle (one of the set) of no use beyond the purpose for which data was gathered and to the principle of not collecting data superfluous to the purpose.

Collection principles are discussed in the ALRC Report 22 (1983), Volume 2, Paragraphs 1209-1221. This report recognises that these principles apply to deliberate collection, with protection for the record-subject in the case of other acquisitions being provided by a combination of right of access, right of challenge and principles governing usage.

The principle of non-disclosure may also prevent the introduction of other data and the drawing of inferences where the second set of data was provided by a third party (who may have legitimately collected it from the subject). Non-disclosure and use for a particular purpose has received consideration in the question of anonymisation of data collected for research as discussed by the 1973 Stockholm Conference on "User Techniques to Prevent the Invasion of Privacy" and given a technical base in the work of Horvitz (1973). The question is also discussed in Paragraph 17 of the UK White Paper (1975) on Computers and Privacy, and in Chapter 7 of the French Report on Informatique et Libertes.

The set of principles listed in Part II of the Schedule attached to the Draft Privacy Bill (ALRC 22 Vol. 2, pp. 265-6) essentially encompasses the OECD principles and form a set which together avoid the problems suggested by Thom and Thorne.

Researchers would also be well advised to study the US Senate Report (1975) on Privacy Developments in Europe prepared by G.R. Pipe as this contains a summarisation of earlier broad discussions out of which later particular solutions have arisen.

L.G. Lawrence,
18 Carlotta Avenue,
Gordon, NSW 2072


UNITED STATES SENATE (1975): Privacy and Protection of Personal Information in Europe.

AUTHORS' REPLY

Dr. Lawrence's main points of criticism appear to be:

(a) that we give "a distorted view of the aims and capabilities of Privacy Legislation" because we do not treat the principles "as an indivisible set of rules" and looked at the right of access in isolation;

(b) that the issues we raise regarding derivable facts were already identified as long ago as 1971;

(c) that the existing set of principles are adequate and avoid the problems raised in our paper.

We wish to refute all these claims.

(a) We deliberately concentrated on "the right of access" principle, because it is "generally regarded as perhaps the most important privacy protection safeguard" (Australian Law Reform Commission, 1980). However we did state that "although the right of access is an important principle, it should be remembered that it is only one of a number of rights which attempt to give the individual control over his or her personal information" (Thom and Thorne, 1983, p. 146).

A number of other privacy protection principles either require or assume the operation of an effective right of access. If all the information which may be related to an individual is not accessible as part of a right of access then these other principles which depend on an effective right of access, become harder to enforce.

Furthermore, the definitional problems related to implicit or inferable information are no less relevant to the definition of, and enforcement of, other principles which also refer to personal information.

(b) We agree that many others in the past have referred in a general way to the dangers to personal privacy raised by "the ability of the computer to reorganise information it stores, to evaluate it in novel ways, [and] to use sophisticated techniques of association and correlation" (Niblett, 1971, p. 18). Although this general problem was raised, as Dr. Lawrence points out, over a decade ago, there appears no evidence that privacy legislation, enacted or proposed since that date, has substantially addressed this problem.

Controls (based on the principles concerning the purpose of collection) which limit cross-correlation of more than one database neither cover the situation of implicit information within one database, nor cover the situation raised by Clarke and Greenleaf (1984) concerning free text retrieval.

(c) The definition of personal information is fundamental to considerations of access rights, requirements of relevance, and obligations to notify the data-subject. Thus the question as to whether or not the definition should embrace implicit or inferable information is fundamental to

the effectiveness of what Dr Lawrence describes as "an indivisible set of rules". For instance, notification of the data subject of implicit information stored about them, may be, in many cases, an infeasible or unreasonable requirement. Yet this implicit information may be of no less importance to the privacy of the data subject than information contained in an explicit record.

Certain assumptions appear to underlie much of existing discussion in this area. The first of these is that information privacy issues only emerge in relation to purpose built databases, separate from each other. It is our concern that not all databases fit this primitive stereotype.

Our paper drew attention to the implications of deductive databases; Clarke and Greenleaf (1984) extend this by drawing attention to further similar problems raised by free text retrieval systems. In these systems the interrelation of separate data items may occur within a single database. These interconnections between items may not be predictable a priori, but may emerge as the consequence of specific, unique queries. Thus the range of implicit information may be indefinitely large and unpredictable unless all conceivable queries are exercised over the whole of the stored data – a task which would usually be impossible, especially in a dynamic system.

The problems of interrelation within one database go much deeper than those concerning cross-correlation between different databases referred to by Dr Lawrence. He asserts that the principles restricting data use to purposes stated prior to collection and to the restrictions on the collection of superfluous data provide a sufficient safeguard against cross-correlation.

A second assumption appears to be that the conduct and control of a personal information system may be confidently vested in a responsible body which can oversee the input, storage and access of records and ensure compliance with legitimate procedures and practices.

In reality, databases may be established with very broad objects thus making the principles relating to purposes of collection and use meaningless. Furthermore, the growing availability of large-scale public access databases greatly increases the range of users who may construct queries. These users may owe no duty, legal or fiduciary, to the database owner or administrator. The responsibility of maintaining reliable data and informing the data subject may well become diffuse in such circumstances.

It is therefore not clear that the present principles of the Australian Law Reform Commission (1983), taken as a set or individually, will provide an effective privacy protection in such circumstances. Nor is there any significant evidence that the issues we raise have been effectively addressed in the formulation of these or earlier principles. These principles appear to have remained effectively unaltered since privacy legislation was first contemplated as a consequence of the emergence of primitive databases.

P.G. Thorne,
University of Melbourne,
Parkville, Vic 3052

References


FROM THE EDITOR

In the May issue, I published an appeal for a copy of Volume One, Number Three of this Journal in order to complete the editor's set of back issues. I am pleased to relate that some original members of the Australian Computer Society still read the Journal, and that the appeal has been successful. Thanks to the generosity of P.A. McFarlane of Box Hill, Victoria, the set has now been completed and will soon be bound. Thank you, Mr McFarlane.

This is the sixth issue of the Journal that has been designated as a special issue on a particular theme. I am certain that these special issues have helped to strengthen the Journal and to focus its interests. Before planning more such issues, I would like to seek readers' views on special issues in general, and on likely topical areas in particular. Your suggestions and comments will be most welcome.

John Lions


The Software Catalog represents an ambitious undertaking to collect information about packaged software. The full catalog (sic) is published twice a year in two parts: minicomputers and microcomputers. An update volume for each part is published between each full edition, although the Spring 1983 updates are combined in a single volume. The catalog is derived from the International Software Database which is accessible via the Lockheed DIALOG system.

The catalog is organized primarily by vendor with indexes by computer system, microprocessor (where applicable), operating system, source programming language, application area, package name and keywords. It is not obvious why separate listings for minicomputers and microcomputers are published. Information is duplicated, and an arbitrary boundary based on system price has been introduced. I feel that it would be more convenient to have a single listing. This could be split into two volumes if necessary, with the primary listing in one and the indexes in another. This form of organization would assist cross-referencing, as the index could be left open at an appropriate page while the primary listing was consulted.

Cross-referencing is via so-called International Standard Program Numbers. These seem unduly restrictive with only eight digits: five for the vendor and three for the vendor specific package. Already, in the Spring 1983 update, one vendor has one thousand programs listed which is the maximum permitted under this scheme. If the Software Catalog does achieve success, a change in the package numbering appears necessary. As with any general reference source, the catalog requires a little practice to use effectively.

The Software Catalog represents a bold step towards making software package information more widely available. While it can not claim to be comprehensive, the description of more than ten thousand programs is a valuable resource for users of packaged software. The publishers are obviously hoping that the catalog will generate its own momentum with vendors being interested in supplying package details. As the library expands, a companion volume containing software package reviews might prove popular although much harder to produce. As the cost of an annual subscription to both sections with updates exceeds $380, libraries and companies are likely to be the major purchasers.

D. Carrington, University of New South Wales


In The Slumbering Sentinels, subtitled "Law and human rights in the wake of technology", Professor C.G. Weeramantry of Monash University presents a simple but sweeping thesis. "Science and technology" he says "have burgeoned in the post-war years into instruments of power, control and manipulation. But the legal means of controlling them have not kept pace". Part of the reason, he argues, is that lawyers are more out of their depths with new technologies than sociologists, philosophers and others. So the "sentinels" that Weeramantry looks to — legal institutions and lawyers — are slumbering.

Much of Weeramantry's book concentrates on nuclear technology and biotechnology, but I will only attempt to deal with the third technology considered, information technology, and with the solutions Weeramantry proposes his sentinels should adopt to cope with all three. It would be reasonable to expect this book to offer some discussion or analysis of the nature of the new information technology and the likely parameters of its future development, but other than such commonplace assertions that "there is an increasing dependence on the computer to assemble, classify and store information which is too vast to be dealt with efficiently by people" this is not to be found.

As for the future, Weeramantry tells us on the authority of the Encyclopedia Britannica 1974 Edition that "Data in the advanced electronics ... the electronic brain may never entirely replace humans. Its limitations are too numerous to make this possibility anything but fanciful for the moment, although an entirely new computer, built as an analogue to the human nervous system, may bring us close to this". In 1984, this is hardly an adequate discussion of the future of artificial intelligence.

Along with inadequate analysis, this part of the book also suffers from unsupported generalisations. We are told that "Clearly, computers will turn out results which are slanted in favour of those who control them". An interesting hypothesis, perhaps correct, but certainly not "clear" without argument.

The main threat by computers to privacy is perceived to be in


73

Book Reviews


Anyone who, like this reviewer, has strayed into the world of microcomputers and specifically into the territory occupied by CP/M may have encountered the same problems. One problem that an experienced computer user has is locating technical information. There are many publications on microcomputing and on systems such as CP/M, and most of it is of low quality. It is the spectrum of users. Each book will be written to accommodate the complete novice, not a bad thing in itself, but if you want an explanation of the instruction set of an XZY 999, you will probably have to purchase a book that explains all about machines and bytes and assemblers and so on; all for the few pages which give the information you want.

And so it is also with CP/M. Before long the experienced programmer will want a programmer who wants on CP/M: what is the structure of CP/M? Where is console input processed? How do I add a new device driver? What is the format of a CP/M disc directory? What is the organisation of a disc under CP/M? Most of the books on CP/M will not answer all of these questions; indeed some of the more detailed ones will be lucky if more than one of those questions are answered in a single book, the rest of the book is likely to be devoted to a description of the external characteristics of CP/M, or even to an elementary introduction to the microcomputer.

So when a book arrives with the title Mastering CP/M there is hope that here is a book which will deal with all the technical questions and will save purchasing the Digital Research Manuals.

For the reader without detailed knowledge of CP/M the system consists of three components: the command and processor (CCP); the basic disc operating system (BDOS) and the basic input output system (BIOS). The BDOS implements the system primitives and is machine independent, while the BIOS contains the device drivers and is, of course, machine dependent. These components are described well enough in the book, and also a good description of how to modify and install the BIOS is given. That takes us to the end of chapter 3; there are 8 chapters and 10 appendices in the book.

By the end of the book all the questions raised at the beginning of this review have been answered, but the other content of the book raises important questions. A more appropriate title for the book would have been Mastering Assembler Programming under CP/M, as the author establishes a library of macros for use with CP/M and uses the discussion of various aspects of CP/M to motivate the writing of a number of utility programs in assembler. Thus, amongst others, the following programs are given: a 4 page program to rename a file; a 3 page program to delete a file; and a 15 page program to display the disc parameters and block allocation map. Anyone who has been in computing for long enough is now seeing the wheel turn full circle for the third time. What is the point of writing such utilities in assembler? There is an excuse for writing the system in assembler and certainly the part of the book concerned with modifying the CP/M system itself must be concerned with assembler, but there are Pascal and C compilers which run under CP/M and it makes much more sense to write utilities such as those listed above in one of those languages.

In conclusion the book does contain probably all the technical information which a programmer will want on CP/M, but that information is submerged in a mass of assembler programs which are simply a distraction for all those who don't believe that Mastering CP/M is a synonym for Mastering Assembler.

Ken Robinson, University of New South Wales

Ken Robinson, University of New South Wales
that if information about individuals which is held on previously separate data systems is "put together, that information is a com-
office and the motor registry offices, but this is what Weeramantry
mean that the merging of these data banks in different gov-
systems has connections with at least
can also be collected in a central repository. People's credit ratings
are established from just such centralised information". Unless
collected by credit agencies, insurance corporations and employers
are stored in computers in Singapore" which, if it is true, certainly
obtain this information from the credit applicant, not the employ-
and while they do record where a credit applicant has worked, they
Weeramantry has unearthed practices in other states totally differ-
er. Attempts to establish such general purpose data systems have, in
obtained from credit agencies, insurance corporations and employers
Weeramantry is simply the adoption of the OECD's eight prin-
illustrates an objectionable trans-border data flow, but as no foot-
the provision of credit reports for insurance purposes illegal. In the
Weeramantry concludes by hoping that "some of the prin-
theses ... to continue the economic and political victimisation of the
CONTROL'S to make it look respectable. The five-line bibliography con-
tions presented later, by arrays and by pointers, the two versions
manner that does them no credit.
Chapter 3 deals with trees. Once again, the ADT definition is
apparently pulled out of a hat, begging a whole range of important
questions. We are then presented with an algorithm for the non-
recursive preorder traversal of a tree. The use of four goto's is justi-
ified with the bland statement that the traversal is "one of those
awkward tasks that can be best described in terms of 'states' and
resequencing. They show that one part of the algorithm is \( O(e) \) and
the other \( O(n \log n) \), so their conclusion is
by the use of a priority queue. They therefore conclude that the run-
algorithm begins with a test
last
grams and its non-trivial errors. There are, maybe, a few counter-balancing merits, particularly in the case studies and the exercises. But as a treatment of data structures and algorithms, it leaves much to be desired.

The printing is done from camera-ready copy supplied by the authors and the preface acknowledges the "excellent UNIX TM, based text preparation and data communication facilities that" signific-
antly eased the preparation of the manuscript by geographically separated authors". The printing of each individual line is indeed quite good; if your standards are low enough, you might even agree that it is "excellent". On the other hand, the page make-up is unbelievably bad. The algorithm for inserting figures and diagrams is a crude one and frequently places them in the section following the one to which they belong — a very confusing arrangement. This reflects the poor state of the art in automating page make-up; but there is no need to inflict the results on the readers when better, more traditional methods are available. We should not be blind to the defects of our own technology.

This review is longer than the book deserves. But it is written as a point of view of professionals who trade on past reputations and who sacrifice quality for quantity. The same authors, in varying combinations, have produced several other books recently. In future, I shall look at their products much more carefully before adopting them for any courses.

J.B. Hext, Macquarie University


This is a useful unpretentious book that has quite a lot of practical advice for managers and computer professionals. It is aimed at the manager of the small business who wants to install a computer system which will increase company profitability. The good advice includes pithy statements such as "don't automate a manual mess", "be prepared for 'people's resistance"", and "short reports are more likely to be read and understood than long reports". Silver's Wall is explained as the division between revenue-earning activities such as sales and overhead tasks such as maintenance of customer records. Keeping the action on the right side of the business is a key point and management is where it all starts (and ends, for that matter). The book has four sections. The first three describe what on-line computer systems really area, explain how they can be useful and offer guidance on how to select software packages or design custom-built software. The fourth section gives examples of specific techniques such as inventory control using a case study approach. Although It is written in the U.K., the ideas developed are just as relevant in Australia. The need for demystification of computer applications in both countries. Many small business managers are still bombarded by sales representatives who discuss 'bytes' and 'point of sales terminals' on the assumption that this will be of interest to any small retailer. The actual problem is a brief tutorial on the subject and a paper on standards which emphasises the importance of the conceptual schema and the open systems interconnection architecture. The remaining papers address research topics and a brief comment on each under the first author name is included below. A paper of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

Most half of the book (165 pp) is taken up by five papers which comprise descriptions of the architecture of some of the prototype systems which have been developed, POREL developed at the University of Stuttgart and the SIRIUS systems developed at INRIA in France are described in two lengthy papers, System R developed at IBM, San Jose is described from the aspect of distributed queries at co-operating System R sites, MULTIBASE is a retrieval only system developed at the Computer Corporation of America and this is designed to handle pre-existing, heterogeneous, distributed databases. Finally ADDAM is a research vehicle developed by Software Sciences in the U.K. for real time command and control applications distributed on a local area network.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

The first half of the book (165 pp) is taken up by five papers which comprise descriptions of the architecture of some of the prototype systems which have been developed, POREL developed at the University of Stuttgart and the SIRIUS systems developed at INRIA in France are described in two lengthy papers, System R developed at IBM, San Jose is described from the aspect of distributed queries at co-operating System R sites, MULTIBASE is a retrieval only system developed at the Computer Corporation of America and this is designed to handle pre-existing, heterogeneous, distributed databases. Finally ADDAM is a research vehicle developed by Software Sciences in the U.K. for real time command and control applications distributed on a local area network.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

The first half of the book (165 pp) is taken up by five papers which comprise descriptions of the architecture of some of the prototype systems which have been developed, POREL developed at the University of Stuttgart and the SIRIUS systems developed at INRIA in France are described in two lengthy papers, System R developed at IBM, San Jose is described from the aspect of distributed queries at co-operating System R sites, MULTIBASE is a retrieval only system developed at the Computer Corporation of America and this is designed to handle pre-existing, heterogeneous, distributed databases. Finally ADDAM is a research vehicle developed by Software Sciences in the U.K. for real time command and control applications distributed on a local area network.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

The first half of the book (165 pp) is taken up by five papers which comprise descriptions of the architecture of some of the prototype systems which have been developed, POREL developed at the University of Stuttgart and the SIRIUS systems developed at INRIA in France are described in two lengthy papers, System R developed at IBM, San Jose is described from the aspect of distributed queries at co-operating System R sites, MULTIBASE is a retrieval only system developed at the Computer Corporation of America and this is designed to handle pre-existing, heterogeneous, distributed databases. Finally ADDAM is a research vehicle developed by Software Sciences in the U.K. for real time command and control applications distributed on a local area network.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.

This book contains the papers of the Second International Symposium on Distributed Data Bases held in Berlin on 1-3 September, 1982. Hence it is not a text on the subject but a collection of papers of interest mainly to researchers and to those who wish to keep informed of advances in this area of database technology. The quality of the print is variable because the production has been offset from individual author submissions, and the combination of size of font and fairness of reproduction in some papers would strain many readers.
for implementing data base systems — proposes a distributed operating system.

This book should only be considered for purchase by computer science libraries. Taking into account the delay between the conduct of the research reported and the publication of this review, I wonder whether it is worthwhile continuing to review such conference proceedings.

John Smith, 
CSIRO Division of Computing Research


Sixty-one papers cannot each be properly appraised by one reader, and especially not when they range from accounts of abstract work in Computer Science and Statistics, through practical details of implementation on microcomputers of specific details of statistical procedures, to net-to-soft selling of commercial packages. This range also means, however, that few prospective readers will fail to find material of direct interest, even though the substantial majority of the papers will be — at most — skimmed by any specific reader.

The Keynote Address, The Future, by R.W. Hamming provides one man's view of possible developments — in particular, he notes (with some alarm) that the traditional view of Statistics as a condensate of data is tending to be reversed under the influence of computers: 15 data points can easily provide 15 pages of output, especially if it is graphical.

Several examples will serve to illustrate the variety of papers exhibited [they are selective indeed]:

— A Package for Solving Large Sparse Linear Least Squares Problems, with details of the implementation, and claimed execution times better by a factor like 50 for large problems when compared with BMD, SAS and SPSS executions.


— Methods for Bandwidth Choice in Non-parametric Kernel Regression, dealing with smoothing.

— Statistical Packages on Microcomputers, an extensive (but inevitably incomplete) annotated list of software principally for micros with hardware and software costs less than $US8000.

As with earlier such Proceedings, this volume provides a snapshot of the state of the interface, and is well worth library acquisition.

J.B. Douglas
University of New South Wales


This book is disappointing. The aims stated in its preface sound good, but it fails to achieve them.

It contains too much general information about microcomputer hardware and immediate system software; the first two chapters are mainly on these two topics and comprise over half of the main text. The remaining four chapters which would be of particular interest to the business user are short and superficial and are padded with trivia.

Figure 2.1 compactly demonstrates the lack of attention to detail throughout the book. The example given of "user oriented" software is a section of VISICALC with two lines of exclusively machine oriented code, nothing to do with a user application. The example given of Pascal, if it did not contain other minor errors, would enter a non-terminating loop repeatedly printing identical information; it also contains a redundancy that no Pascal programmer except a novice would write. The example given of machine code is for a microprocessor with eleven bit words.

A repeated theme of the book is that small businesses should avoid purchasing microcomputers with 40 character wide VDU screens and with cassette tapes for data storage. Certainly very sound advice, but it seemed to be overemphasised in a book where comparatively large applications are suggested.

This book would not be suitable as a textbook, except perhaps for a one day computer introductory course. It is out of place in a series of Computer Science texts.

A businessman contemplating a microcomputer purchase could probably obtain most of the information given in this book from one or two issues of a suitable popular computer magazine. However there is a reasonable bibliography and the glossary of terms is comprehensive and reasonably accurate.

D.Q.M. Fay, 
Western Australian Institute of Technology


This soft-covered book is presented as an introductory text, directed to the requirements of first and second year undergraduate engineers, but it is also of value to advanced mathematics and science students in schools. It presents some fundamental topics in numerical mathematics and illustrates computational techniques with programs written in BASIC. The book is written in an easy-to-read style and is attractive for teaching purposes with many examples, exercises and practical notes.

Chapter 1 gives a brief description of BASIC, Chapter 2 discusses sources of error in applying numerical methods, and Chapter 3 gives examples of BASIC programming of some elementary mathematical calculations. The final chapters, 4 (solution of non-linear algebraic equations) and 5 (interpolation, differentiation and integration) are more substantial and contain a useful collection of numerical techniques and corresponding 'interactive' BASIC programs, which are presented together with algorithms, sample runs and program notes. Each chapter (except the first) is concluded with a collection of about 15 problems on both the mathematical and computational aspects covered.

The author's use of BASIC programming, despite its disadvantages in numerical analysis, is clearly a response to the increasing availability and use of microcomputers. However, the programming language used is probably not important in this elementary treatment; indeed, this thoughtfully presented book is still worth a look even for courses in which BASIC is not used.

M. Davidson, 
CSIRO, Lucas Heights Research Laboratories


This is a useful book for the growing number of people wishing to be able to program micros in BASIC.

Of course the number of books available on this subject is already very large, and continues to grow. Jeff Maynard's book is pitched at a level above that of most books on "BASIC for beginners", and is intended for those in business or in the home who wish to write useful programs — whether for data-processing or for playing games.

The book assumes minimal familiarity with micros, and explains the concepts of programming using simple program examples. There is an attempt to confine the discussion to elements of BASIC common to most interpreters and compilers found on micros, although some examples contain specific references to the Apple. Perhaps because of the compatibility problem, the discussion on programming disk files is kept very short.

Features of the book include some useful routines for sorting and merging data in arrays, and some short game programs.

Because the book explains the concepts progressively in a way intelligible to beginners, it would not be suitable as a reference book on the BASIC language. But for those wishing to learn the concepts and who have a micro available, it will be a useful book indeed.

Rich Jameson, 
Jameson & Associates, Sydney


This volume is a collection of twenty papers presented at the fifth International Conference on Computers and the Humanities held in Michigan (USA) during four days of May 1981. The scope of such a conference represents a wide range of human endeavour and this is reflected in the diversity of topics covered in this book.

The papers are, in the main, sufficiently technical to be of interest largely to those workers and potential workers in the particular fields. One gets the impression that whatever is said in a review such as this, collections of conference papers will continue to be bought by the libraries of research and tertiary institutions. A
librarian friend at a well known CAE recently told me, during a discussion about this book, that his lecturers love it. In the face of comments like this, the publisher must feel right about the book.

For my part it goes beyond my task to comment on the papers individually but we can present a brief summary of the flow of topics:

- Heading the papers is one entitled “Computing in the Humanities”, a well-written article, by the editor of the series, on where the topic has been over the years on where it is up to now.

- Next follows a paper on a programming language called ICON which appears to be a hybrid of SNOBOL4 and PASCAL.

- J.E. Huntsman presents a philosophical discussion of human language versus computer languages, which makes easy reading.

- Following we have what appears to be a debriefing from a situation where Brigham Young University took an Apple II micro into the field to collect foreign language data for a dictionary. Interfacing between IBM’s EBCDIC and Apple’s ASCII and strange Apple Writer formats are problems which were honestly brought out, and it’s good that they are since these are practical problems which often rear their ugly heads.

- Without going into any detail, we next have three papers on what might broadly be called CAI.

- The computer was called on to analyse literary style of two authors in the following two papers.

- Then we have an interesting application, using the computer to form statistics from an experiment on how we get to know others by what they say and how they say it.

- We have one paper on computers applied to music. The first attempts to decide whether Bach chorales are models of Harmony, Counterpoint or something different. The second study concerns how we perceive and recognise music.

- Next is a collection of three papers where the computer is used as an aid in linguistics. The first of these papers is an attempt to model vocabulary richness by studying the so-called type-token relationship in the works of S.Kierkegaard.

- The second is entitled “Cluster Analysis and the Taxonomy of Words in Old French”. The third is entitled “Processing Morphology: Words and Cliches”, where an interesting data structure called a ‘trie’ is used to store sequences of words in a dictionary.

- Then there is a paper describing a project where an attempt is made at resolving of interdocumental cross-references in legal texts.

- H. Van Dyke Parunak presents an interesting idea of applying relational database techniques to biblical texts.

- “Computer Simulation Methodology for Archaeology” presents a method of modelling Bronze Age village sites and their assumed activity. Easy reading.

- We have a paper on computers applied to music. The first attempts to decide whether Bach chorales are models of Harmony, Counterpoint or something different. The second study concerns how we perceive and recognise music.

- Next is a collection of three papers where the computer is used as an aid in linguistics. The first of these papers is an attempt to model vocabulary richness by studying the so-called type-token relationship in the works of S.Kierkegaard.

- The second is entitled “Cluster Analysis and the Taxonomy of Words in Old French”. The third is entitled “Processing Morphology: Words and Cliches”, where an interesting data structure called a ‘trie’ is used to store sequences of words in a dictionary.

- Then there is a paper describing a project where an attempt is made at resolving of interdocumental cross-references in legal texts.

- H. Van Dyke Parunak presents an interesting idea of applying relational database techniques to biblical texts.

- “Computer Simulation Methodology for Archaeology” presents a method of modelling Bronze Age village sites and their assumed activity. Easy reading.

- We have a paper on computers applied to music. The first attempts to decide whether Bach chorales are models of Harmony, Counterpoint or something different. The second study concerns how we perceive and recognise music.

- Next is a collection of three papers where the computer is used as an aid in linguistics. The first of these papers is an attempt to model vocabulary richness by studying the so-called type-token relationship in the works of S.Kierkegaard.

- The second is entitled “Cluster Analysis and the Taxonomy of Words in Old French”. The third is entitled “Processing Morphology: Words and Cliches”, where an interesting data structure called a ‘trie’ is used to store sequences of words in a dictionary.

- Then there is a paper describing a project where an attempt is made at resolving of interdocumental cross-references in legal texts.

- H. Van Dyke Parunak presents an interesting idea of applying relational database techniques to biblical texts.

- “Computer Simulation Methodology for Archaeology” presents a method of modelling Bronze Age village sites and their assumed activity. Easy reading.

- We have a paper on computers applied to music. The first attempts to decide whether Bach chorales are models of Harmony, Counterpoint or something different. The second study concerns how we perceive and recognise music.

- Next is a collection of three papers where the computer is used as an aid in linguistics. The first of these papers is an attempt to model vocabulary richness by studying the so-called type-token relationship in the works of S.Kierkegaard.

- The second is entitled “Cluster Analysis and the Taxonomy of Words in Old French”. The third is entitled “Processing Morphology: Words and Cliches”, where an interesting data structure called a ‘trie’ is used to store sequences of words in a dictionary.

- Then there is a paper describing a project where an attempt is made at resolving of interdocumental cross-references in legal texts.

- H. Van Dyke Parunak presents an interesting idea of applying relational database techniques to biblical texts.

- “Computer Simulation Methodology for Archaeology” presents a method of modelling Bronze Age village sites and their assumed activity. Easy reading.

- We have a paper on computers applied to music. The first attempts to decide whether Bach chorales are models of Harmony, Counterpoint or something different. The second study concerns how we perceive and recognise music.

- Next is a collection of three papers where the computer is used as an aid in linguistics. The first of these papers is an attempt to model vocabulary richness by studying the so-called type-token relationship in the works of S.Kierkegaard.

- The second is entitled “Cluster Analysis and the Taxonomy of Words in Old French”. The third is entitled “Processing Morphology: Words and Cliches”, where an interesting data structure called a ‘trie’ is used to store sequences of words in a dictionary.
interesting paper deals with a set of integrated (computerised) tools for the conceptual design of data base schemas and transactions. The book is relevant for most data base administrators, persons in the information centre, postgraduate students specialising in data bases and academics involved in information processing. The reviewer finds it worth its price.

G.M. Nijssen, University of Queensland


Cobol for Micros is an introduction to Cobol programming in a microcomputer environment. It will be most useful for programmers who can already program, but who are new to Cobol. Cobol programmers moving from mainframes to micros will also find several chapters of interest. The book is not intended for beginners to programming. Readers looking for descriptions of specific microcomputer Cobol systems will be disappointed, since only one such implementation is described, and the presentation is as machine independent as possible.

The book begins with a brief history of Cobol. The author argues strongly for Cobol's superiority over Basic for microcomputer business applications. A microcomputer programming environment is then described for the benefit of mainframe users. The particular environment used is the CP/M operating system running on the CISC Cobol. The main differences between this system and a typical mainframe Cobol are lack of report-writing and sorting facilities and (surprisingly) lack of Boolean expressions. The author also points out that, because of difficulty of considerations, special care is required with the use of files in a microcomputer environment.

The remaining thirteen chapters of the book present a description of the Cobol language. This description is largely implemented independently. Only in such areas as forms-oriented input, debugging and program segmentation are CISC Cobol facilities specifically discussed.

The author has taken pains to make the presentation easy to follow. It is filled with clear, useful examples, drawn from the kind of small business applications for which microcomputer Cobol would be most used. Particularly good, I thought, were the chapters on file handling and interactive input. These two topics are precisely those that will seem most strange to programmers in other languages. Each chapter concludes with a set of self-test questions, for which answers are given in an appendix. Programmers used to Pascal and related languages could have some trouble with Chapter 9 (Sequence Control), where understanding of the semantics of nested Cobol IF statements is not helped by the indentation in some examples, and where the peculiar Cobol usage of the word 'proced­ure' is only very briefly explained.

Readers who, like the author of the book, have background in Basic, will not share these difficulties. This book would be a valuable acquisition for microcomputer users who have realised the limitations of Basic as a business language and who are ready to make the move to Cobol. It is also a useful general text from which Cobol may be learned for use in other situations.

W.P. Beaumont, University of Adelaide


This book describes a number of computer network architectures and protocols and compares them with the Open Systems Interconnection (OSI) model.

The first section of the book presents first an intuitive and then a more formal description of the OSI model. As examples of the protocols for the lower layers of the model, X.21, HDLC, X.25 and a German proposal for a transport protocol are described in some detail.

The second section of the book describes the network architectures developed by various manufacturers. There are fairly detailed descriptions of IBM's Systems Network Architecture (SNA) and Digital Equipment Corporation's DECENT, and briefer descriptions of Univac's and Burroughs' network architectures. The descriptions include detailed comparisons with the OSI model.

The third section of the book, titled "Public Data Networks" discusses some of the CCITT X-series recommendations, describes a mapping between IBM's SNA and X.25 and finally briefly discusses Local Area Networks and some future developments in networking.

The ISO draft standard for the OSI Reference Model is included as an appendix. The appendices also include a list of services and functions of each layer of the model, a table of various architectures with the OSI model and a list of the CCITT X-series recommendations for public data networks.

The authors are employees of IBM and most of the material was prepared for lectures at IBM's European Research Institute. As the authors state in their introduction, the book is not a textbook on network architecture. Its strength lies in its description of the network architectures of a number of computer manufacturers, and particularly its comparison of them to the OSI model. As such it is a very useful addition to computer network literature.

The worst thing about this book is probably its title, which leads one to expect a more comprehensive treatment of the subject matter. Perhaps a subtitle such as "Proprietary Architectures and the OSI Model" would have been appropriate. This reviewer was stunned to find no mention of the ARPANET in a book with this title. The authors in their introduction suggest they are excluding "experimental and transitional environments", because of the exclusion of the ARPANET. The section on "Public Data Networks" does not describe the architecture of Tymnet, Datapac, Telenet or Transpac. There are also some surprising exclusions in the section that describes computer manufacturers' network architectures. For example Hewlett Packard and Pinme get no mention.

Despite these shortcomings, the book is a useful reference, providing a perspective on the network architectures of commercially available products.

J.L. Fernandez, Logica Pty Ltd, Sydney, NSW


I am not sure how one should characterise this volume — at more than 1000 pages of A4 size, it is clearly not bed-time reading (and is much more reasonably priced than most conference pro­ceedings from this publisher). If you are anxious to keep abreast with the state of the art in telecommunications, and you have a day or two to spare, this is well worth browsing. It must be classed as a "best buy" for libraries and perhaps a few individuals whose need to keep up with the latest trends in data communication is paramount.

It is the proceedings of a very large conference: 47 sessions in four parallel streams, resulting in no less than 175 published papers, many with multiple authors. Clearly it will not be possible even to mention all the session titles here — in the book itself, the table of contents (admittedly set in large type) occupies no less than ten pages.

Marshall McLuhan once observed that "the medium is the message". The message of this volume is clear: the PTTs (PTT being a generic term for Public Telephone & Telegraph Corpor­ation) are no longer ambivalent in their attitudes to data communi­cations; never is it just an unwelcome intrusion into the voice telephone network, or just a rapidly growing new service that could be exploited for its revenue potential. PTTs are now facing squarely the problems of integrating their voice services into the new national and international telecommunication networks. If I may select one paper that epitomises the changes taking place, then the paper by W.J. Murray of British Telecom, entitled "The Emerging Digital Transmission Network", suggests the urgency and great extent of the changes now occurring in much of the world (the pace in Australia seems to be much less breathless).

Clearly the PTTs aim to remain in the pilot's seat: the computer industry may have precipitated the communications rev­olution, but the PTTs are now making maximum the revolution. The telephone voice channel may remain as a basic unit of service, but in the future it will be characterised by a capacity of 64 kilobits per second, rather than a bandwidth of 4 kilohertz. For local services, the existing local telephone loops may be converted to provide an 80 kbps service to the local exchange. This channel may be subdivided into one 64 kbps channel, and two 8 kbps subchannels, one for data and one for control. The arrival of ISDN (integrated services digital network), and with it the demise of the basic data channels, cannot come too quickly as far as I am concerned.

Look out for this book and its successors.

John Lions, University of New South Wales

The Australian Computer Journal

Information for Authors

The Australian Computer Journal invites original contributions relating to the design, understanding and application of digital computers. The range of material invited includes research papers, tutorial and review papers, industry case studies, short communications and letters to the editor. The journal appears quarterly, in February, May, August and November each year.

Papers should comply with accepted practices for the style and organisation of scientific papers. Each paper must be in English and must be certified by the author that it is his or her own original work, that it has not been copyrighted, published or submitted for publication elsewhere and that, if the work described has been sponsored, the paper has been cleared for publication by the sponsoring organisation. The author is asked to agree, if the paper is accepted for publication, to assign copyright in the paper to the Australian Computer Society Incorporated in accordance with normal practice. Where there is more than one author, each should sign the letter of transmittal and make the same certifications and agreement regarding copyright as described above.

Each paper submitted will be reviewed by the editor in the first instance. Provided the form of submission is satisfactory, it will then be evaluated for accuracy, originality and relevance by at least two referees or members of the editorial committee. Revisions may be requested. Every effort is made to notify the author(s) promptly whether the paper will be accepted for publication and the expected date thereof.

Four to six printed pages is the preferred size for full papers, including all diagrams and tables. (One printed page equals about 950 words of text.) Longer papers may be accepted if justified.

Short communications (one or two printed pages) describing, for example, novel applications, new ideas, work in progress or summarising post-graduate research theses accepted for degrees are also invited. The aims of the short communications are to inform readers of the Journal of current work of interest, to provide abstracts of current Australian research for overseas readers, and to provide a medium for quickly reporting new ideas in Computer Science.

Letters may be sent to the editor for publication. These should be typed and clearly indicate the matter for concern (e.g. a particular journal article). Any tables or mathematical formulae must be in a form suitable for printing. The writer’s full name and address must be given (not necessarily for publication).

Book reviews are normally prepared at the invitation of the Editor only.

Manuscript Preparation. Authors should observe carefully the directions for manuscript preparation given below. Deviations from these may complicate the editor's task and may introduce unnecessary delays. Papers should be typed double spaced with 30 mm wide margins, on one side only of A4 size paper (210 mm x 297 mm). Three identical copies on good quality paper are required; clear photocopies on bond paper will be acceptable.

The first page should contain the title, author information, an abstract and cross-referencing information.

The title of the paper should characterise the contents of the paper in as short a space as possible. (The title, or a shortened version of it, plus a page number, should appear on the top line of each subsequent sheet of the text.) Below the title of the paper should appear the author's name (initials and last name only — no titles or degrees should be shown). If there is more than one author, the principal author should be listed first. For each author, a sufficient postal address (possibly including the name of an employer), suitable for printing as a footnote, should be supplied.

The abstract on the title page should be informative and summarise the paper as concisely as possible in about 100 words. Since the abstract may be republished separately, it should contain no uncommon acronyms, footnotes or references to other published works.

Finally, the title page should include standard content indicators: lists of Key words and Phrases, and a classification of the paper according to ACM’s “Computing Reviews” categories as published in Computing Reviews, 24, No. 1, January, 1983.

The main text should be clearly written and follow the English conventions for spelling and punctuation. Text prepared using a word processor should not be hyphenated or right-justified. Subheadings and paragraphs should be clearly shown. Major and minor sections may be numbered at the author’s discretion.

Footnotes should not be used unless absolutely necessary (they should not be used for reference citations). If a footnote is unavoidable, then it should appear in the text of the manuscript immediately below the line to which it refers and should be separated from the rest of the text by full width lines immediately above and below the note.

Tables should be numbered consecutively. Each table should have a general but brief self-explanatory heading shown at the top. Column headings should be brief and clearly shown. Preferably tables will be constructed to fit neatly into a single column (83 mm x 229 mm) or exceptionally, across two columns (width 178 mm). Large tables should be typed on separate sheets from the text. More than one table may be typed on a single sheet, but tables should not be split between sheets. The proper location for each table should be noted in the margin of the text.

Camera-ready copies for each figure, diagram or flowchart should be supplied. These should be drawn in Indian ink on stiff white paper or Bristol board. Drawings should be about twice the expected finished size (single column width is 83 mm, and double, 178 mm). Legends and labels should be clearly drawn, positioned suitably and large enough to remain legible after reduction. Each drawing should be on a separate sheet, packed flat and have on the reverse side, the author’s name and the figure number and caption. In the case of photographs, please submit glossy prints with good contrast. Avoid using clips or writing heavily on the reverse side when recording author name, figure number and caption. Full-colour photographs may be submitted in appropriate cases (prior consultation with the editor is advised). Programs that illustrate the text or that convey important algorithms may be published. It will usually simplify matters considerably if camera-ready listings of such programs can be submitted. Such listings should be printed on good quality white paper using a letter-quality printer with a single-strike carbon ribbon.

Unusual mathematical symbols, subscripts and mathematical formulae can be difficult to set in type. Please choose symbols carefully, avoid excessive use of subscripts, plan formulae to fit within the standard column width and use alternative forms that are easier to set in type, e.g. exp(−x) rather than e−x. Avoid using the letters...
References. Careful adherence to the Journal's style for citation and quotation of references is required. This style is the so-called Harvard style and has several advantages.

Reference citations in the main text should be made as in the following examples: "It was shown by Curtis and Osborne (1966) that . . ." or "It has been shown elsewhere (Paine, 1966) that . . ." or "It may be shown (e.g., see Knuth, 1973a) that . . ." Note that the citation is composed of the author's last name and the year of publication. Where this is ambiguous, different works by the same author(s) in the same year are distinguished by adding a lower case letter to the year. Only one level of parenthesis is used. The author's name appears outside the parenthesis in the references is direct, or inside if the reference is indirect. In the latter case, the name and year are separated by a comma. If there are two authors, both names are used as part of the reference. If there are three or more authors, then all author last names should appear in the initial citation but subsequent citations may be abbreviated by replacing the second and later author names by the phrase "et al." Thus, for example, a second citation of the book "Newey, Stanton and Wolfendale (1978)" may be made as "Newey et al. (1978)". In this style the citations are not intrusive, they are often sufficient for the reader to recognise the work without further effort and they may be added or deleted easily during drafting without disturbing any pre-arranged numbering scheme.

References to unpublished works or private communications should be avoided as far as possible. If these are to appear, they should appear within the main text as e.g. "Lone and Ryder (to appear)" or "F.G. Smith (private communication)". They should not appear in the final reference list unless publication has already been arranged.

A section at the end of the paper headed "References" should quote full details for all references cited in the text (and no others). These must be sufficient to allow an ordinary reader to locate the reference without undue difficulty, and they should be arranged so that the textual citations can be located readily, i.e., the reference list should be ordered alphabetically by author name and then year. To this end, the publication year (together with any alphabetic letter suffix) is listed immediately after the author's name and before the title of the work and the remaining details. The following items illustrate the style of a reference list:


Authors are responsible for ensuring the accuracy of all details of all references quoted. Guidelines to be observed during the preparation of the reference list are: author names are capitalised (for prominence) with initials following the family name; the year of publication is enclosed in parentheses and followed by a colon; for references to books and monographs, the colon is followed by the title (underlined), a page reference (if appropriate), the name of the publisher and the place of publication; for references to journal articles, the colon is followed by the full title of the article (without quotation marks), the name, or a standard abbreviation for the name, of the journal (underlined), the volume number, the issue within the volume, and the page number range (introduced by the symbol "pp.").

Journal titles appearing in citations should be abbreviated in accordance with accepted standards. (See ISO 4, Documentation — International code for the abbreviation of titles and periodicals and ISO 833, Documentation — International list of periodical title word abbreviations. A relevant summary of these can be found in Computing Reviews, 22, pp. 418-427, September, 1981.)

Titles consisting of a single word are never abbreviated, and a leading The is usually omitted. For longer titles, the following abbreviations should be used: ACM (Association for Computing Machinery), Abstr. (Abstracts), Appl. (Applied Applications), Aust. (Australian), Bull. (Bulletin), Commun. (Communication[s]), Comput. (Computers), Des. (Design), Eng. (Engineering), J. (Journal), Math. (Mathematics), Proc. (Proceedings), Program. (Programs), Programming, Rev. (Review), Sci. (Science[s]), Scientist, Softw. (Software), Surv. (Surveys), Syst. (Systems), Trans. (Transactions). Thus sample abbreviations are Aust. Comput. J., Datamation and IEEE Trans. Softw. Eng.

Biographical Note. Each author of a paper is asked to supply a short narrative biographical note (about one hundred words) for publication with the paper.

Proofs. Papers that are accepted for publication will be typeset and, shortly before publication, a proof copy will be sent for proof-reading and correction to the author, or to the first author if there is more than one, or to his nominee. (Overseas authors are particularly requested to nominate a representative in Australia who can perform this task.) The symbols used for correcting proofs should be those defined in the Style Manual published by the Commonwealth Government Printer, Canberra, pp. 76-79. Authors should note that proof reading is intended to correct errors introduced by the editing and typesetting processes. Extensive changes or additions to an article introduced during proof-reading may delay publication and be charged to the author. Corrected proofs must be returned promptly or publication cannot be guaranteed.

Reprints. Fifty reprints of articles will be supplied free of charge. Information for ordering additional reprints will be provided by the printer with the proof copy.

Submissions. All papers and correspondence should be addressed to:
Associate Professor J. Lions,
Editor, Australian Computer Journal,
Department of Computer Science,
University of New South Wales,
Kensington, NSW 2033, Australia.

being developed taking into account a new criteria — "Foot Traffic Fatigue".
This characteristic was not evident in the computer room, but its recognition in the office environment is causing some earlier office installations to fail through their inability to withstand high frequency traffic in aisle ways and entrance areas and accents by the introduction of mail mobiles now in common use.

Queensland computer manufacturer David Hartley Computer Australia Pty Ltd put out a strong challenge to the big guns of the computer world with the official launching of a new personal computer.
The company aims to capture a major slice of the Australian and overseas market with the new product.
The computer — the 3902 Hartley PC — is described by its makers as "beyond the capability of its competitors".
Hartley managing director, Mr Tony Paranthoienie, said the launching of the new computer "marks a very significant technological advance for Queensland".
Mr Paranthoienie said, production of the new computer and other major developments "in the Hartley pipeline" reflected a strong depth of computer manufacturing talent in Queensland.
"The Hartley company has the corporate structure and the design excellence to take on overseas manufacturers in the production of both hardware and software," Mr Paranthoienie said.
The computer was given a dress rehearsal at the Hartley Users' Congress on the Gold Coast.
"It was enthusiastically received and first orders already have been written," Mr Paranthoienie said.
Its designer, chairman of the company, Mr David Hartley, personally demonstrated the computer at the Congress.
The company has enjoyed an international reputation for its software programmes for accountants and other business applications. Until now its computers have been 'dedicated' to Hartley software.
But the new 3902 Hartley PC has "turned the tables" on competitors, according to Mr Paranthoienie.

Our new computer will combine the Hartley programme system known as HAPAS together with the MS/DOS system used by the majority of other personal computers," he said.
"What this means is that our new unit will be able to utilise a wide range of software generally available, together with Hartley software. Our own software — which is the best in its field — has a world-wide market which is growing steadily.
"Now that we have produced hardware to take both Hartley and MS/DOS applications we believe we have a computer able to uniquely extend its capability beyond all competitors."
Hartley also point out the 3902 PC has designed elements which give it greater disc storage than its 5¼" disc design competitors and offers data processing capabilities "at least four times faster than anything else around".

CALL FOR PAPERS
A conference on Computer-Aided Learning in Tertiary Education will be held from 30 August to 1 September 1984.
Submissions are invited for papers to be presented at the above conference. Submissions are to be received at the address below by 30 May 1984.
The organising committee reserve the right to accept or decline all submissions.
Abstracts should be approximately 1½ pages long on any theme which may be regarded as relevant to the use of computers as an aid to tertiary education.
An exhibition of hardware, software, courseware and publications will be held in conjunction with the Conference. Potential exhibitors should contact Ms Mary McGregor, UniQuest Conference Systems, University of Queensland, St Lucia, 4067.
For registration forms and inquiries concerning the conference, contact Ms Mary McGregor, UniQuest Conference Systems, University of Queensland, St Lucia, 4067. Telephone 377-4093.

WICAT/DATISI COMBINATION FOR LEGAL APPLICATIONS
The marriage of Datisi and WICAT has produced an effective environment for legal offices and was the reason for WICAT's first two sales to legal practices in Sydney and Parramatta.

Datisi, designed by Sashalom Pty Ltd, handles the management of legal practices' clients and financial information through the use of 27 modules; modules that allow customisation to be a standard feature and recognise the dynamics of each user. Datisi is a large scale system running under Datacat, WICAT's Databus TM interpreter which provides a sophisticated operating system with considerably higher throughput than competing Datapoint systems.
Datisi is applicable to all law firms, regardless of size.

An expanding legal firm with offices in Sydney and Parramatta has installed a WICAT 155 with 45 MB Disk storage, tape backup and input from 2 to 16 terminals, running the Datisi package. The system was installed at a time when the practice's expansion was creating a large pool of client information that needed managing and the facility for fast, comprehensive access. The Datisi package handled that requirement and as a by-product, also supplied up-to-date reports at all levels leaving the way clear for evaluation and assessment to be performed quickly.
DEPARTMENT BUYS NEW AUSTRALIAN COMPUTER

The Minister for Science and Technology, Mr Barry Jones, took delivery of an Ortex Net 186 computer from its Canberra manufacturer. The computer, the first of several to be supplied, will be used by the Technology Development Division in the Department of Science and Technology's central office.

The acquisition of the Ortex Net 186 system, a high-speed 16 bit unit, is significant as it was developed with financial assistance from the Australian Industrial Research and Development Incentives Scheme, administered by the Department. The decision to purchase the Ortex Net 186 system was made after a study of a number of Australian-made computers. While several met the operational requirements of the Department, the Ortex was found to be the most cost-effective for current applications and compatible with proposed automated office plans.

While the Department of Science and Technology is the first Commonwealth authority to purchase the Net 186 system, others have them on order. It is the Department's policy to support local industry.

All Ortex equipment is manufactured from components supplied by Australian distributors.

TEKTRONIX EXPANDS ITS RANGE OF DEVELOPMENT TOOLS

Tektronix Australia has announced further expansion in its offering to the microprocessor design engineer. Just announced are two new systems: the 856140 and the V-Series.

The 856140 Systems are turn-key colour development systems packaged for 8-bit support. This package offers significant savings in building up a custom system for 8 or 16-bit designs.

The V-Series are packaged 16-bit systems geared to the DEC-VAX environment. Tektronix now offers complete software and hardware development tools for the VAX environment for a number of microprocessors. To further enhance this expanded series, several new emulators have been added to bring the total of supported chips to thirty five. These are: the 68008, 68010, 80186, 80188, NSC-800. A new emulator for the NEC 7720, 7809/10/11/16, 78C05/06 family has also been introduced. For military applications, the 1750A emulator has been released to support the board or single chip versions.

For further information, contact your local Tektronix office.

BABCOCK'S NEW COMPUTER PLANS

Computers to run engineering design programmes developed in Australia and overseas by Babcock Australia and Babcock International, have been ordered as part of a five year development programme.

In recent years, Babcock Australia has had its computer services through links to a central bureau in Sydney. This has been of great value during a period of evaluation of computer needs, and development of programmes locally, and by Babcock's associated companies in other parts of the world.

The equipment ordered is an IBM 4331 model 11. Starting with four terminals and two personal computers, the system is expected to grow into a network of some 40 work stations and CADAM terminals. Initially, this will be to run the Computer Aided Engineering programmes, some of which are already developed to take design requirements as the main input, and produce a complete design including the drawings. More of these programmes will be written over the next five years.

This will enhance the production of all drawings, including preliminary design drawings. A major benefit will come when drawings have to be changed. The changes can be entered in the computer, and new drawings printed immediately.

The Managing Director of Babcock Australia, Mr Hugh Weir, said this five year development programme was of great importance to the efficiency and competitive abilities of Babcock in the years to come.

NEW MULTI-PURPOSE DOT MATRIX PRINTERS AVAILABLE

A new series of versatile dot matrix printers from NEC is being released in Australia through Datascape.

The Sydney-based computer peripheral supplier will take delivery of the P2/P3 series this month. The P2/P3 is one of the new printers which can perform various functions. It combines high speed data processing with lower speed word processing -- and all in the one compact unit.

An 18-wire staggered array print head allows high speed printing at 180 cps, medium speed enhanced quality at 90 cps, and premium quality printing at 30 cps.

A range of plug-in interfaces are available, including standard serial, parallel, and a special unit for the IBM PC.

NEW COMPUTERISED FILING SYSTEM

The storing of information has long been a space and time consuming problem for anyone who needs to keep records. Computer Classics solves this problem with the Data Base Management Series from Abraxas Software.

The series, for the Commodore '64' personal computer, consists of eight packages containing a programme (disk) and one of eight user manuals, each designed to help you make the most of the programme for your specific needs.

There are two basic parts to each programme, including a file processor which enables you to set up and manipulate files, and a word processor which enables you to process reports and letters with information from a file.

Each package enables you to store important information which can be easily accessed at any time. You can use information from any file to compile documents such as personalised letters and reports, stock lists, price lists and quotations.

GREAT IMPRESSION BY NEW "IMPRESSIONS"

Software Australia, a major supplier of OEMs and the retail computer shops, has announced the release of its new "Impressions" series of printers.

According to Marketing Manager, Mr Warren Shaw, the three new dot matrix printers, with speeds ranging from 100-300 cps, have been selected on three main criteria -- excellence in product, performance and price. Software Australia put each printer through an exhaustive 90-day evaluation before release.

The three new printers are all made in Japan by reputable companies, and good stocks of the three models will be available from Software Australia.

For the big thinker.

The biggest range of advanced microcomputers made for business. Onyx.

Onyx Computer Systems are proud to announce the addition to their family of microcomputers of no less than seven new, enhanced models. Which makes Onyx the producer of the biggest range of advanced business microcomputers in the computer industry.

Which vastly simplifies the decision-making process for you in choosing the best microcomputer system for your current operations needs and future growth.

It's the Onyx technology that helps you to think bigger about daily operations and growth. For example an Onyx system can provide you with an edge in speed of data handling that's almost twice as fast as your competitor without an Onyx.

Unique advances like these flow right through the broad Onyx range. From the Sundance desk top microcomputer supporting up to five users to the C8002, which stores up to 40 megabytes of data, expands to 160 megabytes and supports up to 16 users. And although Onyx systems are in many cases technologically superior to other microcomputers they are, surprisingly, often less expensive.

Its range of choice and price/performance benefits like these that can put the person choosing an Onyx, the bigger thinker, in a better frame of mind.

ONX

the small computer for the big thinker.

For further information send this coupon to:
Onyx Computer Systems, 7-9 Merriwa St., GORDON, N.S.W. 2072.
A Division of Horizon Computer Corporation Pty. Ltd.

Name ________________________________
Company ____________________________________________
Address ____________________________________________
Postcode _______________________ Phone ________________________
The Australian Computer Journal is an official publication of the Australian Computer Society Incorporated.


SUBSCRIPTIONS: The annual subscription is $20.00. All subscriptions to the Journal are payable in advance and should be sent (in Australian currency) to the Australian Computer Society Inc., Box 4944, GPO, Sydney, 2001. A subscription form may be found below.

PRICE TO NON-MEMBERS: There are now four issues per annum. The price of individual copies of back issues still available is $2.00. Some are already out of print. Issues for the current year are available at $5.00 per copy. All of these may be obtained from the National Secretariat, Box 4944, GPO, Sydney NSW 2001. No trade discounts are given, and agents should recover their own handling charges.

MEMBERS: The current issue of the Journal is supplied to personal members and to Corresponding Institutions. A member joining partway through a calendar year is entitled to receive one copy of each issue of the Journal published earlier in that calendar year. Back numbers are supplied to members while supplies last, for a charge of $2.00 per copy. To ensure receipt of all issues, members should advise the Branch Honorary Secretary concerned, or the National Secretariat, promptly, of any change of address.

MEMBERSHIP: Membership of the Society is via a Branch. Branches are autonomous in local matters, and may charge different membership subscriptions. Information may be obtained from the following Branch Honorary Secretaries. Canberra: PO Box 446, Canberra City, ACT, 2601, NSW: 1st Floor, 72 Pitt Street, Sydney, NSW, 2000. Qld: Box 1484, GPO, Brisbane, Qld, 4001. SA: Box 2423, GPO, Adelaide, SA, 5001. WA: Box F320, GPO, Perth, WA, 6001. Vic: PO Box 98, East Melbourne, Vic, 3002. Tas: PO Box 216, Sandy Bay, Tas, 7005.

AUSTRALIAN COMPUTER JOURNAL
Subscription/Change of Address Form

Name..................................................
Current Address.................................
.......................................................
□ Please enrol me as a subscriber for 1984. I enclose a cheque or bank draft for $420.00.
□ Please record my new address as shown above. I attach below the mailing label for the last received issue.

ATTACH LABEL HERE

Send all correspondence regarding subscriptions to the Circulation Manager, Box 4944, GPO, Sydney, 2001, Australia. Photocopies of this form are acceptable.

CONTRIBUTIONS: All material for publication should be sent to: Associate Professor J. Lions, Editor, Australian Computer Journal, Department of Computer Science, University of New South Wales, Kensington, NSW 2033. Prospective authors may wish to consult manuscript preparation guidelines published in the May 1984 issue. The paragraphs below briefly summarise the essential details.

Types of Material: Four regular categories of material are published: Papers, Short Communications, Letters to the Editor and Book Reviews. Generally speaking, a paper will discuss significant new results of computing research and development, or provide a comprehensive summary of existing computing knowledge with the aim of broadening the outlook of Journal readers, or describe important computing experience or insight. Short Communications are concise discussions of computing research or application. A letter to the Editor will briefly comment on material previously appearing in the Journal or discuss a computing topic of current interest. Descriptions of new software packages are also published to facilitate free distribution.

Refereeing: Papers and Short Communications are accepted if recommended by anonymous referees. Letters are published at the discretion of the Editor, and Book Reviews are written at the Editor’s invitation upon receipt of review copies of published books. All accepted contributions may be subject to minor modifications to ensure uniformity of style. Referees may suggest major revisions to be performed by the author.

Proofs and Reprints: Page proofs of Papers and Short Communications are sent to the authors for correction prior to publication. Fifty copies of reprints will be supplied to authors without charge. Reprints of individual papers may be purchased from Associated Book Publications, PO Box 440, Broadway, NSW, 2007. Microfilm reprints are available from University Microfilms International, Ann Arbor/London.

Format: Papers, Short Communications and Book Reviews should be typed in double spacing on A4 size paper, with 2.5 cm margins on all four sides. The original, plus two clear bond-paper copies, should be submitted. References should be cited in standard Journal format, and generally diagrams should be ink-drawn on tracing paper or board with stencil or Letraset lettering. Papers and Short Communications should have a brief Abstract, Key word list and CR categories on the leading page, with authors’ affiliations as a footnote. The authors of an accepted paper will be asked to supply a brief biographical note for publication with the paper.

This Journal is Abstracted or Reviewed by the following services:

Publisher
ACM Bibliography and Subject Index of Current Computing Literature.
ACM Computing Reviews.
AMS Mathematical Reviews

ENGINEERING INDEX INC.
Engineering Index.
INSPEC Computer and Control Abstracts.
INSPEC Electrical and Electronic Abstracts.
ISI Current Contents/Compumath.
ISI Compumath Citation Index.
SPRINGER-VERLAG Zentralbaltur Mathematik und ihre Grenzgebiete.

Copyright ©1984. Australian Computer Society Inc.

Production Management: Associated Business Publications, Room 104, 3 Small Street, Ultimo, NSW 2007 (PO Box 440, Broadway, NSW 2007). Tel: 212-2780, 212-3780.

All advertising enquiries should be referred to the above address.

Printed by: Ambassador Press Pty Ltd, Parramatta Road and Good Street, Granville, NSW 2142.