DISTRIBUTED COMPUTING SYSTEMS

ANNUAL REPORT

Sept 78 - Sept 79

Issued by
The Computing Science Committee
of the
Science Research Council

1979

THE

COORDINATED PROGRAMME

OF

RESEARCH

IN

DISTRIBUTED COMPUTING SYSTEMS

ANNUAL REPORT

Sept 78 - Sept 79

Science Research Council

1979

Table of Contents

1 INTRODUCTION

This is the second Annual Report of the SRC's Distributed Computing Systems Programme. Its purpose is to:

(1) Outline the objectives of the Programme.

(2) Define the Programme's scope and the mechanisms for implementing it.

(3) Describe the projects to an audience outside the Programme.

(4) Report on the Programme's progress to September 79.

In particular, it is intended to stimulate contacts between researchers within the Programme and those in Industry, the Research Establishments and Universities in the UK and abroad who are working in or interested in Distributed Computing.

2 BACKGROUND

Projects related to Distributed Computing Systems have for some time been supported by the SRC's Computing Science Committee in its conventional responsive mode. The Committee recognised the importance of the area and appointed a Panel in June 1976 to consider what action, if any, was necessary to encourage, coordinate or direct research into Distributed Computing. This Panel was asked to take particular account of the potentially high cost of research and the possibility of unnecessary duplication of effort.

In its report to the Committee in October 1976, the Panel recommended that a coordinated research programme should be established and that additional funds be sought for the programme. When a draft programme was circulated to relevant academic departments for comment, more than 50 replies were received, the great majority expressing a desire to participate and offering useful criticisms of the proposed mechanics and content of the programme. A one-day Workshop (hosted by the Department of Computing and Control of Imperial College) was held in March 1977 and provided an opportunity for the 32 departments represented for a direct communication of views on the original proposal and on the research problems to be addressed.

The Panel, in response to these helpful interchanges of views, revised a number of its original proposals, particularly in respect of the topics for research and the detailed mechanisms for coordination.

The modified proposal to set up a coordinated programme of research into Distributed Computing Systems was warmly welcomed by the Engineering Board of SRC. Approval in principle for the revised programme was given by the Board in June 1977. The Distributed Computing Programme was therefore initiated in academic year 1977-78. It is now funded to a total of 2.3M pounds and is expected to run until 1984.

3 OBJECTIVES OF THE PROGRAMME

The primary objectives of the programme are to seek an understanding of the principles of Distributed Computing Systems and to establish the engineering techniques necessary to implement such systems effectively. In particular, this requires an understanding of the implications of parallelism in information processing systems and storage, and devising means for taking advantage of this capability.

The more general objectives of the programme may be described as follows:

(1) To achieve results of practical value to UK industry by directing research to a key area for the future.

(2) To promote relevant Computing Science research of high quality in a positive manner in academic departments by coordinating the efforts and achievements of individual research teams.

(3) To ensure the best use of funds at a time of financial stringency.

Within the context of the proposed programme, a Distributed Computing System is considered to be one in which there are a number of autonomous but interacting computers cooperating on a common problem. The essential feature of such a system is that it contains multiple control paths executing different parts of a program and interacting with each other. Such systems might consist of any number of autonomous units, but it is anticipated that the more challenging problems will involve a large number of units. Thus, the spectrum of Distributed Computing Systems includes networks of conventional computers, systems containing sets of microprocessors and novel forms of highly parallel computer architecture with greater integration of processing and storage.

The motivations for and importance of research into distributed computing systems are many and varied. The major ones are:

(1) Performance: eventually it will be impossible to increase the speed of a single processor and retain commercial viability. Several processors, cooperating on a single task, will be the only way to greatly enhance performance.

(2) Reliability: a fully distributed system should be able to tolerate faults caused by either software or hardware. Hardware faults might be tolerated by having more than one of each critical element. Software faults might be reduced by running different algorithms in parallel and checking the validity of results.

(3) Clarity: many problems are naturally parallel. Some problems are inherently simpler if expressed as a set of interconnected and communicating processes. If a problem's solution is expressed in this way it might be easier to provide a proof of correctness for the whole solution by breaking the proof task down into proving the correctness of individual processes and then proving the correctness of their interconnection, as opposed to a 'monolithic' approach.

(4) Distribution: in areas such as real time control it is often important that processor power is available where it is required in order to minimise the bandwidth requirements of data paths.

(5) Cost: the low cost of microprocessors will allow certain tasks to be performed more economically on sets of microprocessors than on a single main frame processor.

4 RESEARCH TOPICS

The Panel has categorised the research into five major topic areas, representing a progression from fundamental theory to socio-economic implications and novel applications. The areas are:

(1) Theory and Languages

(2) Resource Management

(3) Architecture

(4) Operational Attributes

(5) Design, Implementation and Application

Each of these topics represents a perspective of Distributed Computing Systems from a different viewpoint and each researcher should find one such viewpoint most relevant to his experience and way of thinking. All the perspectives converge on the central problem of parallelism and its implementation, and as a result there is often an overlap between the topics even though the terminology is different.

The topics are intended as a guide for investigators and are not a definitive description of the research problems, nor of their potential solutions. At this time no priorities have been ascribed. The Panel has, however, tried to maintain an overall balance within the programme. Each area is discussed in greater detail below.

Theory and Languages

The creation of an adequate theoretical basis for Distributed Computing Systems is regarded as having great importance, in particular to underpin work on language and system design. The problems revolve around establishing adequate methods for representing complex asynchronous systems and the identification, formal definition and mathematical analysis of the properties of such systems. It is also expected that such work could have intrinsic merits by extending the notions of formal systems. Two projects at Edinburgh University (Milner and Plotkin) are involved with defining a suitable framework to represent concurrent computation and to study the foundations of such computations. Dr Lauer of Newcastle University has developed a formalism (the COSY notation) for describing systems as a set of sequential processes and distinct resources. Systems can be analysed

for their adequacy, non-starvation, degree of concurrency and distribution.

Research is required to identify methods of representing concurrency within programming languages and to evaluate the effect of asynchronism at different levels. There is a need to investigate the extent to which concurrency needs to be implicit or explicit and to determine the constraints a language should place on concurrency to ensure good programming practice.

Present programming languages are highly sequential. Concurrency, where it is achieved, is usually obtained by the use of operating system facilities not included in the programming language. There is also a requirement to consider how existing programming languages need to be modified or extended in order to provide facilities in a distributed environment. The group at Warwick University (Whitby-Strevens and May) have developed a programming language, EPL, which can be used to write programs to be mapped onto a number of processors. A major issue is whether the creation of processes and their allocation to processors should be static or dynamic.

Hoare at Oxford and Coleman and Hughes at UMIST are investigating the techniques of programming by interconnecting networks of commmunicating processes to achieve simpler solutions which are more amenable to proof as well as efficient, parallel implementations. More radical approaches to the expression of provable parallel programs are the single assignment language ideas of Wadge and the data flow schemata.

Resource Management

Distributed Computing Systems do not necessarily eliminate the need for operating systems. The pressure to over-optimise operating systems code may be reduced but new dimensions are added to the problem of resource management. The most important are distribution of control and allocation of logical and physical resources, the scheduling of resources given the additional constraints of access, delay and bandwidth to remote resources, and the organisation of heterarchical systems in which there is no fixed delegation of control. Dr L Casey and N Shelness of Edinburgh University have developed a distributed domain model for systems with no shared memory which allows processes to migrate from one processor to another as a computation proceeds.

Access to information controlled and organised by other processors is a key problem. Distributed Computing Systems exacerbate many of the current database problems, such as control of access, consistency and naming, as well as providing a more severe environment for access synchronisation.

Two contrasting approaches are emerging. One is the server, a concentrated resource such as a filestore, which is accessed by or 'serves' a community of users. Examples are the file servers at Edinburgh and Cambridge. The second, contrasting approach is the truly distributed resource. For example, Dr Bennett at Keele is investigating a distributed filestore which appears as a conventional, local filestore to the user but is implemented as a system wide set of migrating files.

Architecture

A variety of new architectural organisations using multiple computing elements are being evaluated. The use of a collection of computers or micro-computers to provide facilities similar to those on a conventional main frame computer, by distributing identified operating system and user tasks to individual computer elements, is regarded as an intermediate stage of development. Such developments will enable existing operating systems and software concepts to be implemented with a greater degree of parallelism on specially organised hardware.

A variety of different architectures are being considered within the programme. N Shelness has a 3-processor system based on Argus 700G computers with an Argus 700F processor acting as an interface to the interconnection bus. This will allow a variety of connections to be simulated. Prof Wilkes and Dr Needham at Cambridge University have developed a wide-band digital communication ring attached to which are a number of processors for specific functions. Prof Aspinall at UMIST is investigating the properties of a 16 processor shared memory system.

A collection of computing elements organised into a regular array, or some other structure, might also be used to provide a general computing facility. Research is necessary to determine the desirable properties of such elements and the form of their intercommunication and interconnection. The potential efficiency of such systems needs to be studied, together with methods of organising computations to exploit the structure and the types of computation for which they are appropriate. It may also be possible to increase the degree of parallelism within a processing element, for example, by the use of data-flow techniques. Four groups (Gurd, Treleaven, Sleep, Osmon) are looking at non-von Neumann architectures. Drs Gurd and Watson at Manchester University are building a prototype ring-structured data flow computer system.

Distributed Computing Systems will increase the emphasis on interfaces and interworking. There is a need to evaluate communication protocols, like X25, and to place the design of intercommunication techniques on a sound theoretical footing. The relevance of telecommunication protocols and principles requires investigation. For the longer term, there is a need for a more general consideration of the requirements for effective interchange of information between computers. Within the programme, Dr Sloman at Imperial University is particularly interested in how low-cost X25 chips, available shortly, can be used in real-time applications. Dr Paker at PCL is developing modelling techniques to help understand the problems of complex intercommunication.

Operational Attributes

Distributed Computing changes many aspects of system reliability in that the provision of parallelism poses new problems whilst solving others. General methods of using distributed techniques to enhance reliability need investigation. Areas requiring re-consideration in the new context are error detection, damage assessment, fault treatment and error recovery, particularly with respect to performance, extensibility and flexibility. Theoretical and practical studies to define and measure the performance of Distributed Computing Systems are also required. The major grant in this

area is the one awarded to Prof Randell at Newcastle University to study the reliability and integrity of distributed computing systems.

Design, Implementation and Application

Hardware and software techniques for developing and implementing Distributed Computing Systems are needed. The availability of low-cost computing may facilitate the implementation of more sympathetic interfaces for the human user. The study and evaluation of the human interface is therefore regarded as an important aspect for the effective design of Distributed Computing Systems.

The approach to implementing computer applications may be changed by the advent of Distributed Computing Systems. Comparative studies are required to determine the relative advantages of distributed and centralised techniques in various areas. The use of Distributed Computing Systems may enable the application of computing techniques in new areas. Theoretical and practical studies are needed to assess the potential of new applications, particularly in the areas of instrumentation and office automation. Distributed Computing Systems can change the cost of computing and the way in which it can be delivered to the user. Such changes may be expected to have long-term social and economic consequences which need assessment. Two grants at Queen Mary College are aimed specifically at office automation. Prof Coulouris is interested in effective man-machine communication and is trying to identify the hardware and software requirements for highly interactive information processing systems. He is using the 'paperless office' as a vehicle for this research. I Page has a grant to develop a high quality display system for this application.

Improved techniques for specifying the requirements, by a language or otherwise, which a given Distributed System is to meet are needed, as well as improved hardware and software techniques for translating this specification into a complete system.

The modularity inherent in Distributed Computing Systems will provide for improved implementation. However, new problems will arise in system testing, and investigations will, therefore, be required into practical methodologies for testing.

5 MANAGEMENT OF THE PROGRAMME

The arguments for adopting a coordinated approach in the case of Distributed Computing Systems are two-fold:

(1) The importance of the subject to the future of computing. This is considered by the Committee to be an area in which academic research can be of help to industry in taking advantage of the recent developments in semi-conductor technology. A coordinated approach will ensure both a reasonable balance of SRC support to the whole area, and a framework for the take-up of results by commerce, industry and Government Research Establishments.

(2) The limited funds available to the Committee, together with the relatively high cost of supporting research in this field, make it essential to provide support for the area in a cost effective way, whilst preserving the necessary freedom of investigators to pursue fundamental research.

It is the Committee's intention that coordination be achieved at two levels. Firstly, within the programme itself, by working out proposals with investigators prior to the submission of a formal application and by holding regular Workshops to review progress on particular topics, disseminate information and promote communication and collaboration between individual research teams. Secondly, by developing an ongoing coordination of effort between the SRC programme, the Government Establishments, industrial interests, and any EEC initiatives , to ensure that the programme is placed in a proper national context.

The general principle adopted is that of a positive programme of research into Distributed Computing Systems, managed by a Panel appointed by the Computing Science Committee, with coordinators appointed to assist the Panel in its task. The Panel has no delegated powers and acts as an advisory body to the Committee in all matters related to the programme, in particular by making formal recommendations to the Committee in respect of individual research grant applications.

The coordinators have been appointed on a part-time basis. They have responsibility for maintaining close liaison between the Panel and investigators, and for monitoring progress on the research projects within the programme. The industrial coordinator has, in addition, responsibility for promoting contact between investigators and appropriate research teams within the Government Research Establishments and with potential commercial and industrial end-users, with the objective of ensuring a high degree of technology transfer.

The coordinators report directly to the Panel and act as 'roving technical ambassadors' for the Panel. They are not authorised to take financial, administrative or technical decisions which might affect the future directions of individual research projects. In other words the principle of peer judgement is maintained.

6 THE COORDINATION ACTIVITIES

The precise mechanics of coordination are evolving as the programme develops, but the essential features are as set out below.

Within the programme, as its scope is very broad, the degree of coordination appropriate to each research project varies according to its nature. Effort is concentrated on the following aspects:

(1) The minimisation of duplication of funding by holding informal discussions with investigators prior to submission of formal applications.

(2) The promotion of ongoing liaison betwen research teams and the Panel, via the coordinator.

(3) The ongoing review of the objectives of each project in the light of the results obtained, and the progress of the programme as a whole.

(4) The review of progress, dissemination of information and promotion of communication and collaboration between investigators via the regular SIG meetings, Workshops, Colloquia, Mailshots and communications facilities (see Appendix).

The Committee also wishes to foster industrial interest in the programme, with the objectives of promoting collaborative projects and facilitating a high degree of technology transfer. It is felt that this may be best achieved via:

(1) Invitation of interested researchers funded by other bodies to appropriate technical meetings and conferences run by the programme.

(2) Development of informal links between specific companies and research groups.

(3) Subsequent development of links such as research contracts or collaborative programmes between University and Industrial or government organisations.

The industrial coordinator's role is to provide information on the research programme and to advise on mechanisms available for supporting collaborative research.

Within the SRC itself, there is liaison with the recently created research programme on Information and Communication Systems, run by the Communication and Data Systems Sub Committee of the Electrical and Systems Engineering Committee. The Programme also enjoys close links with the Joint Network Team, collaboratively funded by the SRC and the Computer Board.

The Coordinators, Prof Hopgood and Mrs Ringland, made a significant contribution to the Programme by establishing a style and scale of coordination activity that has become the model and the desired standard for other specially promoted programmes. During 1979 Prof Hopgood became the head of the Rutherford Laboratory's Computing Division and Mrs Ringland joined Inmos Ltd. They have been replaced by Mr Witty of the Rutherford Lab. and Mr Tucker of Logica. Dr Duce was appointed Technical Secretary. Dr Wharton acted as Panel Secretary until the end of August 79 when Mr Monniot assumed this responsibility.

7 RUTHERFORD LABORATORY SUPPORT

The SRC supports academic research by awarding grants to investigators for staff, equipment and travel. Investigators may also apply for practical support from the various SRC laboratories which provide specialised services and facilities. In the context of DCS research, investigators may apply to use such SRC facilities as the FR80 colour microfilm recorder and phototypesetter, large mainframe computers such as 360/195s, 3032, DEC-10, interactive computers such as Primes and GECs, a Unix system, a national communications network, an Electron Beam Lithography unit and CAD packages for ciruit design. Investigators may ask SRC laboratories to help them design and build experimental hardware and software. For example, the SRC's Rutherford Laboratory (RL) has, via its CAD software facility, been helping Cambridge produce the LSI version of the Cambridge Ring. Joint SRC laboratory/universtiy research projects may also be established.

The DCS Programme has been supported by the RL since January 1978. The Academic Coordinator and the Technical Secretary are on the staff of the Rutherford Laboratory. The RL has placed EMR contracts (see appendix), on behalf of the Programme, for Unix X25 communications software and a Pascalplus compiler. The RL has purchased, distributed and maintained a pool of equipment, for the use of DCS investigators, which currently contains 51 items worth over 144K pounds (see appendix).

8 SUMMARY OF PROGRESS, SEPT 78 - SEPT 79

As of September 78 the DCS Programme consisted of 24 projects totalling 1.4M pounds. During the year Sept 78-Sept 79 the SRC, acting on the DCS Panel's recommendations, awarded a further 16 grants and 2 SVFs worth 906K pounds, bringing the Programme up to 40 grants (31 distinct projects) totalling 2.3M pounds. The DCS Programme was originally allocated 1.6M pounds. The extra funds awarded are an indication of the importance attached to the DCS initiative.

With 4 grants awarded in 1977, 17 in 1978 and 16 in 1979 the Programme overall is very much in its initial phase with the majority of projects only just getting into their stride. It is thus unreasonable to expect too many concrete results to have been produced as yet. However, progress has been substantial (see individual progress reports in the appendices) in spite of the general difficulty of obtaining suitable numbers of good research staff.

Amongst the more established groups significant progress is easier to identify. Such examples include the successful commissioning of the 16 processor CYBA-M system by Prof Aspinall's team despite the move from Swansea to UMIST; Prof Randell's team were paid the compliment of being invited to California to present their course on reliability; York's Modula compiler has been distributed world wide and has been selected for the KSOS secure operating system project and an industrial project; Mr Shelness is well on the way to producing the DCS's first implementation of a dynamic process/processor system; Prof Evans' discovery of new algorithms suitable for parallel execution has led to the award of a more powerful four processor system for Loughborough University and the theoretical projects of Drs Lauer, Plotkin and Milner have continued to produce papers of the highest quality.

In conclusion, the end of the 1978/79 academic year has seen the DCS Programme established as a substantial, coordinated portfolio of research projects well set to produce significant results during the next few years.

9 APPENDICES

9.1 RESEARCH GRANTS

Investigators and Titles

Prof D Aspinall, Dr E L Dagless
The Use of Microprocessors in Information Processing Systems.

Dr K H Bennett
A Feasibility Study of Loosely Linked Computers.

Dr K H Bennett
A Distributed Filestore.

S E Binns, Prof P J Brown, Dr E B Spratt
Compiling Servers for the Cambridge Ring (at Kent).

D Coleman, J W Hughes
Developing a Program Methodology for Multiprograms.

Prof G F Coulouris
Distributed System Requirements for Effective Man-Machine Interaction.

Dr R D Dowsing, (Dr P W Grant)
The Specification and Implementation of Programs on a Multi-microprocessor.

Prof D J Evans, Dr I A Newman
An Investigation of the Relationship between Algorithm Structure and Parallel Architectures.

Prof D J Evans, Dr I A Newman, Dr M C Woodward
A Research Vehicle for Investigating the Use of Closely Coupled Distributed Systems.

Dr P W Grant, (Dr R D Dowsing)
The Specification and Implementation of Programs on a Multi-microprocessor.

Prof R L Grimsdale, Dr F Halsall
The Design and Implementation of a Multi-microprocessor System.

Dr J R Gurd, Dr I Watson
A Ring-Structured Data Flow Computer System.

Dr F K Hanna
Distributed Processing Systems for Interactive Knowledge Bases.

S E Hersom
Development of Optimisation Algorithms for Parallel Computation.

Prof C A R Hoare, J E Stoy
Software Engineering.

Prof C A R Hoare
Workstations for Software Engineering.

Dr J R W Hunter, Dr K D Baker, Dr A Sloman
Interactive Software Tools for Distributed Computing Systems with an Application to Picture Interpretation.

Prof P T Kirstein
Communication Protocols in the Context of X25 Computer Networks.

Prof P T Kirstein
Communication Protocols in the Context of X25 Computer Networks - Supplementary Proposal.

Dr P E Lauer
Design and Analysis of Highly Parallel Distributed Systems.

A J R G Milner
Applications of Flow Algebras to Problems in Concurrent Computation.

Dr I Mitrani
Modelling and Performance Evaluation for Distributed Computing Systems.

Dr P E Osmon
Implementation of a High Level Data Flow Programming Language.

Dr P E Osmon
Provision of Facilities for Implementation of a Data Flow Programming Language.

I Page
A High Quality Display for Effective Man-Machine Interaction.

Dr Y Paker
Computer Aided Multi-microprocessor Systems Modelling, Simulation and Performance Evaluation.

Dr G D Plotkin, A J R G Milner
Semantics of Non-Deterministic and Concurrent Computation.

Prof I C Pyle
Real Time Programming Languages for Industrial and Scientific Process Control.

Prof B Randell
Reliability and Integrity of Distributed Computing Systems.

Prof B Randell
A Project to Investigate the Design of Highly Concurrent General-Purpose Computing Systems.

N H Shelness
An Architecture for a Multiple Computer System.

Dr M R Sleep
Instruction Sets for Data Flow Architectures: A Comparative Study.

Dr M S Sloman
Communications for Distributed Process Control.

Dr W W Wadge
Distributed Implementation of Nonprocedural Languages.

Dr I C Wand
MODULA Distribution and Promulgation.

Dr I C Wand
MODULA DIstribution and Promulgation - Supplementary Proposal.

Dr I C Wand
Distributed Operating System for Time-Sharing.

Dr J Welsh
The Design, Implementation and Application of Languages for Distributed Computing.

Dr C Whitby-Strevens, M D May
A Building-Block System for Distributed Computing.

Prof M V Wilkes, Dr R M Needham
Distributed Computing using Wide Band Communications.

Funding

GRANT HOLDER	AWARD K pounds	DURATION	RAs
Prof D Aspinall, Dr R D Dowsing and Dr E L Dagless	90.0	Oct 77-Sept 79	6
Prof D Aspinall, Dr E Dagless	257.6	Oct 79-Sept 83*	6
Dr K H Bennett	6.5	Oct 78-Sept 79	1
Dr K H Bennett	67.5	Oct 79-Sept 82	2
S E Binns, Prof P J Brown, and Dr E B Spratt	10.1	July 79-March80	EO
D Coleman, J W Hughes	12.5	Oct 78-Sept 80	1
Prof G F Coulouris	72.0	Oct 77-Sept 80	2
Dr R Dowsing	51.3	Oct 79-Sept 82	2
Prof D J Evans, Dr I A Newman	19.0	Jan 79-Dec 81	1
Prof D J Evans, Dr I A Newman and Dr M C Woodward	105.7	April79-March82	2
Dr P W Grant	1.4	Oct 79-Sept 82	0
Prof R D Grimsdale, Dr F Halsall	127.3	Oct 79-Sept 83	3
Dr J R Gurd, Dr I Watson	71.5	July 78-Sept 81	2
Dr F K Hanna	25.0	Oct 78-Sept 81	1
S E Hersom	10.0	Oct 79-Sept 81	0
Prof C A R Hoare, J E Stoy	81.0	April78-March82*	2
Prof C A R Hoare	48.6	July 79-March80	EO
Dr J R W Hunter, Dr K D Baker and Dr A Sloman	66.9	April79-March82	2
Prof P T Kirstein	205.0	Oct 78-Sept 82*	4
Prof P T Kirstein	10.0	July 79-March80	EO
Dr P E Lauer	37.5	Jan 79-Dec 81	2
A J R G Milner	9.0	Jan 78-Sept 79	1
Dr I Mitrani	14.0	Oct 78-Sept 80	1
Dr P E Osmon	23.0	July 78-Sept 81	1
Dr P E Osmon	6.8	July 79-March80	EO
I Page	25.5	July 78-June 80	1
Dr Y Paker	26.3	April79-March81	1
Dr G D Plotkin, A J R G Milner	39.0	Oct 78-Sept 81	2
Prof I C Pyle	60.5	April75-Dec 78	3
Prof B Randell	226.5	Oct 78-Sept 82*	5
Prof B Randell	13.5	July 78-Sept 80	1
N H Shelness	89.5	Oct 77-Sept 80	3
Dr M R Sleep	1.5	Oct 78-Sept 79	0
Dr M S Sloman	55.0	Sept 78-Aug 81	1
Dr W W Wadge	2.3	April79-April80	0
Dr I C Wand	15.5	Jan 78-Dec 80	1
Dr I C Wand	6.6	July 79-March80	EO
Dr I C Wand	27.8	Oct 79-Sept 81	2
Dr J Welsh	84.7	Oct 79-Sept 81	2
Dr C Whitby-Strevens, M D May	123.0	Oct 77-Sept 82*	3
Prof M V Wilkes, Dr R M Needham	34.0	Aug 78-July 81	1
Total	2260.4		

* denotes a 4 year rolling grant.
EO denotes a grant which provides funds only for the purchase of equipment

Individual Project Descriptions

PROF D ASPINALL and DR E L DAGLESS

UNIVERSITY OF MANCHESTER INSTITUTE OF SCIENCE AND TECHNOLOGY

THE USE OF MICROPROCESSORS IN INFORMATION PROCESSING SYSTEMS
Oct 77 - Sept 83*

The main objective of this research programme has been to create a multi-microprocessor development environment to permit the study of distributed processing. The facility comprises a machine, CYBA-M, a design notation, CYBA-L, debugging and testing aids, a performance monitor and a documentation system. This facility provides a foundation for a wide range of performance and application studies to be researched aimed at providing a better understanding of how to design products based on future VLSI microelectronics.

The machine, CYBA-M, comprises sixteen 8080 microprocessors each having access to a private memory of 16K bytes, a common global memory of 16K bytes, and a shared image memory to which peripherals are attached. The processors are controlled from the image memory and are considered as resources just like any other hardware peripheral device. One thousand process flags, or binary semaphores, which are set when writing, cleared when read, are located in the image memory to resolve any contention for a software resource. Each peripheral has a similar facility for resolving contention for a hardware resource. Serial and parallel input/output devices are operational and a 5M byte disc is due to be commissioned early in 1980. The machine has been operational with sixteen processors for over a year.

The design notation, CYBA-L, consists of a control driven scheme that activates interconnected data modules. Translators exist which convert the design into STAB-12 objects (STAB is a derivative of BCPL) which are interpreted by a simulator running on a PDP-11 or by CYBA-M. Other forms of translator systems for CYBA-L are being explored and alternative languages are being investigated.

The program development system is provided by hardware in CYBA-M, a separate control language (JCL), which runs on the command processor. This system allows programs created on the PDP-11 to be loaded from floppy discs into CYBA-M. Running of the program, modification and examination of the machine state are effected by the user using the JCL. The JCL can load binary, hex and STAB objects from RT-11 generated floppy discs. Remote user access will be provided in the near future enabling dial-up use of CYBA-M.

A performance monitor is being developed for studying the activity of both hardware and software events in the multiprocessor.

Much effort in the last year has been expended in producing documentation for supporting the hardware and software facilities. The machine documentation has been kept to a high professional standard and maintenance documentation for software is being upgraded. All the machine specifications and software reports and documents generated by the group are on disc under a UNIX system provided by the DCS. This system is to be enhanced by an organised directory and indexing system for all material,

including drawings, created on the project. User manuals for the services and facilities available to outside users will be made available in the Spring of 1980.

A few small applications programs have been developed. The basic strategy is based on a static mapping of processes to processors, where the number of processes is less than sixteen. The investigators' interests lie in understanding design methodologies for creating a specified product whose function is defined by a program which may be stored in a read only memory. Instrumentation and process control systems are typical of the application areas presently being examined.

PROGRESS TO SEPT 79

The CYBA-M project began at Swansea University. The main design and hardware construction efforts took place there. During the 1978/79 academic year the three main investigators moved from Swansea - Prof Aspinall and Dr Dagless to UMIST, and Dr Dowsing to East Anglia. CYBA-M itself was moved to UMIST and has been recommissioned. To date, the system is working with a full complement of 16 processors. The successful establishment of the CYBA-M project in UMIST and recommissioning the hardware have been the year's major activities. Funds have recently been made available to build up the software team so that the potential of the 16 processor vehicle may be fully exploited.

STAFF

The following staff were employed on the project:

J Nordvedt	May 1974 - April 1975
J Proudfoot	May 1974 - Sept 1977
B Davies	Oct 1975 - Sept 1979
D Harvey	Oct 1975 - Sept 1979
B Balogh	Oct 1975 - March 1977
M Edwards	Oct 1977 - Sept 1979
N Graves	Oct 1977 - March 1978
A Munro	Oct 1977 - present
P Burkimsher	Feb 1979 - present

Dr M Edwards is now an SRC research fellow working in conjunction with the group.

Four new appointments have recently been made and they will start in December 1979 and January 1980.

REFERENCES

(1) D.Aspinall & E.L.Dagless, UMIST, 'Overview of Multi-Microprocessor Development Environment'. Microprocessors and Microsystems, Vol 3, No 7, Sept 1979.

(2) R.Dowsing, 'Software for CYBA-M: The Design and Implementation of Software for the Multimicroprocessor System CYBA-M'. Microprocessors and Microsystems, Vol 3, No 7, Sept 1979.

(3) E.L.Dagless, 'A Multimicroprocessor - CYBA-M'. Information Processing 77, B Gilchrist (ed), IFIP North Holland Pub. Co. (1977).

(4) D.Aspinall, E.L.Dagless & R.D Dowsing, 'Design Methods for Digital Systems including Parallelism'. Electronic Circuits and Systems, Jan 1977, Vol 1, No 2.

(5) R.D Dowsing & E.L.Dagless, 'Design Methods for Digital Systems: I - Concurrency Constructs'. Computers & Digital Techniques, June 1979, Vol 2, No 3.

(6) R.D Dowsing and E.L.Dagless, 'Design Methods for Digital Systems: II - Structures Involving Control and Data'. Computers and Digital Techniques, August 1979, Vol 2, No 4.

(7) R.D Dowsing and E.L.Dagless, 'Control and Debugging in the CYBA-M Multimicroprocessor System'. 1st European Conference on Parallel and Distributed Processing, Toulouse, France, Feb 1979.

(8) L.Pugh, 'Colour Television Graphics Simulation'. Microprocessors and Microsystems, Vol 3, No 7, Sept 1979.

The above papers are available as a single bound report, available from the principal investigators.

DR K H BENNETT

KEELE UNIVERSITY

A FEASIBILITY STUDY OF LOOSELY LINKED COMPUTERS
Oct 78 - Sept 79

This project, a one year feasibility study, concentrated on distributed computer systems in which several computers are logically and physically interconnected; the machines cooperate under decentralised control to execute application programs. It has been particularly interested in MIMD systems which have two important properties: firstly, the system is homogeneous in that all sites support the same basic executive using the same primitive level interface; and secondly, the system as a whole appears as a black box to the user, who is unaware of its internal structure (ie there is a single system-wide operating system).

A further feature is that intersite communication is along so-called `thin-wire` (relatively low band width) channels, in contrast to high band width `thick-wire` channels, where physically the link is a parallel bus. Thus the location of machines is permitted to be geographically separate if required.

The near future is likely to bring a complete 32 bit processor, with substantial primary memory, on the same circuit board (and perhaps in the more distant future, on the same chip). For many users and applications, the computer is I/O bound, not CPU bound, and technology will soon provide them with all the computational power they require at negligible cost compared to the rest of the installation. For this type of user, the key shareable resource is data.

PROGRESS TO SEPT 79

When the investigators began the study, they were anxious to keep in mind the potential applications for the type of distributed system in which they were interested. Hence they started by investigating such possible applications, especially in commercial D.P. The main conclusions were:

(1) Almost all existing commercial 'distributed' systems are of the star network type.

(2) A very strong motivating factor for the adoption of this type of network is the current trend to decentralised management.

(3) There is a growing need for 'rim' sites to intercommunicate without involving the central machine.

(4) Almost all commercial systems are being developed in an ad hoc way; almost all are resource sharing networks rather than true distributed systems (ie with distributed control).

(5) A growing emphasis is being placed on availability and access to computers, especially for on-line work. Like the telephone, effectiveness is not necessarily measured in per cent utilisation of the equipment.

They were concerned about (4) and subsequently they investigated suitable architectural models for system-wide operating systems. They have found that the capability/domain model has many attractive features and are currently giving attention to the use of this model in applications in which data management is of major concern in distributed systems.

It had been the intention to develop analytical and simulation models of such distributed systems. Due to staff shortages, this aspect of the work has been delayed but has now restarted.

Nevertheless, this feasibility study has enabled them to isolate and begin to investigate a fruitful area for further research, and they have successfully applied to the SRC for funding to enable them to undertake a further three year`s work on the modelling, design and implementation of a distributed filestore (see following project description).

Nineteen internal reports and working documents were produced, plus ten trip reports.

DR K H BENNETT

KEELE UNIVERSITY

A DISTRIBUTED FILESTORE
Oct 79 - Sept 82

This project has been motivated by studies carried out during a recent feasibility study of loosely linked computers, supported by the SRC under a one-year DCS grant (GR/A/74746, expiring 30 September 1979).

The direction of research has been strongly influenced by the investigation of potential applications for distributed systems. At the present time, almost all existing commercial 'distributed' systems are of the star network type, and design, development and expansion are being carried out largely in an ad hoc way (eg by adapting RJE type protocols). There is a growing need for sites in these systems to intercommunicate without involving a central machine. For example, banks and insurance companies are moving towards storing local data at regional offices instead of maintaining a single centralised data bank. Many enquiries will need purely local access to data stored at the branches. At the other end of the scale, the 'automated office' will have analogous requirements.

The key characteristic of this type of application is that the data required at any site will be distributed throughout the whole system. It is this aspect that is to be studied, and this project is concerned with the management of data as a resource in a distributed computer system with decentralised control. The specific application to be studied is the design, implementation and evaluation of a distributed filestore, to support an interactive Pascal system.

The project envisages a number of interlinked sites, each with its own filestore, CPU and main memory, and user terminals. The feature of the system is that it appears as a single 'black box' to the user, who is unaware both of the location of his files and the internal structure of the box.

Dr Bennett has chosen Pascal because there is already considerable implementation expertise at Keele, and because it is Keele's major teaching language there will be a large user population available when they evaluate their design.

This type of system is representative of several multi-minicomputer configurations that are being installed at some Universities. More generally, one of course would anticipate that these research results will be applicable to larger scale applications.

The proposed three year research programme will be in two phases, analogous to those of a classical scientific investigation. In the first phase, Keele propose to establish a theoretical model of a design, using simulation and analytical modelling as aids. The second phase will be to implement the design and to validate it by performance monitoring under conditions of practical use.

S E BINNS, PROF P J BROWN, DR E B SPRATT

KENT UNIVERSITY

COMPILING SERVERS FOR THE CAMBRIDGE RING (AT KENT)
July 79 - March 80

This award is for equipment, consisting of 3 microcomputers, to allow the Kent team to investigate the design and construction problems associated with building a special purpose Pascal compiling server to act as a resource available via the Kent implementation of the Cambridge ring system.

D COLEMAN and J W HUGHES

UNIVERSITY OF MANCHESTER INSTITUTE OF SCIENCE AND TECHNOLOGY

DEVELOPING A PROGRAM METHODOLOGY FOR MULTIPROGRAMS
Oct 78 - Sept 80

The aim is to develop programming methods applicable to the construction of multiprograms (a program consisting of a set of time independent concurrent processes) and to discover when they should be used. The work is aimed at developing design methods in which parallelism is used to simplify the programming process. This is possible as very many programs are unnecessarily complex when expressed in a strictly sequential fashion. The intention is to produce efficient methods for developing correct and readable multiprograms. The group is interested in commercial applications and mapping these onto low cost microprocessor networks. They see their research as leading to the reduction of software costs, improvement of programmer performance and more cost-effective use of hardware. They hope to achieve this by using a systematic design process which leads to modular programs exhibiting a structure related to the problem. These programs would be implemented in a high level language, such as Concurrent Pascal, with a high degree of compile time protection for modules and it should be possible, using a rigorous and checkable design process, to aim for some degree of proving program correctness.

The design methodology is an extension of the Michael Jackson Method which considers programs from the viewpoint of the translations they perform between input and output data. The structure of such programs should parallel the transformations applied to their data. Whereas the Michael Jackson Method uses diagrams applicable only to data expressable as regular expressions, Coleman and Hughes use translation grammars applicable to data expressable as context-free languages. This is more rigorous and more powerful than the informal Jackson Method. For example, the standard stock update problem in commercial data processing has several inputs (master file, transactions etc) and several outputs (new master file, lineprinter record of transactions etc) and this is decomposed into about 12 processes forming several pipelines of cooperating sequential processes. A decomposition into multiple processes becomes appropriate if the translations are inherently complex, concern complex input/output structures, or involve multiple input/output streams.

PROGRESS TO SEPT 79

Programs consisting of pipelines of simple processes are characteristic of the method and reflect the use of processes to implement abstract data types and data access mechanisms. The use of processes is complementary to the use of classes (subroutines) to achieve structuring. The group have already implemented a number of small programs using these methods (for example, stock control, critical path, an assembler). These programs, each consisting of about a dozen processes, are written in Concurrent Pascal and developed under the Brinch Hansen SOLO System which has been transported on to the Department`s single processor Modular One computer.

This work showed the limitations of handcoding translation grammars and of using Concurrent Pascal as the implementation language. Consequently a translator writing system has been constructed as a tool for their future software development. The work also highlighted:

(1) the advantages of translations for identifying the processes suitable for pipelining (LL(1) grammars, L-attributed translations);

(2) the weakness of translation grammars in expressing semantic information;

(3) their limitations for describing processes with more than one input or output, or those which do not perform an L-attributed translation between streams (eg PERT, symbol table handling).

Simultaneously with the practical work, an investigation was mounted into the problems of verifying the correctness of pipeline programs. This led to a study of program transformations, functional programming languages and their relation to attributed translation grammars. The investigators are hopeful that this approach will produce a powerful method of verifying multiprograms, but much will have to be done before this is a practical possibility. Functional languages also overcome the limitations of attributed translations described above and could therefore prove useful in the design methodology as a supplement to attributed translations for specification. They are maintaining contact with people working in this area, eg John Darlington (IC), Martin Feather (Edinburgh).

The arrival of CYBA-M at UMIST has meant that the investigators have access to a multiprocessor computer. They are currently in the initial stages of applying this methodology to design a software system for the machine. This will provide the necessary practical experience to evaluate their design methodology.

The group sees the result of their research as the development of a practical methodology for the design of multiprograms with the necessary theoretical foundations to allow constructive proofs for multiprograms. A high level language in support of the methodology is being developed based on the translator writing notation.

REFERENCES

(1) D.Coleman, J.W.Hughes, M.S.Powell, 'Developing a Programming Methodology for Multiprograms'. Computation Departmental Report 218, March 1978.

(2) J.W.Hughes, 'A Formalisation and Explication of the Michael Jackson Method of Program Design', Software - Practice and Experience, vol 9, pp 191-202.

(3) D.Coleman, J.W.Hughes, M.Powell, 'A Method for the Syntax Directed Design of Multiprograms', UMIST Computation Departmental Report 243, Sept 1979.

(4) D.Coleman, J.W.Hughes, M.Powell, 'Engineering Data Processing Programs for Multiple Microprocessors', Microprocessors and Microsystems, vol 3, no 2, March 1979, pp 83-85.

PROF G F COULOURIS

QUEEN MARY COLLEGE, LONDON

DISTRIBUTED SYSTEM REQUIREMENTS FOR EFFECTIVE MAN-MACHINE INTERACTION
Oct 77 - Sept 80

The project is motivated by a goal that can be characterised at least in part by the concept of the 'paperless office' or the 'integrated personal information processing system' which enables electronic processing to be effectively applied to most everyday information-based activities.

The main aim of the research is to identify the hardware and software requirements for the development and operation of highly interactive information processing systems based on distributed microprocessors. The automated office will be used as the test-bed for the evaluation of results.

The results will be presented as papers describing a philosophy and techniques suitable for use in a wide range of interactive applications, including software and hardware methods for their effective implementation. Eventually it is intended to collect the results together in the form of a handbook for designers of interactive systems based on distributed microcomputers.

There are a number of subsidiary goals motivated by the intention to evaluate the requirements in the context of a real system running one or more trial applications. The subsidiary goals include:

(1) To establish criteria for evaluating the 'ease of use' of interactive information systems.

(2) To investigate the design of a file management and inter-process communication system suitable for a set of distributed workstations incorporating local file stores.

(3) To investigate techniques for the distributed execution of application and system functions in interactive systems amongst (a small number of) processors in a workstation and for the distribution of application processes amongst several workstations in a network.

Achievement of these goals calls for the development of a network of workstations (personal computers). The user characteristics of such workstations, both with respect to the interactive hardware and software techniques used and with respect to the 'user model' supported must be very much more effective than the usual interactive computer system.

Work already completed includes the design and implementation of an interactive coloured text display with dynamic windowing. This display will be used in the initial experiments on workstation software. A companion research project under I.Page, also funded by the Distributed Systems Panel, is expected to lead to the development of an exceptionally effective text and graphical display for use in office workstations.

A low-cost broadcast packet network (CNet) has also been developed and is now in service use in the Laboratory.

A workstation provides a dedicated processing facility available with minimum delay and at low cost. These characteristics are required in interactive applications to support a sophisticated model of the application constructs and processes as the user conceives them. 'Instant processing' is required to enable changes in information in the application context to be reflected by immediate changes in displayed information, and to enable information to be rapidly retrieved and displayed in any of several possible formats (for example, abstracted, page formatted, selectively filtered etc).

Perhaps the most challenging problems arise in the design of a general-purpose personal information system to run on the workstations. These problems are being attacked on several fronts, as an activity in parallel (with strong cross-links) with the hardware and system developments for workstations. Work is in hand at several levels. The use of displays and interactive techniques has received much attention producing some generally applicable results on windowing and menu selection. An experimental system that supports a data structuring concept analogous to the 'form' as it is used in paper-based information systems has been developed and is being evaluated and extended.

A more ambitious activity is aimed at the development of an 'intelligent' data base for office information. In this model, every information entity (form) is an active process, able to react to communication from processes representing forms and other activities. The processes are defined in a high-level notation, potentially suitable for use by a naive user wishing to define new tasks and activities in an office or other user environment.

Activities such as the latter impose heavy demands on the facilities of the workstation and the network and can, therefore, be expected to have a strong influence on the design of the relevant functions in a distributed workstation system.

PROGRESS TO SEPT 79

Progress has been made in the formulation of principles for user interface design, in display system design, and in formulating models for office information handling. This has led to the conclusion that the research vehicle originally proposed does not meet the requirements of these models, and that technological changes that have occured justify a re-assessment of the research vehicle.

Some studies of office activities aimed at gaining an understanding of formal and informal procedures in the office have been initiated.

In the area of display systems, experience gained with the Text Terminal is being ploughed into the specification of a more powerful character-map display more closely suited to their present requirements [6].

A target has been defined: to develop a useful office information system as a test-bed for the evaluation of hardware and software ideas. A research plan for achieving this goal is described in a report by Lamming [8].

The work done to date has been aimed at defining the constructs and the functions needed in an office environment and system techniques for their implementation. The problems have been divided into three separate but related areas for study:

(1) The formulation of models for information processing in the office and their realisation in experimental systems.

(2) The development and classification of techniques for constructing interactive systems suitable for the office environment.

(3) The specification and, where appropriate, the design and implementation of workstation and network hardware and software suitable for the effective application of the above techniques as they are developed.

In all three areas, current techniques are inappropriate or inadequate in some respects. It has therefore been necessary to pursue the three topics concurrently.

To date, work has been done on:

- The man-machine interface for text editing [15].

- The philosophy of man-machine interface design [16].

- Studies of office procedures [13,14].

- The formulation of models for office information handling [7,9]

- Workstation and network architecture for office systems [12].

- Display system design [6].

- Programming language and operating system design for workstations [4,5].

Studies have been done and reports or papers produced, based in some cases on experimental developments, in each of the above areas.

The conclusions on workstation hardware requirements have led the investigators to modify their views on the research vehicle. The original proposal specified the use of LSI-11 based workstations. In the event, funds were awarded for only one such station. Further studies have shown that the use of the LSI-11 would severely constrain the information processing models they have devised, and that a considerably more powerful and flexible workstation system is needed to provide a basis for work likely to remain relevant for a number of years. Since the LSI-11 has also already been overtaken in sophistication by current microprocessor architectures, the workstation has not been purchased.

Instead, a proposal was submitted in December 1978 for a major extension of the grant to enable them to develop and apply a research vehicle suitable for the next phase of the work [12]. This was not accepted, but the discussion that took place is likely to lead to a further proposal based on off-the-shelf hardware. Meanwhile, they have decided to produce an interim workstation system based on the PDP 11/34 purchased under the grant. To this end, they are investigating ways of extending or adapting it to their

immediate requirements, particularly by the addition of adequate display hardware and system development facilities.

During the period covered by this report, many visits have been made to the Computer Systems Laboratory by other researchers, both in the UK and particularly from the US. Many of these visits can be attributed to the success of a workshop on Message-Based Operating Systems, organised and held at QMC last summer. Their links with Xerox, IBM, BBN and many other institutions in the US have been strengthened and they have formed new ties with several European establishments [IBM and ETH at Zurich, IRIA - Rocquencourt, Phillips - Eindhoven]. In view of the number and calibre of visitors to QMC, most of their visits have been confined to institutions within the UK, with the exception of one non-SRC supported visit by Coulouris to IRIA.

REFERENCES

(1) G.F.Coulouris, I.Page, A.M.Walsby, 'The Potential for Integrated Office Information Systems'. QMC Computer Systems Lab,1977 (Published in association with Butler Cox and Partners Ltd).

(2) I.Page,A.M.Walsby, 'The QMC Text Terminal'. Electronic Displays 78, London 1978.

(3) A.West, A.Davison, 'CNET - A Cheap Network for Distributed Computing'. QMC Computer Systems Lab Report, June 1978.

(4) S.Abramsky, 'PASCAL-M: An extension of PASCAL which supports processes and messages', QMC CSL No 228, March 1979.

(5) S.Abramsky S, 'A PASCAL-M Interpreter', QMC CSL No 223.

(6) P.Adams, 'A Character-Mapped Display for the PDP 11/34', QMC CSL Report No 229, June 1979.

(7) G.F.Coulouris, 'QDESK - An Expedrimental Office System', QMC CSL No 230, 23.3.79.

(8) M.G.Lamming, 'Progress Towards the Design of a Personal Information Processing System', QMC CSL No 205, Sept 1978.

(9) M.G.Lamming, 'The Arguments for an Entirely Process-based Programming Environment', QMC CSL No 206.

(10) M.G.Lamming, 'A Proposal to Conduc t an Office Survey', QMC CSL No 226.

(11) M.G.Lamming, 'A Technique for Analysing the Office', QMC CSL No 227.

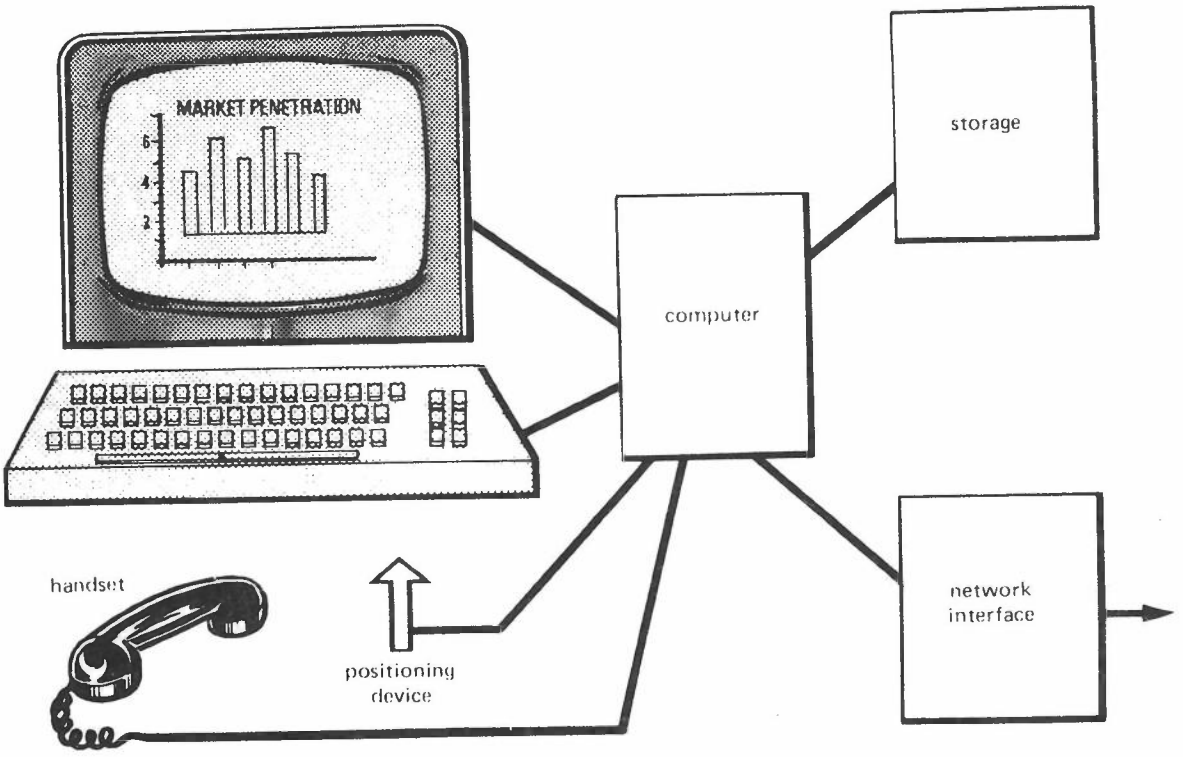
(12) P.Adams et al, 'Distributed Systems Requirements for Effective Man-Machine Interaction - Proposal for the Development of a Research Vehicle', QMC CSL No 212, December, 1978.

(13) B.Skittral, 'A Case in Point', QMC CSL No 219, March 1979.

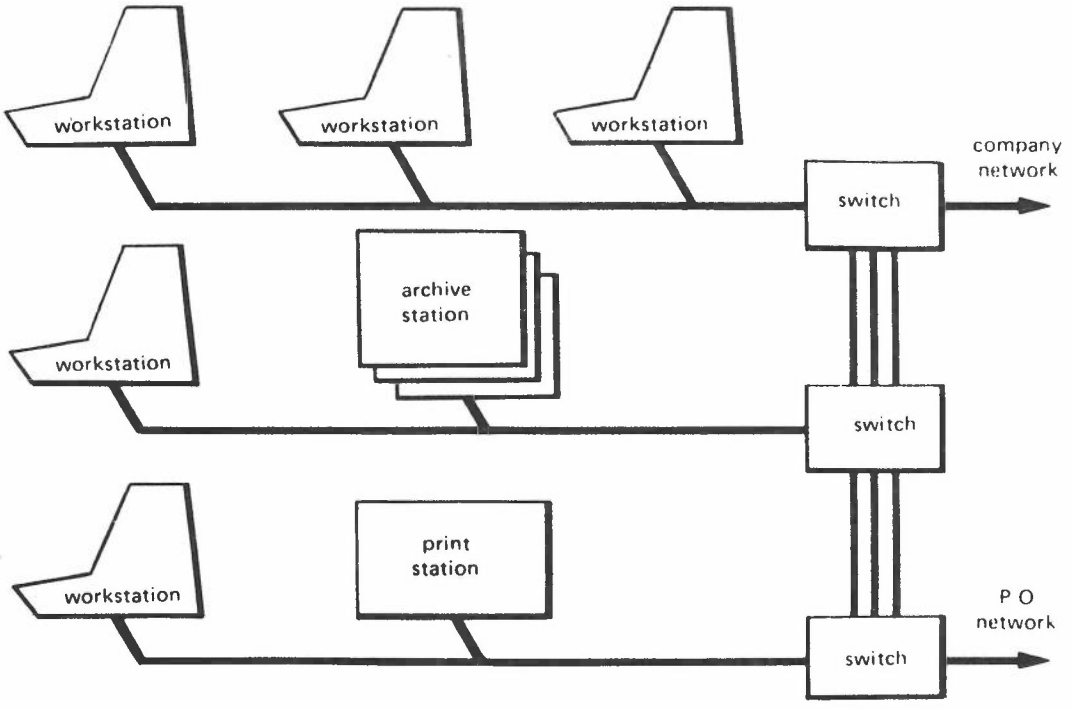
(14) B.Skittrall, 'The Computer Form and the Automated Office', QMC CSL No 225, June 1979.

(15) H.Thimbleby, 'Character-oriented text editing', QMC CSL Report No 195, September 1978.

(16) H.Thimbleby, 'INTERACTIVE TECHNOLOGY: The role of PASSIVITY', QMC CSL No 232.



A workstation



A typical office network

DR R D DOWSING and DR P W GRANT

UNIVERSITY OF EAST ANGLIA and
UNIVERSITY COLLEGE OF SWANSEA

THE SPECIFICATION AND IMPLEMENTATION OF PROGRAMS ON A MULTI-MICROPROCESSOR
Oct 79 - Sept 82

There are two main areas of research in the project: operating systems and specification and implementation of applications. Both of these topics rely on the use of the CYBA-M multimicroprocessor.

The present software system for CYBA-M assumes a static mapping of objects on to the hardware; that is, the designer has to specify, at design time, where all the code and data modules reside in CYBA-M - in global memory or in a processor's local memory. For a 'good' mapping to result, the designer has to predict the execution profile of his design. This is normally difficult, or impossible, since a loop count is frequently dependent on the data input at run time. For this reason, with the present static system, there is usually some degree of trial and error with different mappings and input conditions to produce a satisfactory result. Also, with the present system, the software cannot cope efficiently with varying load conditions since it must be designed to cope with the maximum load not the average.

For these reasons, it will be useful to provide an operating system on CYBA-M to study the systems dynamic behaviour. There are many interesting features of this hardware, relevant to an operating system, which means that useful operating system studies can be conducted as well as the production of a working user system. Some of the interesting features which need to be studied in the implementation of an operating system for CYBA-M include:

(1) A study of the dynamic and static aspects of the system and how they affect implementation. For example, moving information about in memory is slow. This means that the decision to map a process to a processor has to be made at load time with processes which need to be shared or which need to share a processor being mapped onto global memory. This observation also applies to the operating system itself so it would probably reside in two or three dedicated processors.

(2) A study of the high level policies to be used in resource utilisation. If the design parallelism/concurrency is greater than the number of available processors, how should the design be restricted? How is this affected by the fact that CYBA-M is a dedicated system? How can the operating system predict future usage of resources?

(3) A study of the useful features which a designer could specify to help the operating system make policy decisions, for example, the amount of concurrency/parallelism. This also includes a study of the extensions to the design language proposed for CYBA-M to include dynamic properties.

As can be seen from the foregoing, there are many facets of the implementation of an operating system which could provide useful information for the designers of future systems. Dr Dowsing and Dr Grant also hope to establish those features of an operating system which could usefully be incorporated in the hardware of a closely coupled system and the intention is that the team at UMIST would incorporate some of the suggestions into CYBA-M at a later date.

(Note: This collaborative project is formally two grants - one held by Dr Dowsing of East Anglia and one by Dr Grant of Swansea)

PROF D J EVANS and DR I A NEWMAN

LOUGHBOROUGH UNIVERSITY

AN INVESTIGATION OF THE RELATIONSHIP BETWEEN ALGORITHM STRUCTURE AND PARALLEL ARCHITECTURES
Jan 79 - Dec 81

The production costs of central processing units has fallen rapidly during the past few years due to the emergence of integrated circuit technology and one can now purchase minicomputers which are in every way better than second and many third generation machines. A number of such central processing units grouped together to form a general purpose multi- and parallel processor system could, in theory, exceed the price performance specifications of all but the largest computers currently available with the additional advantages of existing software, expandability as demand grows and less catastrophic degradation of performance when hardware malfunction occurs.

An earlier project investigated both the development of algorithms to fully utilise tightly coupled multiprocessor configurations, and the problems involved in the provision of a single user interface to a loosely linked, heterogeneous network. The first part has formed the basis of the ongoing work described. On the other side, a particular scheduling algorithm for distributing processes amongst loosely linked processors has been designed and implemented and a machine independent job control language has been implemented on the three machines (2 Interdata and a PDP11/40) forming the network. The whole system is operational experimentally and permits jobs to be presented at any terminal in the same format and allows some jobs to be moved between machines, subject to the scheduling criteria. Also associated with the earlier research is a simple modelling technique for estimating the maximum amount of power that can be obtained from a given multicomputer system.

The aim of the new research is to investigate the design and implementation of algorithms to solve a number of numerical and non-numerical problems using machines with different parallel architectures. The research is intended to produce algorithms which would fully utilise the special features of each architecture and provide a more general theoretical framework from which the efficiency of different algorithmic structures on various parallel architectures could be predicted.

The group has a working two-processor Interdata shared memory system. At the programming level, they have a version of Fortran which includes FORK and JOIN constructs and DO PARALLEL. The system is well instrumented to allow logging of time spent waiting for processors, system overheads etc. Resource allocation and scheduling for the system are provided by locally-produced software. All processors requiring a resource are connected in a ring with, at any one time, one process being the resource master. The current resource master must choose its successor before relinquishing the resource. Scheduling is distributed and the system can handle processor failure with graceful degradation.

The work on algorithmic design, so far, has been mainly concerned with matrix operations, Fast Fourier Transforms and eigenvalue problems. Already, some novel algorithms including a matrix quadrant interlocking Choleski factorisation algorithm and a task-splitting one for eigenvalue determination have been devised.

The group expect that the detail of an algorithm's structure will depend critically on the particular hardware arrangement, the cost of synchronisation, the speed of communications as well as the number of processor elements and their relative speeds. It is also likely that algorithms can be categorised in terms of the amount of available parallelism, the number of synchronisations required and the number of accesses to shared data.

By measuring the maximum parallelism possible compared with the execution speed on a single processor system, it is possible to define some bounds for the computation time on particular architectures.

The group hope to use several of the systems available to the Distributed Computing programme with the aim of classifying the architectures and the algorithms to be used. The problems to be used will include numerical ones such as partial differential equations, linear algebra, quadrature, sorting and searching.

PROGRESS TO SEPT 79

This year saw the successful completion of the orginal distributed computing systems contract at Loughborough; the commencement of a second contract investigating algorithm structure; and the preliminary work for establishing a new multiprocessor system (see following project description).

This report is divided into three sections:

(a) Algorithms work on the original contract and its continuation on the second contract;

(b) Work on the existing multimachine system;

(c) Planning for the new multimachine system.

(a) Theoretical and experimental studies on serial computer algorithms (numerical and non-numerical) have been carried out to determine what factors and critical regions exist if the algorithm is to be restructured for use on such a multi instruction multi data (MIMD) parallel computer system. To date it has been shown that those algorithms which have asynchronous properties have the best performance ratio on MIMD machines and our recent endeavours have been to develop new algorithms and re-design algorithms in linear systems, eigenvalue determination, quadrature, sorting and searching in order to eliminate synchronous properties in algorithms as much as possible and to obtain timings and bench mark tests on this two-processor system.

(b) The parallel processing system utilising the two Interdata 70 machines, which was developed earlier in the contract, has continued to work successfully throughout the year. It has been used by the lecturers, research assistants, and research students in the department and by two external research groups. The modified Interdata Disc Operating system, with roll-in roll-out multiprogramming, which was developed in the department to support the research work, has been given to the departments of Management Studies and Production Engineering in the University and is being used, full-time, on their departmental machines (Interdata 7/16s).

The file transfer system, which permits text files to be moved between the PDP 11 and the Interdata machines, was completed early in the year, and has been in regular use since then. The job transfer system, together with the enhancements to the scheduling algoritms to permit job transfer decisions to be taken, was also completed during the year.

(c) In order to extend the work on algorithms for multiprocessor systems, which had already been completed in the earlier contract, it was necessary to configure a multiprocessor system with more than two processors. The two processor-system had allowed measurements to be made which confirmed the theoretical performance predictions in that case, and the obvious extension appeared to be a four-processor system since this allows testing of both three and four-processor results, and also it is the minimum configuration which permits a closed interconnection which is not complete. The latter criteria was important if a complete test was to be made of the resource allocation and control algorithms which have already been developed.

REFERENCES

(1) R.H.Barlow, D.J.Evans, 'A Parallel Organisation of the Bisection Algorithm'. Comp. J. 22(1979), pp 267-269.

(2) D.J.Evans, R.H.Barlow, 'Analysis of the Performance of a Dual Minicomputer Parallel Computer System'. Report 43 of Computer Studies Dept 1977.

PROF D J EVANS, DR I A NEWMAN and DR M C WOODWARD

LOUGHBOROUGH UNIVERSITY

A RESEARCH VEHICLE FOR INVESTIGATING THE USE OF CLOSELY COUPLED DISTRIBUTED SYSTEMS
April 79 - March 82

The project will involve the establishment of a four processor research vehicle in which each processor has both substantial private memory and access to shared memory, using, as a basis, directly purchasable hardware and uniprocessor software. Each processor will be capable of independent operation, having its own multiprogramming operating system, with access to disc storage, and its own interactive terminals. Software will be developed to enable the four processors to cooperate in the management of shared resources (memory, disc and possibly some terminals). The vehicle will be used for studies in algorithm structure and processor utilisation by members of the DCS programme and specifically for an investigation of dynamic process allocation strategies where the processors are essentially autonomous.

Programme

(1) Investigate the structure of the operating system to determine the modifications that should be made.

(2) Implement the synchronisation primitives.

(3) Implement the parallel processing task scheduler.

(4) Implement a preprocessor for FORTRAN which allows parallel paths to be easily described.

(5) Modify the operating system and loader to force the variables used in the parallel sections of code, into the shared memory and permit the processors which are to cooperate access to these variables.

(6) Test the system with various existing FORTRAN implementations of parallel algorithms to ensure that it is functioning correctly and to give performance figures for one, two, three and four processors.

(7) Make the system available generally as a service to outside users.

(8) Compare the results obtained in (6) above with the theoretical predictions.

(9) Implement the modified synchronisation algorithm and management routines which permit processor "death" detection and recovery.

(10) Investigate the system performance with these modified routines, compare with the results of (8) to determine the overheads.

(11) Develop a modified scheduling algorithm which allows programs to be replicated and run on more than one processor dynamically or allows additional processors to cooperate in executing an algorithm for part of the time as required, possibly while executing other work (multiprogramming); the modified system being intended to increase overall work throughput by maximising the utilisation of resources over the complete system while not sacrificing the advantages of reliability.

(12) Develop the shared memory management algorithm for allocating and deallocating segments in the shared memory. Develop different algorithms for allocating access to the shared disc drives and other shared peripherals.

(13) Implement the algorithms in (11), (12) and test them, comparing the results with those obtained in (6) and (10) above.

The system will be based on Texas Instruments 990 systems.

PROF R L GRIMSDALE and DR F HALSALL

SUSSEX UNIVERSITY

THE DESIGN AND IMPLEMENTATION OF A MULTI-MICROPROCESSOR SYSTEM
Oct 79 - Sept 83

This project is for work on the design and implementation of a multi-microprocessor system in which proper attention is to be given to the integration of software and hardware. This work will result in a better understanding of the problems arising across the hardware-software engineering boundary and lead to simplified design methods for multi-microprocessor systems.

It is intended to develop an architecture (implemented jointly in hardware and firmware) which will support users' application program modules, written as processes in a PASCAL-like language and provide the necessary inter-process communication mechanism.

The design will be based on the assumption that any parallelism in the system will exist at the process level, where processes are reasonably-sized program modules. The basis for this decision is to minimise the hardware/software overheads of frequent process switching. However, there are no inherent built-in limitations on the size of processes and users will have the freedom to partition the application at their discretion.

There are a number of languages based on PASCAL which will provide a useful basis for this project, including Modula, Concurrent Pascal and Pascalplus. Further work is necessary, however, to investigate the problems of mapping programs on to multi-processors. It is intended that maximum use will be made of the work of other groups, and there is particular interest in compilers written in Pascal. It is intended to modify such an existing compiler to incorporate inter-process communication features and to make use of the intermediate output (in P-code form).

It is intended to examine the possibility of developing and testing individual user modules by emulating the system on the mainframe computer. Such an emulation would require skeleton definition of the other modules to allow compile-time checks on imported variables and exported expressions.

The detailed choice of the system architecture will form part of the investigation. The arrangement outlined below is intended to indicate the general directions in which the design is expected to develop. It is intended to exploit standard microprocessor families of components and VLSI modules whenever possible and final decision will depend on the availability of the modules at the appropriate time and their suitability.

The overall architectural scheme is based on some general considerations relating to the inter-dependencies between processors and the throughput of the individual processors. A balance is necessary between the requirement to provide adequate communication between processors and the cost of providing that communication in terms of the reduction in performance due to contention for the shared resources.

A full system will consist of a number of stations linked by a serial data communication network. Each station will consist of a minimum of two processors, one of which is the Communication and Control Processor and the other, the main Processor. The station will also include a File Processor and a Peripheral Processor. It would be possible to include further processors in the station, in particular, a second main processor. Each processor is equipped with a memory access controller which is used to define the resources for the running process and to direct the access of the processor to local ROM or to shared memory.

DR J R GURD and DR I WATSON

MANCHESTER UNIVERSITY

A RING-STRUCTURED DATA FLOW COMPUTER SYSTEM
July 78 - Sept 81

Since their initial conception, much of the research into Computer Systems has been directed towards higher computational speeds. However, there are many problems in scientific computing of which weather forecasting is a prime example, where the power of conventional machines is still inadequate. Most of these problems contain large amounts of parallelism and several types of architecture have been proposed which exploit this feature to increase computational speed.

Current parallel architectures can be divided into three major types, Vector Machines (STAR-100,CRAY-1), Array Machines (ILLIAC IV, DAP) and Multi-processors (C.mmp,Cm*). The first two architectures require a high degree of regularity in the problem parallelism to achieve efficient execution. Current user experience suggests that this is present in only a small class of problems. The third is more flexible, but there are indications that the overheads of processor communication could severely limit the speed potential of these machines.

Data Flow is a new approach to computation, where the parallelism is implicit in the problem specification. Instead of following a pre-specified sequence of instructions, any instruction is activated by the availability of its data. Very flexible parallel machine architectures can be designed using this principle requiring little or no regularity in the problem parallelism. The Manchester Data Flow project involves the design and construction of a prototype machine together with an investigation of the programming techniques and language requirements of Data Flow Computing.

The architecture overcomes the requirement of other proposed designs for duplication of code for procedures, iterative structures and parallel data paths. This results in more efficient execution and considerably smaller store sizes. The prototype will be constructed of ten 32 bit 'Schottky bit slice' microprocessors each with an average instruction execution time of 3 microseconds giving an overall maximum rate of 3 MIPS. The design allows parallelism in both storage and processing power, although the prototype will contain parallelism only in processing. The modular structure will however permit the introduction of storage parallelism and extension to a machine of significantly greater power at a later date. The architecture is at present being evaluated by simulation.

The data flow computer consists of a number of distinct units in a ring. A data flow instruction consists of at most two input operands, the operation to be performed and the destination of the result. The ring contains a processing unit which includes a set of processors. The result from an operation joins the 'result queue' waiting for access to the 'matching store' which is associatively accessed by the names of stored results. Its operation is central to the working of the computer. An item from the result queue will either generate an entry in the matching store waiting for another operand or it will complete an existing item which is then ready to be dispatched to the processing unit for execution.

A preliminary version of a compiler for a Single Assignment Language, LAPSE, has been written and is currently under development. There is also interest in other types of parallel languages which may be suitable for Data Flow computation.

PROGRESS TO SEPT 79

The project involves the production of five hardware modules, namely: (i) a processing unit; (ii) a switch; (iii) a result queue; (iv) a matching store; and (v) an instruction store; comprising a total of approximately 40 printed circuit boards of some 15 different kinds.

Most of the work to date has involved overall design decisions such as the size of board and type of racking to be used, and the asynchronous communication protocol between modules. A detailed description of the functional behaviour of the ring-structured pipeline will be ready in the near future.

In addition to this, detailed board designs for the result queue and the matching store have been undertaken. The three-board types for the result queue will be ready for construction next month. As expected, design activity is now reaching its peak, and board construction is about to commence. Four of the 15 different board types are now almost completely defined.

To date, no unexpected difficulties have arisen, although several strategical changes of direction have been made. Perhaps the most important of these has been the decision to use a more modern interconnection technology and to dispense with employing a technician for wiring of backplanes and modules. This decision was influenced by (a) the unforeseen availability, at relatively low cost, of high reliability ribbon cable interconnection systems; and (b) the difficulty of attracting a suitably qualified technician to a short-term contract post on the University technician scales.

Another important decision has been to change from the Varian 620/i to an LSI-11 front-end computer which is to be loaned to the project from the DCS equipment pool. A modem link to the ICF DEC-10 at UMIST has been established so that the system software being developed on simulators there can be transferred conveniently to the prototype system.

REFERENCES

(1) J.R.Gurd, I.Watson, J.Glauert, 'A Multilayered Data Flow Computer Architecture'. Internal Report, Jan 1978.

(2) J.Glauert, 'A Single Assignment Language for Data Flow Computing'. M.Sc Dissertation, Univ of Manchester, Jan 1978.

(3) I.Watson, J.R.Gurd, 'A Prototype Data Flow Computer with Token Labelling'. Proceedings of the 1979 AFIPS National Computer Conference Vol 48.

DR F K HANNA

KENT UNIVERSITY

DISTRIBUTED PROCESSING SYSTEMS FOR INTERACTIVE KNOWLEDGE BASES
Oct 78 - Sept 81

The aim is to consider the problem of distributing a very large and irregularly structured non-numerical processing task, an interactive knowledge base, over a potentially very large set of loosely interconnected processors in such a way as to localise data access and realise the maximum processor concurrency.

Over the past decade, steady progress has been made with the development of interactive knowledge bases. An interactive knowledge base is a large and generally complex computer program which attempts to deal with a body of 'real world' knowledge, to be able to manipulate it, ask and answer questions on it and, in a word, to 'understand' it. A typical example of such a system is one which deals with the identification of chemical compounds from mass-spectroscope data, or one which performs medical diagnosis of eye disease. In both of these cases the level of performance obtained currently rivals that of skilled practitioners in the field.

One of the principle features limiting the development of such systems is that they all tend to be computationally intensive. As the scope of the problem domain is widened, so the amount of computation power required to handle problems rises combinatorially. The microprocessor provides one possible way by which such potentially unlimited amounts of computation power may be obtained by coupling together large numbers of these elements. Technical considerations imply however that it is considerably easier to connect them loosely (so that data transfer between processors is slow compared with the speed of data manipulation within a processor). The problem which this research is addressing is whether it is possible to embed a large and complex task, such as an interactive knowledge base on a large, loosely-coupled network of processors, in such a way as to effectively utilise a 'reasonable' proportion of the processing power potentially available.

It is proposed that the overall computational task be subdivided into 'modules' each of which is implemented on a single processor. The modules will be chosen so as to allow the maximum number of them to be concurrently executed, and to be as self-contained as possible. The first of these objectives is not difficult to achieve as interactive knowledge base programs are inherently parallel. Achievement of the second one is a more open question, and it is this that is expected to provide the focus for most of the research effort.

A prototype system on which these ideas can be tested will use a loosely coupled system of five LSI-11`s, on each of which will be implemented a specialised language system which will allow parallelism and interprocessor communication.

PROGRESS TO SEPT 1979

This project has two separate strands. The first, initiated in late 1977, has been pursued by the investigator and a research student, and has led to the design and implementation (hardware and software) of a multiprocessor, multiprocess list-processing system, named PLEIADES. This has recently become fully operational, and its characteristics are described below.

The other strand of the project, that sponsored under SRC grant GR/A/74760, has only just begun (July 1979) with the appointment of Dr G Ritchie.

In many ways, the PLEIADES strand of the project may be seen as a pilot study of the research to be carried out under the DCS grant. The PLEIADES system is a set of loosely coupled processors, on each of which an identical multiprocess interpreter is implemented. Interprocess communication is essentially effected by passing arguments to procedures located in remote environments. Parallelism within a process is programmed either implicitly (by parallel evaluation of procedure arguments) or explicitly (by FORK statements).

The system is currently implemented on a set of four loosely coupled eight-bit microprocessors, using a departmental UNIX system to provide supporting facilities. It was recently brought to a fully operational state, and preliminary studies for AI applications are being commenced. The PLEIADES system is informally described in [3].

REFERENCES

(1) G.Coghill, 'A Multi-microprocessor System for Knowledge Manipulation'. Internal Report, Electronics Laboratory, 1975.

(2) F.K.Hanna, W.T.Harwood, 'A Multiprocessor System for Knowledge Representation and Manipulation'. AISB Conference, Hamburg, 1978.

(3) G.Coghill, F.K.Hanna, 'PLEIADES: A Multiprocessor Interactive Knowledge Base'. Microprocessors and Microsystems, 3(2), March 1979.

S E HERSOM

NOC, HATFIELD POLYTECHNIC

DEVELOPMENT OF OPTIMISATION ALGORITHMS FOR PARALLEL COMPUTATION
Oct 79 - Sept 83

With the advent of relatively cheap central processor units, computer systems with many CPUs operating in parallel can be confidently expected to be readily available in the near future. The NOC is to investigate how optimisation algorithms should be developed in order to make full use of a parallel computation facility. In the context of this project, NOC are considering parallel processing with multiple instruction, multiple data facilities and are excluding the single instruction devices such as DAP.

Optimisation algorithms written for sequential machines often include sections of code which could be executed in a parallel fashion, for example, the calculation of the partial derivatives of a function when its gradient is required. The improvement in performance, measured in computation time, is predictable and NOC's own work has confirmed that it is attainable in practice. Such "parallelisation" would certainly be implemented in this proposed project, but would not be considered as part of the research programme.

Algorithms for global minimisation, however, are often based on calculations made at random values of some optimisation vector and it may readily be seen that calculations of the objective function for different values of this vector could be carried out in parallel. Research is required into the effect of adopting such procedures and on how best to implement them.

The previous work carried out by the NOC included the implementation of a least-squares method for finding a local minimum (as opposed to a global minimum) on a parallel system [1] and it is proposed to extend this work under this research grant.

The following are the main objectives of this project:

(1) To design parallel versions of a number of existing algorithms for global optimisation and to implement them on the available parallel systems. These algorithms would include Price's method [2] and Torn's method [3,4]. Considerable work has already been carried out at the NOC on the sequential form of such algorithms. The Price algorithm was designed for use on small computers, and has features which make it attractive as the basis for a simple parallel asynchronous technique. The Torn algorithm is more complex, and is probably the best of the global optimisation techniques investigated at the NOC; it could be parallelised in a number of different ways.

(2) To carry out a programme of tests on the above programs, using test problems already available [5].

(3) To compare the results of such tests, wherever possible, with predictions based on simulation models of, for example, the queueing behaviour of the various algorithms/system combinations. A suitable simulation program is currently being developed at the NOC.

(4) In addition to the above work on global optimisation, the parallel programs for nonlinear least squares problems already implemented [1] on the dual-processor system in the Department of Computer Studies at Loughborough University would be extended. Such tests would not be confined to that system as the effect of a larger number of parallel processors should be investigated. The performance of the asynchronous algorithm mentioned in [1] on a larger system would be of particular interest.

REFERENCES

(1) McKeown, J.J. (1979) Experiments in implementing a nonlinear least-squares algorithm on a dual-processor computer. N.O.C. Technical Report No. 102.

(2) Price, W.L. (1978) A controlled random search procedure for Global Optimisation. In 'Towards Global Optimisation 2' L.C.W. Dixon & G.P. Szego (Eds), North Holland.

(3) Torn, A.A. (1978) A search-clustering approach to Global Optimisation. (Ibid)

(4) Gomulka, J. (1977) A user's experience with Torn's clustering algorithm. N.O.C. Technical Report No. 90.

(5) Dixon L.C.W. & Szego, G.P. (1978) The Global Optimisation Problem: an introduction. In 'Towards Global Optimisation 2' L.C.W. Dixon & G.P Szego (Eds), North Holland.

PROF C A R HOARE and J E STOY

OXFORD UNIVERSITY

SOFTWARE ENGINEERING
April 78 - March 82*

This research aims to reduce the cost of special purpose and general purpose computer software by scientific publication of programs of high quality and general usefulness. Within the next twenty years, one would hope that quite substantial software systems could be purchased for no more than the price of a book which contains the documentation and source text.

There are hopeful signs that such an ideal may be technically achievable. For relatively small scientific subroutines, publication is an accepted method of dissemination. For larger nuclear codes, the Computer Physics Communications provides a means of dissemination. The Numerical Algorithms Group markets a large library of numerically sound algorithms. Unfortunately, the source text of these programs is not usually presented in a well-engineered form; it is only tested and not inspected by referees; and it is not adaptable to meet variable needs of its users. The investigators are discussing these problems with NAG.

In the longer term, we hope to show the way to publish non-numerical software of general utility, for example editors, peripheral controllers, word processors, filing systems, and perhaps even compilers and operating systems. These are most urgently required for low-cost computer systems, and local networks. Again, there are promising signs: personal computing magazines publish small-scale programs in low level languages. Stoy and Strachey's OS6 is designed for a personal computer, and was published in 1972. But the most spectacular recent development has been the UCSD PASCAL (TM) system, developed by Ken Bowles at the Institute for Information Science. This is written in PASCAL and provides support for PASCAL programming on microcomputers of the most widely available varieties. Although commercialisation prohibits publication of the complete source texts, this system provides excellent support for execution of published applications programs written in PASCAL.

A very serious obstacle to the successful publication and use of non-numerical software is the absence of any consensus on its specification. In the case of trigonometric subroutines, it is generally agreed the objective is to compute a known single valued function of a single argument. In the case of a text editor, there is no such agreement on the functions to be provided or even on the number and types of their arguments and results. It seems that we must develop a tradition for publishing software specifications, preferably using some of the formal tools which are currently the subject of research and development.

PROGRESS TO SEPT 79

For various reasons work on the grant did not effectively start until the beginning of 1979. Nevertheless some valuable preliminary work has been covered. The most tangible achievement has been the importation of the UCSD PASCAL (TM) system and its first implementation on a British computer, the DICOLL LSI-11. Five of these machines have now been installed at the Programming Research Group, and they are connected in a star network to a

central printing and file server. In addition, a word processing system has been developed for a high quality low cost dot matrix printer. Two additional machines are due to be installed in other laboratories of the University. After further development, it is hoped that this system, used by experienced students, will serve as a valuable test bed for publishable software. Such developments absorb a lot of effort, but it is believed that this will be justified in the long run, particularly if other Universities are willing to participate in the exercise.

Progress on the problem of program specification is necessarily less tangible. The investigators have started by considering a specification of a particular example, the editor described in Kernighan and Plauger's book 'Software Tools', which was published without any kind of specification. A formal specification has been constructed, using the methods of Denotational Semantics, which were pioneered at Oxford, and implemented by Peter Mosses under the name SIS. Considerable effort has been expended to reimport SIS back to Oxford on the Group's own computer. This has required the installation of extra storage and a consequential restructuring of the local operating system. When this is available, it is hoped that it will be used by a number of students for the rigorous specification of software products.

An alternative, more abstract, approach to program specification is taken by Professor Abrial. His methods are firmly based on logic and set theory, couched in a notation (Z) similar to that of a programming language. However, it is deliberately intended to be not executable by computer. Professor Abrial is currently an SRC visiting fellow at the Group.

In summary, the investigators are approaching the topic of software engineering from the very high level of abstract specification and from the very low level of bit pushing protocols of microprocessors and cheap peripherals. The gap between these two approaches is distressingly wider than ever. It is the task of software engineering to span this gap in a reliable, efficient, and cost-effective fashion. To these criteria, the investigators have added a novel and even more difficult one - that all design decisions must be documented at publication standard, and that the eventual product should itself be published.

Outside the project itself, the investigators have contributed fully in the aims of the Distributed Computing Programme and in the transfer of technology.

(1) The investigators have organised/participated in the presentation of the following courses and seminars:

- EEC course at Copenhagen on Abstract Software Specification, January.
- Seminar at Oxford - Dijkstra on Distributed Computing, 8/9 January.
- Seminar Series at Oxford on Distributed Computing, January-March.
- STL workshop on Formal Design Methodology, April.
- Seminar at Oxford - Ken Bowles on PASCAL and Personal Computers, July.
- Summer Course at Santa Cruz on Programming Methodology, August.
- SRC CREST Course at Belfast on the Construction of Programs, September.
- ICL Professional Development Course at Oxford, September/October.
- IBM Professional Development Course at Oxford, September/Octobet.

(2) Good working relations have been established with the following industries: - IT&T, Dicoll, SOL, HPAC, Inmos, Siemens, Wavin, Infotech.

(3) Working software has been distributed to:
- The Open University, the Rutherford Laboratory, and Universities at Edinburgh, Belfast, Sussex, QMC, Manchester, etc.

PROF C A R HOARE

OXFORD UNIVERSITY

WORKSTATIONS FOR SOFTWARE ENGINEERING
July 79 - March 80

This award is purely for equipment to support Oxford's software engineering research. Funds have been granted to enable Prof Hoare to purchase six workstations, each consisting of a display, microcomputer and floppy discs.

DR J R W HUNTER, DR K D BAKER and DR A SLOMAN

UNIVERSITY OF SUSSEX

INTERACTIVE SOFTWARE TOOLS FOR DISTRIBUTED COMPUTING SYSTEMS WITH AN APPLICATION TO PICTURE INTERPRETATION
April 79 - March 82

This is an interdisciplinary project which brings together groups working in Artificial Intelligence and Computer Science. The project has two distinct but mutually dependent goals:

(a) Development of interactive tools for implementing and validating software for distributed computing systems.

(b) Implementation of a distributed picture-interpretation system, applying the results of (a).

One of the starting points is an AI project in which a group led by Dr Sloman has developed a POP2 program, known as POPEYE. POPEYE combines knowledge about different levels of structure in picture interpretation: the chosen pictures highlight some of the problems of combining several kinds of knowledge in vision. At present, it is implemented on a single processor, and processes corresponding to different layers of analysis have to be time-shared. The investigators now wish to distribute these processes on to functionally dedicated processors in order to explore the benefits of parallelism and develop techniques for designing more ambitious image interpretation systems in which many knowledge systems cooperate. Such systems are distinct from, but may eventually be combined with image processing systems in which essentially the same relatively simple operations are performed at many locations in parallel, eg on a distributed array processor.

The second starting point is work carried out in recent years by Dr Baker on multi-computer systems and the recognition that, as hardware costs continue to fall, software costs are becoming increasingly significant. There is therefore a need to produce aids for the economic development of software for multiple micro-processor systems, using high-level languages, and allowing interactive symbolic debugging.

The implementation of 'distributed POPEYE', and the development of software aids will proceed in parallel, each supporting the other. A series of increasingly complex subsets of POPEYE will be implemented on appropriate multi-micro-processor systems, linked to a 'host' computer containing the software development facilities. However good the programming language and compiler, powerful interactive run-time debugging aids will always be necessary to minimise implementation and software validation time, especially for distributed systems. An existing programming language will be modified for use in the distributed processing environment, and enhanced to provide interactive development and validation aids.

The two goals are therefore complementary. The usefulness of the software aids will be tested in a meaningful and demanding application; the implementation of distributed POPEYE would be very difficult without a suitable high-level language and interactive debugging aids.

PROF P T KIRSTEIN

UNIVERSITY COLLEGE, LONDON

COMMUNICATION PROTOCOLS IN THE CONTEXT OF X25 COMPUTER NETWORKS
Oct 78 - Sept 82*

The main aim of the project is to investigate the features and performance of the X25 protocol and of the High Level protocols needed above X25 in a computer network. The correctness, ease of implementation, adequacy of function, independence of level, level structure, dependence on subnet properties and protocol performance will all be studied. Where necessary, modifications will be suggested. It is particularly important to determine whether the digital telephone network, introduced by digital transmission and System X, will have a profound effect on this hierarchy.

A large part of this activity will consist of paper studies and specifications. The applicability of formal specification and verification techniques will be investigated. In addition, simulation (particularly of error recovery and throughput under specified error conditions) will be made. Experimentation will be restricted to simulated or idealised situations using just two computers.

A subsidiary aim of the project is to investigate the relevance of X25 and higher level protocols on local networks and the problems of connecting local networks to X25 networks. The technology applicable to Local Access Networks(LANs) can have different features from that used in Wide Area Networks(WANs). The protocols may need to be modified to take account of specific features such as broadcast capability, high transmission bandwidth, error performance, low transit times and specific interfacing techniques. The HDLC options used in X25 need some modification to communicate with terminals attached through a cluster controller to a LAN.

The work on connection of LANs to X25 WANs will require substantial paper studies on the changes needed in X25 for this application, on the status and error control information required across the gateways and on the whole gateway structure. These studies will be coordinated with other UK activities and with others on the ARPA Internetwork programmme.

Implementation and experimentation associated with the research will use initially the ARPA Packet Satellite Network (SATNET) and the UK Experimental Packet Service (EPSS) in forms which support fully the X25 Network Access Protocols. Later some of the experimental activity will be transferred to EURONET and PSS, the commercial European and Post Office networks. Efforts will be made to uncover the awkwardness of many protocol features, problems in the interface between levels, unanticipated side effects, performance difficulties and ambiguities in the specification.

This research is relevant in a much wider area than just the Distributed Computing programme.

PROGRESS TO SEPT 79

PDP-11 Hardware and Systems Software

The new SYSTIME 5000 (PDP-11/34) running UNIX has become the main development vehicle. In addition to the main processor, memory and asynchronous interfaces , a Plessey 60M byte disc, an EMI Magnetic Tape Deck, a Terminal 340 printer, and a DMA interface have been obtained and connected. It has been impractical to make network attachments directly in the PDP-11. Instead, it will be necessary to front-end that machine with an LSI-11. At least one more quarter of work will be needed before the PDP-11 is a proper network machine, running the X25 software in the front-end. At the same time, other network software will also be supported in the front-end, including the interface to the UCL copy of the Cambridge Ring and ARPANET.

Because it is necessary for various LSI-11s and PDP-11s to communicate with each other for testing purposes, UCL have brought up a multiplexor, called a Port Expander. This uses the same software that is required for connecting a host to ARPANET, but it does allow each of five LSI-11s or PDP-11s to communicate with each other.

A HDLC line card using program interrupt has been developed in a PCB version. Four production boards are being tested.

The Cambridge Ring

UCL have deliberately chosen to take over directly the technology developed at Cambridge University for their Ring System. They have produced PCBs for the repeater and station logic of the Cambridge Ring. These have proved very popular, and a number of universities have bought copies of these PCBs. UCL have also built a simple monitor station and three PDP-11/LSI-11 interfaces, again based on the Cambridge University designs. A prototype ring has started operation.

X25 Tests

UCL have had considerable difficulty in getting the Post Office to connect them to either IPSS or EURONET for X25 activities. Though some experience with the Post Office tester was obtained early in 1979, towards the end of this reporting period they became connected to IPSS. Tests have started of both X25 Level 2 and 3 with that network, using the Binary Synchronous mode at Level 2, currently supported by IPSS. They have also successfully tested with the Post Office tester in the true HDLC mode. In addition, they borrowed the EMU-SNAP and EMU-3 testers from the European Informatics Network Executive. In collaboration with EIN, the EMU-SNAP tester was made to work for the Post Office X25 tester (in the true HDLC mode). They have also tested X25 Levels 2 and 3 between the UCL LSI-11 and the EMU-SNAP machine. Thus, while much of the infrastructure of Level 2 and Level 3 X25 software has been developed, it will not be really complete until the line to EURONET comes in; that is currently scheduled to be at the end of 1979.

High Level Protocols

UCL have gone much further in the development of the Network Independent File Transfer Protocol (NIFTP) and have this working in some form on the UNIX System and a PDP-10 on ARPANET. Successful tests have been performed between NIFTP hosts on EPSS and those on the PDP-10s, using the EPSS-ARPANET protocol converters developed in the UCL PDP-9s. These tests have shown up a number of incompatibilities in the NIFTP implementations, and have been extremely useful in debugging both their and other people`s implementations. Some measurements on the PDP-10 version have been done. Tests of the UNIX version have been done internally, and using an ad hoc connection between UNIX and ARPANET. Proper network connection of UNIX has not yet been achieved.

UCL have started work on the problems of interconnecting the Ring Net to other wide-area networks, and of translating between datagram types of networks and X25 ones. Some of the principal problems are concerned with the difference between pure datagram gateways, with or without flow control, and ones where the connection is either at the Virtual Call or Transport Level. A number of notes have been written on some of these problems, and are listed in the references.

Theoretical Work

The theoretical work started rather more slowly. However, there has been considerable simulation work on the problems in buffer management. The different models of transport stations and problems of buffer management between gateways using different algorithms for flow control have been studied.

REFERENCES

(1) P.T.Kirstein et al: 'SATNET Application Activities', Proceedings: National Telecommunications Conference, 1979.

(2) P.L.Higginson: 'Packet Switches and Networks', Proceedings: AIM Digital Systems Conference on Teleprocessing, 1979.

(3) P.T.Kirstein, 'The Development of the UCL System - the Picture in October 1979'. INDRA 811.

(4) P.T.Kirstein, 'Some Implications of the UK International Tariff Structure for Packet Data Service'. INDRA 810.

(5) A.J.Mayne, 'Report on a Visit to California'. INDRA 796.

(6) C.J.Bennett, 'TENEX NIFTP User`s Guide'. INDRA 794.

(7) S.R.Wilbur, 'Interfacing the Ring to UNIX'. INDRA 793.

(8) S.R.Wilbur, 'Definition of Low Level Ring Protocols'. INDRA 792.

(9) A.J.Hinchley, 'Internetworking of Local and Wide Area Networks'. INDRA 791.

(10) C.J.Bennett, 'Clean and Simple'. INDRA 790.

(11) P.L.Higginson, 'Visit to NBS'. INDRA 787.

(12) P.L.Higginson, 'Report of Meetings at ISO/TC97/SC16 and its Working Group - June 4-15, 1979 in London'. INDRA 786.

(13) P.L.Higginson, 'Proposal for a Synchronized Attention in Transport Service'. INDRA 871.

(14) P.T.Kirstein, 'UCL Activities in Connection with SRC Grant GR/A/5695 for the First Half of 1979'. INDRA 777.

(15) S.R.Wilbur, 'A Design for a Terminal Multiplexor for the Cambridge Ring'. INDRA 776.

(16) C.J.Bennett and D.N.Frost, 'Comparison of the UNIX and TENEX Implementations of the NIFTP'. INDRA 772.

(17) C.J.Bennett et al, 'The UCL NIFTP Implementations - Progress Report'. INDRA 771.

(18) P.L.Lloyd, 'The Problem of Telnet - XXX Conversion'. INDRA 765.

(19) A.J.Hinchley, 'Overview of the TCP/X25 Protocol Converter'. INDRA 761.

(20) R.Cole, 'Translating C Programs for Use with the MOS Operating System'. INDRA 757.

(21) C.J.Bennett and P.L.Higginson, 'Comparison of the DIN FTP and the NIFTP'. INDRA 754.

(22) C.J.Bennett, 'Boston Trip Report'. INDRA 752.

(23) S.Yilmaz, 'EMU-SNAP Tester - User's Guide'. INDRA 750.

(24) S.R.Wilbur, 'Local Area Network Research'. INDRA 748.

(25) S.W.Edge, 'Report on the International Symposium in Flow Control in Computer Networks'. INDRA 741.

(26) P.L.Higginson, 'What is the Position on Character Terminal Connection to a Public Packet Switched Network'. INDRA 738.

(27) C.J.Bennett, 'Versailles Flow Control Symposium - Trip Report'. INDRA 737.

PROF P T KIRSTEIN

UNIVERSITY COLLEGE, LONDON UNIVERSITY

COMMUNICATION PROTOCOLS IN THE CONTEXT OF X25 COMPUTER NETWORKS - SUPPLEMENTARY PROPOSAL
July 79 - March 80

The award is for equipment to help Prof Kirstein`s existing project. The equipment supplied includes a Logic State Analyser, 3 VDUs and a high quality printing terminal.

DR P E LAUER

NEWCASTLE UNIVERSITY

DESIGN AND ANALYSIS OF HIGHLY PARALLEL DISTRIBUTED SYSTEMS
Jan 79 - Dec 81

Many of the problems facing the designer of systems may be regarded as being, in essence, purely synchronic, in the sense that what is required is that the design establishes certain desired relationships between occurrences of certain events. An example of such a synchronic problem is the reader/writer problem, in which one is not concerned with what is being written or read but only with ensuring, as a consequence of a design specification, that occurrences of the (uninterpreted) events 'read' and 'write' satisfy certain synchronic restrictions. Other such problems are those involving the phenomena of deadlock and starvation. In general, these problems involve 1) isolating from a specification or description those aspects which have to do with determining behaviour considered merely as an ordered set of occurrences of uninterpreted events, 2) understanding the relationships between specification and that which is being specified, and 3) using this understanding to verify and/or modify one's specification vis a vis some given design desideratum.

A formal study of such problems clearly requires some means of formally representing that part of a specification which has to do with the establishment of desired inter-relationships between occurrences of events, a way of formally representing behaviour as an ordered set of event occurrences and the possibility of formally relating the two for the purpose of analysis and verification.

The COSY (COncurrent SYstem) notation is a formalism which may be used to describe systems in their synchronic aspects. A system from the COSY point of view consists of a number of sequential processes and a number of distinct resources. Each process is described by a process expression, essentially a regular expression which prescribes the order in which the process may execute an operation named in it. With each resource is associated a set of path expressions [3], also regular expressions, which express constraints on the order of execution of operations associated with the resource. Thus, the activity of the system will consist of the processes activating their particular operations according to their process description and in such a way as to obey, taken together, the constraints on execution order enshrined in the resource descriptions.

Furthermore, a number of ways have been developed to associate with each COSY program a collection of objects which formally model possible behaviours permitted by the program. This means that the notation and its interpretation provide a framework within which may be carried out a formal study of the specification of concurrent systems and of that which they specify, in abstraction from those aspects which are not essential to a consideration of synchronic structure - aspects such as implementation details. This is perhaps analogous to the difference between verifying that an algorithm does evaluate some function (a problem in pure mathematics) and verifying that some specific program in some specific language implements the algorithm (a problem in informatics). This should not be interpreted as meaning that the Newcastle group is not ultimately concerned with implementation - indeed implementation is a problem to which

they intend to address themselves - but that it regards the two considerations as capable of being usefully studied independently.

It is, of course, of little use to know merely that an object, say a COSY program, may be regarded as an abstract specification because it defines objects which may be interpreted as describing behaviour. A system is supposed to do something in particular. We require to understand what particular specifications specify. Given the scope of concern of the group to be that of abstract specification, it follows that it has a commitment to the development of formal results for expressing this understanding. But not only that; the work of the group should not be merely the development of abstract mathematics for its own sake. If it is to be of use in the solving of real problems, the notation should be rich enough to express conventional structuring methods (abstraction, hierarchy, modularity) and should be appealing to use as a tool for specification and analysis [17].

Thus the group is interested in structure in a number of ways.

(a) Structuring a program to improve ease of writing it, to represent such things as abstraction, hierarcy, modularity and so on; the research effort in this area has led to the development of a macro notation [7,9,17] to support concise definitions of regular structures in programs.

(b) Structure of objects in the 'behaviour domain' - as pertinent to important synchronic properties such as deadlock and starvation - and how this is determined by the structure of the specification objects. The formal theory of path expressions and adequacy is a development in this direction.

There is a fruitful interaction between these two initiatives. For example, (b) may influence (a) in suggesting certain structures as useful in helping the designer to see what his specification does; (a) influences (b) in suggesting directions in which to direct the analysis of the relationship between particular structures of a specification and useful properties of their corresponding behaviours.

Both the design notation and the theory have been used to analyse and compare some existing operating systems for parallel and distributed systems with regard to their adequacy, non-starvation, degree of concurrency and degree of distribution.

Furthermore, the design methodology has been used to develop new and more concurrent and distributed systems to perform the same function as more conventional operating systems [1,7,9]. The intention is to continue these endeavours and especially to explore mechanisms for directly implementing systems specified in the design notation without loss of concurrency or distribution. By the end of the project, it is hoped to have:

(1) A fairly high-level design tool based on the COSY notation and of a high degree of flexibility and reliability.

(2) An analytic apparatus capable of dealing with objects produced by this design tool.

(3) Some notion of the ways in which a specification written using this methodology might be implemented. In particular it is expected to have generated a collection of non-trivial example systems capable of implementation.

PROGRESS TO SEPT 79

At the present time the project consists of Peter Lauer, Mike Shields and a Ph.D. student, J.Y.Cotronis. They have had a number of visitors from abroad who have spent between three weeks to two months with the project. These visitors have made a number of contributions to the development of the notation and results obtained during the project in the past and recent visits by P.R.Torrigiani (GMD Bonn) and R.E.Devillers (Universite Libre, Bruxelles) were particularly beneficial.

In the first nine months of the project since January 1979, the investigators have been occupied with the following tasks:

(1) They have completely revised the final report of the previous project during the period 1976-78 and it is shortly to appear as a 130 page technical report. It is an exhaustive record of the COSY notation and the formal results concerning it obtained during that period. It replaces all Newcastle departmental technical memos from ASM/0 - ASM/45.

The report has been extensively circulated and in particular it formed the supporting text for a series of four lectures by Peter Lauer at the EEC/CREST Advanced Course on Abstract Software Specification, Denmark, January 22 - February 2, 1979. This will form a basic text for new members of the project and visitors to the project.

(2) They have written a 50 page paper [17] which has appeared in Acta Informatica this year and which is an Introducion to the COSY Notation developed in the period 1976-78. It contains over 50 example system descriptions.

(3) They have written a 50 page paper which presents in a gentle and didactic manner both the notation and the formal results of the Final Report mentioned above. This paper will appear in Springer Lecture Notes in the autumn.

(4) They have begun a very careful reconsideration of the design notation and this has already led to several useful extensions of the notation. This work is documented in ASM/55 - ASM/63.

(5) They have developed a number of entirely novel, highly concurrent and distributed Banker's algorithms which do much to convince them that the notation developed in the project really does encourage novel system design more than any other notational suggestion they are aware of at present. These Banker's algorithms are documented in ASM/55 - ASM/57 and in a technical report by Devillers, Lauer and Torrigiani [10].

(6) They have substantially strengthened their formal analytical tools and obtained a number of new results enhancing their ability to demonstrate the presence or absence of important global system properties like partial and total system deadlock [14].

It has been mentioned that COSY programs may be interpreted as abstract system specifications in the sense that each such program determines a collection of objects which model behaviour. Behavioural properties, such as deadlock, could then be defined in terms of these behavioural objects, and rigorously studied. A large part of the work of the project during its initial period was concerned with the study of such a property, the property of adequacy, which corresponds to an absence of partial system deadlock. Several new results were obtained and applied to demonstrate the proof of adequacy of a number of programs written in the notation [8].

The development of these techniques involved treating programs in the notation as grammars generating string languages, whose elements, called firing sequences, could be thought of as sequentialised forms of possible histories of systems specified by these grammars. The problem then could be expressed as one of generating techniques of deducing properties of a string language from the textual structure of the program generating it.

This linguistic approach was used to devise and verify a grammar generating a subset of the notation which had the property that every program generated by this grammar would be adequate.

Technical details may be found in the final report of the previous period of the project.

The ideas underlying both the grammatical approach to programs and the syntactic generation of correct systems have been further developed and extended in the period since the end of the project's initial period. Take the grammar approach first.

Although a sequential representation of possible histories of a concurrent system is sufficient for certain purposes, such as the study of adequacy or freedom from deadlock, it fails to completely model behaviours; information about concurrency is lost. There is a difference between concurrency and arbitrary interleaving. Witness the existence of traffic lights which 'arbitrarily interleave' streams of traffic as opposed to allowing them to attempt to cross the area at the centre of a junction concurrently. Indeed, traffic lights may be thought of as implementations of a solution of a synchronisation problem involving fair access to a shared resource by streams of processes from two classes, where two processes from the same class might be allowed concurrent access to the resource. (Compare with the various subspecies of the reader-writer problem.) It is hard to see how a synchronisation problem such as the above - and indeed any synchronisation problem whose point is that concurrency causes problems - could be even stated formally using merely a sequential model of behaviour. To take another example, in [13], it was demonstrated that programs using only Agerwala's extended semaphore primitives [15] could be given a formal semantics in terms of COSY, it had to be shown that an ESP program and its corresponding COSY program gave rise to the same set of concurrent behaviours. A formal means of representing non-sequentiality was therefore necessary.

The model of COSY programs as grammars was therefore modified and they are now interpreted as defining languages whose objects are not strings but vectors whose coordinates are strings, the vector firing sequences. These may be shown to have the same modelling power as more conventional models of concurrent behaviour, such as occurrence graphs [16] but have the advantage that they may be manipulated in the same manner as strings. This has certain technical advantages also. Vector firing sequences were used in [13].

Vector firing sequences were also used in [14], which contains results greatly extending those behind the syntactic generation of adequate programs. The central idea here is that of a compound substitution. Essentially, a compound substitution is a textual modification which involves the replacing of an operation at various points of a program by regular expressions. The operation may be replaced by two different expressions at two different points in the program. They are interested in those substitutions which preserve essential properties, such as adequacy. There are two advantages to be gained from an understanding of such matters: firstly, such results may be used to reduce large specifications to smaller ones having the same adequacy properties - leading to a simplification of verification by, say, simulation - and secondly, an understanding of adequacy preserving transformations of this kind allow one to structure one's specification in a manner that guarantees that it will possess certain properties. This illustrates the fruitfulness of the interaction between notational and theoretical development. The analysis of a specification has the effect of implicitly decomposing it into parts with specific properties. One might well conceive of a catalogue of such parts, together with rules determining how they might be composed together into systems - the reverse of the aforementioned decomposition. Using this catalogue, one may build only adequate systems. It is heartening that notational forms ideal for expressing such composition have already been defined - the class-like objects of [3,9,17].

REFERENCES

(1) R.E.Devillers, P.E.Lauer, 'A General Mechanism for avoiding Starvation with Distributed Control'. Information Processing Letters, Vol 7, No 3, April 1978.

(2) P.E.Lauer, E.Best, M.W.Shields, 'On the Problem of Achieving Adequacy of Concurrent Programs'. Proceedings of IFIP Working Conference on Formal Description of Programming Concepts, Aug 1977, North Holland.

(3) P.E.Lauer, R.H.Campbell, 'Formal Semantics for a Class of High-Level Primitives for Coordinating Concurrent Processes'. Acta Informatica, 5, 1975, pp297-332.

(4) P.E.Lauer, M.W.Shields, 'Abstract Specification of Resource Accessing Disciplines: Adequacy, Starvation, Priority and Interrupts'. Tech Report 117, University of Newcastle upon Tyne, 1977. SIGPLAN Notices.

(5) P.E.Lauer, M.W.Shields, 'Design and Analysis of Highly Parallel and Distributed Systems'. ASM/46, University of Newcastle upon Tyne, April 1978.

(6) P.E.Lauer, M.W.Shields, E.Best, 'On the Design and Certification of Asynchronous Systems of Processes: Final Report: Part I: COSY - a System Specification Language based on Paths and Processes'. ASM/49, June 1978. 'Part II: Formal Theory of the Basic COSY Notation'. ASM/45, March 1978, University of Newcastle upon Tyne.

(7) P.E.Lauer, P.E.Torrigiani, 'Towards a System Specification Language based on Paths and Processes'. Tech Report 120, University of Newcastle upon Tyne, 1978.

(8) M.W.Shields, P.E.Lauer, 'On the Abstract Specification and Formal Analysis of Synchronisation Properties of Concurrent Systems'. University of Newcastle upon Tyne, ASM/44, Feb 1978, Proceedings of the International Conference on Mathematical Studies of Information Processing, Aug 1978, RIMS Kyoto, Japan, Springer Verlag.

(9) P.E.Torrigiani, P.E.Lauer, 'An Object Oriented Notation for Path Expressions'. AICA (Associazione Italiana per il Calcolo Automatico), 1977, Annual Congress 3rd Vol, Software Methodologies, pp349-371, Oct 1977.

(10) R.E.Devillers, P.E.Lauer, P.R.Torrigiani, 'A Distributed Banker'. Technical Report Series, Universite Libre de Bruxelles, Belgium, 1979.

(11) P.E.Lauer, M.W.Shields, E.Best, 'Formal Theory of the Basic COSY Notation'. To appear as Technical Report 143, Computing Laboratory, University of Newcastle upon Tyne, 1979.

(12) P.E.Lauer, M.W.Shields, E.Best, 'The Design and Analysis of Highly Parallel and Distributed Systems'. Advanced Course on Abstract Software Specifications, Lyngby, Denmark. Also as Tech. Report 142, Computing Laboratory, University of Newcastle upon Tyne, 1979. To appear in Springer Lecture Notes.

(13) M.W.Shields, P.E.Lauer, 'A Formal Semantics for Concurrent Systems'. Proc 6th International Colloquium, on Automata, Languages and Programming, Graz, July 16-20, 1979, Springer Lecture Notes, Springer Verlag. Also Tech Report 132, Computing Laboratory, University of Newcastle upon Tyne, 1979.

(14) M.W.Shields, 'Adequate Path-Expressions'. Proc. International Symposium, Semantics of Cuncurrent Computation, July 2-4, 1979, Evian-les Bains, France, Springer Lecture Notes. Also Tech. Report 141, Computing Laboratory, University of Newcastle upon Tyne, 1979.

(15) T.Agerwala, 'Some Extended Semaphore Primitives'. Acta Informatica Vol8, pp 201-220, 1977.

(16) A.W.Holt, 'Information System Theory Project'. Tech Report RADC-TR-68-305, Applied Data Research Inc., 1968.

(17) P.E.Lauer, P.R.Torrigiani, M.W.Shields, 'COSY: a system specification language based on paths and processes'. Acta Informatica, Vol 12, pp 109-158, 1979.

A J R G MILNER

EDINBURGH UNIVERSITY

APPLICATIONS OF FLOW ALGEBRAS TO PROBLEMS IN CONCURRENT COMPUTATION
Jan 78 - Sept 79

This project takes as its starting point the work by Milne and Milner[2] on a process model of concurrent computation. This mathematical framework has been produced to enable a person to reason about computing systems in terms of the mathematical objects which model them. The question as to what is the most suitable framework to represent concurrent computation remains open but an apparently successful step has been made by the Flow Algebra of Processes. In [3] the process algebra is used in the construction of a model of a card reader and in the representation of communication scheduling. In both these examples correctness proofs have been found. As well as adopting an algebraic approach, the model uses the semantic concepts of Scott and Strachey[4] whereby the meaning of a system (or net) of communicating agents is an abstract object (a process). The meaning of such a system is constructed from the composition (via suitable operations) of the meanings of the system components.

The purpose of this project is to investigate the suitability of processes in particular, and other Flow Algebras in general, as models of concurrent computation. Application of the theory to real problems of concurrency should reveal those features of concurrency not suitably represented in the theory, and should also suggest changes to produce a 'better' theory.

The first investigation has been in the area of distributed database systems. A real problem of such database systems arises when one wishes to keep copies of databases at various nodes (to minimise access delays). The problem is to produce a control algorithm to reside at each node of a computer network to control access to its own and to other databases. The protocols should control the communication and transfer of information between nodes in such a manner that the system does not deadlock, that the data remain consistent (at all times after the system has settled following an update all copies of a database are identical), that data accessing delays are minimised and that the system remains partially operable when one or more nodes has failed. Ellis [1] has produced a number of algorithms which aim to deal with these problems. The solution which he believes to be most satisfactory has been modelled using processes. By altering the definition of a process algebra slightly, a concise process formulation of Ellis's algorithm was found. The alteration expresses, more naturally than using the previous process algebra, the notion of a simultaneous broadcast communication between one process and a number of others. The database network was modelled by combining a number of processes (each of which models a database manager) using a 'synchronising' conjunction operator. A proof that the network is free from deadlock has been carried out but needs checking. Such a result gives some evidence that the model is 'on the right track'.

Future work will be concerned not only with providing the consistency and partial operability of the above chosen example but also with examining a completely different level of concurrent computation- that of the problems of hardware representation and design. Many difficult problems exist concerning modelling of hardware and it needs to be seen whether the

process algebra (or some similar Flow Algebra) can go some way towards satisfying them. An obvious problem is that of modelling an inherently continuous system by a discrete model. Also the present model assumes asynchronous communication, and it must be seen whether strongly synchronous communication can be suitably represented.

What has been achieved so far is not only a notation in which the behaviour of systems of communicating computing agents can be represented in a concise, intuitive manner as it evolves through time, but also a mathematical framework for formal reasoning about such systems. Certain proof techniques have already been developed but others need to be found. It has also become apparent that when modelling different concurrent systems and carrying out proofs on them, different algebras may need to be used. Guidelines governing the choice of algebra must be found. It must also be determined which properties of concurrency are preserved when mapping from one algebra into another.

PROGRESS TO SEPT 1979

This project is the work of one researcher, George Milne, under the direction of Robin Milner.

Many challenging examples of concurrency present problems which are mainly concerned with the control features of systems, rather than with the flow of data within. 'Syn' algebras and Synchronised Behavours were introduced to capture this pure synchronisation. This new framework is a natural variation of Flow Algebras, but does not model the passage of values between components of a system; on the other hand, it allows for multi-way synchronisation and does not restrict one to communications between two components as in Flow Algebra.

A major example tackled was that of a distributed database system. This system comprises a number of nodes containing databases, some of which hold identical copies. The investigators modelled the control portion of each node which controls access to its own and other databases. They in fact abstract away from the databases themselves, and use the models of the components to produce a model of the whole communicating control system, using their '&' combinator. Broadcasting of signals, which is inherent in this system, is easily represented by an n-way synchronisation between the sender and the n-1 receivers, all n cooperating in this event. Once modelled, they use algebraic laws to prove a desired property of the system, namely, that it is free from deadlock.

At the other end of the spectrum they have also investigated some of the problems of modelling gate-level hardware using the Syn framework. There is a need for design and specification languages for representing the behavioural features of hardware components and not just their static features. Their investigation leads them to think that the 'Syn' framework may be useful for this purpose, since it allows them to represent as primitive the rising and falling change on lines, together with the simultaneous effect of such a change upon possibly more than two components. Problems with this involve representing inherently continuous behaviour in a discrete model, and dealing with complicated behaviours. The latter means that the object representing this behaviour in their model is also very complex. Problems arise when combining such complex objects using the '&' combinator, and in carrying out proofs such as the

equivalence of a composite system with its design specification. Some mechanisation of this part is probably necessary.

Currently they are investigating some standard concurrency problems to assess the flexibility of their model, including the Cigarette Smokers and the Dining Philosophers problems. Using their model, various solutions to both problems can be produced. These are proved to satisfy the problem specification and thus to be equivalent with each other. This partly supports their claim that the algebraic method helps them to gain insight into different solutions to the same problem, and to compare them.

REFERENCES

(1) C.A.Ellis, 'Consistency and Correctness of Duplicate Database Systems'. Sixth ACM Symposium on Operating Systems Principles, SIGOPS 11,5, Nov 1977.

(2) G.Milne, R.Milner, 'Concurrent Processes and their Syntax'. Report CSR-2-77, Computer Science Dept, Edinburgh, 1977 (to appear in Journal ACM).

(3) G.Milne, 'A Mathematical Model of Concurrent Computation'. Ph.D Thesis, Edinburgh, 1978.

(4) D.Scott, C.Strachey, 'Towards a Mathematical Semantics for Computer Languages'. Proceedings Symposium on Computers and Automata, Microwave Research Institute Symposia Series, Vol 21, Polytechic Institute of Brooklyn, 1971.

(5) G.Milne, 'Scheduling within a process model of computation'. Proc. 1st European Conference on Parallel and Distributed Processing, Toulouse, France, February 1979.

(6) G.Milne, 'Modelling distributed database protocols by synchronization processes', Report CSR-34-78, Computer Science Department, University of Edinburgh.

DR I MITRANI

NEWCASTLE UNIVERSITY

MODELLING AND PERFORMANCE EVALUATION FOR DISTRIBUTED COMPUTING SYSTEMS
Oct 78 - Sept 80

The aim is to investigate the effect of some hardware and software characteristics on the performance of distributed computing systems. The main problem areas addressed are the deterioration of performance due to breakdowns of the equipment and the design of scheduling algorithms which meet certain performance criteria. The analysis and simulation of probabilistic models will be used to study these problems.

PROGRESS TO SEPT 79

Formally, this project started on April 1st 1979, when P.King was appointed Research Associate. However, work on the project had begun before that date. I.Mitrani, in collaboration with K.C.Sevick, of the University of Toronto, has been investigating the trade-off between centralised and distributed computing. A star network consisting of one central and several local sites was considered. The problems arising in this connection concern (a) the distribution of processing power among the sites and (b) the distribution of workload. Concentrating most of the processing power at the central site, and transferring most of the workload there, has the advantage of efficiency: one fast processor provides a better service than a number of slow ones with the same total capacity. On the other hand, such a concentration leads to higher communication costs due to job transfers. To evaluate the trade-off, a cost function was defined, taking into account the costs due to congestion (efficiency) as well as those due to load transfers (communication). That cost function was optimised, first with respect to load routing only (how much should be done locally and how much should be sent to the central site) and then with respect to both load routing and processing capacity allocation (what fraction of a fixed budget should be invested in local processing power and what in central). Moreover, either of the optimisations can be performed individually (seeking maximum benefit for the jobs originating at a given site), or collectively (over all sites). It turns out that often, but not always, the best policy is at one of the allocation extremes: either all locally or all centrally.

The above research is fully described in [1].

Work on an examination of a scheduling problem in loosely coupled multiprocessor systems has recently been started. If a system of N processors offers service to two different classes of jobs, one way of providing a discriminatory control, is to designate a fraction A of the processors 'class 1' and the rest - 'class 2'. Class i jobs have preemptive priority on class i processors. As the control parameter A varies between 0 and 1, one obtains a family of scheduling strategies ranging from 'top priority to class 2' to 'top priority to class 1'. This system can be modelled by a two-dimensional Markov process. The latter can be solved numerically, for given parameter values, without too much difficulty; P.King has implemented a numerical solution package. The problem with a numerical solution is that, since one is dealing with infinite systems of equations, large amounts of CPU time are consumed to

achieve the desired accuracy. It now seems possible to obtain an analytical solution. Dr G.Fayolle, of IRIA (France), who has worked extensively in the area of Markov processes, visited the project in February. Significant progress towards an analytical solution was made during his visit; Dr Fayolle will continue to collaborate on this probem.

Professor E.Gelenbe, of the University of Paris-Sud also visited the project. His work is concerned with consistency in distributed information systems, and a distributed system with fibre optics communications channels. At a future date, Dr Mitrani may help in the modelling and performance evaluation of that system.

REFERENCES

(1) I.Mitrani, K.C.Sevcik, 'Evaluating the Trade-off between Centralised and Distributed Computing'. Proceedings of the First Int. Conf. on Distributed Computing, Huntsville, October 1979.

(2) I.Mitrani, B.Avi-Itzhak, 'A Many-Server Queue with Service Interruption'. Opns. Res. Journal, 16,3, pp 628-638, 1968.

(2) I.Mitrani, J.H.Hine, S.Tsur, 'Satisfying Performance Requirements in Computer Systems'. Internal Report, 1977.

DR P E OSMON

WESTFIELD COLLEGE, LONDON

IMPLEMENTATION OF A HIGH LEVEL DATA FLOW PROGRAMMING LANGUAGE
July 78 - Sept 81

The Westfield group is interested in the fundamental principles underlying Data Flow. They have developed a set of definitions and primitive concepts and have designed a high level programming language, CAJOLE, that is consistent with the model. An intuitive idea of what Data Flow means is given by the definition that a 'data flow program' is one in which, at run time, the order of execution of individual statements is only determined by the data interdependencies between statements. This approach has significant implications for both software and hardware and the following gives some indication of the work that has been done at Westfield in both of these areas.

Many of the commercial prophets of structured programming, in particular Ed Yourdon and Michael Jackson, use some of the Data Flow ideas in the top-down development of programs. Research workers at IBM have recently produced a high level language, EDL, which uses a Data Flow notation. At Westfield the interest is more in scientific programming languages because the large scientific applications lend themselves most readily to a distributed processing solution. The order of execution of a program is determined by the flow of partial results, this means that data dependencies must be mechanically discernible which implies that program statements must be free from side effects. Another implication of Data Flow is that the textual ordering of statements need no longer be related to the order of execution. The absence of any sequencing and the absence of side effects is ensured by the application of the single assignment rule that 'each name used in a program may only be assigned a value once at run time'. This rule prevents the programmer from using loop constructs, but iterative constructs may be implemented by using recursive calls.

Two members of the group have been working on hardware design, although they have not as yet produced a Data Flow architecture. The implications of the primitive concepts are that the architecture should be capable of efficiently exploiting parallelism at the instruction level without any complicated centralised control mechanism. The single assignment rule implies a dynamic recursive machine architecture similar to the designs of Glushkov and Davis (although this is a static model). It is hoped that the group will be able to use the Manchester machine (see Gurd and Watson, above) to evaluate CAJOLE.

Over the next year, it is hoped to implement a compiler for CAJOLE and possibly, as the Manchester machine will not be ready, to implement a simulator for a Data Flow architecture. The group has recently been joined by someone who has had significant experience in Data Processing and he is currently looking at possible commercial applications of Data Flow. On the theoretical side, there is still a large amount of work to be done in developing an acceptable and consistent set of concepts.

PROGRESS TO SEPT 79

The early part of the year was spent introducing new members of the group to the concepts of Data Flow, requiring the established members to re-appraise their views of the subject. This was very helpful in consolidating a consistent view of Data Flow, and has enabled the group to proceed with the more concrete aspects of its work.

A 'quick and dirty' interpreter for the group's language, CAJOLE, is now nearing completion, and a cleaner compiler is in progress. This will provide a front end to both the simulator which is being produced, and also the Manchester Data Flow machine.

In addition to the work directly involved with CAJOLE, work is also proceeding on an architecture, measurement of parallelism in programs, applications, and a comparison of data flow languages.

The group's SRC funded research assistant (H Glaser) who joined the group at the beginning of October spent some time discussing the language considerations, and is now implementing the compiler mentioned above.

Throughout the year the group has maintained contact with many other groups working in related fields. In particular Professor Shriver and Steve Landry from Southwestern Louisiana and Ronan Sleep from East Anglia have visited the project. The group has visited the Rutherford Laboratory to meet Professor Arvind from M.I.T., and presented four papers at the data flow workshop in Toulouse. Individuals have presented papers at Workshops at M.I.T. and Warwick.

REFERENCES

(1) C.L.Hankin, P.E.Osmon, J.A.Sharp, 'A Data Flow Model of Computation'. Submitted for publication.

(2) C.L.Hankin, J.A.Sharp, 'A Functional Data Flow Programming Language'. Submitted for publication.

(3) R.E.Cooley, 'Parallel Data Processing'. Data Flow Workshop, Toulouse (February 1979).

(4) C.L.Hankin, 'Towards a Formal Data Flow Model'. Data Flow Workshop, Toulouse (February 1979).

(5) P.E.Osmon, 'Towards a Data Flow Interpreter'. Data Flow Workshop, Toulouse (February 1979).

(6) J.A.Sharp, 'The Analysis of Data Flow Programs'. Data Flow Workshop, Toulouse (February 1979).

DR P E OSMON

WESTFIELD COLLEGE, LONDON

PROVISION OF FACILITIES FOR IMPLEMENTATION OF A DATAFLOW PROGRAMMING LANGUAGE
July 79 - March 80

This award is for equipment to help Dr Osmon`s existing project. A magnetic tape unit was supplied to facilitate software interchange with other groups.

I PAGE

QUEEN MARY COLLEGE, LONDON

A HIGH QUALITY DISPLAY SYSTEM FOR EFFECTIVE MAN-MACHINE INTERACTION
July 78 - June 80

The aim of the research programme is to investigate and develop a radically new display technology to significantly improve the effectiveness of man-machine interaction, particularly in the context of distributed personal computers. The technology is based on the use of an array of cooperating processing elements to generate a raster description of the displayed picture from a problem-related data structure.

Existing research on raster displays has indicated that it is not possible to generate the raster map for a rapidly changing picture sufficiently quickly, by purely serial processing, to support a fully effective man-machine interface. This is so even for the largely textual information displays occurring in simple office applications. The problem becomes increasingly acute as the display requirements are extended to larger screens, colour, multiple fonts and full graphics.

The display to be produced will consist of a set of 1-bit processing elements interconnected as a rectangular array and controlled by a single instruction stream. Associated with each 1-bit processing element is a significant amount of local semiconductor storage. The array will be used to generate the picture map which will be distributed amongst the stores of the processing elements.

As well as the hardware design, the project will investigate the development of a display file structure and algorithms for the generation of the picture map. The research is being performed in conjuction with the work of Prof Coulouris.

PROGRESS TO SEPT 1979

Prior to this research programme the QMC Text Terminal [1] had been designed and constructed in the Computer Systems Laboratory. The Text Terminal has facilities for presenting coloured text on simulated pieces of paper known as windows. This device has been used to try out a number of techniques for improving the effectiveness of the man-machine interface, particular attention being paid to the capability for 'dragging' windows in real time. These techniques have direct relevance to the design of the 'Disarray', as the new device has been christened.

A novel 'mouse' input device has also been developed [2] which enables the user to interact naturally with the very novel features of this display. This invention has been patented in a number of countries with the help of NRDC, and commercial sponsors for it are currently being sought. The mouse will be a part of the 'Disarray' system.

A bit-per-point display system has been designed, built and interfaced to the PDP 11/34 system. This device is used as a vehicle on which to run software simulations of the hardware to be built. A software environment to support these simulations has been written. The experience gained will be a considerable aid in designing the hardware and the simulator will

enable software development to proceed in parallel with hardware development.

A considerable amount of initial design work has been done, with the emphasis on processor architecture. A number of chips have been investigated; it may prove possible to use the 2900 bit-slice family chips to ease the manufacture of processors, but should these not give the necessary speed, resort will have to be made to simple Shottky SSI and MSI integrated circuits.

STAFF

One full-time RA is employed on this project, A M Walsby.

EQUIPMENT

A bit-per-point display system with a DMA system to the PDP 11/34 has been built. Valuable experience was gained in interfacing the device to the PDP 11/34 DMA board which may be of interest to other groups.

VISITS

Incoming - Many UK and USA research groups have visited the project. Links with Xerox, IBM, BBN, ETH Zurich, IRIA at Rocquencourt and Phillips at Eindhoren have been forged and strengthened.

Outgoing - Sites in the UK.

REFERENCES

(1) I.Page, 'A High Quality Display System for Effective Man-Machine Interaction'. Internal Report, April 1978.

(2) I.Page, A.M.Walsby, 'The QMC Text Terminal', In preparation, accepted for 'Electronic Displays 78'.

(3) I.Page, 'A specification of a mouse graphical input device working on a novel principle', UK Provisional Patent 8791/78 - Visual Display Input Device, March 1978.

DR Y PAKER

POLYTECHNIC OF CENTRAL LONDON

COMPUTER AIDED MULTI-MICROPROCESSOR SYSTEMS MODELLING, SIMULATION AND EVALUATION
April 79 - March 81

Microprocessors and microcomputers are being coupled together in increasingly large numbers, reaching several hundred or more, in a tightly or loosely coupled manner as distributed computing structures which include complex interconnection mechanisms and interfaces to link these to an application. Superimposed on this hardware structure, software is written to provide the communication protocols, synchronisation between sequential processes, application programmes and so on. Considering the complexity of such systems, however, currently there are no adequate methods to understand and evaluate many of the basic architectural and software options that exist when a distributed computer system is designed to satisfy the requirements of a particular application environment.

The objective of the proposed research work is to develop interactive computer aided design methods for modelling and simulation of multi-microprocessor (computer) based distributed computer systems, especially for performance evaluation. The modelling proposed will include individually discernable hardware components such as processor memory and also intercommunication structures such as bus, serial line and so on. The designer will be given the ability, by means of an interactive terminal, to model his system using those standard or user defined building blocks and their interconnections. Once a basic module such as a number of processors sharing a common memory is defined, this can then be used to construct more complex structures. The interconnection mechanisms will include most of the known direct schemes such as bus, shared memory, ring, as well as the store-and-forward type indirect schemes such as HDLC and X25. A number of routing protocols will be provided with the possibility of adding user specified protocols. The application software will be simulated at process level and facilities will exist to introduce control mechanisms such as semaphores and constructs such as buffers, monitors and so on.

Once a distributed computer model is built with its hardware components, interconnections and software modules, then the system can be simulated under user defined conditions. The principal performance figures to be simulated will be delay times, buffer sizes and their distributions, throughput and other relevant user defined quantities. Based on simulation run results, a user will be given facilities to interactively change the system configuration, specifications, message lengths and inter-arrival times, protocol used and so on. The model will also include randomly introduced failure simulation to determine the influence of these on system performance. It is expected that such an interactive design aid will help in evaluating the architectural options that exist in building a distributed computer system that best matches a given application requirement in a cost-effective manner.

DR G D PLOTKIN and A J R G MILNER

EDINBURGH UNIVERSITY

SEMANTICS OF NON-DETERMINISTIC AND CONCURRENT COMPUTATION
Oct 78 - Sept 81

The aim of the research is to study the foundations of nondeterministic and concurrent computation and to develop the appropriate mathematical tools. The eventual aim is to a large extent practical - to understand, design and reason about concurrent computing systems - but in this research the intention is to place great emphasis on achieving a general, elegant and tractable mathematical theory. It is not assumed that the application of the theory is restricted to man-made systems.

Since a considerable amount of work has already been done in this direction, the project needs justification. First, it is believed that there is no consensus yet concerning the choice of basic concepts for a model of concurrent computation. Three examples will support this statement:

(1) The Net Theory of Petri and his followers takes as primitive the 'concurrency relation' - a reflexive, symmetric but non-transitive relation among events and conditions which expresses, approximately, their causal independence.

(2) The actor model of Hewitt takes 'message passing' as primitive - the sending and receiving of a message are not assumed to be simultaneous.

(3) The Flow Algebras of Milner and the Communicating Sequential Processes of Hoare take 'synchronised communication' as primitive. In contrast with the actor model, the passing of information is an act (a hand-shake) executed simultaneously by sender and receiver.

These three models differ in other respects but the different choice of primitive notion is responsible to a large extent for their disparity.

Second, work has been done on operational models of concurrency which has had considerable application but has not gained a sufficient degree of abstraction. Examples are the parallel program schemata of Karp and Miller, and the Data Flow Graphs of Dennis et al. Such models elucidate the (partial) sequencing of basic actions which make up a concurrent computation, but it seems that they do not provide an analogue of the correspondence between evaluation (concrete) and mathematical function (abstract) which is achieved for sequential computation in the theory of Scott, Strachey and their followers. Such a correspondence has practical importance, since it is needed to validate methods for proving properties of the behaviour of computing systems.

Part of the difficulty in obtaining a sufficiently abstract theory is to find the correct analogue of 'mathematical function' in describing the behaviour of a computing agent which, in the course of computation, may communicate with a number of other agents. A function, in the ordinary mathematical sense, describes the behaviour of an agent which takes one or more inputs at the start, and at some later time (with no intervening communication) delivers an output as its final action. This is clearly not

enough. The processes of Milne and Milner are an attempt at providing the right kind of mathematical object. They are amenable to algebraic treatment (they constitute what is called a Flow Algebra), which is a considerable advantage in conducting proofs. The aim of another project at Edinburgh, 'Application of Flow Algebras and Processes to Problems in Concurrent Computation', is to apply processes and other Flow Algebras in practical case studies in order to assess their practical viability; this will provide evidence for the present (more theoretical) project as to the role of algebra in the theory.

One purpose of the present project is to develop the theory of the Flow Algebra model. However, this will not be enough. The aim is also to integrate this work with other approaches - in particular, with Petri net theory and Hewitt actors. In the course of this, mathematical tools must be developed. Already, Plotkin's Powerdomain construction (which explains much of the semantics of nondeterminism) has been needed to define the notion of process. A further problem is that the theory should work at different levels of abstraction - for example, it appears that to prove properties of systems such as freedom from deadlock, and from starvation of individual components, one needs to work at a less abstract level than in discussing, for example, the set of possible outcomes, or in proving the determinancy of a system. For this and other purposes, it is expected that the research will rely heavily on algebra and category theory.

Finally, however good a mathematical theory of concurrency is for 'describing' computation, its value in engineering will be judged by whether its constructions are 'implementable'. Consider again the analogy of sequential computation. It was a major step in designing high level programming languages to discover that an arithmetic expression - which could certainly be used to describe the behaviour of an assembly language program - could also be 'executed' - that is could be written as part of a FORTRAN program. Part of the work must be, therefore, to maintain a link between process expressions - whatever they are - and execution, whether the expressions are to be implemented directly in hardware or via a concurrent programming language.

In summary, the aim is to obtain a foundation for distributed computing which is as primitive and economical in its concepts as existing foundations for sequential computation.

PROGRESS TO SEPT 1979

The earlier work on power domains has been continued, linking them up to non-deterministic domains. These are the kinds of structures usually considered in the Scott-Strachey theory, together with an extra union function satisfying the equations:

1. Associativity $xU(yUZ) = (xUy)UZ$

2. Commutativity $xUy = yUx$

3. Absorption $xUx = x$

Using these new structures, it was possible to obtain full abstraction, in certain senses, for a simple case [4]. Other ideas of some interest are the use of a kind of conditional production system in specifying the operational semantics of the language considered, and the categorical and algebraic tools (especially a tensor product) used to obtain the denotational semantics.

The operational semantics of finite processes have been investigated and a first draft of the results is available [5]. A new notion of operational equivalence is defined using the idea of observers performing experiments on processes. This equivalence is then characterised by a finite set of axioms. This work is continuing and it now seems that similar results are true of infinite processes.

STAFF

Dr M C B Hennessy joined the project as a research fellow on 1 January 1979. G Winskel joined the project as a research associate on 1 October 1979.

VISITS

Discussions have been held with: K Apt (Erasmus University, Rotterdam), B Cohen (Standard Telecommunications Ltd), H J Genrich (GMD, Bonn), P E Lauer (Newcastle University), W P de Roever (Utrecht University), and M W Shields (Newcastle University).

REFERENCES

(1) G.Milne, R.Milner, 'Concurrent Processes and their Syntax'. CSR-2-77, to appear in JACM.

(2) R.Milner, 'Flowgraphs and Flow Algebras'. CSR-5-77, submitted to JACM.

(3) G.D.Plotkin, 'A Powerdomain Construction'. SIAM Journal on Computing, Vol 5 No 3, pp 452-487.

(4) M.C.B.Hennessy, G.D.Plotkin, 'Full Abstraction for a Simple Parallel Programming Language', Mathematical Foundations of Computer Science, Olomouc (1979).

(5) M.C.B.Hennessy, R.Milner, 'On Observing Nondeterminacy and Concurrency', Dept of Computer Science, University of Edinburgh (1979).

PROF I C PYLE

UNIVERSITY of YORK

REAL TIME PROGRAMMING LANGUAGES FOR INDUSTRIAL AND SCIENTIFIC PROCESS CONTROL
April 75 - Dec 78

This project ran from April 1975, to December 1978. The original objectives were to study requirements for real-time programming languages with particular reference to the work of the LTPL-E (Long Term Procedural Language - Europe) committee. It was intended to make trial compilers and to study various features by writing programs to assess them, paying particular attention to tasking and the run-time aspects of the language.

During the period of the project, the LTPL-E committee changed its role, and the impetus of language design for real-time systems left Europe with the establishment of the US Department of Defense High Order Language project (Steelman). Also the programming language Modula was published, as a language for dedicated real-time systems.

Since the languages for the DoD project are based on Pascal, and Modula was designed in the light of Wirth's experience with Pascal, it was decided to implement a Modula compiler as a basis for the studies. This compiler has been very successful, and it is being distributed under a separate grant (see Wand, below). Most of the studies carried out were on Modula, but also include paper programming in Ada, and possible Modula extensions.

Programming in Modula for a variety of projects in the Department gave a basis of experience by about ten people, from undergraduates upwards. The influence of the target machine architecture was studied, particularly for the Intel 8080, Zilog Z80 and Texas 9900 microprocessors. Work on the Z80 was done in association with Linotype Paul Limited, who are planning to use Modula as a system implementation language; they have now developed a Modula compiler for the Z80.

The final work concentrated on using Modula for a substantial real-time programming task - controlling a digital radio receiver to sweep a range of frequencies and detect changes in the ionosphere. This was done in association with the University of Leicester: they want the data which is collected. It provided a real problem with 'strange' input/output equipment and a time-dependent control loop.

This project was fully described in a final report submitted to the SRC [2].

REFERENCES

(1) J.Holden, I.C.Wand, 'Experience with the Programming Language Modula'. 1977 IFAC/IFIP Real Time Programming Workshop, June 1977.

(2) I.C.Pyle, I.C.Wand, 'Real-Time Programming Languages for Industrial and Scientific Process Control'. Computer Science Report No 21, University of York, May 1979.

PROF B RANDELL

UNIVERSITY of NEWCASTLE UPON TYNE

RELIABILITY AND INTEGRITY OF DISTRIBUTED COMPUTING SYSTEMS
Oct78 - Sept 82*

Distributed systems can suffer from reliability problems which do not occur in conventional systems. These problems arise from the existence of concurrent, yet inter-dependent, activities in a distributed system. For example, one part of the system may be taking decisions and performing actions in ignorance of the fact that these decisions have been based on copies of data that are already out of date, or even that some part of the system has already found to be erroneous.

The specific problem that the Newcastle project will concentrate on is that of designing distributed computing systems which can cope effectively with situations in which it is belatedly detected that erroneous data has been allowed to spread from computer to computer in the system. Ideally, of course, such spreading would be prevented. In practice, such prevention can never be guaranteed, if for no other reason than the fact that it is impossible for a system (or a system component) to check that the inputs it receives are indeed correct with respect to the aims of its environment rather than just consistent with data received previously, and valid with respect to specified rules.

The work to be undertaken forms a natural successor to earlier work at Newcastle on fault-tolerant systems [1,2], including work on error recovery amongst sets of processes that are competing for shared resources [3]. A start has in fact already been made on extending this work to distributed computing systems, and one quite general technique for state restoration in distributed systems has been developed [4].

The approach to be taken is centred on:

(1) the development of a 'system-level design language' which provides means of expressing policies and strategies relating to the overall reliability of the distributed system, and the integrity of the global data that the system is expected to maintain, and

(2) the design of efficient decentralised (and in fact highly parallel) mechanisms which can embody such policies and strategies.

Existing project hardware resources (two PDP11/45's) are to be augmented, and used for the project. The work is to be carried out in close consultation with Computer Analysts and Programmers Ltd., so that it can take proper account of the realities of a number of current and likely future actual distributed computing systems, and of the environments in which they are used.

PROGRESS TO SEPT 79

Earlier work on distributed systems, which resulted in the development of the 'chase protocols' as a general scheme for backward error recovery has been developed further. Work has also continued on several of the other concepts which originated in an earlier project such as inclusive and

disjoint recovery in multi-level systems, in order to try and extend the ideas involved to the case of distributed systems.

Stemming from the work on chase protocols, the central notion of an atomic action has been formalised in terms of occurrence graphs (causal nets). This had led Eike Best to attempt to clarify to what extent such graphs overspecify dependencies which are irrelevant for the purposes of chasing and recovery - this in fact is an attempt to provide a much-needed understanding of information flow in asynchronous systems. His other major activity has been the search for a convenient means of generating structured occurrence graphs, i.e. to develop a language which directly and simply admits atomic actions (and concurrency) - the approach being taken is based on Dijkstra's guarded command language.

Complementing this, Santosh Shrivastava has been studying commitment in distributed systems, including such topics as two-phase protocols, the use of time stamps, and the maintenance of multiple copies of data. The aim is to identify the essential components of a minimal recovery system that must be supported by every node of a distributed system. Work is currently being concentrated on extending existing simple commitment strategies to multiple processes, and on auditing requirements.

Pete Lee has contined the investigations of fault-tolerant interfaces in multi-level systems [14], the architectural features required [11,12], and the implications of distributed control on these features. He has also been heavily involved in the development of the Unibus recovery cache [13], the hardware for which has been constructed and is now being commissioned by Din Ghani and Keith Heron.

STAFF

The following staff are currently employed on the project: Dr P.A.Lee, E.Best, K.Heron, Dr F.Cristian, G.Wood. These latter two replace Dr Shrivastava and Dr Anderson who have been appointed to lectureships in the Computing Laboratory.

VISITS

Dr T.Anderson spent eight months leave of absence from the project during 1978/79 at NASA, Langley, Virginia. Several project members visited the States during the summer of 1979, principally to repeat, at the Univestiry of California at Santa Cruz, the Reliability Course first given at Newcastle in 1978. The lectures have been published by Cambridge University Press in the book 'Computing Systems Reliability' (Eds. Anderson and Randell).

REFERENCES

(1) B.Randell, 'System Structure for Software Fault Tolerance'. IEEE Transactions on Software Engineering, SE-1,2 (June 1975) pp 220-232.

(2) B.Randell, P.A.Lee, P.C.Treleaven, 'Reliable Computing Systems'. Lecture Notes in Computer Science, Vol 60, Springer Verlag (1978) pp 282-391.

(3) S.K.Shrivastava, J-P Banatre, 'Reliable Resource Allocation between Unreliable Processes'. IEEE Transactions on Software Engineering, SE-4,3 (May 1978) pp 230-240.

(4) P.Merlin, B.Randell, 'State Restoration in Distributed Systems'. Proc FTCS-8, Toulouse (June 1978) pp 129-134.

(5) T. Anderson, P.A.Lee, 'An Example of Recovery in Multi-level System'. SRM/205, (October 1978).

(6) P.J.Smith, 'Communication Requirements of Parallel Programming'. SRM/206, (November 1978).

(7) E.Best, B.Randell, 'A Formal Model of Atomicity in Asynchronous Systems'. Tech. Report 130, (December 1978).

(8) E.Best, 'On Information Transformation'. SRM/209, (January 1979).

(9) P.A.Lee, J.L.Lloyd, S.K.Shrivastava, 'Some Critical Comments on the paper "An Optimal Approach to fault Tolerant Software Systems Design" by Gannon and Shapiro'. SRM/210, (February 1979).

(10) E.Best, 'A Note on a Concurrent Program'. SRM/211, (February 1979).

(11) P.A.Lee, 'The Provision of Recovery in Interpretive Systems'. SRM/212, (February 1979).

(12) P.A.Lee, 'Inclusive and Disjoint Recovery Revisited'. SRM/213, (March 1979).

(13) P.A.Lee, N.Ghani, K.Heron, 'A Recovery Cache for the PDP-11'. Proc FTCS-9, Madison (June 1979) pp 3-8.

(14) T.Anderson, P.A.Lee, 'The Provision of Recoverable Interfaces'. Proc FTCS-9, Madison (June 1979) pp 87-94.

(15) B.Randell, 'Software Fault Tolerance'. Proc EURO IFIP (1979).

PROF B RANDELL

NEWCASTLE UNIVERSITY

A PROJECT TO INVESTIGATE THE DESIGN OF HIGHLY CONCURRENT GENERAL-PURPOSE COMPUTING SYSTEMS
July 78 - Sept 80

The project on the design of highly concurrent computing systems is investigating computer and program organisations for a tightly-coupled, multi-instruction-multi-data stream (MIMD) computing system which is aimed at supporting highly concurrent general purpose computation. Given such a computing system, its power may be exploited by multiprogramming (ie by running a number of independent programs concurrently) or by utilising the natural structure of a highly parallel program.

The project is in fact a fairly direct continuation of Dr Treleaven's early work on data flow computing, both at Manchester University [1] and subsequently at Newcastle [2,3]. However, investigations have not been restricted solely to organisations based on data flow computation. Initial work was based on a 'generalised' concept of control flow [4] and the current belief is that various methods of evaluation, including data flow, control flow and demand driven schemes, can naturally co-exist in a single concurrent computer.

PROGRESS TO SEPT 79

A simple classification developed by the project for models of computation illustrates the inter-relationships of the various schemes. Two mechanisms are common to all models of computation, firstly the data mechanism by which an instruction communicates data to other instructions, and secondly the control mechanism by which an instruction causes the execution of other instructions. There are two types of data mechanism: by value where data is passed directly between instructions, and by reference where data is passed via a shared memory cell addressable by the source and destination instructions. Also there are two types of control mechanism: by availability where controls, signal that input data is available for use and by need where controls, signal the request for output data.

The following table shows the appropriate data and control mechanisms for various computation models:

model of computation	data mechanism: by value	data mechanism: by reference	control mechanism: by availability	control mechanism: by need	mnemonic
data flow	Y	-	Y	-	Y/A
control flow	-	Y	Y	-	R/A
data-control flow	Y	Y	Y	-	VR/A
reduction	Y	-	-	Y	V/N
combined model	Y	Y	Y	Y	VR/AN

From this classification it can be seen that data flow schemas are 'by value/by availability', control flow schemas are 'by reference/by availability' and demand driven reduction machines are 'by vaule/by need'.

In reaching this classification a number of computer organisations have been investigated. Initially a 'slotted' ring architecture based on a 'generalised' control flow (GCF) model - class R/A was studied. The 'slotted' ring is a circular conveyor belt used for the communication of information between resources (processors, memories) that each have access to one slot on the ring.

To investigate the ring-based computer two simulators were constructed, one to study the logical behaviour of the computer [4], and the other to execute the computer's machine code programs. In addition a single assignment language called VORPAL [5] was designed and implemented. The conclusions from this exercise are that the ring-based computer provides a suitable basis for a MIMD architecture, but that the GCF model of computation, like data flow, is inadequate for general purpose program representation.

To overcome these inadequacies the project has designed a computer that integrates the concepts of data flow, control flow and updateable memory [6]. (This computer is in class VR/A.) The design can be used either as a pure data flow computer or as a control flow computer, or, more interestingly, as some combination of the two. The group is in the final stages of implementing a simple version of this computer using microprocessors and producing documentation [7] that will allow the machine to be quickly built by any other group interested in a research vehicle for concurrent computing.

The conclusions from this experiment are that the 'by value' and 'by reference' data mechanisms may co-exist, but that the 'by availability' control mechanism should be enhanced with the 'by need' mechanism. This leads to machines in class VR/AN.

So as to fully understand the concepts of 'by need' driven computing (ie class V/N), the group have gone through the exercise of designing a reduction machine [8]. In the next few months research students will implement this machine in the same way as the data flow-control machine has been implemented.

In summary, having studied the various models of computation and their implications in terms of computer architecture, we believe that the two data mechanisms and two control mechanisms naturally go together and that a suitable concurrent architecture for their support will be a ring-based computer. With this thesis the group is pursuing a computer design in class VR/AN which it is hoped will satisfy the aims of the project and allow an application to be made to the Science Research Council for a modest grant to build a prototype.

STAFF

One full-time research assistant, Dr P.C.Treleaven is employed under this grant.

REFERENCES

(1) P.C.Treleaven, 'Exploitation of Parallelism in Computer Systems'. Ph.D. Thesis University of Manchester (Feb. 1977).

(2) P.C.Treleaven, 'Principle Components of a Data Flow Computer'. Proc. Euromicro Symp. (Oct 1978) pp 366-374.

(3) P.C.Treleaven, 'Exploiting Program Concurrency in Computing Systems'. IEEE Computer, Vol 12, No. 1 (Jan 1979) pp 42-50.

(4) E.P.Farrell, N.Ghani and P.C.Treleaven, 'A Concurrent Computer Architecture and a Ring Based Implementation'. 6th Annual Symp. on Computer Architecture (April 1979) pp 1-11.

(5) R.P.Hopkins, 'VORPAL - A Single Assignment Language for Programming the GCF Simulator'. M.Sc. Dissertation, University of Newcastle upon Tyne (1978).

(6) R.P.Hopkins, P.W.Rautenbach and P.C.Treleaven, 'A Data Flow Computer with Addressable Memory'. Data Flow Workshop in Toulouse (Feb. 1979).

(7) R.P.Hopkins, P.W.Rautenbach and P.C.Treleaven, 'A Computer Supporting Data Flow, Control Flow and Updateable Memory'. Tech. Report No 144, Computing Laboratory, University of Newcastle upon Tyne (1979).

(8) P.C.Treleaven and G.F.Mole, 'A Multi-processor Reduction Machine for User-defined Reduction Languages'. Internal Report ARM/6, Computing Laboratory, University of Newcastle upon Tyne (1979).

N H SHELNESS

EDINBURGH UNIVERSITY

AN ARCHITECTURE FOR A MULTIPLE COMPUTER SYSTEM
Oct 77 - Sept 80

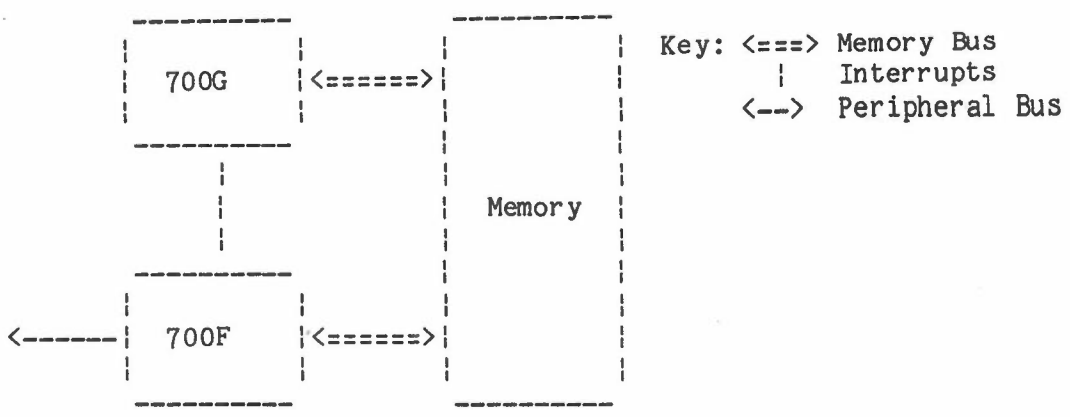
This project has been running in one form or another since 1973. It sprang from a supposition common to a number of other projects in the program, that there exists a broad class of computer applications for which hardware containing more than one processor (CPU) would form a more effective base than hardware containing only a single processor. Increased effectiveness can be measured in terms of improved reliability, throughput, response, ease of growth or economy, the relative weighting of which will depend upon the nature of the application.

An early study of how these various improvements could be effected in a system containing more than 2 or 3 processors led rapidly to a conclusion that such systems would need to make only minimal use of a shared common memmory and that, this being the case, a more effective solution was to dispense with shared memory and to employ an inter-machine communications sub-system in its stead. An immediate result of dispensing with shared memory is that an application will have to be programmed differently. The application will require a structure during execution that allows the code and data required by a sub-computation to be firstly identified and secondly transfered between and assigned to particular processors and their private memories.

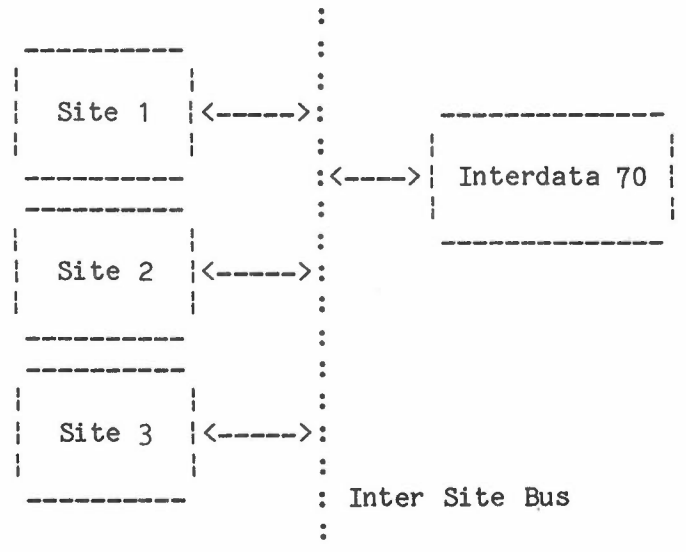
From 1974-76 the group, especially Dr Liam Casey, devoted its efforts to developing and evaluating through simulation a structure (the distributed domain model) that would satisfy this requirement. During 1977 and early 1978 a number of improvements to this model were developed and evaluated. This led to the elaboration of a new, simpler and more powerful structure (the segment flow model) based on a generalisation of the distributed domain model.

In parallel with this latter effort, the group approached the SRC to fund the construction of a multi-computer test-bed on which the proposed models could be implemented and evaluated by successfully constructing a number of realistic pilot applications.

The test-bed, which was installed in mid-1978, is built from Ferrant Argus 700 components. There are three identical computers or sites. Each site consists of an Argus 700G CPU, 64 Kbyte of memory and an Argus 700F CPU which serves as interface processor. The inter-connection of a site is shown below.



The three sites are linked via the peripheral bus of each of the Argus 700Fs to a fast inter-site bus into which is also connected an Interdata 70 computer. This Interdata 70 which serves as a peripheral and maintenance processor for the entire system runs existing software and is linked through a high speed local network to standard peripherals and the Department's file processor. The interconnection of the sites is shown below.



The basic software required at each site is under development, and the group expects to be constructing and evaluating applications from mid-1979.

PROGRESS TO SEPT 79

A considerable amount of effort during the first year of the project was devoted to the acquisition of the Ferranti ARGUS 700 computing equipment, which was delivered and installed in July 1978.

Since acquiring the equipment a number of instances of four additional hardware modules have been designed and constructed. These are:

(1) A dual EIA/CS LINK interface card (7 instances).

All peripheral equipment and computers in this department have been provided with one or more high speed, receiver controlled, asynchronous, serial, interfaces referred to as a CS LINK.

This card interfaces two CS LINKS (one for input and one for output) to the ARGUS 700 parallel PERIBUS. A number of additional chips on the card allow it to operate alternatively as an RS 232C to ARGUS 700 PERIBUS interface. Each ARGUS 700G is provided with two of these cards, and there is one general spare.

(2) A peripheral interrupt unit (4 instances).

The ARGUS 700 PERIBUS has no interrupt facility. As this was considered to be a shortcoming, a special card has been constructed which interfaces both to the ARGUS 700 PERIBUS and to the interrupt system.

(3) An inter-machine bus controller (2 instances).

(4) An inter-machine bus interface card (4 instances).

This interfaces a port on the inter-machine bus to ARGUS 700 SERIAL PERIBUS. Each ARGUS 700F is provided with one interface, and there is one general spare.

A complete set of design and construction documentation is available for each of these modules.

It is not intended to design or construct any more hardware during the remaining project period.

SOFTWARE

The initial phase of software design and implementation is now drawing to a close. During this period, the investigators have developed a number of tools that were needed to pursue the next stage of project research, which will involve the construction of multi-computer applications and control procedures. The tools already constructed include:

(1) An assembler.

The assembler accepts a straight forward high level assembly language (HAL 700) which is used for writing code that interfaces to the bare machine.

(2) A compiler.

This consists of a two pass back end which accepts intermediate code generated by the IMP 77 compiler. The investigators have chosen IMP 77 because of the ease with which they could both construct a compiler and insert special features into the language.

(3) A linker.

This is a straight forward multi-area linker capable of combining and satisfying references in multiple code modules produced by both the assembler and the compiler.

(4) A set of diagnostic utilities.

Post mortem dump decoders, hard binary to assembly language decoders, etc.

(5) A simple single domain kernel.

This kernel was constructed to allow the checking out IMP 77 programs on ARGUS 700 prior to construction of a full system. Both the assembler, the compiler, the linker and numerous other programs have been successfully run under it.

These facilities have been well documented, and, in the case of both HAL 700 and IMP 77, manuals exist. All of the software, with the exception of one module in the kernel (5 above) has been written in IMP 77 and has been cross-compiled, both on a VAX 11/780 and an Interdata 70, as well as compiled on ARGUS 700 itself. Compatible IMP 77 compilers are also available for the PDP 10, the PDP 11, the ICL 4/75 and the Interdata 32 bit range. An additional compiler is under development for the Intel 8086 16-bit microprocessor.

RESEARCH

It is intended to construct a three domain implementation of the three pass IMP 77 compiler to be run in parallel on the three ARGUS 700 systems by the end of the summer. Domain templates and capabilities for this application will be constructed by hand. This will facilitate the evaluation of the impact of certain kernel constructs prior to their detailed implementation.

As long as this phase progresses satisfactorily (as the investigators have some confidence it will), it should be possible to demonstrate a fully working system capable of running additional applications with ease by late autumn. At this time performance evaluation will begin. It is hoped to have initial results ready for publication by the summer of 1980.

STAFF

Dr L Casey was unfortunately recalled to New Zealand. Three research associates are employed on the project, Dr H J Jeanronde (from July 1978), Dr M J King, and K T Howard (both from October 1978). A University Demonstrator, A Vernon, has also made a considerable contribution to the project.

REFERENCES

(1) L.Casey, N.H.Shelness, 'A Domain Structure for Distributed Computing Systems'. ACM Operating Systems Review, Vol 11, No 5 and CSR-8-77, Edinburgh Computer Science Dept.

(2) L.Casey, 'Structures for Distributed Computing Systems'. CST-2-77, Edinburgh Computer Science Dept.

(3) L.Casey, 'On Kernels for Distributed Computing Systems'. CSR-27-78, Edinburgh Computer Science Dept.

DR M R SLEEP

UNIVERSITY OF EAST ANGLIA

INSTRUCTION SETS FOR DATA FLOW ARCHITECTURES : A COMPARATIVE STUDY
Oct 78 - Sept 79

The ultimate goal of the project is the design of an instruction set for a packet communication architecture which maps upwards onto the expressive power of Concurrent Pascal and downward onto an economically realisable hardware configuration which is extensible.

Preliminary work will involve a comparative study of the more stable designs (for example, the Manchester machine being developed by Gurd and Watson). The expected outcome of this phase of the work is an abstract model at the ISP level which is capable of describing the more interesting proposals, and highlighting their differences. A simulator based on such an abstract model in principle enables users to switch from one design to another allowing comparative studies of performance to be carried out.

It is expected that various natural features of a good instruction set for a packet communication architecture will emerge which have important implications at the language level. It is clear, for example, that it is easy to implement BCPL-like local memory, but more difficult to implement local memory in the Algol sense.

In pursuit of the longer term goals, close collaboration will be maintained with Treleaven's group at Newcastle who are developing a novel form of general purpose MIMD architecture.

PROGRESS TO SEPT 79

Significant progress can be reported with respect to both goals of this investigation into instruction sets for novel packet communication architectures.

Visits to Manchester, Toulouse, Westfield College in London and Newcastle have provided invaluable details of the work of these groups. As a result an abstract model at the ISP level of a typical dataflow instruction set has been developed. This model has emerged from a particularly fruitful series of meetings with Newcastle.

Perhaps the most important result of the project to date is the recognition that pure dataflow, with its insistance on a data-driven evaluation regime, raises considerable problems when applied to the new generation of high level languages such as CAJOLE and LUCID. For whilst a fully-concurrent, data-driven regime may be used, in general some element of demand-driven evaluation is required if termination is to be guaranteed.

With most conventional dataflow instruction sets, which tend to be designed in a somewhat ad-hoc manner, it is difficult to approach such problems with any confidence. A much firmer basis, advocated by Backus and Berkling, replaces the notion of instruction execution by the notion of reduction. Because reductions never change the meaning of a computation, they may be performed in any order, or even concurrently, and the result of a computation which terminates will be unique for a suitable language,

together with its reduction operators.

One such system has been implemented by David Turner of Kent University. The language, SASL, is something like CAJOLE with the infinite sequences of LUCID. The system converts a SASL program into a labelled binary tree of constants, some of which are logical combinators. The tree may be attached to an argument, and the whole reduced to yield the result.

The abstract model mentioned above has very recently been applied to the problem of translating Turner's reduction operators into a form suitable for packet communication architectures. Unless any unforeseen difficulties arise, it should be possible to move Turner's system to the Newcastle Unix machine. The result should be a useful software vehicle for exploring evaluation regimes for packet communication architectures.

VISITS

Dr Sleep visited the Toulouse workshop earlier in 1979 and Arvind at MIT in September 1979.

DR M S SLOMAN

IMPERIAL COLLEGE, LONDON

COMMUNICATIONS FOR DISTRIBUTED PROCESS CONTROL
Sept 78 - Aug 81

The advent of cheap microprocessors has made economic the possibility of implementing local networks of microcomputers for the distributed control of industrial plant and laboratory apparatus. In these local networks individual computers need to communicate with each other to coordinate their actions and achieve cooperation.

The traditional centralised control approach can also benefit from the use of a microprocessor based communications network. The amount of information transmitted from the point at which it is generated can be reduced by partially analysing and formatting the data and, possibly, by using data compression techniques. In addition the memory of the microcomputer can act as a buffer to smooth out bursts of traffic. The resulting lower capacity data transmission requirements will permit the use of serial rather than parallel transmission techniques between a computer and plant interface. Also, the use of appropriate packet error control techniques can achieve very reliable communication.

The Imperial College Research Project will investigate the issues involved in the design of local networks. A small network of four LSI-11 microcomputers linked to a GEC4080 will be used for a typical laboratory automation application - the distributed control of an electron probe microanalyser.

The research objectives are first to develop a methodology for the assessment of local computer networks. This work is being undertaken in collaboration with the European Distributed Intelligence Study Group (EDSIG) which is funded by the Commission of the European Community. A set of criteria are being defined which can be used to assess the suitability of a communication system for process control application. These criteria will be used to produce a survey of communication techniques which are available commercially or are in use in research laboratories or industry.

The second objective is to investigate the relevance of the CCITT X25 Recommendations for control and automation applications. The X25 recommendations are defined as an interface to a public packet switched network, but can also be used as a computer to computer communications protocol. The intention is to define a suitable subset of the full X25 specification and assess its suitability for a typical distributed control application.

The final objective is to develop communication techniques which are suitable for both broadcast and store-and-forward networks. A broadcast system (for example, highway or ring) is one in which a common transmission medium is shared by a number of stations. A store-and-forward system (for example, star or mesh) has multiple point-to-point lines interconnecting stations, so a message or packet may have to be routed via an intermediate node to its destination. An industrial site may contain both types. Broadcast serial highways could be used for controlling a local function and these sub-networks could be connected to a larger computer which may in

turn be connected, possibly over Post Office lines, or by a public packet switched network to a central computer which performs management functions etc.

The intention is to adapt X25 for use as a communication protocol which is suitable for use in both categories of networks. It is hoped that the communication techniques developed for synchronisation and transfer of information will be applicable to other distributed processing applications and even to resource sharing local networks.

The industrial relevance of this project is:

(1) Use of X25 LSI Circuits. In general, the process control industry does not have a large enough market to develop its own LSI circuits, and so must use existing circuits. The potential market for intelligent terminals with communication interfaces is very large, and so semiconductor manufacturers are already producing LSI circuits to support level 2 of X25 (HDLC). In the near future VLSI circuits will be produced which will support more of the X25 functions. Even if X25 is not ideal for the process control environment, it will be more cost effective to use these hardware circuits than to implement protocols in software.

(2) Operating System Support for X25. With the adoption of X25 as an international standard, and with the increasing LSI circuit support for it, most computer manufacturers are likely to provide support for X25 in their operating systems. It would then be feasible to use a single operating system interface to access both local and remote peripherals, thus reducing the complexity of the operating system.

(3) 'Black Box' Communication System. Software for reliable communications is complex and costly to produce. It is hoped that this work could result in a 'black box' communication system. Microprocessors could be plugged into the communication system via standard simple interfaces. This would simplify the design of distributed control systems.

(4) Communication Standards. At present, various manufacturers are working on communication systems for their industrial control equipment, based on 'ad hoc' designs. It is desirable to standardise the communication system to allow compatability between equipment produced by different manufacturers. It would be better to base such a standard on existing ones from the Data Processing environment, than to come up with something completely new. It is hoped that our work will indicate whether or not an X25 based standard could be appropriate for the process control environment.

PROGRESS TO SEPT 79

The first objective, that of developing a technique for assessing communication systems for distributed process control, has been completed. The result is a structured questionnaire which elicits a description of the system in a common format and terminology. It is considered to be of value as a learning aid and so is being published as an EDISG report by the European Camac Association.

It is hoped that EDISG will act as a repository for descriptions of as many different communications systems as possible, which will be published as a booklet. This would be of use to people working on local networks for other applications as well as control.

An attempt is being made to define and clarify the functional requirements of distributed control systems for using a communication system. When this is complete a small sample of industries will be surveyed in order to obtain quantitative values for performance and reliability requirements.

A survey of the design problems posed by distributed control systems is in hand. The survey reviews the state of the art of present methodologies and solutions and, most important, the research directions in the use of tools and methodologies from control theory in computing science and vice versa.

STAFF

One full-time RA, Dr A Nader, has been employed with effect from October 1978.

VISITS

Several meetings relating to EDISG activities have been attended, in particular, the November 1978 and May 1979 EDISG meetings in Brussels, the Purdue Europe TC5 meeting in January 1979, and the Purdue Europe Workshop in Zurich in April 1979.

Discussions have been held with the NCB, ICI Corporate Laboratory, Warren Spring Laboratory, Smiths Industries and the CEGB research laboratory.

REFERENCES

(1) M.S.Sloman, B.K.Penney, T.A.Marsland, 'A Communications Protocol for Distributed Microprocessors'. Proceedings of Conference on Distributed Computer Control Systems, Birmingham 1977, IEE Publ. No 153.

(2) M.S.Sloman, 'The Relevance of the CCITT X25 Recommendations To Process Control Applications'. Real Time Data '79 Conference, Berlin October 1979.

(3) EDISG 'A Technique for the Assessment of Data Communication Systems for Process Control'. EDISG/COM/01 September 1979. Available from ESONE/ECA Secretariat of the European Communities, JRC-CBNM, 2440 Crel, Belgium.

(4) S.M.Prince and M.S.Sloman, 'The Communication Requirements of a Distributed Computer Control System'. Research Report No 79/33, Imperial College.

(5) A.Nader, 'Distributed Computer Control Systems - An Overview'. Research Report No 79/24, Imperial College.

(6) A.Nader, 'The Design of Distributed Compter Control Systems'. IFAC Workshop on Distributed Computer Control Systems. Tampa Florida, October 1979.

(7) A.Nader, 'Petri Nets for Real Time Control Algorithm Decomposition'. IFAC Workshop on Distributed Computer Control Systems, Tampa Florida, October 1979.

DR W W WADGE

WARWICK UNIVERSITY

DISTRIBUTED IMPLEMENTATION OF NONPROCEDURAL LANGUAGES
April 79 - April 80

This project is investigating implementations of nonprocedural languages (especially Lucid [1]) which make good use of parallel and distributed modes of computation. The main problem is the development of sophisticated program analysis techniques, most of which take the form of type checking and assignment. This involves basic research in semantics and the theory of data types.

REFERENCES

(1) E.A.Ashcroft, W.W.Wadge, 'LUCID - A Non-Procedural Language with Iteration'. CACM, July 1977.

DR I C WAND

UNIVERSITY of YORK

MODULA DISTRIBUTION AND PROMULGATION
Jan 78 - Dec 80

The programming language Modula is unusual in that it attempts to combine relatively conventional multiprocessing facilities with the references to the hardware of particular input/output device in one coherent language structure. The language also contains a sophisticated module mechanism with a strict security system for visibility control across module boundaries. The overall form of the language is based upon Pascal.

PROGRESS TO SEPT 79

A compiler for Modula has been written at York in BCPL to run on a PDP-11 computer running the UNIX operating system. Code can be generated for a target PDP-11 or LSI-11 with or without the Extended Instruction Set Option. The grant was awarded to enable York to develop, document and maintain the compiler, to make copies available to other workers and to enhance it in response to requests from users.

The second release of the compiler was made in March 1979 to about 60 users both in the UK and overseas. From reports received we know that the compiler is being used successfully in a wide variety of projects. Full details of the first year of the project are given in [2]. The only language restriction remaining in the implementation is that all names must be declared before use.

Effort is now concentrated upon:

(a) making small changes to the compiler in response to requests from users, including bug fixing.

(b) preparing a version that will run on the DECsystem 10 as a cross compiler. This version is being constructed so that only limited effort is required to put up a DECsystem 10 version when a new UNIX compiler is completed.

(c) rewriting the second pass of the compiler so that forward referencing and separate compilation are possible.

Collaborators at other Universities are preparing versions that run under RT-11 (Bath) and RSX-11M (Southampton). A code generator for the INTEL 8086 has been produced in Australia.

REFERENCES

(1) J Holden, I C Wand, 'An assessment of Modula', Department of Computer Science report number 16, University of York (1978). To be published in Software - Practice and Experience.

(2) I C Wand, 'Modula distribution and promulgation 1978', Department of Computer Science report number 17, University of York (1979).

(3) I D Cottam, 'Functional Specification of the Modula compiler release 2', Department of Computer Science report number 20, University of York (1979).

DR I C WAND

UNIVERSITY of YORK

MODULA DISTRIBUTION AND PROMULGATION - SUPPLEMENTARY PROPOSAL
July 79 - March 80

This award is for equipment only. It provides terminal equipment, both for text editing and printing, to help the staff on the SRC DCS Modula distribution project.

DR I C WAND

UNIVERSITY of YORK

DISTRIBUTED OPERATING SYSTEM FOR TIME-SHARING
Oct 79 - Sept 81

This project will study the design of a time-sharing operating system with the functional properties and interface as far as possible the same as UNIX but using an indefinite number of identical microprocessors. The initial investigation will consider the process distribution among the processors and the use of inter-processor connections forming a ring. An alternative approach will assume each processor acting as a personal computer. The objective is a feasibility study with comparison involving complexity, cost and performance, of a centralised and alternative distributed version of the same functional system. No implementation is proposed at this stage.

DR J WELSH

QUEEN`S UNIVERSITY, BELFAST

THE DESIGN, IMPLEMENTATION AND APPLICATION OF LANGUAGES FOR DISTRIBUTED COMPUTING
Oct 79 - Sept 81

This project aims at making a high quality implementation of Pascal Plus available on a number of major mainframes such as DEC 10, CDC7600 and IBM 370. This will help to propagate the use of Pascal Plus and form the basis for further research into languages for distributed computing.

DR C WHITBY-STREVENS and M D MAY

WARWICK UNIVERSITY

A BUILDING-BLOCK SYSTEM FOR DISTRIBUTED COMPUTING
Oct 77 - Sept 83*

The increased use of distributed computing systems has made the development of new techniques for designing and programming them essential. The aim of this research proposal is to provide a set of building blocks to support a top-down design process and to experiment with various topologies for a system containing a number of processors. This will provide the hardware and software facilities necessary to allow the system designer to realise his design as a distributed system without becoming involved in the low-level implementation details.

Current work at Warwick centres around the design and implementation of a programming language called EPL. EPL is intended to be used for the implementation of distributed database systems, real-time control systems etc.

Apart from its facilities for distributed computing, EPL has the appearance of a conventional sequential programming language. This is because it is intended to be easily implemented on existing computers, and also because such languages are familiar to programmers.

EPL is directed towards the programming of systems which, although consisting of a number of processors, are programmed as one system. This is unlike a computer network, for example, where each machine is programmed separately. In the latter case, the coordination (at an administrative level) of the independent operation of different systems, often with differing objectives, presents a difficult problem. It remains to be seen what programming aids are useful in this case, and whether they can be embodied in a programming language.

EPL is an experimental language and it evolves as limitations in its specification become apparent. For this reason, it contains very few primitives apart from those relating to parallel processing and distributed computing. EPL is intended to be useable on systems with different architectures, such as a single processor/memory, a closely coupled network, or a set of processor/memory pairs coupled via a bus. In fact, it is intended that programs written in EPL should be portable between systems with different architectures. For example, a program might be moved from a single processor system to a multiprocessor system to improve its performance. It remains to be seen to what extent such portability can be realised, since it seems that many special purpose distributed systems will require special programming techniques.

EPL Description

EPL is based on the idea of actors, which communicate only by sending and receiving messages. EPL separates the description of a computation from the actor which performs it. (Such a description is often called an act.) Thus many actors may be created to perform the same act (on different

data). Although an act may not refer to any global variables, it may have parameters, which are initialised when an actor is created from the act, and are then treated as local variables of the actor. This provides a means to inform actors about each other's existence, thereby allowing them to communicate.

EPL allows the declaration of structures of actors, such as arrays. Once such a structure is created, all the component actors start executing simultaneously, probably exchanging messages with each other. Note that each of the actors may itself be expressed in terms of other actors. This provides a means of 'top down' programming, since any actor can be decomposed into a set of intercommunicating sub-actors. In fact, an actor may have no existence at run-time, serving only to encapsulate a group of smaller actors.

The message sending primitives contain no buffering. Thus, when an actor tries to receive a message, it waits until another actor sends it one, and when an actor tries to send a message, it waits until the target actor receives it. This provides a conceptually simple mechanism for the programmer, and is relatively easy to implement on different architectures. However, there are arguments in favour of other schemes. For example, single buffering enables a communication system to proceed with the transmission of a message while the sender continues executing. These issues may become clearer in the light of attempts to implement some of these primitives.

EPL Implementation

One of the design aims of EPL has been to minimise the number of 'hidden' overheads. The creation of actors and the sending of messages is intended to be very efficient - this is one of the reasons why arbitrary length queues are not provided by the message sending primitives. The programmer can, of course, implement such facilities explicitly in terms of actors, where necessary. The EPL primitives may easily be used to model a variety of traditional programming techniques such as procedures, coroutines, classes, monitors.

Currently under development is a minimal distributed system (three processor/memory pairs linked by serial communication lines) which will be programmed in EPL. The allocation of actors to processors will be performed statically. This system will be used to explore a number of model applications.

It is hoped that this system will be expanded to about 10 processors, some of which will be able to be more closely coupled. It is also intended to investigate the construction of an 'EPL processor', in which the creation of actors and the sending of messages will be given hardware support.

EPL Issue

A current issue is whether the creation of actors and their allocation to processors should be static (performed by the compiler) or dynamic (performed at run-time). This seems to depend on the application. For example, a real time control system requires static allocation whereas a multi-access operating system requires dynamic allocation. It may turn out that a distributed computing language will have to provide both of these facilities.

Further work on EPL is required to make it more suitable for real-time programming. There are two complementary approaches. One is to provide compile-time feedback to the programmer about the expected running time of EPL actors between message transfers. The other is to permit the programmer to make assertions about performance requirements which the compiler could use to ensure that sufficient processor power is available to construct a priority schedule.

Another necessary feature is the provision of EPL primitives to aid error recovery. This might involve a statement of resources permitted to an actor, and a method of checking and handling violations.

PROGRESS TO SEPT 79

Development of the EPL programming language has continued throughout the year. However, a version was 'frozen' last December and a full implementation made available on the LSI/11. The 'EPL Programming Manual' [1] forms the first of a new series of technical reports specifically designed to reflect work on the project. This version of EPL both compiles and runs on the LSI/11 - with non-prioritised non-pre-emptive round-robin scheduling of actors. The runtime system operates in a 'bare' machine.

Dr Taylor has experimented with a small range of applications and the accumulated experience of programming these examples in EPL is being collected for a substantial technical report [3]. Dr Whitby-Strevens has also been studying the application of EPL to decentralised control algorithms [6].

D.May is continuing to work on the development of the language, particularly in the area of determining the sending/receiving actors for message transmission [4]. In the original EPL, messages are always sent to a specified destination. This is now being relaxed, so that one of an alternative set of destinations can be selected. This will allow, for example, further exploration of interesting classes of decentralised control algorithms.

Progress towards a distributed runtime system is being conducted in two stages. The first, which has been completed, is to implement a new code generator and runtime system on the TI9900 microcomputer. This is considerably more sophisticated than its LSI/11 counterpart and a great deal of care has been taken over its implementation. It has been specifically designed as the basis for a multi-machine system, it has a prioritised scheduling mechanism, and its uses interrupts for peripheral communication. Compilation still takes place on the LSI11.

The distributed runtime system will be developed from this. Particular issues to be resolved include the system-wide naming scheme, facilities for initialisatin and debugging, and tools for monitoring the behaviour of the 'distributed' EPL program.

John Lowe (appointed on 1 May 1979) has been working on an independent compilation scheme for EPL. In this scheme, information about the parts of a program already compiled are maintained on disk and used in later compilations. He is also working on mechanisms for allocating actors to processors. This is performed statically (actors are not moved from one processor to another), and forms an essential part of the distributed implementation.

The project welcomed Prof Ed Balkovich (University of Connecticut) for the summer. He worked on performance modelling of systems described in EPL [6]. The aim was to provide a prediction of the real-time behaviour of an EPL program, taking into account data-dependencies, actor-processor allocaton, and delays due to communication and synchronisation. The resulting model was combined with data produced by monitoring, providing some surprisingly encouraging results concerning the behaviour of programs written as a large number of small communicating actors.

REFERENCES

(1)M.D.May, R.J.B.Taylor, C.Whitby-Strevens, 'EPL- An Experimental Language for Distributed Computing'. Proceedings Trends and Applications: 1978 Distributed Processing, IEEE Computer Society, May 1978.

(2) M.D.May and R.J.B.Taylor, 'The EPL Programming Manual'. Distributed Computing Project Report No. 1, University of Warwick, May 1979.

(3) R.J.B.Taylor, 'Notes on Programming in EPL', Distributed Computing Project Report No. 3, University of Warwick (in preparation).

(4) M.D.May, 'EPL Systems Programming for Distributed Computing'. (in [5] below, pp12-18)

(5) C.Whitby-Strevens (ed) 'Record of a Workshop on Programming Languages for Distributed Computing; September 1978'. Distributed Computing Project Report No 2, University of Warwick, June 1979.

(6) C.Whitby-Strevens, 'On the Performance of Dijkstra's Self-Stabilising Algorithms in Spite of Distributed Control'. 1st International Conference on Distributed Computing Systems, IEEE (October 1979), pp 582-592.

(7) C.Whitby-Strevens, 'Towards the Performance Evaluation of Distributed Computing Systems'. COMPSAC '78 (in expanded form in Computer Centre Report No 18, University of Warwick, August 1978).

(8) E.E.Balkovich and C.Whitby-Strevens, 'A Model for the Behaviour of Distributed Systems', (in preparation).

PROF M V WILKES and DR R M NEEDHAM

CAMBRIDGE UNIVERSITY

DISTRIBUTED COMPUTING USING WIDE BAND COMMUNICATIONS
Aug 78 - July 81

In the past Cambridge have always thought of a large computer centre as being based on a main frame organised as shown in Figure 1. The main frame both provides the computing power of the system and also acts as a switching centre connecting the various peripherals. Figure 2 shows an alternative form of organisation which is now becoming possible and which is likely to become of increasing importance during the course of the 1980s. Here the user's consoles, processors, disc files, magnetic tapes, and other components are all interconnected through a wide-band local area communication system. These various components interact as 'users' and 'servers'. The primary users of a system will be human beings sitting at consoles of varying degrees of complexity. These consoles will be users of processors and the processors will themselves use other servers, for example, file-servers and hardcopy-servers.

The processors, or 'processing-servers' as they may better be termed in this context, will vary in speed and in the amount of high-speed memory with which they are provided. Some of them might be quite powerful, rivalling in speed the main frames of today, and one does not to have to project far into the future to see them having very large memories - even as much as a million words. Their design will, however, reflect the fact that they are not free standing, but form part of a distributed system. In particular, they will have one connection only to the outside world, namely, that to the wide-band communication system.

A user wishing to log-in would be put in touch with a program in one of the processing servers that would check his credentials and allocate to him the resources that he needed, including a processing-server on which to do his work. The processing-server might be allocated for his exclusive use or he might share it with a number of other people. The assumption lying behind the design of the system, however, is that the cost of processors will become sufficiently low for sharing to be the exception rather than the rule, perhaps being confined to situations in which a number of users are interacting with the same program.

The design of processing-servers is a subject calling for a serious research effort. The fact that they are to be used serially rather than simultaneously by a large number of people, coupled with the fact that they are likely to have large memories, means that fresh considerations must be given to memory organisation and memory protection. Also open as a subject for investigation is the extent to which a user, upon securing possession of a processor, could personalise it to his own needs by loading his own microprogram. Present day micro-programmable computers provide the user with only very limited scope in this respect, since the registers and their inter-connections have been designed with instructions of a particular format in mind. The microprogrammer has no control over such matters as how many bits are allocated to the function code, how many are allocated to the control of index registers and base-limit registers, and whether word or byte addressing is provided. Developments in micro-electronics have made it possible for the designer of a processor to contemplate much more

elaborate register structures than were formerly possible and thereby to give the programmer control over parameters such as those just mentioned which have hitherto been fixed in the design. It should not be necessary to pay any appreciable speed penalty in order to acquire this flexibility. The processor would not be completely 'soft' in the sense that it could emulate any given instruction set, but would differ from a conventional processor only in having a more complicated register structure that would support a wider range of micro-programmed instruction formats. The processor would be like a man who puts on many different kinds of clothes, all of which fit him, rather than like a man who wears other peoples' clothes.

The operating system would itself be distributed. Much work has been done in recent years, particularly, in Cambridge, on the subject of protection within an operating system. This subject acquires another dimension when the operating system is distributed. In particular, there is the question of whether the protection barriers should be located within the individual processors or in the interfaces to the communication system.

A program of research inspired by the ideas outlined above is in progress. A wide-band digital communication ring has recently been constructed and will form the backbone of the system. A plotter-server already exists and work has begun on a universal file-server. Until resources are available for the construction of an experimental processing-server based on the above principles, the CAP computer, constructed as part of a project in memory protection supported by the SRC, will be used as the principal processing-server, although other computers will also be connected.

PROGRESS TO SEPT 79

This section outlines the Computer Laboratory's work in Distributed Computing.

The work centres around the Cambridge Ring which provides a rapid means of communicating data from one machine to another. The CAP computer is being used as a 'home base' for much of the development. In particular it is being used for the development of protocols for the use of the ring. The ring itself transports very small packets carrying 16 bits of data. For almost all purposes these are aggregated into basic blocks, which carry up to 2k bytes of data together with a checksum and logical port number. Basic blocks may be sent and received at the maximum rate the ring will carry, but it is very important that they need not be handled at that speed. This makes it easy to connect to the ring devices which cannot support a high data rate even for a short time. As will be shown below, this makes it considerably easier to connect microprocessor-based systems to the ring than it would be to connect them to communication systems in which substantial blocks must be received and sent synchronously.

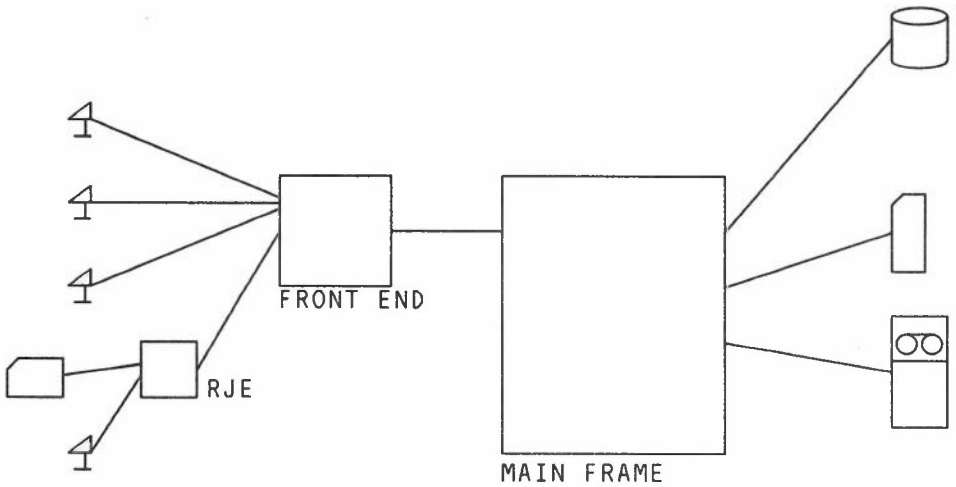


Figure 1

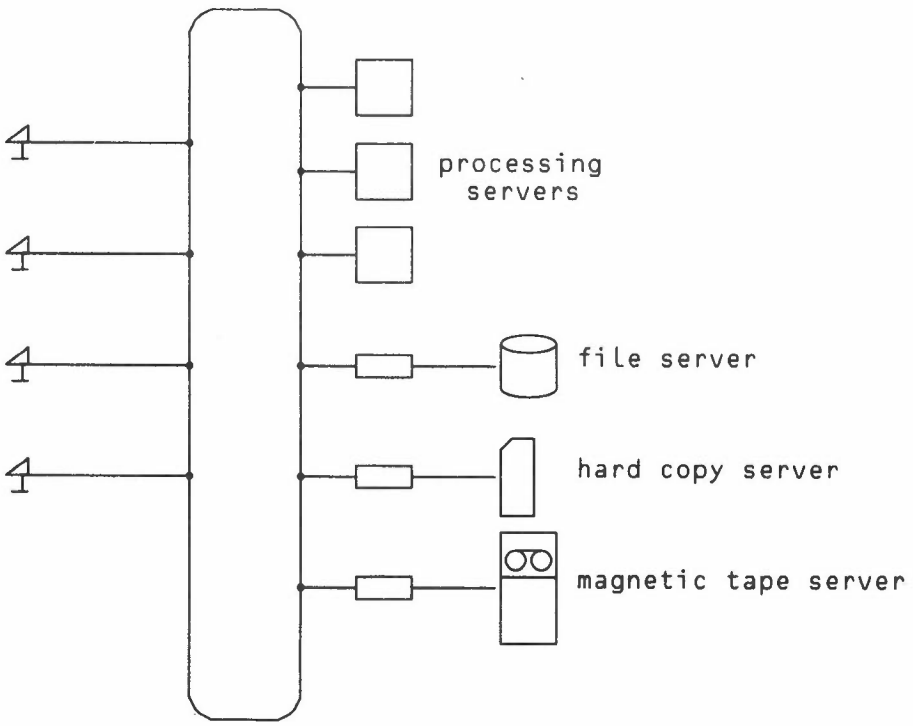


Figure 2

The error rate of the ring is low, so that retransmission units can safely be large. In order to exploit this property of local communications, a particular attitude has been taken to questons of higher levels of protocol. Wherever practicable, transactions are carried out in such a way that state is only retained at one end. For example, one may say to a file server 'send me words 0 to 5673 of file F on port P' thus indicating a commitment to take these words as fast as the file server can send them. If anything goes wrong, one tries the whole procedure again, probably with a different port number. The file server retains no state information about the transaction at all, and the safe arrival of the material is not acknowledged; where flow control is necessary one must use a more conventional protocol, and for this purpose a byte-stream protocol has been developed. The semantics of this have been decided with an eye on those of the Transport Service proposed by the Post Office Packet Switching Study Group 3 but the characteristics of the Cambridge Ring have been exploited in the implementation. In general the protocol efforts have been directed towards providing the maximum freedom for diverse implementation and sufficient uniformity of appearance for the systems on the ring to function as a coherent group where necessary. The basic block protocol for example is implemented in microcode for the CAP, in software for the PDP11, and mainly in an autonomous channel for the Computer Automation LSI4. Some designers prefer to reject material for the wrong logical port before it ever gets into memory (CAP) and others to accept everything and then decide what to throw away (LSI4). It is believed that in local communication systems of the future it will be essential to take advantage of diversity in this way.

Temporary protocols have been implemented for such functions as file transfer, logging in from one machine to another via the ring, disc archiving, and so on. These have been very useful in practice, and have contributed greatly to the design of more lasting protocols which are currently being brought into service.

Development of the file server on an LSI4 is under way; the design is complete and programming work is now going on. The implementation is being done under the Tripos portable operating system developed by Dr Richards' group, and this is expected to be used also with other ring-based servers, at any rate those implemented on minis rather than micros. A further 6 LSI4 machines have been ordered, of which one has so far been delivered.

In parallel with this activity a simple microprocessor system has been designed for ring use. It consists of a Z80A processor, some memory chips, and a very simple ring interface. There is enough room left on a 7" square board for ad hoc logic to control devices. These micros are connected to the ring in the simplest way possible so that their software polls the signals from the ring station. They can only transmit or receive at about 320K bits/sec, which is entirely adequate for most purposes; the flexible timing of basic blocks, mentioned above, makes this simplicity possible. Four such systems have been built. One is used as a name server, giving machine number, port, and function when looked up with a service name. It also controls a PROM programmer. Another serves to connect 4 VDUs to the ring, and a third to monitor ring errors. The final one to date is to be used to connect a line printer. These systems are loaded from the ring, typically at present from the CAP. In the future they will when switched on consult the name server to find where to bootload themselves from.

Plans have been made and implementation is underway for a radical change to the CAP system. The file server was designed with the intention that it could not only function in support of local filing systems, but as the total file system for some other machine or machines, and that it could carry out these rather different activities at the same time. It is intended to remove from the CAP system all trace of disc management, relying entirely on the file server used via the ring. A substantial amount of work has been done in this direction, and it is hoped that when the file server is ready the experiment will be made fairly easily. The exercise has proved very instructive already in revealing unexpected complexities, all soluble, in the distribution of function in a new way. The use of remote virtual disc would have been relatively straightforward, but we are also trying to remove some higher levels of management from the CAP. This line of work has been very helpful in sharpening up the specification of the file server and aiding understanding of some of the naming issues.

9.2 SENIOR VISITING FELLOWSHIPS

Senior Visiting Fellowships may be awarded as follows:

Incoming Fellowships - to enable senior scientists of distinction from recognised centres abroad (or in the UK) to visit by invitation the proposer's institution as Senior Visiting Fellows to give full-time advice and assistance in research in the fields in which the visitors are eminent; in introducing new techniques and new developments that may advance research work in the proposer's and other institutions in the UK; or in connection with specific research projects supported by SRC. These fellowships may be awarded independently; but where a fellowship is linked with a project for which an SRC Research Grant has been applied for or awarded, the fellowship would be considered as part of the project application or grant.

Outgoing Fellowships - to enable investigators to visit recognised centres abroad (or in the UK) to study new techniques and research developments that may advance research work in specialised scientific fields in his own and other institutions in the UK and/or directly benefit his own specific research projects. Where an investigator is requesting funds for travel in connection with a research project, however, such expenses will normally be provided under the Travelling and Subsistence heading of the grant.

The DCS Programme has obtained the following Senior Visiting Fellowships.

INCOMING

Host	Visiting Fellow	Award K pounds	Duration
Prof C A R Hoare Oxford University	J R Abrial Paris	7.2	Sept 79-Mar 80
Dr J D Roberts Reading University	Prof S Owicki Stanford University	0.2	July 79-July 79
Dr C Whitby-Strevens Warwick University	Prof E Balkovich Connecticut	2.7	June 79-Aug 79
Dr C Whitby-Strevens Warwick University	E D Jensen Honeywell	0.1	Oct 78-Oct 78

OUTGOING
-none

9.3 EMR CONTRACTS

Introduction

The aims of the DCS Programme include the reduction of duplicated effort and the production of software and hardware which are of general benefit to the majority of DCS investigators. Often the production of such items is more of a development task than a pure research exercise. In such cases, the DCS Panel can ask the Rutherford Laboratory to draw up a suitable contract between Rutherford Laboratory and a university to develop such a specific product. These agreements are called EMR (Extra Mural Research) contracts.

The contracts placed so far by the Panel are described below.

EMRS AWARDED

EMR HOLDER	PROJECT	AWARD K pounds	DURATION
Dr D Bustard	Pascalplus compiler	10.0	Nov 78-Nov 79
Software Sciences Ltd	Pascalplus tools	16.0	Dec 78-Feb 79
Dr I Wand	Unix X25	25.0	Sept 79-Aug 81

Descriptions of the contracts follow below.

DR D BUSTARD

QUEEN'S UNIVERSITY, BELFAST

PORTABLE PASCAL PLUS COMPILER
Nov 78 - Nov 79

The objective of this contract is to produce a compiler-interpreter system to allow Pascal Plus to be easily implemented on a variety of computers. This will facilitate software interchange within the DCS community.

The project has suffered two setbacks. The first is that the project is one staff member short. The second was the disruption of the University's 1906A computer, which is back in service now though somewhat unreliable.

The intermediate code for the P-machine has been defined and a working syntax analyser produced. Work on the code generator is now in progress. An LSI-11 and RT-11 kit from the DCS Equipment Pool are currently on loan to Dr Bustard to use as the first target machine for porting the compiler from the 1906A.

The estimated delivery date for the software is December 1979.

SOFTWARE SCIENCES LTD

FARNBOROUGH, HANTS

PASCAL PLUS DEVELOPMENT ENVIRONMENT
Dec 78 - Feb 79

The objective of this contract was to report on the software tools needed to create a good Pascal Plus development environment. The report was delivered in May 1979.

DR I C WAND

UNIVERSITY of YORK

UNIX X25 INTERFACE
Sept 79 - Aug 81

The objective of this contract is to link the Unix systems of the DCS research groups via the Post Office's forthcoming PSS Network to facilitate more direct cooperation and communication and a greater sharing of resources.

9.4 DCS EQUIPMENT POOL

INTRODUCTION

Equipment in the pool is loaned to investigators following authorisation from the Panel. The pool serves three functions:

(1) To provide improved communications facilities between the research groups in the Programme. Initially magnetic tape is the recognised communications medium, but the Post Office PSS Network is expected to play a key role in the future.

(2) To provide commercial UNIX licences.

(3) To provide a bank of LSI-11 computer systems. Many research workers have a requirement for a large number of such systems for a short period, typically towards the end of a project when checking that results found to hold for say a 3 processor system also hold for a 10 processor system. In such a case 7 LSI-11s from the pool would be loaned to the investigator. By sharing these LSI-11s several projects can benefit from a large resource which could not be provided to each of them.

The LSI-11 equipment pool is a good example of the benefits of a coordinated research programme in the realm of cost-effective usage of equipment.

In the last financial year (78/79) the Council approved the expenditure of 144K pounds by the DCS Panel to set up such an equipment pool to be managed by the Rutherford Laboratory. The Council has recently approved the DCS Panel's bid for 162K pounds of grant-related capital expenditure to extend the equipment pool.

EQUIPMENT DETAILS

Item	Purchased FY 78/79	To be purchased FY 79/80	Total
Newbury VDU	10	-	10
EMI tape decks	6	4	10
Diablo printers	7	4	11
Dicoll LSI-11s	10	5	15
RT-11 kits	5	-	5
UNIX licences	8	6	14
Modems	5 (rented)	-	5
'Teraks'	-	7	7

CURRENT PLACEMENT

The equipment has been allocated and shipped to the various DCS investigators as shown below.

DATE SENT	EQUIPMENT TYPE	SITE	USER
20-10-78	1620-DIABLO	SWINDON OFF.	MONNIOT J.
04-12-78	1620-DIABLO	RUTHERFORD	WITTY R.
20-10-78	1620-DIABLO	U.C.LONDON	KIRSTEIN P.
16-02-79	1620-DIABLO	IMPERIAL COLL.	SLOMAN M.
16-02-79	1620-DIABLO	KENT UNIV.	HANNA F.
20-10-78	1620-DIABLO	NEWCASTLE U.	LEE P.
30-11-78	1620-DIABLO	UMIST	ASPINALL D.
29-01-79	7002-NEWBURY(MP)	KEELE UNIV.	BENNETT K.
29-01-79	7002-NEWBURY(MP)	EAST ANGLIA	SLEEP R.
18-01-79	7002-NEWBURY(MP)	RUTHERFORD	WITTY R.
18-01-79	7002-NEWBURY(MP)	RUTHERFORD	WITTY R.
18-01-79	7002-NEWBURY(MP)	RUTHERFORD	WITTY R.
16-02-79	7002-NEWBURY(MP)	LOUGHBOROUGH	EVANS D.
16-02-79	7002-NEWBURY(MP)	NEWCASTLE	LEE P.
16-02-79	7002-NEWBURY(MP)	NEWCASTLE	LEE P.
21-02-79	7002-NEWBURY(MP)	RUTHERFORD	WITTY R.
06-04-79	7002-NEWBURY(MP)	YORK UNIV.	PYLE I.
24-04-79	EMI SE8000 M\T DECK	U.C.LONDON	KIRSTEIN P.
24-04-79	EMI SE8000 M\T DECK	NEWCASTLE U.	LEE P.
24-04-79	EMI SE8000 M\T DECK	KENT UNIV.	HANNA F.
24-04-79	EMI SE8000 M\T DECK	WARWICK UNIV.	WHITBY-STREVENS C.
	EMI SE8000 M\T DECK	MANCHESTER U.	GURD J.
24-04-79	EMI SE8000 M\T DECK	RUTHERFORD	WITTY R.
24-04-79	DICOLL LSI-11	U.C.LONDON	KIRSTEIN P.
24-04-79	DICOLL LSI-11	U.C.LONDON	KIRSTEIN P.
24-04-79	DICOLL LSI-11	EAST ANGLIA	DOWSING R.
24-04-79	DICOLL LSI-11	UMIST	ASPINALL D.
24-04-79	DICOLL LSI-11	Q.M.COLLEGE	COULOURIS G.
11-04-79	DICOLL LSI-11	OXFORD UNIV.	HOARE C.
24-04-79	DICOLL LSI-11	SUSSEX	HUNTER J.
	DICOLL LSI-11	MANCH. UNIV.	GURD J.
24-04-79	DICOLL LSI-11	RUTHERFORD	WITTY R.
14-05-79	DICOLL LSI-11	Q.UNI.BELFAST	BUSTARD D.
00-00-79	DATEL 13 MODEM	KENT	HANNA K.
00-00-79	DATEL 13 MODEM	EAST ANGLIA	SLEEP R.
00-00-79	DATEL 13 MODEM	RUTHERFORD	BLANSHARD P.
00-00-79	DATEL 13 MODEM	KEELE	BENNETT K.
00-00-79	DATEL 13 MODEM	LOUGHBORO UN	EVANS D.
01-05-79	RT-11 KIT	SUSSEX	HUNTER J.
01-05-79	RT-11 KIT	BELFAST	BUSTARD D.
01-05-79	RT-11 KIT	MANCHESTER	GURD J.
01-05-79	RT-11 KIT	EAST ANGLIA	DOWSING R.
01-05-79	RT-11 KIT	UMIST	ASPINALL D.
	UNIX LICENCE	Q.M.COLLEGE	COULOURIS G.
	UNIX LICENCE	YORK UNIV.	PYLE I.
	UNIX LICENCE	NEWCASTLE UNIV.	RANDELL B.
	UNIX LICENCE	LOUGHBORO UNIV.	EVANS D.
	UNIX LICENCE	RUTHERFORD	WITTY R.
	UNIX LICENCE	WESTFIELD COLL.	OSMON P.
	UNIX LICENCE	U C LONDON	KIRSTEIN P.
	UNIX LICENCE	UMIST	ASPINALL D.

9.5 MEETINGS

INTRODUCTION

The DCS Panel sponsors a series of meetings to promote cooperation between individual DCS funded researchers and communication between DCS funded projects and industry, Government Establishments and other research groups.

The DCS Panel sponsors three types of meeting, Special Interest Groups (SIGS), Workshops and Colloquia.

SPECIAL INTEREST GROUPS

These are small informal one-day meetings of up to 30 people drawn from DCS projects plus a few invited guests. The purpose of SIGS is to provide a vehicle for lively, technical discussions of state-of-the-art problems and current research ideas.

WORKSHOPS

Workshops are one or two-day meetings for up to 100 people drawn primarily from the DCS Programme but containing a significant (say 30%) number of outside participants. Workshops are aimed at presenting work on current DCS projects to a more general audience i.e. those researchers in the DCS Programme not directly studying the topic and other academic and industrial interested parties.

COLLOQUIA

No DCS Colloquia have yet taken place. A Colloquium is intended to be a meeting of up to 250 people at which some major achievements of the DCS Programme are presented to the world at large.

Two IEE sponsored colloquia have been jointly organised by the DCS and the IEE.

MAJOR MEETINGS TO DATE

Date	Subject	Host	Type
May 78	Real Time	York	SIG
May 78	Ironman	York	Workshop *
June 78	Dataflow	Newcastle	Workshop
Aug 78	Dataflow	RL	SIG
Sept 78	Languages	Warwick	Workshop *
Oct 78	'Jensen'	Warwick	SIG
Jan 79	'Dijkstra'	Oxford	Workshop
Jan 79	Academic DCS	IEE	Colloquium *
Apr 79	Networks	UCL	SIG
May 79	Industrial DCS	IEE	Colloquium *
July 79	Ada	York	Workshop
July 79	'Bowles'	Oxford	Workshop

PROCEEDINGS

SIGS do not usually produce proceedings. Workshops usually produce proceedings in the form of a university departmental report. These are circulated to participants and the DCS investigators via the "mailshot". Those workshops marked * above have associated proceedings.

9.6 DCS MAILSHOT

A monthly 'mailshot' is sent to all investigators and research assistants in the Programme. A typical mailshot will include such items as trip reports, drafts of papers for comment, vacancy notices.

The mailshot provides an informal mechanism by which information can be disseminated within the DCS Programme and ideas can be debated in a critical forum. The mailshot is a valuable adjunct to the series of Meetings organised within the Programme.

9.7 ADDRESSES

Coordination Team

Coordinators

R W Witty
Academic Coordinator, DCSP
Computing Division
Rutherford Laboratory
Chilton
Didcot
Oxon
OX11 0QX
0235-21900 Ext 6218

J Tucker
Industrial Coordinator, DCSP
Logica Ltd
64 Newman Street
London
W1A 4SE
01-637 9111 Ext 467

Technical Secretary

Dr D A Duce
Technical Secretary, DCSP
Computing Division
Rutherford Laboratory
Chilton
Didcot
Oxon
OX11 0QX
0235-21900 Ext 511

DCS Panel Secretary

J Monniot (previously Dr E J Wharton)
Secretary
Distributed Computing Systems Panel
Computing Science Committee
Science Research Council
P O Box 18
Swindon
Wilts
SN1 5BW
0793-26222 Ext 2401

Grant Holders

Prof D Aspinall
Dept of Computation
University of Manchester Institute of Science and Technology
P O Box 88
Manchester
M60 1QD
061-236-3311

Dr K D Baker
School of Enginerring and Applied Sciences
University of Sussex
Falmer
Brighton
BN1 9QT
0273-606755 Ext 128

Dr K H Bennett
Dept of Computer Science
University of Keele
Keele
Staffs
ST5 5BG
0782-621111 Ext 338

S E Binns
Computing Laboratory
The University
Canterbury
Kent
CT2 7NT
0227-66822 Ext 636

Prof P J Brown
Computing Laboratory
The University
Canterbury
Kent
CT2 7NT
0227-66822

D Coleman
Dept of Computation
University of Manchester Institute of Science and Technology
P O Box 88
Manchester
M60 1QD
061-236-3311 Ext 2341

Prof G F Coulouris
Dept of Computer Science and Statistics
Queen Mary College
Mile End Road
London
E1 4NS
01-980-4811 Ext 751

Dr E Dagless
Dept of Electrical Engineering
UMIST
P O Box 88
Manchester
M60 1QD
061-236-3311

Dr R D Dowsing
School of Computing Studies
The University of East Anglia
Norwich
NR4 7TJ
0603-56161 Ext 204

Prof D J Evans
Dept of Computer Studies
University of Technology
Loughborough
Leicestershire
LE11 3TU
0509-63171 Ext 297

Dr P W Grant
Dept of Computer Science
University College of Swansea
Singleton Park
Swansea
SA2 8PP
0793-25678

Prof R L Grimsdale
School of Eng. and Applied Sciences
University of Sussex
Falmer
Brighton
BN1 9QT
0273-606755 Ext 74

Dr J R Gurd
Dept of Computer Science
University of Manchester
Oxford Rd
Manchester
M13 9PL
061-273-7121 Ext 5553

Dr F Halsall
School of Eng. and Applied Sciences
University of Sussex
Falmer
Brighton
BN1 9QT
0273-606755

Dr F K Hanna
The Electronics Laboratory
University of Kent at Canterbury
Canterbury
Kent
CT2 7NT
0227-66822 Ext 396

S E Hersom
Numerical Optimisation Centre
The Hatfield Polytechnic
19 St Alban`s Road
Hatfield
Herts
070-72-66298

Prof C A R Hoare
Oxford University Computing Laboratory
Programming Research Group
45 Banbury Rd
Oxford
OX2 6PE
0865-58086

Mrs J W Hughes
Dept of Computation
UMIST
Sackville Street
Manchester
M60 1QD
061-236-3311

Dr J R W Hunter
School of Eng and Applied Sciences
University of Sussex
Falmer
Brighton
BN1 9QT
0273-606755 Ext 133

Prof P T Kirstein
Dept of Statistics and Computer Science
University College
Gower St
London
WC1E 6BT
01-387-7050 Ext 815

Dr P E Lauer
Computing Laboratory
University of Newcastle upon Tyne
Claremont Tower
Claremont Rd
Newcastle upon Tyne
NE1 7RU
0632-29233

M D May
(Formerly University of Warwick)
INMOS Ltd
Whitefriars
Lewins Mead
Bristol
0272-290861

A J R G Milner
Dept of Computer Science
James Clerk Maxwell Building
The King's Buildings
Mayfield Rd
Edinburgh
EH9 3JZ
031-667-1081 Ext 2753

Dr I Mitrani
Computing Laboratory
University of Newcastle upon Tyne
Claremont Tower
Claremont Rd
Newcastle upon Tyne
NE1 7RU
0632-29233 Ext 271

Dr R M Needham
Computer Laboratory
University of Cambridge
Corn Exchange St
Combridge
CB2 3QG
0223-52435

Dr I Newman
Dept of Computer Studies
University of Technology
Loughborough
Leicestershire
LE11 3TU
0509-63171

Dr P E Osmon
Westfield College
University of London
Kidderpore Av
Hampstead
London
NW3 7ST
01-435-7141 Ext 521

I Page
Dept of Computer Science and Statistics
Queen Mary College
Mile End Road
London
E1 4NS
01-980-4811 Ext 751

Dr Y Paker
School of Engineering & Science
Polytechnic of Central London
115 New Cavendish Street
London
W1
01-486-5811 Ext 6301

Dr G D Plotkin
Dept of Computer Science
James Clark Maxwell Building
The King's Buildings
Mayfield Road
Edinburgh
EH8 3JZ
031-667-1081 Ext 2775

Prof I C Pyle
Dept of Computer Science
University of York
Heslington
York
YO1 5DD
0904-59861 Ext 485

Prof B Randell
Computing Laboratory
University of Newcastle upon Tyne
Claremont Tower
Claremont Rd
Newcastle upon Tyne
NE1 7RU
0632-29233 Ext 235

N H Shelness
Dept of Computer Science
James Clerk Maxwell Building
The King's Buildings
Mayfield Rd
Edinburgh
EH9 3JZ
031-667-1081 Ext 2752

Dr M R Sleep
School of Computing Studies
University of East Anglia
University Village
Norwich
NO4 8BC
0603-56161 Ext 205

Dr A Sloman
School of Social Sciences
University of Sussex
Falmer
Brighton
BN1 9QN
0273-606755 Ext 1012 or 1002

Dr M Sloman
Dept of Computing and Control
Imperial College
London
SW7 2BZ
01-589-5111

Dr P C Treleaven
Computing Laboratory
University of Newcastle upon Tyne
Claremont Tower
Claremont Rd
Newcastle upon Tyne
NE1 7RU
0632-29233

Dr W W Wadge
Computer Science Department
University of Warwick
Coventry
CV4 7AD
0203-24011 Ext 2647

Dr I C Wand
Dept of Computer Science
University of York
Heslington
York
YO1 5DD
0904-59861 Ext 5570

Dr I Watson
Dept of Computer Science
University of Manchester
Oxford Road
Manchester
M13 9PL
061-273-7121

Dr J Welsh
Department of Computer Science
Queen`s University of Belfast
Belfast
BT7 1NN
0232-45133 Ext 3221

Dr C Whitby-Strevens
(formerly of University of Warwick)
INMOS Ltd
Whitefriars
Lewins Mead
Bristol
0272-290861

Prof M V Wilkes
Computer Laboratory
University of Cambridge
Corn Exchange St
Cambridge
CB2 3QG
0223-52435 Ext 256

9.8 DISTRIBUTED COMPUTING SYSTEMS PANEL MEMBERSHIP

PANEL MEMBERS 1977-1978

I M Barron	Chairman
Prof D Aspinall	
A C Conway	Department of Industry
Dr F K Hanna	
D Hebditch	
Prof C A R Hoare	
C J Hughes	
J McNeil	
Prof I C Pyle	
Prof B Randell	
M G Shortland	Control Engineering Committee
Dr R Barrett	Electrical and Systems Engineering Committee
R Landeryou	Secretary, CSERB

PANEL MEMBERS 1978-79

Prof I C Pyle	Chairman
Prof D Aspinall	
R Brown	Department of Industry
Dr F K Hanna	Control Engineering Committee
D Hebditch	
Prof C A R Hoare	
C J Hughes	
D Pearce	
Prof M Rogers	
G Scarrott	
Dr R Barrett	Electrical and Systems Engineering Committee

PANEL MEMBERS 1979-80

Prof I C Pyle	Chairman
Prof D Aspinall	
R Brown	Department of Industry
S L H Clark	
Dr F K Hanna	
Prof C A R Hoare	
C J Hughes	
Dr R Needham	
D Pearce	
Prof M Rogers	
Dr R Barrett	