CONTENTS

An Environment for Distributed Systems Development

CONTENTS

An Environment for Distributed Systems Development

Introduction

This proposal is directed towards the construction of an integrated environment for the design and implementation of large distributed, concurrent or embedded systems. The environment will include tools to support the entire design cycle from system specification through to hardware implementation. The interface seen by the system designer will be uniform at all levels. That is, similar actions performed by the designer will have similar effects regardless of context; graphical and textual feedback will be presented in common formats and based on a uniform model. In particular, functional and behavioural specifications will be communicated in a common target language at all levels.

The design and implementation of large software systems, particularly distributed systems, is an arduous task. Systems which are geographically distributed, or which involve multiple processes, perhaps executed in parallel on different processors, are inherently more difficult to develop because of their non-deterministic behaviour. Few tools are available to assist the designer of such systems. Those that do exist are limited in their scope, addressing only isolated parts of the design and development cycle. The designer's problems are worse if the target system is ultimately to be embedded in hardware different from that on which it was developed. The operation of the proposed environment is illustrated by the following scenario.

A 'driver' for an operating system is specified and initial code for it is written. The code is proven correct with respect to the specification. It is then prototyped within an environment which provides tools for inspecting the executing code, for statistically measuring its performance, for performing timing analyses and for simulating real world devices with which the driver will be connected. The code is then inserted into a real operating system. While much less friendly than the prototyping environment, some tools are available for inspecting it and monitoring its execution. Eventually the driver is found to be crucial to the performance of the operating system and migrates to a processor of its own which communicates with the rest of the operating system via special links or shared memory. Finally it is enshrined in VLSI and this circuitry is integrated into the operating system.

Many variations on this scenario can be envisaged. The target for the code might be a geographically distributed remote sensing problem with many processors communicating over slow communication links, or the target might be a tightly coupled array multiprocessor. The designer might wish to proceed directly to silicon implementation from the prototyping stage. As bugs surface it may be necessary to return to earlier stages to rework the code.

In the current state of the art, each stage of this process requires a different notation or language. Notations exist for specification; for prototyping or simulation; for code which can be run in an operating system; for code to run on tightly coupled multiprocessors; for code to run on geographically distributed systems; and for the design of VLSI chips. Even when the target code is written and the system is running there are barriers to monitoring and debugging. For example a number of processes may be running in parallel but the tools for examining the messages between them will be completely different from the tools used to examine bugs within one process. Ultimately this proposal aims at a single integrated system which removes these barriers to code migration and testing.

A major thrust of the proposal is to build on and focus the momentum generated by the Jade distributed prototyping environment, developed at the University of Calgary under an earlier strategic grant [Unger *et. al.*, 1982a] and now nearing completion. A full report on the current status of the Jade project is appended to this proposal. In brief, the achievements most relevant to this proposal are:

- an interprocess communication protocol, Jipc, providing interaction between processes written in Ada, Simula, C, Lisp and Prolog
- a multiprocessing kernel for single user workstations supporting Jipc, a window manager and a virtual terminal protocol to Unix
- a device-independent graphics system with associated graphical editor supporting hierarchical graphical structures and automatic modelling
- a extensible, distributed Jipc monitoring system with central control providing multiple user consoles for textual and graphical display, deadlock detection and system control and statistical display
- a simulation tool for distributed system prototyping permitting parametric control over the simulated target architecture
- the largest ever specification and verification of a practical VLSI design, and participation in the design and implementation of two of Canada's largest chips
- a design and prototype of a VLSI design capture language.

Many of the facilities of Jade can be used immediately to help construct the proposed environment; indeed it will be built on top of the already existing Jade environment. Other tools, such as those for monitoring and prototyping, will, after some enhancement and redesign of their interfaces, form part of the target environment. In addition, a number of other research projects supported independently and already underway within the Department of Computer Science will be drawn on to supply tools, techniques and experience to support the proposed research. These include research on specification and proofs, user interface prototyping and graphical animation tools, and a substantial VLSI research and development effort.

The format of this document is as follows. We begin by briefly stating our objectives; firstly in terms of the impact this project will have on computing in Calgary and in Canada and secondly in terms of the research results we aim to achieve. We then briefly review previous work on vertically integrated environments and outline the overall methodology to be used in meeting these objectives. Concluding sections give biographical details on the individuals who will be involved in this research effort, describe our planned group organisation, and discuss the impact of the project on the outside world. Three appendices deal with: (A) the system design components we plan to research; (B) research on tools for integrating these components into a uniform environment; and (C) the applications we will use to test and demonstrate the target environment. Within each appendix, and for each component (design tool, environment construction tool, or application), we review the current literature relating to the component, detail our plans for constructing it and describe the status of our research on that component. Included are details of which research personnel will be working on the component and what their experience is in relation to it. References are collected in a fourth appendix.

Objectives

The strategic objectives of the project proposed are:

A. To make a significant contribution to the development of *Fifth Generation* computing skills and technology within the Canadian professional community.
B. To further develop and apply the distributed software development environment (Jade).
C. To integrate software and hardware development techniques within a framework of specification and proof.
D. To ensure that the research reaches a stage suitable for commercial exploitation.

The enabling technologies for the above objectives to be realised include software engineering, distributed systems, human-computer interfaces, artificial intelligence and VLSI design systems. This proposal addresses problems in all of these areas, which reflect strengths within our department and a continuation of work under a previously highly successful strategic grant proposal (Jade). It should be noted that the Jade system is currently being tested at beta test sites in industry.

Our research objectives are:

A. Target environment components: to solve fundamental problems which form the bottleneck in the development of distributed and parallel systems, including the implementation of such systems in both software and VLSI. The proposed research, based on work already in progress, will address the following areas:

- specification, prototyping, simulation, monitoring and implementation of distributed systems
- specification, verification, and implementation of VLSI systems.

B. Environment support tools: to design and implement a software and silicon design environment that integrates all the tools needed by a developer (including those listed above) under a uniform user interface. The environment will draw upon the already existing Jade environment and associated projects, based on new research results obtained. The research problems to be tackled include:

- support of specification and proof within an interactive programming environment
- prototyping and management of user interfaces
- creation, management, and use of on-line documentation.

C. Structured design and implementation of a number of research-level applications.

Our plans for achieving each of the research objectives are discussed in corresponding appendices A, B and C. The main body of the proposal treats the integration of the various components into a unified environment.

Review of Literature

System development environments is an increasingly active research area and much work is now being published. Two recent collections [Henderson, 1984; and Barstow, 1984] provide a wealth of information about seminal work in this field. However, a large proportion relates to specific tools for program construction, such as structure editors [e.g. Neal, 1980; Teitelbaum & Reps, 1984; Fischer *et. al.*, 1984; Garlan & Miller, 1984], or code control [e.g. Leblang & Chase, 1984; Tichy, 1982; Estublier *et. al.*, 1984]. Many integrated programming environments now exist, for example Smalltalk [Goldberg, 1984], Interlisp [Teitelman & Masinter, 1984] and PECAN [Reiss, 1984]; but few address the particular problems of creating distributed or embedded systems. We discuss in detail some of the systems that relate most closely to our own work.

Helmhold & Luckham [1985] describe an environment for debugging Ada tasking programs. A general transformation tool is used to generate a modified Ada program from another that collects and presents Ada task interaction monitoring information. The transformation tool is general in that many possible software development tools can be created, the task monitoring tool being one example. This work is closely related to ours in that process interactions are monitored for multiple purposes. Their purposes are primarily to detect deadlock and task synchronisation errors. The former can be done in general, the latter requires modifying the monitor for specific applications. The Jade environment, however, is not limited to one language, and focusses on process interactions in a way which can be extended arbitrarily to different, but complementary, views into a systems operation that also enables control over non-determinism. Deadlock detection is straightforward in both schemes, but synchronisation errors can be pursued on a number of levels within the Jade monitoring scheme. On the higher levels, process interactions can be verified against a protocol specification, performance can be examined, and user defined graphical representations of the application system can be animated.

Cuenet [Radhakrishnan & Grossner, 1985] is a loosely coupled matrix of micros that can be interconnected by the programmer in a variety of topologies. Alternative problem decompositions can be evaluated by implementing and testing them. However, we use simulation techniques to represent parts of the target

that do not exist in the development environment. Not only can a much wider variety of target systems and configurations be explored, but particular decompositions can be studied without implementing all software components. Only the key modules need be implemented initially and parts that are not likely to affect performance strongly can also be modeled.

Eden [Almes *et. al.*, 1985] is directed at goals similar to ours, however, the focus is on transparently mobile objects in a distributed environment which is both the development environment and target. The Eden approach also is based on a single language, an extended Concurrent Euclid. Eden objects are quite general consisting of multiple processes in a shared address space. Jipc processes are smaller, finer grained objects on which higher level protocols and objects can be implemented and explored, as in the implementation of Virtual Time [Jefferson & Sowizral, 1982; 1983] by building Tipc upon Jipc [Cleary *et. al.*, 1985]. The Spice project [Fahlman & Harbison, 1984] is similar to Eden in that it is directed at creating a distributed programming environment with workstations. Our focus is, however, is on tools for specifying, prototyping and evaluating distributed programs.

The Waterloo Port [Malcolm & Dyment, 1983] network operating system's goals are also similar to ours, particularly the focus on an interactive graphical user interface to distributed systems. The Port project attempts to provide a graphical user interface that is integrated with a single implementation language — an extended version of C. Our objectives differ from Port's in an attempt to use a single implementation language for the specification and prototyping phases, as well as for the implementation and testing phases. This is related to the approach taken in Mayflower [e.g. Craft, 1983].

Our goals are also similar to those of Shoshin [Tokuda & Manning, 1983] in that Jade supports prototyping and evaluation experiments for distributed programs. The Shoshin interprocess communication model is designed specifically to enable experimentation with a wide variety of interprocess protocols. Our approach is to use a simple, fixed, low-level synchronous protocol upon which higher levels can be built (e.g. Tipc, above) and to focus on prototyping and performance evaluation at levels above the base operating system message passing level, e.g. the distributed transactions of TABS [Spector *et. al.*, 1984]. In contrast with Shoshin, our approach also makes extensive use of simulation and hierarchical monitoring for prototyping distributed systems with objectives very similar to Lauer [1983]. Jade/2 is also currently restricted to workstations that have no secondary storage, (c.f. Lazowska, [1984]). The proposed environment, however, will support workstations having Unix managed secondary storage.

Methodology

Our overall strategy will be to divide our efforts into five overlapping phases: (1) design and implementation of the individual components of the target environment; (2) design of the target environment itself and development of tools to support implementation of the environment; (3) construction and integration of the target environment; (4) testing of the target environment by using it to design and build several substantial applications; (5) documentation of the target environment, dissemination of results, transfer of technology to industry and critical review of our achievements.

Phases 1, 2 and 4, which correspond to the three research objectives A, B and C, will operate concurrently from the beginning of the project, with the principal focus of our attention being given to the first two. We anticipate being able to begin phase 3 (integration) at about the end of the first year, although not all the components comprising phases 1 and 2 need be ready immediately. Phase 4 (testing) will then receive increased attention, and we expect to have at least one application completed and demonstrable by the end of year 2. The final six months of the project will be principally devoted to phase 5 (documentation, dissemination and review), although this activity will be pursued throughout the life of the project.

Phase 1 consists of work in two major problem areas: formal specification of systems and interactive proof of the correctness of refinements in both hardware and software; and interactive simulation, prototyping, and instrumentation of distributed and parallel systems. The result of our work in these areas will be a number of component subsystems, for example: an interactive tool for specifying and refining distributed

systems; a tool for defining the environment within which a distributed system will operate and simulating its behaviour in that environment; a graphical monitor for interactions between processes.

Research in this phase will concentrate on two areas: VLSI specification, verification and implementation; and system specification and prototyping. Work will build as much as possible on tools already developed within Jade and other research projects. For efficiency and convenience, the languages used for implementation of the components are likely to differ; the components will be linked together using Jipc. However, each component will eventually interface to the outside world via a common target language. All personnel working on phase 1 will share responsibility for developing a specification of this language and for ensuring that the interface to each component can later be tailored so as to conform to the target language. Full details of the plans, current status and personnel for each of the areas are given in appendix A.

During *phase 2* we will be designing the target environment itself in preparation for phase 3. Phase 2 also involves the development of three major tools which will be used in building the target environment: an interactive programming environment which supports specification and proof; a system for prototyping user interfaces for the target environment; and documentation presentation and maintenance facilities supporting both interactive perusal and generation of high quality printed copy. More information about these tools is given in appendix B.

A good user interface and (especially) the possibility of constraining the design of user interfaces to meet certain basic standards, whilst also facilitating the design process, are fundamentally important goals of the environment, as we envisage it. The environment itself must have a good interface, and tools must be created to meet the second goal. Estimates suggest 40-80% of applications code is concerned with implementation of the user interface. In a programming environment, it is the programmer who is the user, but the environment is then used to create applications for non-programmers. To obtain consistency among the interfaces to the various tools that constitute the environment, we believe it necessary to specify rigorous constraints on how the tools will be presented to the user. Our research plan is to produce initial functional definitions for the interfaces to each of the components of the target system, based on analyses of application scenarios, and elaborate these definitions into detailed descriptions of user interactions, incorporating a specification of the target language. The methodology for refining these descriptions relies heavily on user interface prototyping and experimentation, and we anticipate going through several cycles of prototyping and redefinition before finally specifying the target environment user interface.

Work on phase 2 will be coordinated by a single group. This group will be responsible for elaborating initial functional definitions of the components of the target environment; however, these initial definitions, and criticism of the later elaborations, will be supplied by the research groups of phase 1.

In *phase 3* the user interface prototyping system will be refined so as to enforce the decisions made in phase 2 about the desired user interface. This system will be used to reimplement interfaces to each of the tools in the target environment, including both the newly developed components and support utilities. Support utilities will, where possible, be imported from elsewhere. The user interface component will be designed to communicate with each of the tools in the environment using the Jipc message protocol. Phase 3 is expected to begin at about the end of the first year, even if some of the phase 1 components are not complete. We will proceed by incorporating tools incrementally, using stubs to simulate tools that have not yet been added.

In *phase 4*, we will test the environment by using it to design and build several substantial applications. We are currently considering three applications for this phase: design and prototyping of software for the mesh computer, certain parts of the target environment itself, and a distributed graphics protocol. These applications are detailed in appendix C. Use of the environment to design, formally specify, and construct the applications will begin as soon as a partial environment becomes available. However, preliminary work on analysis, informal specification and design will commence at the inception of the project. In fact

considerable progress is already evident, and, in particular, a prototype of the Mesh computer, designed by hand, is already available. We expect to be able to demonstrate a prototype, designed using the new environment, by the end of the second year.

We regard *phase 5*, documentation and dissemination of results, as crucial to the success of the project; phase 5 will be be continually active throughout the life of the project and, in addition, the final six months of the project will be principally devoted to this phase. We plan to produce both a user manual and a reference manual as part of our effort. The user manual will arise naturally out of the interface design process in phase 2 and we expect to have a substantially complete version available before the environment itself.

Phase 5 will also include packaging the environment so that it can be distributed to test sites. This in itself is a non-trivial task for a large system and will be an important consideration during each of the phases of the project. Formal control over the system will be exercised using a source code control system from the beginning of the project, and supervision of this phase will be one of the principal concerns of the project steering committee.

Further details of our plans for disseminating research results can be found in the section *Plans for Promoting Project* below.

Research Personnel

We give here brief cameos for each of the thirteen investigators who will be working on the proposed project. In the appendices, we indicate which faculty members will be working on which sub-projects, and give relevant biographical details for the graduate students and research associates involved.

Ian Witten is to be the Principal Investigator for the proposed project. His research interest in man-machine interaction has developed over a number of years (10 in the UK, 5 at Calgary) and is recorded in some 75 refereed publications and a number of books. He is one of the key Faculty in the Jade project, although his day-to-day involvement has been curtailed by the administrative obligations of his position as Department Head for the past three years. Despite this he has remained closely in touch with the Jade research. Witten's recent research has been in the areas of casual-user human interfaces, documentation and dynamic documentation, adaptive user modeling, expert help systems, and natural language interaction. Graduate students are completing (or have just completed) theses in all these areas under his supervision. He has also (in conjunction with John Cleary) developed new methods of coding and data compression, both for text and pictures. He has gained considerable experience with logic programming, and the use of object-oriented paradigms within the logic programming framework.

Graham Birtwistle's academic career has included a term as a researcher, and later as a project leader in systems programming at the Norwegian Computing Center in Oslo, between 1968 and 1973. The Simula compiler produced by that group is still on the market today. Noted for his research activities in the areas of programming languages, discrete event simulation, and VLSI design tools, Dr. Birtwistle has worked within industry, including consulting with such firms as Burroughs, Hoogovens, Scalant, and Transport Canada. He has published a total of 40 refereed academic papers and four books.

John Cleary has had five years experience in industry including senior technical positions on large (20 man year) projects. After completing his Ph.D. in 1980 he took up a faculty position at Calgary in 1982. He has been actively involved in a number of areas of the Jade project; in particular, the development of Jipe and graphical interfaces for Prolog and a distributed implementation of Concurrent Prolog. Other related work with graduate students has been the construction of a graphical debugger for Prolog and proof rules for showing the completeness of Prolog programs. During the same period he has published in the areas of data compression, applications of hash coding, and graphics. He has also directed the construction of a prototype multiprocessor Mesh computer. Current research interests include discrete event simulation using "Time Warp", a logic programming interpretation of interval arithmetic, expert systems for

interpreting geophysical seismic data, and deadlock free distributed communication protocols.

Brian Gaines has just been appointed to the Killam Research Chair at the University of Calgary, and will become a full-time member of our Department in July 1985. Dr Gaines has extensive research experience, both university and industry, in areas including software development, man-machine interaction, typesetting, expert systems, system theory, and computer architecture. He has published broadly in all these fields. He has been Professor and Head of the Department of Electrical Engineering Science, University of Essex, UK; Professor of Computer Science at York University; and has close associations with the Department of Industrial Engineering at the University of Toronto. He has founded several high-technology companies (eg City Computer Systems, London, and Cadre Information Systems, Toronto) and held senior executive posts in others (e.g. Monotype International). He has very many industrial and research contacts throughout Canada, the US, Japan, and Europe. The project will benefit from his technical expertise in all areas of research, as well as his extensive experience of both software and research management, and university/industry technology transfer.

David Hill brings more than 20 years experience of research in human-computer interaction and artificial intelligence to the project. Commissioned as a pilot in the RAF, experienced as a research engineer in exploratory research, and then as a project leader with ITT, before taking up a faculty post at Calgary, he has seen the problems of human interaction with complex systems both as a user and as a researcher; he has an industrially oriented view as well as an academic view. He has also been Editor of the International Journal Of Man-Machine Studies since 1970, and has been involved in the assessment of research and allocation of research funds at a National level. His research has centered upon speech generation and understanding by machine, dialogue design, special input/output devices, computer graphics and VLSI design and application. Apart from a large NSERC individual operating grant, he holds a $30K Esso Resources three year grant to carry out research on ways of improving access to computers for disabled university students (the MAWD project). His graduate supervision includes 6 M.Sc. and 1 Ph.D. student in his research areas over the last 6 years, with 5 theses currently in progress. He has a total of 27 refereed publications to his credit, as well as numerous invited talks and reports. He is principle author of two patents.

Tom Keenan has a strong research interest in computer security and teleconferencing. This interest extends to local area networks and distributed computing since much of computer security is really communications security. He has a number of refereed papers in these areas, and will be used as a resource person in the project.

John Kendall joined the Department of Computer Science in 1984 after spending the preceding four years as Manager, Electronics and Instrumentation Department, NOVAL Technologies; a department involved in the development of microprocessor instrumentation and electronics for the oil and gas industry. He is Vice President and Chairman of the Electronic Industry Association of Alberta and a member of the Board Directors of the Alberta Microelectronics Centre. He has been active as a consultant to industry since 1965, in areas ranging from silicon nitride and amorphous semiconductor studies for the British Admiralty, to multiphase flowmeters for both the National and Alberta Research Councils. A recognised authority on VLSI, Dr. Kendall is the holder of two patents and has written three books and a total of 45 refereed academic papers.

Danny Levinson will be working in the Department of Computer Science under a University of Calgary Research Fellowship, awarded for the investigation of user interfaces and User Interface Prototyping Systems. He obtained his Ph.D. in Automatic Speech Recognition at Calgary in 1982, and has been working as Senior Research Associate on the Jade project since July 1983. He has been principally responsible for developing the Jade graphics system. This has led to his present interest in UIPSs. He is currently working on the implementation of a graphical interface generator.

Jon Rokne has been working extensively in the area of algorithms and implementations of algorithms in interval arithmetic. Presently he is working on the adaptation of logic programming techniques to

numerical analysis algorithms, in particular implementation in Prolog. A multivariate polynomial manipulator is being used as a test case. He has also worked extensively on resource industry related simulations.

Brian Unger is the Principal Investigator for the Jade project, currently supported by NSERC Strategic, Major Equipment, and Infrastructure awards. This has involved leading a collective effort to: define project goals, assemble an excellent research team, coordinate co-investigator research towards project goals, and administrate the development of the Jade/2 environment. He has also furthered the exchange of research on an international level as the general chairman and programme chairman of several conferences, and as an editorial board member of two journals. Unger's current research interests are in the monitoring, prototyping, simulation, and animation of distributed systems. Recently this work has focussed on developing techniques for distributed simulation and the application of logic programming to simulation. During the past six years he has published 10 journal papers and one book, and supervised 10 graduate students in these areas.

Vasu Vasudevan is presently completing his PhD in Computer Science at the University of Waterloo, and will return to his faculty position at Calgary in the summer of 1985. His research is in the area of network operating systems and highly relevant to the proposed project, and his experience of the Port and Shoshin systems at Waterloo will be of great benefit. He has published a number of papers on real-time and network operating systems.

Walter Vollmerhaus, who has had a long-term research interest in graph theory, has been associated with the Jade project for two years. He has also been instrumental in providing the VLSI research group with a sufficient background in theoretical areas, such as recursive function theory, fixpoint theory, and interval temporal logic, which have been essential for their work on hardware verification and proof. He will continue this role as theoretical resource person in the proposed project.

Brian Wyvill has been actively involved in computer graphics since 1971 and has 16 refereed publications in this area. He has been a faculty member for four years at Calgary and was one of the originators of the Jade graphics system. Wyvill's recent research has been in the area of graphics languages for 3D animation, image synthesis and the investigation of alternative strategies for distributed rendering algorithms. One graduate student is currently completing his thesis on particle graphics under his supervision.

In addition to the investigators listed here, there are other research faculty whom we expect to be joining the Department very shortly and who will also contribute significantly to central areas of the project.

Group Organisation

The project will be led by a small steering committee consisting of one senior representative from each of the three major research groups, and chaired by the Principal Investigator. The individuals will be: Birtwistle (VLSI and specifications), Hill (user interfaces), Unger (system prototyping), and Witten (Principal Investigator). Our experience in the Jade project suggests that this smaller steering committee will be more effective.

During phase 1, Birtwistle and Unger will coordinate tool development for their respective areas; Hill will coordinate phase 2; and phases 3, 4 and 5 will be coordinated by individuals defined by the steering committee. Evidence of our ability to collaborate closely together is manifest in the success of the Jade project.

Ian Witten has been selected as Principal Investigator for the present proposal for the following reasons:

- he is a senior researcher with broad interests throughout the project area
- his major research interest in documentation systems and man-machine interaction emphasises the focus on the "integrated system" approach which is so vital for success of the project
- his term as Department Head ends on 1985-06-31, and he is an experienced administrator
- we felt that this role should be rotated among the senior investigators.

Against this, Witten will have one year's sabbatical from 1985-07-01, and will be out of the country for much of this time. Brian Unger will coordinate the project, chairing the steering committee, until 1986-06-31. This is particularly appropriate since it will ensure a smooth transfer into the new project from the existing Jade organisation (of which Unger is Principal Investigator).

Relevance and Socio-Economic Impact

Canadian industry needs to take full advantage of advances in information technology in order to remain, or become, competitive. The small size of the home market relative to the USA and the geographic dispersion of industry in Canada makes it difficult for information and experience to be generated and made widely available. In particular, potential users of high-technology systems face the dual disadvantages of having few national suppliers at the forefront of the technology, and of having few local innovators in the use of the technology.

The extension of the Jade project proposed is aimed at alleviating these problems. First, by making tools and expertise in system integration widely available to the suppliers of high-technology systems in Canada. Second, by designing these tools specifically against the system integration needs of major end users and giving them the opportunity to becoming self-supporting in innovation of applications.

The major socio-economic benefits of the project will be:

- encouraging the development of Canadian information technology industries supplying integrated systems targeted at specific markets.
- supporting potential users of information technology in Canada in the development of integrated systems specific to their needs
- demonstrating that "fifth generation" projects involving the integration of diverse technologies are both feasible, and have practical consequences, for Canadian universities and industry.

Our strategic objectives meet Canada's needs, as expressed by many agencies, including the Natural Sciences and Engineering Research Council in their last five year plan. Namely, the training of highly skilled manpower and development of new technology within areas of economic importance; the transfer of the technology and skills to industry; and the performance of excellent research, by highly skilled researchers, with adequate dissemination of results. The group has a proven track record of ability to meet the goals of an ambitious, practically oriented research project, aimed at strategic goals.

Anticipated User Community

At present we see the primary users of the environment being the research and advanced development people withing the following areas:

- university faculty, graduate students, undergraduate students and research staff. These would principally be using application systems built using the environment.
- the energy resource industry. We have several contacts developing already, primarily in the human interface and distributed system areas.

- the communications industry. Particularly in the intra-office area, the interactive graphics for education and training areas, and the tele-conferencing areas.
- the information management industries. This includes the information management aspects of medicine and law. We have had serious enquiries from a law organisation that is embarking on the development of distributed software for law firms.

Plans for Promoting Project

We intend to involve industry in this project from the outset. A number of papers on Jade, its tools and techniques, have been presented at national and international conferences and this will continue. In addition, the completion of the first phase of the Jade system development makes it possible to set up workshops from the start of this project aimed at industrial organizations concerned to use VLSI and distributed systems. We will use these workshops not only to present techniques arising out of the Jade project but also to ensure that future developments are targeted against industrial requirements.

Details of workshops already held and to be held. As an indication of the level of activity in this area during the current project (1982-1985), we have held:

- two external reviews of the project (June 1983 and December 1983)
- two internal workshops (March 3-4, 1984 and April 9-10, 1984)
- a combined conference and external review (September 21-23, 1984)
- two one-day presentations to local industries (April 13 and June 15, 1984)

In addition, we have given regular demonstrations to representatives from various industries at the approximate rate of one per week in the last two years of the project. Demonstrations have been strictly curtailed by the amount of research time we have been able to devote to them. We expect activity to continue at a significantly higher level due to the inclusion of a half-time industrial liason officer in the present application.

Our plan for disseminating the results of the project is strongly coloured by our experience with Jade, both positive and negative. We intend to organise one user workshop in the summer of each year of the project; the first will mainly showcase the accomplishments of the present Jade system. The later workshops will present new achievements as they become established in the environment. We have asked for half support for an industrial liason officer. In addition to being responsible for organising the workshops, the officer will coordinate our new "Industrial Affiliates Program" by initiating and maintaining contacts with potential users, coordinating production of presentation and demonstration materials, and supervising distribution of sources to beta test sites.

Appendix A. Target Environment Components

As described in the main text, research in phase 1 of the project is aimed at producing components for eventual incorporation into the target environment. This section gives details of the plans, current status and composition of the two major phase 1 working groups: VLSI specification, verification and implementation; and system specification and prototyping.

An environment sufficient to support the entire system development cycle requires many components besides those that we can expect to produce ourselves. Our phase 1 research is directed only at those areas in which we believe important deficiencies exist in presently available tools and where we believe we have the experience and capacity to make a significant contribution to the state of the art. We intend to supplement the environment by importing components whose development is outside the scope of this project. Each such component will have its interface redesigned, as described above, to conform to the overall specification of the target environment. Examples of components we plan to import are Bell Laboratories' Source Code Control System (SCCS) [Rochkind, 1975] and Berkeley's data base management system *Ingres* [Stonebraker *et. al.*, 1976]. Both are presently available at Calgary, and we have already gained some experience at writing new interfaces to the latter.

Specification, verification and implementation of VLSI systems

In order to construct large designs, we must be able to argue their correctness with proofs that do not explode in size as complexity increases. One way of doing this is to partition a design into sub-designs, or sub-components, and operate with a composition rule defining how the latter are connected together. As we detail a design, we partition each component in turn into an interconnected group of sub-components which we think are equivalent. Working from the top downwards, we are given the specification of the original component and we have to construct interconnected sub-components whose aggregate behaviour conforms to this specification. Given the specification of the sub-components and the composition rule, we are able to deduce the behaviour of an implementation. If this aggregate behaviour is equivalent to that of the original component, then we have a valid implementation.

We can use the same technique to deduce the behaviour of an aggregate of connected modules with known behaviours. This is very useful, because designers often work in first top-down mode and then bottom-up as they iterate towards a satisfactory design.

If the amount of formal reasoning required for this deduction is not to increase with the size of a component, then the specification of a component should not reflect its internal structure but only how it looks from the outside. The determination of the effect of a large design would become too complicated if we had to take the internal structure of sub-components into account. Two components with equal specifications may be very different internally, but we do not wish to use this information when we employ them to build larger components.

The technique can be used to prove that a design conforms to its specification working in bottom up or in top down style. But it does not guarantee that the specification is correct nor that a specific chip will be flawlessly fabricated. Simulation may be used to give confidence in the correctness of specifications.

Review of Literature. Gordon [1981; 1983] used denotational semantics to reason about clocks, feedbacks, instruction set implementation and buses at the register transfer level. Gordon's proof model does not handle very low level primitives. Buchanan [1982], Malachi and Owicki [1981] and Halpern *et. al.* [1983] have shown how to use temporal logic to give detailed reasons at fine grains and how to handle asynchronous circuits. Recently, Gordon has extended his LCF-LSM theorem prover to Higher Order

Logic (HOL) and has investigated the different styles of proof required above and below the register transfer level. HOL can bridge this gap. Barrow [1984] has rewritten Gordon's techniques in Prolog and added in automatic composition techniques.

Plans and methods. We have used Gordon's LCF-LSM theorem prover to show the correctness of a local area network box (we believe this to be the largest practical design yet verified). Our major current tasks in this area are the specification driven design of an SECD chip and the investigation of hierarchical simulation techniques. Further experience in chip verification and specification will include building a specification library and library assistant, studying the possibilities for parameterised specifications, and specification driven design. Contacts in this area include Gordon and Melham at Cambridge, Milne at Edinburgh, Arnborg at KTH Stockholm, and Barrow at SPAR.

Current status. We have established fruitful working relationships with Silicart and with the Universities of Cambridge, Edinburgh, Stockholm, and Montreal. We have placed students at BNR (16 months), SPAR (27 months), and Xerox PARC (3 months). We have extensive experience with the Electric system. Kroeker and Liu have made additions to the SPAR original which are now distributed as part of the standard package. Schediwy (April to August 1985) will be developing the ideas of his master's thesis into a full cell library for Electric. This too will be distributed by SPAR. Coates has already fabricated the main floor plan elements for his ISM chip. The ISM chip, 150,000 transistors, was designed with Electric and is expected to be fully prototyped this summer.

In addition to Electric, we have constructed our own LAP packages for NMOS and CMOS, completed a SHIFT design capture language prototype, and are currently interfacing SHIFT to a multitude of low level validation tools (extractors, routers, design rule checkers, CIF, plotters, SPICE, etc).

Finally, substantial progress towards a Prolog architecture has been achieved by Liu who has completed a software interpreter at the register transfer level. Joyce has completed a hardware design for the SECD machine and formally specified a register-transfer level implementation of that design. Bus traffic in the SECD machine has been studied by means of a register-transfer level simulation. Our development of Prolog and SECD machines is one aspect of a department wide interest in Fifth Generation computers and knowledge based systems and exploring their architectures via simulation.

Personnel. The VLSI group numbers 3 faculty (Birtwistle, Kendall and Vollmerhaus) and 8 graduate students. Coates will be responsible for the mesh computer chip, and Liu will be working on Prolog and Concurrent Prolog chips. Coates has 12 months industrial experience working with Al Davis at SPAR on a floor plan element of the latter's FAIM-1 chip. Liu has a master's in digital filter architectures, 3 months experience at SPAR, and has completed a silicon compiler for random logic targetted at Weinberger arrays. The system, which is an extension to Electric, includes a technology library, automatic design verification, and optimises layout using simulated annealing.

Specification, prototyping and instrumentation of distributed systems

In the following subsections we review the literature and discuss our plans and the current status of our research in the areas of: specification, refinement and proof of software; prototyping and simulation; distributed simulation; and monitoring of distributed systems. A final subsection gives details about the personnel working in this research group.

Specification, refinement and proof of software. One technique for gaining confidence in the behaviour of a program is to prove it correct with respect to some specification [Floyd, 1967], [Hoare, 1969]. Practical techniques to do this have been difficult to derive; however, the use of logic programming languages may alleviate some of the problems [Clark & Tarnlund, 1977; Hogger, 1978, 1979, 1982]. In logic programming the specification and final program are couched in the same language (first order predicate calculus). This allows the original specification to be systematically transformed to the final program without any need to translate between formalisms.

Initially two problems will be tackled: the creation of an interactive proof system for Prolog, and a proof system for Concurrent Prolog (CP) programs. The Prolog proof system will be based on Hogger's [1979] proof techniques as refined and extended by Kornfein. As befits the intended environment it will have a strongly graphical interface based on the work done by Dewar on a graphical Prolog debugger and will borrow heavily from the techniques used in LCF-LSM [Barrow, 1984]. The user will point to objects on the screen and direct the proof system to apply more or less complex proof techniques to transform the code or prove the result correct. The major research problems to be solved are the representation and graphical organisation of the material being proved and the form of the rules to be used for proofs.

Distributed programs which can be realised in VLSI circuitry are significantly easier to transform and prove correct than general distributed programs. Drawing heavily on the work by the VLSI group a system for proving and transforming Concurrent Prolog programs will be constructed. It will initially be restricted to the "VLSI subset"; however, this may later be extended to a larger class of programs.

Cleary and Kornfein (as part of Kornfein's M.Sc. thesis work) have refined and extended the work of Hogger on proving and transforming logic programs. The major extension has been to include completeness as well as correctness proofs within the scheme. A number of algorithms have been proven correct and complete using this system (these proofs were done "by hand"). As described above, Birtwistle and others in the VLSI group have made significant progress in proving electrical circuits correct.

Prototyping and Simulation. Software prototyping promotes the rapid development of application systems to enable early experimentation and so that the results of experiments can be incorporated into successive versions of the system. Boehm *et. al.* [1984] compare "prototyping" with the more traditional "specification" approach to general software development and the value of prototyping, particularly in the human interface portions of a system, seems clear. However, important questions remain as to how this technique can best be incorporated into the software development cycle, its impact on product quality and maintainability, and what tools are needed to support prototyping [Carey & Mason, 1983; Dearnley & Mayhew, 1983].

Previous approaches to software prototyping have concentrated on using and enhancing programming languages such as Ada, Apl, Lisp, and Prolog which have powerful data manipulation capabilities. When prototyping distributed software not only do we need access to powerful, well supported languages for writing individual processes, but we also need tools which address the unique characteristics of distributed systems. We must be able to efficiently prototype: 1) software and hardware architectures; 2) alternative decompositions of system components into processes; and 3) communication and synchronisation protocols. To this end, we see simulation as an integral part of prototyping distributed software.

Boari *et. al.* [1984], Schiffenbauer [1981], and Ward [1979] describe tools which use simulation to support particular phases in the development of distributed software. Our research differs from this previous work in two major respects: 1) we plan to support all phases of the software development cycle; and 2) distributed, concurrent simulation techniques will be used to maximise the correspondence between prototype and actual implementation and to increase efficiency. In particular, simulation techniques will be used to:

- create models of important aspects of target system behaviour including non-computer hardware devices within the development environment
- ensure that timing relationships between components of the system can be preserved and controlled in the simulation
- collect performance statistics so that problem decomposition and process/processor assignment strategies can be assessed
- exercise the system under conditions which would be difficult to induce in the target environment
- permit the software developer to interact with and control the execution of the developing system for testing and debugging purposes at varying levels of detail

An initial design of a prototyping tool that addresses these issues is described in Lomow & Unger [1984a and 1984b] and in Lomow's M.Sc. thesis. This work builds on previous computer system and software simulation approaches presented in Unger & Bidulock [1982], and Unger *et. al.* [1982b]. Although these results towards a general prototyping tool for distributed systems are still preliminary, we are making exciting progress on a number of related fronts — see Birtwistle *et. al.*, [1984a, 1984b, 1984c, 1984d & 1984e], Cleary & Dewar [1984], Inkster *et. al.* [1984], Lomow and Unger [1982], Pooley, Williams, & Birtwistle [1984], and Unger, Lomow & Birtwistle [1984].

Distributed Simulation. Our research in prototyping is currently focussed on the design and development of a distributed simulation technique based on Jefferson & Sowizral's "Time Warp" mechanism [Jefferson & Sowizral, 1982; 1983]. Time Warp involves each application process maintaining its own local virtual time and receiving time stamped messages, and when a message is received time stamped in the receiving processes past, the process is "rolled back" to a previous state with a virtual time at least as old as the received message. Two versions are under development: a distributed Concurrent Prolog (CP) system that accomplishes "rollback" using backtracking [Cleary *et. al.*, 1985], and a procedural discrete event simulation system that accomplishes rollback with M68000 process state saving. Both will be built on "Tipc" which extends Jipc; preliminary work on the procedural approach was reported in Unger, Lomow & Andrews [1985].

Monitoring Distributed Systems. Research into distributed system prototyping and hierarchical monitoring will be supported by the Jade monitoring system which is already in place. Very little work has been done in the monitoring of executing distributed and concurrent programs [Baiardi *et. al.*, 1983; Garcia-Molina *et. al.*, 1984; Snodgrass, 1982]. The problems of non-determinism and the impact monitoring necessarily has on the behaviour of distributed systems are challenging.

Because prototyping distributed systems can be accomplished in either a top-down or bottom-up fashion it is important for the developer to be able to monitor and debug a system at multiple levels of abstraction. In a complex system the events of interest may not be represented by single interprocess events but may instead be composed of several events. By recognising a sequence of interprocess events as a higher level event we are able to create abstractions for low level detail. Just as we can construct an abstraction in terms of primitive events, we can construct yet higher level abstractions in terms of events defined at the previous level. Ideally, each level of abstraction will correspond to a level in the original problem decomposition. Event abstraction is the subject of Bates & Wileden [1983] and is discussed in Henneman & Rouse [1984] and Model [1979]. The focus of our work is on the hierarchical specification of process interactions. For example, user "sessions" consisting of a sequence of retrieval and update transactions for a distributed data base might be the highest level view. A next level in this distributed application might be the data base "transactions", and finally the lowest level protocol may be the interprocess "messages" used to implement transactions. The monitoring tools must support moving between these three views during testing and performance evaluation.

Closely related to moving among protocol layers while monitoring is the testing of interprocess interactions against a protocol specification. Just as composite events can be posted to a monitor, events which cannot be parsed as part of a specified interaction can be posted as potentially erroneous process interactions. The hierarchical monitoring and protocol verification work is currently being explored in the context of the Jade/2 monitoring scheme.

The Jade/2 monitoring system consists of an extensible monitoring scheme which collects process interaction information from a multi-lingual distributed application program. This information is sent via unmonitored Jipc messages to channels, where one channel resides on each computing node on which the application is running. These channels then direct monitoring information to one or more "consoles" which interact with the developer of the application program. Different consoles provide different interpretations, or views, of an executing distributed application program.

A number of different kinds of consoles have been explored including: 1) a basic one line of text per interprocess event view; 2) an animated display with icons representing processes, and messages moving among these processes; 3) a statistical console for measuring traffic density among processes; 4) a distributed deadlock detector, and 5) a prototype protocol verifier which reports deviations from a protocol specification. Additionally, tools are provided for directing the non-deterministic execution of a distributed system based upon either user control or histories collected during previous runs of the same system. These techniques make it possible to reproduce error situations and to test possible, but improbable, situations. The monitor scheme also provides facilities for filtering event information and for setting and detecting breakpoints. These ideas have been described in several papers [Joyce & Unger, 1985; Joyce *et. al.*, 1985; Joyce, Birtwistle, & Wyvill, 1984; and Dewar & Unger, 1984].

Personnel. The specification and prototyping group now consists of three faculty (Unger, Cleary and Vollmerhaus), six graduate students (Kornfein, Lomow, Goh, Inkster, Schack, and Yen), three visiting scientists (Dong Zhixin, Li Xining, Xiao Zhong-e), and two research staff (Dewar and Slind). Yen, Dong, Dewar and Slind have been working in monitoring systems and animating graphical representations of both arbitrary distributed programs and Prolog programs. Goh, Schack and Xiao are actively pursuing alternative schemes for using Time Warp for distributed simulations. Lomow, Inkster and Li are working on new distributed software prototyping concepts. Kornfein is working on the transformation of logic programs.

Appendix B. Environment Support Tools

Research in phase 2 concentrates on the design of the target environment and target language, and the design and implementation of support tools for constructing the target environment. This section gives details on the plans for and current status of the three major tools to be used in constructing the environment, and on the personnel who will be involved with each.

Interactive programming environment for the target language

The system will have at its heart a target language which will be used at all levels. Although we expect the tools for the environment to be written in a variety of languages — mainly for efficiency and historical reasons — the tools will communicate between themselves, and with the user of the system, in the common target language. The target language must (a) be sufficiently expressive, (b) support concurrency, (c) embody simple and elegant semantics (because tools will need to process it in a variety of ways), (d) encourage logical proof and stepwise refinement of specifications, and (e) be potentially compilable down to silicon. The current presupposition is that it will be some form of parallel logic programming language.

It is clear that we will need to provide excellent support for the target language, including: efficient, distributed implementations; an interpreter with a well-engineered human interface; advanced (preferably graphical) debugging tools; and program browsing and documentation support. This subsection discusses work on the implementation and distributed implementation of parallel logic languages; other aspects of the environment are considered in following subsections.

Relation to previous work. Recently there has been considerable interest in the parallel execution of logic programs [Furukawa *et. al.*, 1982; van Emden & Filho, 1982]. Studies have included special purpose hardware [Eisinger *et. al.*, 1982] and message passing techniques [Bowen, 1982].

Most attempts at parallel logic programming languages use some form of Prolog with annotations to control the parallelism, e.g. Gallaire & Lasserre [1982], Hogger [1982], Monteiro [1982], Pereira [1982], and Clark *et. al.* [1982]. Of particular interest are the languages Concurrent Prolog (CP), described by Shapiro [1983] and Parlog described by Clark & Gregory [1984]. Both these languages use shared variables for communication, and draw heavily for their ideas on the earlier Relational Language described by Clark & Gregory [1981]. They differ mainly in the constructs used to achieve delay and synchronisation of goals. At least one recent uniprocessor implementation of CP by Mierowsky [1984] is restricted to a "flat" form of CP with only a very simplified form of or-parallelism. This was done both to simplify the implementation and to improve efficiency.

Plans and methodology. The major thrust of this work will be the successive refinement and extension of Concurrent Prolog (CP). The initial aim is to prove that CP can be extended into the target language used in the final phase of the project. The implementations will (approximately in order of completion): be fully distributed and based on an efficient local implementation including garbage collection (probably obtained from elsewhere); contain time delays to support discrete event simulation and prototyping; be distributed and time based (using Jefferson's "Time Warp" mechanism); use shared memory efficiently and run on the mesh computer; and support execution of CP programs compiled down to silicon. Graphical tools for debugging CP will be built on top of these implementations as well as an operating system for the mesh computer. Parallel work in VLSI will be to develop support chips for the execution of Prolog and CP, and to provide a path for the migration of software into silicon.

Current status and personnel. Work to date has produced a fully distributed implementation of CP using Jipc [Cleary, 1985]. The implementation is designed around a fixed set of processes (with the intention that they be mapped one to one with physical processors). In this it may be contrasted with the implementations of CP by Chun Man Tam [1984] and by Lee [1984], which both use one process per CP goal. The result is that both systems are very slow and make very heavy use of process creation and destruction. A version of CP including time delays has been developed as a discrete event simulation system by Cleary and Unger (Broda and Gregory [1984] report a similar system in Parlog). Work is currently proceeding on integrating CP with the "Time Warp" system being built by Unger, Xiao, Goh and Cleary. This will use Prolog backtracking as a rollback mechanism. Dewar has recently completed an M.Sc. thesis in which he constructed a graphical debugger for (standard) Prolog. This displays the and/or tree and data structures of the executing Prolog program graphically to produce a very novel and effective debugging tool. This work will form the basis for graphical debuggers for distributed CP systems. Erwin Liu has recently completed a register level design for a Prolog interpreter realisable in VLSI and a silicon compiler for random logic targetted at Weinberger arrays. He will be working on a silicon compiler for CP. Kusalik is currently producing an operating system based on CP as part of his PhD. He will be working on the production of a CP operating system for the Mesh computer.

System for prototyping user interfaces

Relation to previous work. A user interface prototyping system (UIPS) or management system (UIMS) offers the interface designer the opportunity to develop and test interaction techniques rapidly and obtain useful evaluations of their performance. It permits experimentation with different interfaces to the same application, allows the interface and the application to be designed independently, may generate a more consistent and hence more easily learned interface, and provides a language for discussion of requirements with the user community. Despite these and other advantages, only a very few systems to assist the interface designer existed before 1980 and papers on the subject were mostly relegated to journals on computer graphics. The situation has now changed dramatically, and interest in the user interface and human factors in computer environments is growing rapidly. Two major conferences on UIPSs and UIMSs have recently taken place [Seattle, 1982; Seeheim, 1983]. Good examples of UIPSs are the Interactive Dialogue Driver of Hill & Irving [1984]; the building blocks catalogue of Green [1982]; the TIGER Interactive Command and Control Language of Kasik [1982]; and the Menulay courseware building system of Buxton *et. al.* [1983]. Work on such systems features prominently in the flood of conferences now appearing on the human-computer interface.

Plans and Methodology. The UIPS will be organised as a toolbox of special purpose interactive prototyping tools. Each unit will allow generation and manipulation of a component of the user interface, for example: static displays; animations of displays; menu layouts; interaction techniques; dialogue design; and application interfaces. So that the UIPS will become useful as soon as possible, the component prototyping tools will be implemented first. Later on, a top layer will be added through which the interface components can be created, accessed, edited and checked for consistency. Particular research issues that will be addressed include: formal specification of dialogues and interaction techniques; scripting; the incorporation of expertise; de-skilling of interface design; and the use of multi-modal interactions such as speech.

Each component of the UIPS will communicate with the others using Jipc. In phase 3 of the project, the designer's choices will be restricted by constraining the toolbox so that interfaces developed using the UIPS will conform to the target environment user interface definition. We will also couple the components generated by the UIPS more tightly by using procedure calls instead of message passing where necessary so that the environment interface offers better response.

A major goal for the UIPS is to support a *browser* for the target language [Goldberg, 1984; Delisle *et. al.*, 1984]. The browser will enable a user to create and manipulate modules of the target language efficiently. It will also support the ability to organise modules into hierarchies, the ability to create structured documentation about a module, fast access to modules referencing and referenced by the module under

examination, and access to the other components of the target environment. The browser will be resident on the Jade workstation to provide fast and guaranteed response time, and will, in common with other interfaces developed using the UIPS, utilise the interaction facilities supplied by the workstation software. The browser, built on top of more primitive UIPS elements, will constitute an valuable test of the adequacy of our system to generate real, as opposed to toy, interfaces.

A crucial step in our research plan is the creation of a script-driven interface prototyping and management unit that abstracts and formalises the various components and resources that make up a functioning interface for an interactive system. Some work has already started on this [Hill & Irving, 1984; Levinson, Joyce & Hill, 1984] but there is much research to be done. It is necessary to select a formalism (ATN networks are promising, but extensions are required to deal with matters such as interaction techniques and flexible feedback specification). It is also necessary to resolve questions of internal versus external control. Our current thinking suggests a greatly extended form of external control tantamount to specifying a new kind of operating system component. The scripting of interaction, based on a suitable formalism and this form of extended external control, then allows other areas to be addressed. Of particular importance is the encapsulation of design rules and I/O device abstractions in such a way that the interface designer can concentrate on the function and style of the dialogue, without having to continually dive into details of rule implementation and physical device management. In parallel with this work, we have to develop an adequate intermediate language for dialogue specification, and manage conversion between this and the extended ATN representation.

Current Status. A number of modules which can be directly incorporated into the UIPS are already completed or are nearing completion. Hill & Irving, 1984 and Levinson, Joyce & Hill, 1984 report a prototype scripting system and outline the overall structure developed. The Jade workstation window manager supplies graphics and textual output, cutting and pasting of text, pop-up menu selection, context dependent help, and event-based input [Witten *et. al.*, 1983]. The Jade graphics system combines hierarchical, recursive picture structures, real-time animation and automatic modelling [Wyvill *et. al.*, 1984]. It is supplemented by an interactive editor for static pictures. Greenberg has implemented a comprehensive set of tools providing easier access to window manager features from programs and supporting some interaction techniques including static menus, light buttons, scrollbars, cursor control and menu and window layout. Levinson has created a prototype graphical interface generator for the interactive design of interface modules and the control of animation. A full system is scheduled for completion by September, 1985. A module created by this system converts incoming commands given in terms of the application's requirements into sequences of graphical actions such as animations. Freedman is working on an interactive editor for controlling system execution via Jipc message generation. An overview of problems and some early results from work on the Jade user interface appears in Hill *et. al.* [1984].

Much other work in progress is directly relevant to design of the UIPS but requires some adaptation before it can be incorporated. Greenberg and Witten [1985] have designed a *Workbench Creation System* for interactive creation of menu hierarchies providing access to the Unix environment through multiple independent views using the Jade window manager. This design is currently being implemented. Greenberg has also recently completed an Master's thesis under Witten's supervision on automatic modelling of the user and concomitant adaptation of the user interface [Greenberg, 1984; Greenberg & Witten, 1984; Witten, Cleary & Greenberg, 1983]. Irving has designed and implemented a system for providing more friendly interfaces to already existing programs [Hill & Irving, 1984]. His *Interactive Dialogue Driver* isolates the form of the dialogue from its content and allows easy specification of scripts in a special purpose language.

Several other projects are linked to the work, testing parts or contributing ideas and will serve as a basis for constructive criticism of it. For example, the MAWD project is using Jade as the basis for a multi-media adaptive workstation for the disabled. Lines of research being pursued include touchpad input and speech output [Hill *et. al.*, 1984; Dohrn, 1984], speech and electromyographic input, predictive interfaces using keystroke analysis and ergonomics of workstation design. Dewar is completing a Master's thesis on

a graphical tool for debugging Prolog programs using the Jade graphics system. Lukey is pursuing graduate work on a window-based debugging tool for Pascal that incorporates some expert knowledge.

Personnel. The department of Computer Science boasts a strong Human Computer Systems groups consisting of five faculty (Hill, Witten, Wyvill, Cleary and Gaines), one Ph.D. and six M.Sc. students (Greenberg, Irving, Darragh, Lukey, Jansonius, Dignum and Ariel), four programmers (Esau, Freedman, Polischuk and Mellon) and one post doctoral fellow (Levinson). Four of the faculty have been involved with the Jade project since its inception, as have Levinson, Esau, Greenberg and Irving. Levinson has recently been awarded a University of Calgary fellowship for design and construction of a UIPS and related investigations. Greenberg is beginning Ph.D. research will be on user interface design. As mentioned above, Irving's work is on dialogue design and user interface prototyping. Dignum is working on the formal description of user interfaces using command language grammars.

Documentation presentation and maintenance tools

Relation to previous work. High quality documentation is crucial for any large project. Systems such as TEX [Knuth, 1979] and Scribe [Reid, 1980] are targetted at high quality printed output and user accessibility; other research is aimed at interactive or "WYSIWYG" formatting [Chamberlin *et. al.*, 1981; Hammer *et. al.*, 1981]. Mint [Hibbard, 1983] emphasises interactive previewing of documents intended ultimately for paper. Our research is targetted primarily at interactive viewing, which originated with NLS [Engelbart & English, 1968] and the Hypertext vision [Nelson, 1974], and has more modern incarnations in Zog [McCracken & Akscyn, 1984] and the Smalltalk browser [Weyer, 1982]. We have automatically generated on-line browsable documents from a typesettable source [Witten & Bramwell, 1985], and are working with typeset-quality on-line documentation with flexible indexing and layout facilities [Bonham & Witten, in press].

Plans and methodology. The documentation research will be based on Jot [Bonham & Witten, in press], which maintains high quality presentation on the screen by the use of multiple fonts, variable-pitch characters, and careful page layout. This will be extended to include "active" documents (i.e. they incorporate arbitrary executable programs). Research is planned on more structured ways of integrating interaction with text through the use of forms. The notion of paper business form can be extended to encompass a wide variety of interactive systems, and a prototype generalised forms interpreter has been constructed to demonstrate feasibility in an experimental system [Witten, 1984]. This will be interfaced to an existing relational database system (such as Ingres) to allow project management within the documentation tool itself. The implementation will be multi-lingual (Jot in Lisp, forms in Prolog, screen handler and database in C) with Jipc intercommunication.

Current status and personnel. Several documentation tools have already been developed within Jade by a variety of people (e.g. the browsing system which was the subject of Bramwell's M.Sc. thesis and has been extended since, an Emacs subsystem for split-screen editing/formatting, software and a set of conventions for WYSIWYG document preparation), and all the applicants have extensive experience with a wide variety of document preparation systems. Witten has a research interest in interactive documentation (e.g. [Bonham & Witten, 1982; in press; Corbett & Witten, 1982; Witten et. al., 1982; Witten, 1984; Witten & Bramwell, 1985] and his student Bonham has completed a recent M.Sc. (1985) in the area. Gaines has very substantial experience in typesetting, as Executive Director of Monotype International, in a variety of consulting roles, and through extensive publications on the subject.

Appendix C. Applications

Phase 1 of the project will be the application of the environment to some substantial distributed problems. The major area of application will be 3D and 2D graphics. Two sharply focussed applications have been chosen from this area to act as probes or tests of the final integrated environment. They are discussed individually below. It is expected that they will have been substantially completed at least once early in the project. They will be completely redone from scratch within the completed target environment. A third application, continuous simulation, is beginning to emerge as a possible area of research, and is discussed briefly below. Other applications will continue to be developed throughout the life of the project.

The general area of graphics has been chosen as an application because: it presents challenging research problems in the design of distributed algorithms; the problems are computationally intensive and will respond well to distribution and concurrency; there is significant graphical expertise within the department; and there is significant commercial interest in graphics within the Calgary-based geophysical exploration industry.

High quality 3D graphics

The major problem in the production of high quality 3D graphics images is the huge amount of computation involved (see Hall & Greenberg, [1983]). A promising research area is the use of distributed systems to speed up these calculations, with the ultimate objective being the real-time production of high quality images. Because of the large amounts of data involved and the different forms of coherence available in images, the design of such algorithms presents formidable challenges to the software designer. A number of different approaches have been suggested and a few tried by different workers. The parallel production of graphical objects from their descriptions, for example has been tackled in the 2D case by Lantz and Nowicki [1984] and to some extent in the Jade project where simple strategies are being investigated for distributing a 2D hierarchical data structure between host and workstation [Wyvill *et. al.*, 1984]. The bottleneck for 3D animation is in rendering: the process of taking graphical primitives such as polygons and scan converting them to produce a realistic image. A large number of floating point calculations are required to solve problems of lighting, smoothing, transparency, shadows, reflections, refraction etc. Previous work into distributed ray tracing algorithms has been done by [Dippe & Swensen, 1984]. A system called Links [Nishimura *et. al.*, 1983] uses a special purpose multiprocessor to do ray tracing. This differs from the current work in the special purpose nature of the hardware used (as opposed to the use of a large number of simple uniformly connected processors).

The primary target for implementation of 3D rendering algorithms will be the Mesh computer — a homogeneous array of 64 processors. The memory available on the proposed design (2Mbytes/processor) and the presence of floating point support should allow realistic scenes to be rendered. (Scenes with up to a million graphical objects are not uncommon. This means that machines with small amounts of memory are unsuitable for this type of application). Current trends in microprocessor technology imply that homogeneous systems like the Mesh computer will provide the most cost effective way of obtaining sufficient computational power. This hypothesis is borne out by experience with the "Cosmic Cube" [Seitz, 1985], which is a 64 node homogeneous machine with hypercube interconnection. It has been applied very successfully to a number of problems in physics. The proposed Mesh computer differs from the Cosmic Cube mainly in the memory on each processor (2Mbytes as opposed to 128Kbytes) and the in the simpler interconnection topology used. The mesh interconnects each processor with its four nearest neighbours. This means that the distances between processors are greater than in a hypercube. However, each link is implemented as a dual ported memory, which allows peak transfer rates of 32Mbits/sec. This offsets the greater distances between nodes. Also, the interconnect wiring between processors is

particularly simple greatly reducing the difficulties in constructing the system. The Jade environment also provides a very simple development route for programs run on the Mesh. They can initially be prototyped on a single processor (such as a VAX 11/780) and then run on a network of workstations (the current Corvus Concept network) and only when debugged need they be moved to the Mesh.

A number of distributed algorithms will be developed, including Z-buffer, ray tracing and Warnock's algorithm. They will all use Jade for initial development and will ultimately be transferred to the Mesh computer. Toward the end of the project one of these algorithms, distributed ray tracing, will be reimplemented within the target environment. It will be specified and prototyped and then transferred to the Mesh computer. In this role the problem will exercise:

- the target environment's ability to handle prototypes with large numbers of processes;
- the target language's efficiency for computationally intensive tasks;
- the implementation of the target language on Mesh computer.

Both 2D [Wyvill *et. al.*, 1984] and 3D [Wyvill, Liblong & Hutchinson, 1984] hierarchical graphics systems have been developed and some preliminary work has been undertaken on distributing the 2D system by Levinson and Cleary. Several alternative ray tracing strategies have been analyzed [Cleary *et. al.*, 1983] and work is currently continuing by Pearce. The graphics group has produced a large amount of 3D animation software, known as Graphicsland, and also several minutes of animated film. Others actively involved are Novacek, McPheeters, Jansonius, Dignum and Ariel. A 4 processor prototype of the Mesh computer has been running since the summer of 1984. It is currently being upgraded to 9 processors. The Jipc kernel software has been ported to the system, and one version of distributed ray tracing has been tested on the Mesh. A distributed Z-buffer algorithm has been written, and tested on the VAX 11/780 and Corvus networks.

2D distributed graphics

2D graphics is already available as part of the support for the target environment. However, it presents interesting problems in distribution between host and workstation in order to provide timely response in highly interactive situations.

One recent development has been the design of a distributed graphics system based on a high level graphics language embedded in Prolog. It consists of two components, a high level logic program which generates the picture and a dedicated graphics processor which performs the necessary geometrical transformations and picture generation. The system, as described in Cleary [1984], minimises the communication necessary between these two components. This system will be: specified and proven correct; implemented on a uniprocessor; implemented in a multiprocessor; and finally the remote graphics part will be designed in silicon (it is unlikely that final fabrication and testing can be complete within the three year scope of the project).

The advantages of this application are: a description and simple implementation are already available; it is a problem with significant commercial implications; it draws on the group's expertise in graphics and logic programming as well as VLSI and distributed systems; it can realistically be realised at many different scales of distribution and efficiency; the resulting system would be useful within the environment itself for graphics support; it will exercise all of the proposed integrated environment.

Continuous simulation

The problems of using continuous simulation to solve systems of linear equations have been addressed in the context of hydraulic networks [Porsching *et. al.*, 1971], power systems [Fong & Pottle, 1978]. Research in this area is of significant commercial interest. More recent approaches to solving the general problem are discussed in Arnold *et al.* [1983] and Reed & Patrick [1984].

Trolimenkoff and Mahani in the Electrical Engineering Department are currently working on the problem of solving large sparse linear equations. The context of this work is the continuous simulation of Nuclear Power Reactors via state equations. This problem is computationally very intensive and responds well to distributed and concurrent solutions. The Mesh computer will be used to implement and test a number of algorithms which could otherwise otherwise only be simulated. The size of the mesh computer and the presence of floating point support will allow much larger distributed solutions to be tested than would otherwise be feasible. This in turn will allow checks on the scaling properties of such algorithms with the size of the problem. Mahani is currently doing a Ph.D. in this area and has designed and simulated a number of algorithms for Jacobi relaxation of sparse matrices. A number of small problems have been simulated in a uniprocessor setting.

Appendix D. References

Almes, G.T., Black, A.P., Lazowska, E.D., and Noe, J.D., (1985) "The Eden System: a technical review," *IEEE Transactions on Software Engineering*, vol. SE-11, no. 1, pp. 43-59. January

Arnold, C.P., Parr, M.I., and Dewe, M.B., (1983) "An Efficient Parallel Algorithm for the Solution of Large Sparse Linear Matrix Equations," *IEEE Transactions on Computers*, vol. C-32, no. 3, p. 265. March

Baiardi, F., De Francesco, N., Matteoli, E., Stefanini, S., and Vaglini, G., (1983) "Development of a Debugger for a Concurrent Language," *Software Engineering Notes*, vol. 8, no. 4, p. 98. August

Barrow, H., (1984) "Proving the Correctness of Digital Hardware Designs," *VLSI Designs*, vol. 5, no. 7, pp. 64-77. July

Barstow, D.R., Shrobe, H.E., and Sandewall, E., (1984) *Interactive Programming Environments*, McGraw-Hill, New York.

Bates, P. and Wileden, J.C., (1983) "An Approach to High-Level Debugging of Distributed Systems," *Software Engineering Notes*, vol. 8, no. 4, p. 107. August

Birtwistle, G., Cleary, J.G., Joyce, J., Liblong, B., Unger, B.W., Witten, I.H., and Wyvill, B.L.M., (1984a) "A simulation environment," *Proc. Canadian Information Processing Society National Conference*, Calgary, Alberta. May

Birtwistle, G. and Luker, P., (1984b) "Dialogs for Simulation Programming," *Proc. SCS Conference on Simulation in Strongly Typed Languages*, San Diego, California. February

Birtwistle, G., Luker, P., Lomow, G.A., and Unger, B.W., (1984c) "Process Style Packages For Discrete Event Modelling: Data structures and packages in Simula," *Transactions on Simulation*, vol. 1, no. 1, pp. 75-79.

Birtwistle, G.M., Luker, P., Lomow, G.A., and Unger, B.W., (1984d) "Process Style Packages For Discrete Event Modelling: Transaction, event, and activity approaches," *Transactions on Simulation*. in press

Birtwistle, G.M., Luker, P., Lomow, G.A., and Unger, B.W., (1984e) "Process style packages for discrete event modelling: Simula's class Simulation," *Transactions on Simulation*, vol. 2, no. 1, pp. 175-195.

Boari, M., Crespi-Reghizzi, S., Dapra, A., Maderna, F., and Natali, A., (1984) "Multiple-Microprocessor Programming Techniques: MML, a New Set of Tools," *IEEE Computer*, vol. 17, no. 1, pp. 47-59. January

Boehm, B.W., Gray, T.E., and Seewaldt, T., (1984) "Prototyping versus Specifying: a multiproject experiment," *IEEE Transactions on Software Engineering*, vol. SE-10, no. 3, pp. 290-302. May

Bonham, M. and Witten, I.H., (1982) "A structured procedural document preparation language," *Proc Canadian Information Processing Society National Conference*, pp. 6-12, Saskatoon, Saskatchewan. May

Bonham, M. and Witten, I.H., (in press) "Towards distributed document preparation with interactive and noninteractive viewing," *Infor.*

Bowen, K.A., (1982) "Concurrent Execution of Logic," in *Proc of the First International Logic Programming Conference*, pp. 26-30, Marseille, France.

Broda, K. and Gregory, S., (1984) "PARLOG for discrete event simulation," *Proc. Second International Logic Programming Conference*, Department of Computing, Imperial Colege of Science and Technology, London. (also available as Research Report DOC 84/5, July)

Buchanan, L., (1984) "Scale - A VLSI Design Language," Technical Report CSR-117-82, Department of Computer Science, University of Edinburgh, Edinburgh, Scotland. May

Buxton, W., Lamb, M.R., Sherman, D., and Smith, K.C., (1983) "Towards a Comprehensive User Interface Management System," *Computer Graphics*, vol. 17, no. 3, pp. 35-42. July

Carey, T.T. and Mason, R.E.A., (1983) "Information System Prototyping: Techniques, Tools and Methods," *INFOR*, vol. 21, no. 3, pp. 177-187. August

Chamberlin, D.D., King, J.C., Slutz, D.R., Todd, S.J.P., and Wade, B.W., (1981) "JANUS: an interactive system for document composition," *SIGOA Newsletter (Proc ACM Symposium on Text manipulation, Portland, Oregon)*, vol. 2, no. 1/2, pp. 82-91. Spring/Summer

Clark, K.L. and Tarnlund, S., (1977) "A first order theory of data and programs," *Proc. I.F.I.P. Conf.* .

Clark, K.L. and Gregory, S., (1981) "A Relational Language for Parallel Programming," *Proc. Conf. on Functional Programming Languages and Computer Architectures*, pp. 171-178, ACM. October

Clark, K.L., McCabe, F.G., and Gregory, S., (1982) "IC-PROLOG Language Features," in *Logic Programming*, pp. 253-266, Academic press.

Clark, K. and Gregory, S., (1984) "PARLOG: Parallel Programming in Logic," Research Report DOC 84/4, Department of Computing, Imperial College of Science and Technology.

Cleary, J., Wyvill, B.L.M., Birtwistle, G., and Vatti, R., (1983) "Design and Analysis of a Parallel Ray Tracing Computer.," *Proc. XI Association of Simula Users Conference.*, Paris.

Cleary, J.G. and Dewar, A., (1984) "Interpreters for Logic Programming - A Powerful Tool for Simulation," *Proc. SCS Conference on Simulation in Strongly Typed Languages*, San Diego, California. February

Cleary, J.G., (1984) "A distributed graphics system implemented in Prolog," Research Report 84/173/31, Department of Computer Science, University of Calgary, Calgary, Alberta.

Cleary, J.C., (1985) "A Distributed Implementation of And-parallel Concurrent Prolog," Research report 85/190/3, Department of Computer Science, University of Calgary, Calgary, Canada. (submitted to IEEE Int. Conf. on Parallel and Distributed Systems)

Cleary, J.G., Lomow, G.A., Unger, B.W., and Xiao, Z., (1985) "Jade's IPC Kernel for Distributed Simulation," *accepted Proc. Association of Simula Users*, Calgary, Alberta. August

Corbett, C. and Witten, I.H., (1982) "On the inclusion and placement of documentation graphics in computer typesetting," *Computer J*, vol. 25, no. 2, pp. 272-277. February

Craft, D.H., (1983) "Resource Management in a Decentralised System," *Operating Systems Review*, vol. 17, no. 5, pp. 11-19.

Dearnley, P.A. and Mayhew, P.J., (1983) "In Favour of System Prototypes and their Integration into the Systems Development Cycle," *The Computer Journal*, vol. 26, no. 1, pp. 36-42.

Delisle, N.M., Menicosy, D.E., and Schwartz, M.D., (1984) "Viewing a Programming Environment as a Single Tool," *SIGPLAN notices*, vol. 19, no. 5, pp. 49-56.

Dewar, A. and Unger, B.W., (1984) "Graphical Tracing and Debugging of Simulations," *Proc. SCS Conference on Simulation in Strongly Typed Languages*, San Diego, California. February

Dippe, M. and Swensen, J., (1984) "An adaptive subdivision algorithm and parallel architecture for realistic image synthesis.," *Proc. SIGGRAPH '84*, pp. 149-158. July

Dohrn, C., (1984) "Speech Pad: direct manipulation computer access for the visually disabled based on speech and touch," M.Sc. Thesis, Department of Computer Science, University of Calgary, Calgary, Canada.

Eisinger, N., Kasif, S., and Minker, J., (1982) "Logic Programming: a parallel approach," in *Proc of the First International Logic Programming Conference*, pp. 71-77, Marseille, France.

van Emden, M.H. and de Lucena Filho, (1982) "Predicate Logic as a Language for Parallel Programming," in *Logic Programming*, pp. 189-198, Academic Press.

Englebart, D.C. and English, W.K., (1968) "A research center for augmenting human intellect," *Proc Fall Joint Computer Conference*, vol. 33, pp. 395-410, AFIPS Press, Arlington, VA.

Estublier, J, Ghoul, S, and Krakowiak, (1984) "Preliminary Experience with a Configuration Control System for Modular Programs," *SIGPLAN notices*, vol. 19, no. 5, pp. 149-156.

Fahlman, S.E. and Harbison, S.P., (1984) "The Spice Project," in *Interactive Programming Environments*, ed. D.R. Barstow, H.E. Strobe and E. Sandewall, pp. 546-557, McGraw-Hill, New York.

Fischer, C.N., Pal, A., Stock, D.L., Johnson, G.F., and Mauney, J., (1984) "The POE Language-Based Editor Project," *SIGPLAN notices*, vol. 19, no. 5, pp. 21-29.

Floyd, (1967) "Assigning meaning to programs," in *Mathematical Aspects of Computer Science*, ed. Schwarz, J.J., A.M.S., Providence, R.I..

Fong, J. and Pottle, C., (1978) "Parallel Processing of Power System Analysis Problems via Simple Parallel Microcomputer Structures," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-97, no. 5, p. 1834. September

Furukawa, K., Nitta, K., and Matsumoto, Y., (1982) "Prolog Interpreter based on Concurrent Programming," in *Proc of the First International Logic Programming Conference*, pp. 38-44, Marseille, France.

Gallaire, H. and Lasserre, C., (1982) "Metalevel Control for Logic Programs," in *Logic Programming*, pp. 173-188, Academic press.

Garcia-Molina, H., Germona, F., and Kohler, W.H., (1984) "Debugging a Distributed Computing System," *IEEE Transactions on Software Engineering*, vol. 10, no. 2, p. 210. March

Garlan, D.B. and Miller, P.L., (1984) "GNOME: an Introductory Programming Environment Based on a Family of Structure Editors," *SIGPLAN notices*, vol. 19, no. 5, pp. 65-72.

Goldberg, A., (1984) "The Influence of an Object-Oriented Language on the Programming Environment," in *Interactive Programming Environments*, ed. D.R. Barstow, H.E. Strobe and E. Sandewall, pp. 141-173, McGraw-Hill, New York.

Gordon, M., (1981) "A Model of Register Transfer Systems with Applications to Microcode and VLSI Correctness," Computer Science Report CSR-82-81, University of Edinburgh.

Gordon, M., (1983) "Proving a Computer Correct," Technical Report 42, Computer Laboratory, University of Cambridge.

Green, M., (1982) "Towards a User Interface Prototyping System," *Graphics Interface '82*, pp. 37-45.

Greenberg, S., (1984) "User Modeling in Interactive Computer Systems," MSc Thesis, Department of Computer Science, University of Calgary, Calgary, Canada.

Greenberg, S. and Witten, I.H., (1984) "Comparison of Menu Displays for Ordered Lists," *Proc. Canadian Information Processing Society National Conference*, pp. 464-469, Calgary, Alberta. May

Greenberg, S. and Witten, I.H., (1985 (in press)) "Interactive End-User Creation of Workbench Hierarchies within a Window System," (ICIS Congress 85, Montreal).

Hall, R.A and Greenberg, D.P, (1983) "A Testbed for realistic image synthesis," *IEEE Computer Graphics and Applications*, vol. 3 No. 8, pp. 10 - 19. November

Halpern, J., Manna, Z., and Moskowski, B., (1983) "A Hardware Semantics Based on Temporal Intervals," Proc. Tenth International Colloq. on Automata, Language and Programming, Barcelona, Spain, pp. 278-291, Springer-Verlag, Berlin.

Hammer, M., Ilson, R., Anderson, T., Gilbert, E., Good, M., Niamir, B., Rosenstein, L., and Schoichet, S., (1981) "The implementation of Etude, an integrated and interactive document production system," *SIGOA Newsletter (Proc ACM Symposium on Text manipulation, Portland, Oregon)*, vol. 2, no. 1/2, pp. 137-146. Spring/Summer

Helmhold, D. and Luckham, D., (1985) "Debugging ADA Tasking Programs," *IEEE Transactions on Software Engineering*, vol. 2, no. 2, pp. 47-57. March

Henderson, P., (1984) "Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments," *SIGPLAN notices*, vol. 19, no. 5.

Henneman, R.L. and Rouse, W.B., (1984) "Human Performance in Monitoring and Controlling Hierarchical Large-Scale Systems," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-14, no. 2, pp. 184-191. March

Hibbard, P., (1983) "User manual for MINT -- the SPICE document preparation system," Technical Report, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA.

Hill, D.R. and Irving, G., (1984) "The Interactive Dialogue Driver: a UNIX Tool," *Proceedings CIPS Session 84*, pp. 307-313, Calgary.

Hill, D.R., Witten, I.H., Neal, R., and Lomow, G., (1984) "Jed and Hide: practical questions for the Jade user interface," *Proc. Canadian Information Processing Society National Conference*, pp. 373-380, Calgary, Alberta. May

Hill, D.R., Dohrn, C., Darragh, J., Esau, R., Levinson, D., Unger, B., and Witten, I.H., (1984) "Using Speech Output as a Medium for Human-Computer Dialogue," *Proc. Canadian Information Processing Society National Conference*, pp. 470-476, Calgary, Alberta. May

Hoare, (1969) "An axiomatic basis of computer programming," *Comm. ACM.*

Hogger, C.J., (1978) "Program synthesis in predicate logic," *Proc. A.I.S.B/G.I. Conf. on Artificial Intelligence.*

Hogger, C.J., (1979) *Derivation of logic programs,* Imperial College of Science and Technology, London. PhD. thesis

Hogger, C.J., (1982) "Concurrent Logic Programming," in *Logic Programming*, pp. 189-198, Academic Press.

Inkster, J., Lomow, G.A., and Unger, B.W., (1984) "Combined Continuous and Discrete-Event Simulation in Ada," *Proc. SCS Conference on Simulation in Strongly Typed Languages*, San Diego, California. February

Jefferson, D. and Sowizral, H., (1982) "Fast Concurrent Simulation Using the Time Warp Mechanism, Part I: Local Control," Technical Report, The Rand Corporation, Santa Monica, California. December

Jefferson, D. and Sowizral, H., (1983) "Fast Concurrent Simulation Using the Time Warp Mechanism, Part II: Global Control," Technical Report, The Rand Corporation, Santa Monica, California. August

Joyce, J., Birtwistle, G.M., and Wyvill, B.L.M., (1984) "Andes - An Environment for Animated Discrete Event Simulation," *UK Simulation Conference 1984*, Bath, U.K..

Joyce, J. and Unger, B.W., (1985) "Graphical Monitoring of Distributed Systems," *SCS Conference on AI, Graphics, and Simulation*, San Diego, California. January

Joyce, J., Lomow, G.A., Slind, K., and Unger, B.W., (1985) "Monitoring Distributed Systems," Research Report 84/8/1, Department of Computer Science, University of Calgary. submitted to I.E.E.E. Software

Kasik, D.J., (1982) "A User Interface Management System," *Computer Graphics*, vol. 16, no. 3, pp. 99-106. July

Knuth, D.E., (1979) *Tex and Metafont: new directions in typesetting*, Digital Press and American Mathematical Society.

Lantz, A and Nowicki, W, (1984) *Structured Graphics for Distributed Systems*, University of Stanford. February

Lauer, P.E., (1983) "Users Introduction to BCS: a computer based environment for specifying, analyzing and verifying concurrent systems," Technical Report ASM/107, Computing Laboratory, University of Newcastle upon Tyne, Newcastle, England.

Lazowska, E.D., Zahorjan, J., Cheriton, D.R., and Zwaenepoel, W., (1984) "File Access Performance of Diskless Workstations," Technical Report STAN-CS-84-1010, Department of Computer Science, Stanford University, Stanford, CA.

Leblang, D.B. and Chase, R.P., (1984) "Computer-aided Software Engineering in a Distributed Workstation Environment," *SIGPLAN notices*, vol. 19, no. 5, pp. 104-112.

Lee, R.K.S., (1984) "Concurrent Prolog in a Multi-Process Environment," Research report CS-84-16, Department of Computer Science, University of Waterloo, Waterloo, Ontario. M.Sc. Thesis

Levinson, D., Joyce, J., and Hill D., (1984) "Jedi: the Jade Editor for Interaction - a user interface prototyping system for Jade," Presented at Jade Conference on Distributed Software Prototyping, Calgary, Canada. September 21-23

Lomow, G.A. and Unger, B.W., (1982) "The Process View of Simulation in Ada," *Proc. Winter Simulation Conference*, pp. 77-86, San Diego, California. December

Lomow, G.A., (1984) "Distributed Software Prototyping and Simulation," *M.Sc. Thesis*, Department of Computer Science, University of Calgary.

Lomow, G.A. and Unger, B.W., (1984a) "Distributed Software Prototyping in Jade," *CIPS Session 84 Conference*, Calgary, Alberta. May

Lomow, G.A. and Unger, B.W., (1984b) "Distributed Software Prototyping and Simulation in Jade," *INFOR*. Accepted October, 1984.

Malachi, Y. and Owicki, S.S., (1981) "Temporal Specifications of Self-Timed Systems," in *VLSI systems and computations*, ed. H.T. Kung et. al., pp. 203-212, Computer Science Press, Rockville, MD.

Malcolm, M. and Dyment, D., (1984) "Experience Designing the Waterloo Port User Interface," *SIGPC Notes*, vol. 6, no. 2, pp. 168-175.

McCracken, D.L. and Akscyn, R.M., (1984) "Experience with the ZOG human-computer interface system," *IJMMS*, vol. 21, no. 4, pp. 293-310. October

Mierowsky, C., (1984) "Design and Implementation of Flat Concurrent Prolog," Report CS84-21, p. Weizmann Institute of Science. M.Sc. Thesis, November

Model, M.L., (1979) "Monitoring System Behavior In a Complex Computational Environment," *Ph.D. Thesis, Department of Computer Science*, Stanford University, Also available from Xerox Palo Alto Research Center.

Monteiro, L., (1982) "A Horn Clause-like Logic for Specifying Concurrency," in *Proc of the First International Logic Programming Conference*, pp. 1-8, Marseille, France.

Neal, R.M., (1980) "An editor for trees," *M.Sc. Thesis*, Department of Computer Science, University of Calgary.

Nelson, T.H., (1974) *Dream machines -- new freedoms through computer screens,* Ted Nelson, Publisher.

Nishimura, H, Ohno, H, Kawata, T, Shirakawa, I, and Omura, K, (1983) "Links-1 : A Parallel Pipelined Multimicromputer System for Image Creation," *Proc. 10th Symposium on Computer Architecture, SIGARCH*, pp. 387-394.

Pereira, L.M., (1982) "Logic Control with Logic," in *Proc of the First International Logic Programming Conference*, pp. 9-18, Marseille, France.

Pooley, R., Williams, A., and Birtwistle, G.M., (1984) "A Process Based Simulation of X25 Using Demos," *Proc. Conference on Strongly Typed Languages*, San Diego, California.

Porsching, T.A., Murphy, J.H., and Redfield, J.A., (1971) "Stable Numerical Integration of Conservation Equations for Hydraulic Networks," *Nuclear Science and Engineering*, vol. 43, pp. 218-225.

Radhakrishnan, T. and Grossner, C.P., (1985) "Cuenet - a Distributed Computing Facility," *IEEE Transactions on Microcomputers*, vol. 5, no. 1, pp. 42-52. February

Reed, D.A. and Patrick, M.A., (1984) "A Large Model of Asynchronous Iterative Algorithms for Solving Large, Sparse, Linear Systems," *IEEE International Conference on Parallel Processing*, p. 402.

Reid, B.K., (1980) "Scribe: a document specification language and its compiler," PhD thesis, Carnegie-Mellon University.

Reiss, S.P., (1984) "Graphical Program Development With PECAN Program Development Systems," *SIGPLAN notices*, vol. 19, no. 5, pp. 30-41.

Rochkind, M.J., (1975) "The Source Code Control System," *IEEE Transactions on Software Engineering*, vol. SE-1, pp. 364-370. December

Schiffenbauer, R.D., (1981) "Interactive Debugging in a Distributed Computational Environment," *Ph.D Thesis, Laboratory for Computer Science*, vol. MIT/LCS/TR-264, MIT.

Seattle, (1983) "Graphical Input Interaction Technique Workshop," *Computer Graphics*, vol. 17, no. 1.

Seeheim, (1984) *Seeheim Workshop on User Interface Management Systems,* EUROGRAPHICS-Springer Series.

Seitz, C.L., (1985) "The Cosmic Cube," *Comm. ACM*, vol. 28, no. 1. January

Shapiro, E.Y., (1983) "A Subset of Concurrent Prolog and its Interpreter," Technical Report TR-003, Tokyo, Japan, ICOT - Institute for New Generation Computer Technolgy.

Snodgrass, R.T., (1982) "Monitoring Distributed Systems: a relational approach," Ph.D. Thesis, Carnegie-Mellon University, Pittsburgh, PA.

Spector, A.Z., Butcher, J., Daniels, D.S., Duchamp, D.J., Eppinger, J.L., Fineman, C.E., Heddaya, A., and Schwarz, P.M., (1984) "Support for Distributed Transactions in the TABS Prototype," Technical Report CMU-CS-84-132, Carnegie-Mellon University, Pittsburgh, Penn.

Stonebraker, M., Wong, E., and Kreps, P., (1976) "The Design and Implementation of INGRES," *ACM Transactions on Database Systems*, vol. 1, no. 3. September

Tam, Chun Man, (1984) "The Design of a Distributed Interpreter for Concurrent Prolog," Master's Thesis, Department of Computer Science, The University of British Columbia. Technical report 84-18

Teitelbaum, T. and Reps, T., (1984) "The Cornell Program Synthesizer: a syntax-directed programming environment," in *Interactive Programming Environments*, ed. D.R. Barstow, H.E. Strobe and E. Sandewall, pp. 97-115, McGraw-Hill, New York.

Teitelman, W. and Masinter, L., (1984) "The Interlisp Programming Environment," in *Interactive Programming Environments*, ed. D.R. Barstow, H.E. Strobe and E. Sandewall, pp. 83-95, McGraw-Hill, New York.

Tichy, W.F., (1982) "Design, Implementation and Evaluation of a Revision Control System," *6th International Conference on Software Engineering.*

Tokuda, H. and Manning, E.G., (1983) "An interprocess communication model for a distributed software testbed," *Proc ACM SIGCOMM 83*, University of Texas at Austin. March

Unger, B.W. and Bidulock, D. S., (1982) "The Design and Simulation of a Multicomputer Network Message Processor," *Computer Networks*, vol. 6, no. 4, pp. 263-277. September

Unger, B.W., Bidulock, D., Birtwistle, G., Cleary, J., Colijn, A., Hill, D., Keenan, T., Parker, J., Rokne, J., Vasudevan, R., Witten, I., and Wyvill, B., (1982a) "Proposal for an Environment and Computer Network for Distributed Software Development," Research Report No. 82/92/11, Department of Computer Science, University of Calgary. May

Unger, B.W., Bidulock, D., Lomow, G.A., Belanger, P., Hawkins, C., and Jain, N., (1982b) "An Oasis Simulation of the ZNET Microcomputer Network," *IEEE Micro*, vol. 2, no. 3. August

Unger, B.W., Lomow, G.A., and Birtwistle, G.M., (1984a) *Simulation Software and Ada,* SCS Research Monograph, San Diego, California.

Unger, B.W., Lomow, G.A., and Andrews, K., (1985) "A Process Oriented Distributed Simulation Package," *SCS Conference on Distributed Simulation*, San Diego, California. January

Ward, S.L., (1979) "Simulation and Real-Time System Design: An Integrated Approach," *Ph.D. Thesis, Department of Computer Science*, Northwestern University.

Weyer, S.A., (1982) "Searching for information in a dynamic book," PhD Thesis, School of Education, Stanford University. (Also Report SCG-82-1, Xerox Parc)

Witten, I.H., Bonham, M., and Strong, E., (1982) "On the power of traps and diversions in a document preparation language," *Software -- Practice and Experience*, vol. 12, pp. 1119-1131.

Witten, I.H., Birtwistle, G., Cleary, J.G, Hill, D., Levinson, D., Lomow, G.A., Neal, R., Peterson, M., Unger, B.W., and Wyvill, B., (1983) "Jade: A Distributed Software Prototyping Environment," *ACM Operating Systems Review*, vol. 17, no. 3. July

Witten, I.H., Cleary, J., and Greenberg, S., (1984) "On Frequency Based Menu-Splitting Algorithms," *International Journal of Man-Machine Studies*, vol. 21, no. 2, pp. 135-148. August

Witten, I.H., (1984) "Dynamic documents," *Proc PROTEXT I -- First International Conference on Text Processing Systems*, Boole Press, Dublin, Ireland. October

Witten, I.H. and Bramwell, B., (1985) "A system for interactive viewing of structured documents," *Comm ACM*, vol. 28, no. 3, pp. 280-288. March

Wyvill, B.L.M., Neal, R., , Levinson, D., and Bramwell, R., (1984) "JAGGIES -- a Distributed Hierarchical Graphics System," *Proc. Canadian Information Processing Society Session '84*, pp. 214-217, Calgary, Alberta. May

Wyvill, B.L.M., Liblong, B., and Hutchinson, N., (1984) "Using Recursion to Describe Polygonal Surfaces.," *Proc. Graphics Interface 84, Ottawa.* June

Request for Supporting Equipment

CONTENTS

Request for Supporting Equipment

Introduction

This equipment is requested to support the Strategic Grant Proposal, *An Environment for the Development of Distributed Systems.* The major thrust of that project is the construction of an integrated environment for developing distributed systems. The major parts of this research are: high quality interactive interfaces, including graphical displays and animation; construction of distributed systems including local area networks, a multiprocessor mesh, and VLSI co-processors; languages and tools for distributed systems including their specification, proofs of correctness, prototyping and simulation, final code and VLSI specification.

It is intended that the project be centered around a network of medium to high powered work stations. This has been chosen over a single central computer because of the need for highly interactive and responsive man-machine interfaces and in order to support distributed systems running across a number of work stations. The components of the network will be: a dual processor VAX 11/780 (currently in use); 16 Corvus Concept workstations (to be upgraded from existing equipment); 3 Cadline Suns (currently in use); and 10 Sun 2 workstations (to be purchased).

Experience with the Jade project has shown that for an environment to be used and exercised it is essential that there be enough workstations that a "critical mass" of users has access to the environment. A significant part of the success of Jade is due to the availability of a work station for each individual employed by the project, to involved faculty members and for a number of graduate students. As well the environment has been used by the MAWD project and by senior undergraduate students (using departmental equipment). We have been careful in proposing the equipment below to ensure that there are no single pieces of equipment that will languish because of lack of users and software support. The total number of users (combining staff requested on this and the strategic grant) and current numbers of graduate students is: project staff — 4.5 programmers/technicians; other staff — 3; involved faculty and researchers — 13; visiting scholars — 4; supported graduate students — 6; other involved graduate students — 15; for a total of 45.5 people. These will have a 10 high level workstations (Sun/2) and 19 low level workstations (16 Corvus Concepts and 3 Cadline Suns) for a total of 29 workstations. Given that some users (such as the project staff) will have workstations dedicated to them this seems a very reasonable assignment of equipment. As well it is expected that groups of workstations will be needed simultaneously to support testing and debugging of distributed systems.

The following sections describe the equipment requested as part of the current grant. The appendix describes currently available research equipment, its utilisation, and its relationship to the requested equipment.

Equipment Requested

1. Corvus Concept Workstation Upgrade

It is proposed to upgrade the 16 Corvus Concepts at a cost of $2000 each. This will give each a total of 1Mbyte of memory, which will leave about 700Kbytes available for applications. This will greatly expand the range of applications and prototyping possible and allow significant Prolog and Simula programs to be run locally. Also a floating point processor will be added. This along with the memory increase will provide a very significant improvement in the quality and speed of graphics.

Other advantages of this upgrade path are that: it requires little or no software changes; it will provide immediate benefits to existing programs; and it will allow further significant offloading of the VAX 11/780.

2. Sun Workstations

The major portion of the distributed network will be 10 Sun 2 workstations connected via Ethernet. As the project develops these will form its major computational resource relieving the current VAX 11/780. Each workstation is configured with a high resolution display (1100x920), 4Mbytes of memory, a floating point processor and an ethernet connection. Many of the envisaged interactive tasks require good response time and much floating point intensive graphics. The configuration should be sufficient for such demands.

A gateway between the ethernet and the VAX 11/780 will provide access to other departmental and ACS facilities (for example laser printer and teaching VAXes). The configuration also allows for two independent ethernets with a gateway via one of the Sun 2/170s. This will improve paging response on the network and also allow experimentation on low level protocols on one net while the other continues in production status. Disk space will be provided by three 380Mbyte disks. This provides redundancy in case of hardware failure as well as a good balance between low cost disk storage and response time. As well the VAX 11/780 will be available as a remote fileserver. Three of the workstations will be equipped with high resolution colour displays. These will provide support for VLSI work as well as experiments in more sophisticated graphics and interactive systems.

An important reason for choosing Sun workstations was the ease with which the current Jade environment can be ported to them. This is important to allow maximum progress on new research during the project. The Suns run the same Unix 4.2 as the current VAX 11/780, so the device drivers for intramachine communication via Jipc can be ported directly. Intermachine communication with Jipc uses TCP/IP which is supported on the Suns. The major task would be porting the Jaggies graphics system. This will require rewriting the lowest level display routines. This is estimated to be less than one man month of effort.

3. Mesh Computer

Distributed systems encompass many scales and styles of distribution and communication. The proposed strategic project will address a number of these: interaction within a single processor; interaction between autonomous computers via local area networks (VAX 11/780s and Sun workstations via Pronet and Ethernet); and interaction with and between VLSI components. Another important class of distributed systems are those where a large number of similar processors are connected via high speed links to form a single large computational resource. Current trends in microprocessor technology imply that such systems will supply very cost effective computing as well as high computing power. An important research problem is how to program and effectively use the power inherent in such systems. It is felt that it is very important for the proposed distributed programming environment to address the problems in such systems. The hardware is requested so that significant problems in the areas of graphics (Wyvill), concurrent logic programming (Cleary), discrete event simulation (Unger, Cleary) and the solution of linear equations (Trofimenkoff) can be addressed in a realistic setting. A sufficiently large system is needed to ensure that actual performance measurements can be made and realistic problems solved in the problem domains.

A Mesh has been running since 1984 with four processors (soon to be expanded to nine). The system was designed and built from the board level up. It is distinguished by its simplicity and high bandwidth between processors. This is achieved by using dual ported memories as the communication medium and restricting communication to nearest neighbours. A number of distributed graphics algorithms have already been developed for the architecture.

The proposed mesh will be identical in its basic architecture to the current system. Each processor will occupy one board and will have 2Mbytes of local memory (up from 128Kbytes in the prototype). It will have a 68000 processor and 68081 floating point co-processor as well as four dual ported memories, one connected to each of its four nearest neighbours. An 8x8 mesh of such nodes will be connected in a full torus, with one or more nodes connected to an Ethernet for external communication. All construction will be done with locally designed boards. The current prototype has cost $700 per node for parts. The proposed nodes will be $2000 or less per node (the main uncertainty is the cost of memory chips). Because only one board needs designing and testing and because the interconnection wiring between boards is very simple and regular one year of technician time will be sufficient to complete the design and construction of the system. It will be readily accessible to all users of the Jade environment via Jipe and the Ethernet network.

4. High Quality Printer

A significant emphasis of the strategic grant request is on the provision of high quality documentation for users. This will be in a variety of interactive and non-interactive forms including graphics and good quality typesetting. To support this it is necessary to have good quality hard copy. To this end a laser printer has been requested. It will provide the ability to produce typeset documentation as well as graphics. It will be used by all members of the project for producing documentation and by Witten for specific research on typesetting algorithms.

An existing laser printer is available via Academic Computing Services. This however has very high paper costs and because it is a production system has proved very difficult to use for experiments in typesetting. The proposed Apple laser printer is very flexible and has available a very wide range of fonts not found on the current Imagen system.

Maintenance

Maintenance of the current VAX 11/780 is being covered from University funding. Maintenance of the Corvus and Sun workstations will be carried out locally wherever possible, and provision has been made for purchasing extra boards to help with this. More significant problems will be serviced on a time and cost basis. This is estimated as 3% of the total hardware cost. 687,400 = $20,600). This will be covered by a levy on personal operating grants (10 of the applicants currently receive such grants).

The recent Infrastructure grant will be used to cover support of both Unix and the Jade/Jipe environment on all the departments research computers. Approximately one half person will be available for maintaining the software on the Sun and Corvus workstations. The department is also prepared to provide one half of a technician for hardware maintenance.

Organisation

Access priority to all facilities will be set by the principal investigator of the strategic grant in consultation with a three member steering committee chosen from the three major research subgroups: Birtwistle (VLSI); Hill (human computer interaction); and Unger (distributed systems and prototyping).

Canadian Availability

The workstations and Corvus upgrade are not available in Canada. Most of the chips for the Mesh computer are not available locally, however, miscellaneous supplies and board manufacturing will be done locally.

Appendix A: Present Equipment

1. VAX 11/780

The VAX 11/780 (vaxb) has been purchased partly from departmental funds (approximately 1/2) and partly from a previous Major Equipment Grant (approximately 1/2). The current configuration is a dual processor (in the "Purdue" configuration) with 960Mbytes of disk, and 14Mbytes of main memory. It is connected via a 10Mbit/sec network (Pronet) to the department's teaching computer (vaxa) and to two systems maintained by Academic Computer Services (vaxc, vaxd). These last three are teaching computers available for research only out of term time. However, the Jipc message passing system has been ported to them and all four machines have been used together for testing distributed systems.

Vaxb currently supports the bulk of departmental research. Some 107 people have accounts on the machine including 37 graduate students 20 faculty and 10 full or part time employees of the current Jade strategic grant. It is used as the main machine for programming and documentation as well as computationally intensive tasks such as rendering high quality graphics and proving VLSI specifications correct. The result is a heavily overloaded machine where the number of users logged in regularly exceeds 30 and the load factor (number of jobs waiting to run) often exceeds 10. Under these conditions line editing and character echoing are noticeably slow and more sophisticated interaction is effectively impossible.

The proposal assumes that the current VAX 11/780 will be retained to act as a gateway machine and a source of background machine cycles. The majority of intensively interactive tasks will be handled by workstations. The VAX will form a gateway between different local area networks and to the teaching machines as well as acting as a fileserver.

It is felt that the current configuration is well balanced and consequently no upgrade for it is being requested.

2. Cadline Suns

Three Cadline Suns are currently running the Jade kernel and look functionally identical to the Corvus Workstations. Because of non-delivery of software sources by Cadline and hardware problems with the disks the original software was dispensed with. These machines will be hardware compatible with the proposed Sun workstations.

3. Corvus Concept Workstations

16 Corvus Concept workstations are currently available for research (5 have been purchased as part of the Jade project and 11 from personal operating grants. An additional 16 have been purchased by the department for senior undergraduate teaching). These are used as workstations into the VAX 11/780s via a window system constructed as part of Jade. Most faculty heavily involved in Jade, the Jade staff and some graduate students use them for the bulk of their interactive work. As well the workstations can run downloaded programs written in 'C', (and shortly Simula and Prolog) which communicate via Jipc. In this role they have been used to support the prototyping and execution of a number of distributed systems and applications such as graphics. The major restrictions on their use are a lack of memory and floating point support. After loading support software and allowing for working space for the window manager and the bit-mapped graphics screen about 200Kbytes are available for applications programs (from a total of 512Kbytes).

Appendix B: Summary of Equipment Requested

The equipment requested, together with associated items, is summarised in the table below. An allowance for maintenance, estimated at 3% of hardware costs per annum, is also being requested.

1. Network of Sun 2 workstations, each configured with floating point processor and 4Mbytes of memory and bit mapped high resolution (1100x920) display. This will consist of:

   - Five Sun 2/50 workstations
   - Three Sun 2/160 workstations
   - Two Sun 2/170 workstations/fileservers
   - Three 380Mbyte disk drives and ethernet gateway for fileservers
   - 1/4inch tape drive (for software distribution)
   - Freight charges
   - Spare Sun boards

2. 10Mbit/sec Ethernet connection for VAX 11/780

3. Upgrade of 16 Corvus Concept Workstations

4. Duty on above hardware (excluding disk drives)

5. Prolog compiler and interpreter for Sun workstations

6. Unix source license for Sun workstations

7. Apple laser printer

8. Mesh computer, 64 single board nodes, Ethernet connection, card cages, backplane, board design.

9. Technician for construction of mesh computer (1 year)

10. Miscellaneous software

The Jade Status Report for April, 1985

CONTENTS

The Jade Status Report for April, 1985

Introduction

The University of Calgary's Project Jade is a research project directed towards distributed systems which grew out of collaborative efforts among faculty members and graduate students in the Computer Science department. Several research proposals were prepared during 1982 resulting in four NSERC awards: Major Equipment and Infrastructure grants awarded in June 1982 and Strategic Equipment and Operating grants awarded in November 1982.

The project's primary goals are to pursue research in distributed systems and to construct a programming environment that supports the development of distributed software. The Jade Environment integrates a set of software development tools for specifying, designing, implementing, testing, and evaluating the performance of distributed computer systems.

The first section of this status report outlines the major achievements of the project. Next the extent to which the original objectives have been met are summarised along with a description of changes in the direction of research. The third section discusses the significance of project achievements and current activities involving the application of these results both within the University of Calgary, and by external organisations. Finally, the last section outlines future plans and how they relate to the existing Jade Environment.

Graduate students and theses supervised by Jade co-investigators are listed in Appendix A. Research publications by co-investigators, students, and the Jade research staff are listed in Appendix B.

Major Achievements

Significant accomplishments have been made in a number of research areas, and in the development of systems and tools. These achievements and the key related publications are:

- a multilingual distributed programming environment, including an interprocess communication protocol, a multi-process workstation kernel, a device-independent graphics system and a graphics editor [62, 66, 94]

- new techniques for monitoring distributed systems, including textual and graphical display, deadlock detection, system control and statistical display [43, 52, 95]

- progress toward a prototyping and simulation tool [12, 41, 51, 60, 67]

- distributed Prolog and Prolog animation tools [38, 39, 40, 88]

- a *mesh computer* consisting of a 2D array of computing nodes [37, 87]

- progress toward VLSI specification and proof tools, including the largest ever specification of a practical VLSI design, participation in the design and implementation of two of Canada's largest chips, and the design and prototyping of a VLSI design capture language [11, 42, 45, 57, 58, 83]

- user interface tools and high performance graphics [28, 29, 33, 49, 79, 80, 92, 93]

These achievements are outlined in the following paragraphs.

A Multilingual Programming Environment

The Jade/2 Environment was released in November 1984. A significant part of this release was the publication of the Jade User's Manual [94]. This distributed programming environment supports the development and execution of multilingual distributed programs. The environment is implemented on 4 Vax/Berkeley Unix 4.2 systems connected via a 10Mbs token passing ring (Pronet) and 25 Corvus Concept M68000 based workstations connected via a 1Mbs ethernet (Omninet). Two of the Vaxes form gateways to 2 Omninets of Concepts.

This second version of the Jade Environment supports the development and execution of programs whose individual components can be implemented in Ada, C, Lisp, Prolog, or Simula. Processes written in these languages communicate via a message-passing kernel called "Jipc" (pronounced gypsy). This is accomplished with language interfaces to Jipc for each of the 5 supported programming languages.

Processes implemented in C can reside on any of the workstations or Vaxes, while processes implemented in the other languages are currently restricted to the four Vaxes. Cross compilation and downloading of Ada, Prolog, and Simula components to the workstations is currently under development. The Jipc kernel has also been ported to Cadlink Sun and MTU workstations. Porting of the kernel to a 2x2 array of M68000 based computing nodes, called the "Mesh Computer" is partially complete.

The Jade/2 Environment includes tools that support: monitoring the execution of distributed systems, the creation of 2D graphical pictures that can be animated by executing programs, a window based interactive user interface, and on-line documentation browsing facilities. This environment is unique in its support of multi-lingual distributed system development, and in particular, its powerful tools for monitoring distributed systems. The environment and its major components are described in [62, 66, 80, 94].

Distributed Monitoring Techniques

This work addresses the problems of non-determinism in distributed systems and the unavoidable impact on a distributed system's behaviour caused by monitoring. An extensible monitoring scheme has been defined which collects process interaction information from a multi-lingual distributed application program. This information is sent via unmonitored Jipc messages to one or more "consoles" which interact with the "user", ie. the developer of the application program. Different consoles provide different interpretations, or views, of an executing distributed application program.

A number of different kinds of consoles have been explored including: a basic one line of text per inter-process event view; an animation sequence with icons representing processes and messages moving among these processes; a deadlock detector; a representation of traffic density among processes; multi-level monitoring via a hierarchical protocol specification (eg. Jipc messages at the lowest level, data base transactions which involve a number of different patterns of message interactions, and user/data base sessions which involve patterns of transactions); and a protocol verifier which reports only deviations from a protocol specification.

The basic monitor provides a breakpoint facility and enables collecting interaction histories which can be replayed by controlling non-deterministic choices. The user can also control non-determinism to force improbable execution paths and to support automated testing.

These ideas have been described in [43, 51, 52, 95]. The extensible monitoring scheme has been implemented in Jade/2 along with several consoles. One visiting scientist (Dong Zhixin) is working in this area and two M.Sc. (A.Goh, Schack) and one Ph.D. project (Lomow) are in progress.

Distributed System Prototyping and Simulation

Our prototyping work is directed at tools which support the implementation, testing, and evaluation of distributed programs whose components interact via Jipc message passing. An application can be developed within Jade/2 and then ported to a target network of computers which may be an embedded system. An implementation of Jipc for the target network is assumed, although this implementation may be optimised for that target hardware configuration.

The technique involves the use of simulation models for devices which exist in the target network but not in Jade/2. An application distributed program can be completely, or partially, implemented and then its performance evaluated via simulation. Alternative peripheral devices and the target computer network architecture can also be evaluated.

An initial design and implementation of this prototyping tool is described in [12, 60] and in Lomow's M.Sc. thesis. This work builds on previous computer system and software simulation approaches presented in [22, 23]. Although these results towards a general prototyping tool for distributed systems are still preliminary, we are making exciting progress on a number of related fronts, see [1, 2, 3, 31, 32, 38, 50, 59, 64, 67, 114]. Several theses projects are involved in this work conducted by students (Lomow, A.Goh) and visiting scientists Xiao Zhong-e and Li Xining.

Distributed Prolog

A distributed version of the and-parallel portion of Shapiro's Concurrent Prolog (CP) [Shapiro, 1983] has been designed and implemented. This implementation manages the distribution of logical variable bindings among local implementations of CP and is based on Jipc.

A discrete event simulation system has been implemented in Concurrent Prolog [40]. Previous work on discrete event simulation was reported in [38]. This work builds on the experience of Futo's seminal work [Futo, 1980 & 1982] in the field of simulation and logic programming. Other closely related work includes [Broda, 1984].

The integration of Prolog into Jade/2 has enabled the implementation of a number of novel systems. These include a logic programming language for 2D graphics (GROWL) and a distributed implementation of it. GROWL has been implemented using the Jade graphics system [80] and has been used to construct a graphical debugging system for Prolog programs. In it GROWL is used to draw an and/or representation of the executing program and its data structures. This interactive tool provides a very powerful environment for debugging standard Prolog programs. The distributed implementation of GROWL includes a novel and powerful scheme for reducing the communication between the logic intensive Prolog and remote graphics processors.

Cleary and Kornfein have developed a set of rules for proving partial and, more significantly, complete correctness of Horn clause programs. Other work by Cleary and Rokne has provided a very natural integration of real interval arithmetic into Prolog. Two M.Sc. theses (Dewar and Kramer) have been completed and two are in progress (Kornfein and Yen).

Mesh Computer

A project was begun in 1983 to construct a multi-processor system. Since then the system has been designed from scratch including inter-processor communication and board design. Since late 1984 a four processor system using M68000 processors has been running. The system is easily scaled to much larger sizes because of the two dimensional layout of processors where only neighbouring processors can communicate. This enables a very simple and regular layout of processors and backplane wiring while permitting very high speed interprocessor communication.

The communication between processes is via nearest neighbour shared memories which enable transfer rates of 2Mbs per communication link without involving the receiving processor in any overhead. The Jade/2 Jipc kernel has been ported to the system and used to develop programs for it. The standard technique is to develop and simulate algorithms on Jade/2 and when debugged to cross-compile and run them on the mesh computer. This has been used for a number of test programs and for a distributed version of a ray-tracing program for high quality graphics [37, 87]. One M.Sc. thesis has been completed in this area (Vatti) and one other is in process (Pearce). Work is currently proceeding on a 9 processor configuration and on a design for a successor system with increased memory and integral access to a graphics frame buffer. A technician and programmer (Asaph and Vatti) are currently involved in the mesh computer implementation.

VLSI Specification & Proof

Significant achievements have been made since the formation of our Very Large Scale Integration (VLSI) research group in the Spring of 1983. A Simula based layout package called LAP has been adapted to the Northern Telecom NMOS and CMOS processes. This package was completed in 1983 [57] and has been used as the basis for additional local software packages [45] and a VLSI context [42] – a framework of several pads used to surround medium sized designs, and components for digital filter chips.

We are currently using Electric [Rubin, 1983], a first generation VLSI design tool, to design a cell library. Local extensions to Electric include: a Prolog and theorem proving interface, a prototype design system (EDICT) [83] targeted at Weinberger array layout by Liu, and a front end for a digital filter silicon compiler by Krocker. Additionally, Schediwy has been invited to the Schlumberger Palo Alto Research Center (SPAR) for Summer 1985 to complete the Electric cell library, work arising from his thesis research.

In order to verify one of our larger designs (an associative memory for a local area network box [9] we have used LCF-LSM, a LISP based theorem prover developed at Cambridge University (UK). This project represents the largest specification and verification of a practical hardware design ever completed. Another tool developed is SHIFT, a design capture language. This is a fully working system, implemented in LISP [11, 57]. SHIFT is the only known Canadian initiative; it strongly resembles the recently proposed US standard, EDIF and a number of American firms have expressed interest in our work. An additional noteworthy accomplishment is Coates' recent submission of Canada's largest prototype to fabrication in March, 1985 (150,000 transistors) at SPAR, as part of the Faim project [Davis, 1979 & 1982].

The Jade multilingual programming environment has been a positive factor in our work as the network enabled use of Simula for layout and simulation, C for dialogues, Lisp for SHIFT and hardware verification, and Prolog to develop a random logic and silicon compiler.

One M.Sc. thesis (Liblong) is finished, six more M.Sc. projects are currently underway (Coates, Krocker, Keefe, Ling, O'Byrne, and Schediwy), and two Ph.D. dissertations (Joyce, Liu) are underway.

Human Computer Interfaces

Research results on human interfaces has strongly influenced the design of software development tools outlined above. Achievements in this area also include results in interactive dialogue systems [48], dynamic and distributed documentation tools, the analysis of user interactions with computer systems, workbench-style interfaces, a multimedia adaptive workstation, and 3D high resolution colour graphics.

Innovative systems have been created for browsing through on-line documents that involve text, graphics and animated interactive displays. An interactive documentation tool which provides a highly flexible interface to an on-line reader/browser, with the ability to deal with linked streams of text (footnotes,

figures, and annotations by a variety of people as well as the main text) is implemented as a distributed system within Jade/2 [34, 35]. An on-line Unix browser, which was implemented as part of Bramwell's M.Sc. thesis, now encompasses all Jade software, software documentation, and project information. [28, 36, 86, 103]. Two M.Sc. theses have been completed (Bramwell, Bonham).

One focus for research which underlies several projects is the recording and automatic analysis of user interactions with computer systems. Novel techniques for identifying structural models from behaviour sequences, and using the resulting models for prediction, have been been applied to a variety of problems. One application is the automatic personalisation of a user interface [27, 46, 75, 91] which is also the topic of one completed M.Sc. thesis (Greenberg) and one Ph.D. dissertation (Greenberg) which is in progress. Another application is modeling users' typing behaviour to expedite text entry [70, 71, 73, 89, 101]. A third is text compression; extremely impressive compression ratios have been achieved (2.2 bit/char on English text drawn from a 94-character alphabet, with no prior knowledge of statistics) [4, 74].

The Multimedia Adaptive Workstation for the Disabled (MAWD) project is intended to improve access to computer power for disabled university students and represents a major Jade/2 application. It provides highly usable and innovative interfaces based on touch, speech and graphics that have been built upon Jade/2. One M.Sc. has been completed (Dohrn).

Research results in high performance graphics include the design and implementation of a 3D hierarchical graphics system [33, 78, 79, 81, 82]. At each level objects are defined as geometrical transformations of sub-objects, and may include recursive references. Objects are not limited to a single data type and may refer to a variety of different leaf nodes representing primitives (polygons, particles, fractals). The objects are rendered in one pass facilitating the manufacture of animated film. So far about 200 seconds of film have been made demonstrating these techniques [117]. The project also involves research into distributed rendering algorithms implemented on Jade/2 using Jipc (Warnocks, Z-Buffer and ray tracing). One M.Sc. thesis has been completed in this area (Vatti) and two others are in process (Novacek, Pearce).

Original Goals Achieved & Changes in Direction

All of our original goals have been achieved except that the environment supports only one class of workstation instead of three, and the software prototyping tool and voice I/O have not been integrated into the environment.

The only class of workstation now supported is an M68000 based computer with random access memory, monochrome bit mapped display, and Omninet or Pronet Interface. We originally intended to support standard CRT terminals (eg. VT100 class), however it was decided early in the project that interactive bit-mapped graphics must be an integral part of the user interface to the environment. Standard terminals are clearly supported on our Vax/Unix systems and some of the Jade/2 tools can be accessed. However, no effort has been made, nor is planned, to provide an integrated access to Jade tools via simple terminals.

We had planned, and still hope to support, a higher performance workstation with colour graphics and local off-line storage. The SMI Sun running Unix 4.2 is an attractive candidate which is being explored in conjunction with the Naval Research Laboratory (NRL) in Washington, D.C. (see below in Section IV). Porting to a Unix workstation is a relatively straightforward task. Other candidate workstations include Lisp machines.

The preliminary software prototyping tool [12] has not been integrated into the environment because this effort was diverted by the the closely related distributed system monitoring work. This monitoring work proved to be very interesting and fruitful, and was also discovered to be essential for a general prototyping scheme. Our approach will be to design a new software prototyping and simulation tool based on our monitoring and distributed simulation work.

The work on voice I/O [47] has been primarily associated with the multimedia adaptive workstation effort and has not reached the stage where it can be properly integrated into the Jade/2 production environment.

The "Mesh Computer" work represents a research direction that evolved during the projects first year. This multiprocessor machine has proved to be an important application where distributed software developed, tested, and evaluated within Jade/2 can be ported to this target system for further testing, evaluation, and we hope production computing. This machine has offered very interesting opportunities for exploring distributed 3D graphics algorithms, distributed simulation applications, and hardware support of distributed Concurrent Prolog. Finally, the VLSI specification and proof area constitute another "new" research direction. This work, in the area of hierarchical specification of VLSI designs has proved to be relevant to specifying distributed systems.

Significance of Achievements

The significance of project achievements is reflected in more than 81 refereed journal and conference papers and in the successful use being made of Jade/2. We now have over 100 faculty, staff and student users within the CPSC department at the University of Calgary. Several external organisations are either using the environment or plan to use it.

The Naval Research Laboratory (NRL), Washington D.C. USA, has approached us to acquire internal licences for use of parts of Jade/2. Both the Information Technology Division (ITD) and the Navy Center for Applied Research in Artificial Intelligence will be using Jade/2. NRL has provided $10,000 for installation. A $50,000 collaborative project is being negotiated that involves porting Jade/2 to SMI Sun workstations, and our current distributed simulation work based on Time Warp [Jefferson, 1983].

Corvus Systems has also approached us to acquire the Jipc kernel and window system in exchange for equipment and/or funding. Novatel, Calgary is planning to use Jade/2 to develop a prototype switch that consists of a number of M68000 computing nodes. This kind of target system is a natural application for Jade tools, and Novatel hopes to substantially reduce software development time.

The Jade monitoring and animation facilities that can be applied to processes implemented in a mixture of procedural and applicative languages is unique in North America. The "Port" project (University of Waterloo), "Eden" (University of Washington), as well as, the earlier Unix, Interlisp, and Ada Programming support environments are primarily built around a single programming language.

The ability to mix procedural and applicative languages has a number of advantages. Often certain components of a distributed application program are best implemented in an applicative/interpreted language, such as Lisp or Prolog. This is particularly true of interactive human interface components. Other components, however, are often best implemented in procedural languages such as Ada or C, particularly for embedded applications.

Future Plans

Our current work is directed at: multi-level monitoring with protocol verification support; distributed simulation via a Time Warp mechanism based on the Jipc synchronous protocol; a software prototyping and simulation scheme that builds on this monitoring and distributed simulation work; a specification language that spans the specification, prototyping, and programming phases of software development; and specification tools for VLSI designs.

These objectives and research plans build on past achievements, and support and lead directly into the project defined in the Strategic Proposal, University of Calgary, April 1985.

References

Broda, P.K. and Gregory, S., (1984) "Prolog for Discrete Event Simulation," Research Report DOC 84/5, Department of Computing, University of London. March

Davis, R., (1979) "Interactive transfer of expertise: acquisition of new inference rules," *Artificial Intelligence*, vol. 12, no. 2, pp. 121-157.

Davis, R. and Lenat, D.B., (1982) *Knowledge-based systems in artificial intelligence,* McGraw Hill, New York.

Futo, I. and Szeredi, J., (1980) *T-Prolog user's manual,* Institute for Coordination of Computer Techniques, Budapest.

Futo, I. and Gergely, T., (1982) "A logical approach to simulation." *Proc. of International Conference on Model-Realism*, Springer Verlag, Bad-Honnef, GFR. April

Jefferson, D. and Sowizral, H., (1983) "Fast Concurrent Simulation Using the Time Warp Mechanism, Part I and II: Global Control," Technical Report, The Rand Corporation, Santa Monica, California. August

Rubin, S.M., (1983) "An Integrated Aid for Top-Down Electrical Design," *VLSI '83, Anceau and Aas, eds.*, pp. 63-72, North Holland Publishing Co.,, Amsterdam.

Shapiro, E.Y. and Takeuchi, A., (1983) "Object oriented programming in Concurrent Prolog," *New Generation Computing*, vol. 1, pp. 25-48.

Shapiro, E.Y., (1983) "A subset of concurrent Prolog and its interpreter," ICOT Technical Report TR-003. January

Shapiro, E.Y., (1983) "Lecture notes on the Bagel: a systolic Concurrent Prolog machine," Technical Report TM-0031, ICOT - Institute for New Generation Computer Technolgy. November

Appendix A:
Graduate Student Theses During The Current Award Period

THESES COMPLETED

Name	Full Titles	Supervisor	Completion Date
Chelani, A.	AMOK A Message Oriented Kernel	R. Vasudevan	1982
Levinson, D.	The Well-Tempered Speech Recogniser (Ph.D.)	D. Hill	1982
Ang, T.	Local Area Communications	G. Birtwistle	1983
Bramwell, B.	An Automatic Manual	M. Williams	1983
Brookwell, B.	Representation and Display of Selected Surfaces Using Texture Mapping	D. Hill	1983
Girling, D.	Well-Tempered Speech Synthesizer	D. Hill	1983
Barker, K.	Local Area Network Security	T. Keenan	1984
Dohrn, C.	Speech Pad: Direct Manipulation Computer Access for the Visually Disabled Based on Speech and Touch	D. Hill	1984
Greenberg, S.	User Modeling in Interactive Computer Systems	I. Witten	1984
Liblong, B.	A Structured Hierarchical Intermediate Form for VLSI Design	G. Birtwistle	1984
Lomow, G.	Distributed Software Prototyping and Simulation	B. Unger	1984
Vatti, R.	Multiprocessor Ray-Tracing	D. Hill	1984
Bonham, M.	Viewing and Formatting Documents On-line	I. Witten	1985
Dewar, A.	A Graphical Prolog Debugger	J. Cleary	1985
Kramer, L.	Knowledge Representation in Expert Systems	J. Cleary	1985

THESES IN PROGRESS

Name	Full Titles	Supervisor	Expected Completion Date
Andrews, K.	An Extensible Prolog Interpreter	H. Baecker	1985
Coates, B.	A Fast Instruction Memory	G. Birtwistle	1985
Darragh, J.	Man-Machine Communication	I. Witten	1985
Du Wors, R.	Human Computer Interaction and Expert Systems	D. Hill	1985
Goh, A.	Distributed System Monitoring	B. Unger	1985
Goh, L.	Human Computer Systems	J. Cleary	1985

Greenberg, S.	Intelligent Adaptive Interfaces (Ph.D.)	I. Witten	1986
Harris, S.	Expert Systems for Oil Well Logging	J. Cleary	1986
Inkster, J.	Distributed Simulation	B. Unger	1985
Irving, G.	Interactive Dialog Design Tools (Ph.D.)	D. Hill	1985
Jansonius, C.	Human Computer Interaction	D. Hill	1986
Joyce, J.	Specification Directed VLSI design (Ph.D.)	G. Birtwistle	1987
Kornfein, R.	Mechanization of Reasoning	J. Cleary	1985
Keefe, M.	An Expert System for Placement and Routing	J. Kendall	1986
Kroeker, W.	A Silicon Compiler for LDI Filters	G. Birtwistle	1985
Ling, R.	An Expert System for Minimization of Boolean Expressions	J. Kendall	1986
Liu, E.	An Expert System for Gate Matrix Layout	G. Birtwistle	1986
Lomow, G.	Distributed Systems (Ph.D.)	B. Unger	1986
Lowdon, B.	Expert Systems for Geophysics Applications	J. Kendall	1987
Lukey, T.	Program Comprehension and Debugging	D. Hill	1985
Masrani, R.	Natural Language Processing in Object Oriented Prolog	I. Witten	1985
Novacek, M.	Particle Graphic Systems	B. Wyvill	1985
O'Byrne, R.	Language for the Parametric Description and Placement of Circuits	J. Kendall	1986
Pearce, A.	Multiprocess Ray-Tracing	J. Cleary	1986
Schack, B.	Multi-Level Monitoring via Hierarchical Protocol Specification	B. Unger	1986
Sharman, D.	Expert Systems for Gas Pipeline Control	J. Kendall	1988
Schediwy, R.	Designing Standard Cells	G. Birtwistle	1985
Wu, X.	A Digital Image: Processing & Coding	I. Witten	1986
Yen, Y.	Expert Systems for Simulation	B. Unger	1986
Zissos, A.	An Expert System for Analyzing Editor Interaction Traces	I. Witten	1985

Appendix B: Publications During The Current Award Period

REFEREED JOURNAL PAPERS

1 Birtwistle, G.M., Lomow, G.A., Unger, B.W., and Luker, P. (1984) "Process Style Packages For Discrete Event Modelling: Using Simula's class Simulation" *Transactions on Simulation, 1* (2) 175-195.

2 Birtwistle, G.M., Lomow, G.A., Unger, B.W., and Luker, P. (1984) "Process Style Packages For Discrete Event Modelling: Data Structures and Packages in Simula" *Transactions on Simulation, 1* (1) 61-82.

3 Birtwistle, G.M., Lomow, G.A., Unger, B.W., and Luker, P. (1985) "Process Style Packages For Discrete Event Modelling: Transaction, event, and activity approaches" *in press Transactions on Simulation.*

4 Cleary, J.G. and Witten, I.H. (1984) "Data compression using adaptive coding and partial string matching" *IEEE Trans Communications, COM-32* (4) 396-402, April.

5 Cleary, J.G. and Witten, I.H. (1984) "A comparison of enumerative and adaptive codes" *IEEE Trans Information Theory, IT-30* (2) 306-315, March.

6 Cleary, J.G. (1984) "Compact hash tables using bidirectional linear probing" *IEEE Transactions on Computers, C-33* (9) 828-834, September.

7 Cleary, J.G. and Darragh, J.J. (1985) "A fast compact representation of trees using hash tables" *in press IEEE Transactions on Computers.*

8 Corbett, C. and Witten, I.H. (1982) "On the inclusion and placement of documentation graphics in computer typesetting" *Computer Journal, 25* (2) 272-277, February.

9 Hutchinson, N., Patten, T., and Unger, B.W. (1985) "The Flooding Sink: A New Approach to Local Area Networking" *in press Computer Networks.*

10 Keenan, T. (1982) "The Computer Media" *Journal of the Alberta Association for Continuing Education, 10* (1) 39-44, May.

11 Liblong, B., Melham, T., Birtwistle, G.M., and Kendall, J. (1985) "Towards a VLSI Design Tool System" *in press Canadian Journal of Operational Research and Information Processing.*

12 Lomow, G.A. and Unger, B.W. (1985) "Distributed Software Prototyping and Simulation in Jade" *in press Canadian Journal of Operational Research and Information Processing.*

13 Rokne, J. and Wu, T. (1982) "The circular complex centered form" *Computing, 28,* 17-30.

14 Rokne, J. (1982) "Optimal computation of the Bernstein algorithm for the bound of an interval polynomial" *Computing, 28,* 239-246.

15 Rokne, J. and Wu, T. (1983) "A note on the circular complex centered form" *Computing, 30,* 201-211.

16 Rokne, J., Singh, B.M, and Dhaliwal, R. S. (1983) "Diffraction of torsional waves by a circular rigid disc at the interface of two bounded dissimilar elastic solids" *Acta Mechanica, 40,* 139-146.

17 Rokne, J. and Alefeld, G. (1984) "On the iteractive improvement of approximate triangular factorization" *Beitraege zur Numerische Mathematik, 12,* 7-19.

18 Rokne, J. (1985) "A low complexity explicit rational centered form" *in press Computing.*

19 Rokne, J. (1985) "Including Iterations for the Lambda-Matrix Eigenproblem" *in press Computing.*

20 Rokne, J., Singh, B.M, and Dhaliwal, R. S. (1985) "Diffraction of torsional waves by a penny-shaped crack in an infinitely long cylinder bonded to an infinite medium" *in press Journal of Engineering Fracture Mechanics.*

21 Rokne, J., Singh, B.M, and Dhaliwal, R. S. (1985) "Diffraction of antiplanar shear waves by two moving Griffith cracks" *in press Applied Science Research.*

22 Unger, B.W. and Bidulock, D. S. (1982) "The design and simulation of a multi-computer network message processor" *Computer Networks, 6* (4) 263-277, September.

23 Unger, B.W., Bidulock, D., Lomow, G.A., Belanger, P., Hawkins, C., and Jain, N. (1982) "An Oasis Simulation of the ZNET Microcomputer Network" *IEEE Micro, 2* (3), August.

24 Witten, I.H., Bonham, M., and Strong, E. (1982) "On the power of traps and diversions in a document preparation language" *Software Practice and Experience, 12,* 1119-1131.

25 Witten, I.H. and Neal, R. (1982) "Using Peano curves for bilevel display of continuous-tone images" *IEEE Computer Graphics, 2* (3) 47-52, May.

26 Witten, I.H. and Wyvill, B. (1983) "On the generation and use of space-filling curves" *Software Practice and Experience, 13,* 519-525.

27 Witten, I.H., Cleary, J., and Greenberg, S. (1984) "On frequency based menu-splitting algorithms" *International Journal of Man-Machine Studies, 21* (2) 135-148, August.

28 Witten, I.H. and Bramwell, B. (1985) "A system for interactive viewing of structured documents" *Communications ACM, 28* (3) 280-288, March.

29 Wyvill, B.L.M., McPheeters, C., and Garbutt, R. (1985, in press) "A practical 3D computer animation system" *Journal of the British Kinematographic, Sound and Television Society.*

REFEREED CONFERENCE PAPERS

30 Barker, K.E. and Keenan, T.P. (1984) "Local Area Network Security" *Proc. Canadian Information Processing Society Session '84,* 489-499, Calgary, Alberta, May.

31 Birtwistle, G., Cleary, J.G., Joyce, J., Liblong, B., Unger, B.W., Witten, I.H., and Wyvill, B.L.M. (1984) "A simulation environment" *Proc. Canadian Information Processing Society Session '84,* Calgary, Alberta, May.

32 Birtwistle, G. and Luker, P. (1984) "Dialogs for Simulation Programming" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

33 Birtwistle, G., Wyvill, B., Levinson, D., and Neal, R. (1984) "Visualizing a Simulation Using Animated Pictures" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

34 Bonham, M. and Witten, I.H. (1982) "A structured procedural document preparation language" *Proc. Canadian Information Processing Society Session '84,* 6-12, Saskatoon, Saskatchewan, May.

35 Bonham, M. and Witten, I.H. (1984) "Towards distributed document preparation with interactive and noninteractive viewing" *Proc. Canadian Information Processing Society Session '84,* 365-372, Calgary, Alberta, May.

36 Bramwell, B. (1984) "Browsing around a manual" *Proc. Canadian Information Processing Society Session '84,* 438-442, Calgary, Alberta, May.

37 Cleary, J.G., Wyvill, B., Birtwistle, G., and Vatti, R. (1983) "Design and analysis of a parallel ray tracing computer" *Proc. Canadian Information Processing Society Graphics Interface '83,* Edmonton, Alberta, May.

38 Cleary, J.G. and Dewar, A. (1984) "Interpreters for Logic Programming - A Powerful Tool for Simulation" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

39 Cleary, J.G. (1985) "A Distributed Implementation of And-parallel Concurrent Prolog" *submitted Proc. IEEE International Conference on Parallel and Distributed Systems,* Calgary, Alberta.

40 Cleary, J.G. Goh, K.S., and Unger, B.W. (1985) "Discrete Event Simulation in Prolog" *Proc. SCS Conference on AI, Graphics, and Simulation,* San Diego, California, January.

41 Cleary, J.G., Lomow, G.A., Unger, B.W., and Xiao, Z. (1985) "Jade's IPC Kernel for Distributed Simulation" *accepted Proc. Association of Simula Users,* Calgary, Alberta, August.

42 Coates, W. and Kendall, J. (1984) "A VLSI Design Context" *VLSI'84 Conference,* Edmonton, Alberta, September.

43 Dewar, A. and Unger, B.W. (1984) "Graphical Tracing and Debugging of Simulations" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

44 Ellis, G.B. and Keenan, T.P. (1982) "Microcomputers, Videotex and Educational Teleconferencing" *Proc. International Conference in the Application of Micro- and Mini-Computers,* Tel Aviv, Israel, March.

45 Esau, R., Kroeker, W., and Birtwistle, G. (1984) "The RLC Silicon Compiler" *VLSI'84 Conference,* Edmonton, Alberta, September.

46 Greenberg, S. and Witten, I.H. (1984) "Comparison of menu displays for ordered lists" *Proc. Canadian Information Processing Society Session '84,* 464-469, Calgary, Alberta, May.

47 Hill, D.R., Dohrn, C., Darragh, J., Esau, R., Levinson, D., Unger, B., and Witten, I.H. (1984) "Using speech output as a medium for human-computer dialogue" *Proc. Canadian Information Processing Society Session '84,* 470-476, Calgary, Alberta, May.

48 Hill, D.R. and Irving, G. (1984) "The Interactive Dialog Driver: a UNIX Tool" *Proc. Canadian Information Processing Society Session '84,* 307-313, Calgary, Alberta, May.

49 Hill, D.R., Witten, I.H., Neal, R., and Lomow, G.A. (1984) "Jeel and Hide: practical questions for the Jade user interface" *Proc. Canadian Information Processing Society Session '84,* 373-380, Calgary, Alberta, May.

50 Inkster, J., Lomow, G.A., and Unger, B.W. (1984) "Combined Continuous and Discrete-Event Simulation in Ada" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

51 Joyce, J., Birtwistle, G.M., and Wyvill, B.L.M. (1984) "ANDES - An Environment for Animated Discrete Event Simulation" *UK Simulation Conference 1984,* Bath, U.K..

52 Joyce, J. and Unger, B.W. (1985) "Graphical Monitoring of Distributed Systems" *SCS Conference on AI, Graphics, and Simulation,* San Diego, California, January.

53 Keenan, T.P. (1981) "A Computer Security Taxonomy for the Resource Industries" *ACM Mountain Regional Conference,* Calgary, Alberta, May.

54 Keenan, T (1982) "Strategic Issues in Distributed Data Processing" *Proc. CIPS National Conference,* Toronto, Ontario, November.

55 Keenan, T.P. and Ferris, K.L. (1983) "Information Security" *CIPS/Department of Justice National Consultation on Computer Abuse,* Toronto, Ontario, March.

56 Keenan, T.P. (1984) "Introducing Computers as Evidence" *66th Annual Meeting, Canadian Bar Association,* Winnipeg, Manitoba, August.

57 Liblong, B. and Birtwistle, G.M. (1983) "A VLSI Design Language Based Upon a High Level Intermediate Form" *VLSI '83 Conference,* Waterloo, Ontario.

58 Liblong, B., Melham, T., and Birtwistle, G. (1984) "Exploiting Hierarchies in EDICT" *VLSI'84 Conference,* 270-273, Edmonton, Alberta, September.

59 Lomow, G.A. and Unger, B.W. (1982) "The Process View of Simulation in Ada" *Proc. SCS Winter Simulation Conference,* 77-86, San Diego, California, December.

60 Lomow, G.A. and Unger, B.W. (1984) "Distributed Software Prototyping in Jade" *Proc. Canadian Information Processing Society Session '84,* Calgary, Alberta, May.

61 Masrani, R. and Keenan, T.P. (1984) "Security and privacy in cellular telephone systems" *Proc. AFIPS Conference on Computer Security,* 397-410, Toronto, Ontario, September.

62 Neal, R., Lomow, G.A., Peterson, M., Unger, B.W., and Witten, I.H. (1984) "Experience with an inter-process communication protocol in a distributed programming environment" *Proc. Canadian Information Processing Society Session '84,* Calgary, Alberta, May.

63 Parker, J. (1985) "Simulating a Robot Arm Using Graphics and Animation" *Proc. SCS Conference on AI, Graphics, and Simulation,* San Diego, California, January.

64 Pooley, R., Williams, A., and Birtwistle, G.M. (1984) "A Process Based Simulation of X25 Using Demos" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California.

65 Rokne, J. (1982) "The evaluation of the range of functions" *Proc. 10th IMACS Congress,* 382-385, Montreal, Quebec.

66 Unger, B., Birtwistle, G., Cleary, J., Hill, D., Lomow, G.A., Neal, R., Peterson, M., Witten, I.H., and Wyvill, B.L.M. (1984) "JADE: A distributed software prototyping environment" *Proc. SCS Conference on Simulation in Strongly Typed Languages,* San Diego, California, February.

67 Unger, B.W., Lomow, G.A., and Andrews, K. (1985) "A Process Oriented Distributed Simulation Package" *Proc. SCS Conference on Distributed Simulation,* San Diego, California, January.

68 Witten, I.H. (1981) "Programming by example for the casual user: a case study" *Proc. Canadian Man-Computer Communication Conference,* 105-113, Waterloo, Ontario, June.

69 Witten, I.H. (1981) "Some recent results in non-deterministic modelling of behaviour sequences" *Proc. Society for General Systems Research Annual Conference,* 265-274, Toronto, Ontario, January.

70 Witten, I.H., Cleary, J.G., Darragh, J.J., and Hill, D.R. (1982) "Reducing keystroke counts with a predictive computer interface" *Proc. IEEE Computer Society Conf on Computing to Aid the Handicapped,* 3-10, University of Virginia, Charlottesville, November.

71 Witten, I.H. (1982) "An interactive computer terminal interface which predicts user entries" *Proc. IEEE Conference on Man-machine Interaction,* 1-5, Manchester, England, July.

72 Witten, I.H. (1982) "Non-deterministic modelling and its application in adaptive optimal control" *Proc. International Conference on Mathematical Learning Models — Theory and Algorithms,* Bad Honnef, Germany, May.

73 Witten, I.H., Cleary, J.G., and Darragh, J.J. (1983) "The reactive keyboard: a new technology for text entry" *Converging Technologies: Proc. Canadian Information Processing Society Conference,* 151-156, Ottawa, Ontario, May.

74 Witten, I.H. and Cleary, J. (1983) "Picture coding and transmission using adaptive modelling of quad trees" *Proc. International Electrical, Electronics Conference,* 222-225, Toronto, Ontario, September.

75 Witten, I.H., Greenberg, S., and Cleary, J. (1983) "Personalizable directories: a case study in automatic user modelling" *Proc. Graphics Interface '83,* 183-189, Edmonton, Alberta, May.

76 Witten, I.H. (1984) "Dynamic documents" *Proc. PROTEXT I — First International Conference on Text Processing,* Dublin, Ireland, September.

77 Witten, I.H. and Fremont, D. (1984) "A student information service for a University Computer Science Department" *Proc. 15th Ontario Universities Computing Conference,* Lakehead University, Thunder Bay, Ontario, June.

78 Wyvill, B.L.M., Liblong, B., and Hutchinson, N. (1984) "Using Recursion to Describe Polygonal Surfaces" *Proc. Canadian Information Processing Society, Graphics Interface '84,* Ottawa, Ontario.

79 Wyvill, B.L.M. Liblong, B., Maulsby, D., and McPheeters, C. (1984) "Computer Animation at the University of Calgary" *Proc. Canadian Information Processing Society Session '84,* Calgary, Alberta, May.

80 Wyvill, B.L.M., Neal, R., Levinson, D., and Bramwell, R. (1984) "JAGGIES – a distributed hierarchical graphics system" *Proc. Canadian Information Processing Society Session '84,* 214-217, Calgary, Alberta, May.

81 Wyvill, B.L.M. (1985) "Current Trends in Graphics and Animation" *SCS Conference on Simulation,* San Diego, California, January.

82 Wyvill, B.L.M., McPheeters, C., and Novacek, M. (1985) "Stochastic Objects in a Hierarchical Graphics System" *Proc. Graphics Interface '85,* Montreal, Quebec.

RESEARCH REPORTS

83 Birtwistle, G.M. (1984) "Proposal for EDICT: An Environment for Designing Integrated Circuits" Research Report 84/155/13, Department of Computer Science, University of Calgary, January.

84 Birtwistle, G. and Wyvill, B.L.M. (1984) "Blockaid: A Model for Program Execution" Research Report 84/158/16, Department of Computer Science, June.

85 Birtwistle, G., Liblong, B., and Wyvill, B.L.M. (1984) "Simula Bibliography" Research Report 84/159/17, Department of Computer Science, University of Calgary, June.

86 Bramwell, B. (1983) "Browse: An On-Line Manual System Without an Acronym" *Newsletter of ACM-SIGDOC, 9* (4), Research Report J83/6/6, Department of Computer Science, University of Calgary, January.

87 Cleary, J.G., Wyvill, B.L.M., Birtwistle, G.M., and Vatti, R. (1983) "Multiprocessor Ray Tracing" Research Report 83/128/17, University of Calgary, Department of Computer Science.

88 Cleary, J.G. (1984) "Implementation of Concurrent Prolog using message passing" *in press International Conference on Parallel Processing,* Research Report 84/149/7, University of Calgary, Department of Computer Science.

89 Darragh, J.J., Witten, I.H., and Cleary, J.G. (1983) "Adaptive text compression to enhance a modem" Research Report 83/132/21, Department of Computer Science, University of Calgary.

90 Esau, R. (1983) "A speech input interface to emacs" Research Report J83/7/7, Department of Computer Science, University of Calgary, February.

91 Greenberg, S. and Witten, I.H. (in press) "Adaptive personalized interfaces — a question of viability" Research Report 84/152/10, Department of Computer Science, University of Calgary, April.

92 Hill, D.R. (1984) "Designing for Human-Computer Interaction: some rules and their derivation" Research Report 84/166/24, Department of Computer Science, University of Calgary, June.

93 Hill, D.R. (1984) "Dialogue design notes" Research Report 84/167/25, Department of Computer Science, University of Calgary, October.

94 Jade (1984) "Jade User's Manual" Research Report J84/1/1, Department of Computer Science, University of Calgary, September.

95 Joyce, J., Lomow, G.A., Slind, K., and Unger, B.W. (1985) "Monitoring Distributed Systems" *submitted to IEEE Software,* Research Report J84/8/1, Department of Computer Science, University of Calgary, April.

96 Keenan, T.P. and Ferris, K.L. (1983) "Ways You Can Help Protect Your Corporate Information" *CIPS Review, 7* (3), May/June.

97 Unger, B.W. (1983) "Jade: Project Definition and Plan" Jade Research Report J83/1/1, Department of Computer Science, University of Calgary, March.

98 Unger, B.W. (1984) *Nserc Status Report for SMI-64: Distributed Software Prototyping : Jade.* Department of Computer Science, University of Calgary, January.

99 Vollmerhaus, W. (1983) "On Computing All Irreducible Non-Embeddable Graphs for the Projective Plane That Contain K3,4" Research Report 83/142/31, Department of Computer Science, University of Calgary, December.

100 Witten, I.H., Birtwistle, G., Cleary, J.G, Hill, D., Levinson, D., Lomow, G.A., Neal, R., Peterson, M., Unger, B.W., and Wyvill, B. (1983) "Jade: A Distributed Software Prototyping Environment" *ACM Operating Systems Review, 17*(3), July.

101 Witten, I.H. and Cleary, J.G. (1984) "Fortelling the Future by Adaptive Modelling" Research Report 84/143/1,, Department of Computer Science, University of Calgary, February.

102 Witten, I.H. (1984) "Elements of typography (for computer scientists)" Research Report 84/165/23, Department of Computer Science, University of Calgary, October.

103 Witten, I.H. and Bramwell, B. (1985) "On viewing structured documents" *in press ACM SICOA Newsletter.*

104 Wyvill, B.L.M. (1981) "Some Recursive Techniques in Picture Description." Research Report 82/99/18, University of Calgary, Department of Computer Science, December.

105 Wyvill, B.L.M. and Greenberg, S. (1983) "A GROPER Tutorial" Research Report 83/131/20, Department of Computer Science, University of Calgary, September.

106 Wyvill, B.L.M. and McPheeters, C. (1984) "A Tutorial Guide to the ANI Animation System" Research Report 84/147/15, Department of Computer Science, University of Calgary, October.

107 Wyvill, B.L.M., McPheeters, C., Novacek, M., Pearce, A., and Jansonius, C. (1985) "Practical Graphics for 3D Computer Animation" Research Report 85/189/2, Department of Computer Science, University of Calgary, January.

BOOKS AND FILMS

108 Jade (1984) "Jade Windows, Graphics and Animated Monitoring" 20 minute film, Department of Computer Science, University of Calgary, January.

109 Keenan, T.P (1984) *Business Computer Literacy*. series of eight videotapes produced by ACCESS Alberta; placed in national distribution, Calgary, Alberta.

110 Keenan, T.P. (1984) *IDEAS: Crimes of the Future*. Canadian Broadcasting Corp.: transcript of a three-part radio series, broadcast October, 1984.

111 Rokne, J. (1983) *Introduction to Interval Computations (Translation of Alefeld-Herzberger: Einfuhrung in die Interval Mathematik, 1982)*. Academic Press.

112 Rokne, J. and Ratschek, H. (1984) *Computer Methods for the Range of Functions*. Ellis Horwood (Research Monograph).

113 Rokne, J. (1985) *Discrete Iterations (Translation of Robert: Iterations Discretes)*. Springer Verlag.

114 Unger, B.W., Lomow, G.A., and Birtwistle, G.M. (1984) *Simulation Software and Ada*. SCS Research Monograph, San Diego, California.

115 Witten, I.H. (in press) "Computer Speech" *The Encyclopedia of Physical Science and Technology*, Academic Press.

116 Witten, I.H. (in press) *Making Computers Talk*. Englewood Cliffs, NJ, Prentice Hall.

117 Wyvill, B.L.M., Maulsby, D., and McPheeters, C. (1983) "The War of the Worlds" *A 2 minute computer animated film*, Department of Computer Science, University of Calgary, December.