5.3.3 SUMEX Staff Publications

The following are publications for the SUMEX staff and include papers describing the SUMEX-AIM resource and on-going research as well as documentation of system and program developments. Many of the publications documenting SUMEX-AIM community research are from the individual collaborating projects and are detailed in their respective reports (see Section 9 on page 135). Publications for the AGE and AI Handbook core research projects are given there.

[1] Carhart, R.E., Johnson, S.M., Smith, D.H., Buchanan, B.G., Dromey, R.G., and Lederberg, J, _Networking and a Collaborative Research Community: A Case Study Using the DENDRAL Programs_, ACS Symposium Series, Number 19, Computer Networking and Chemistry, Peter Lykos (Editor), 1975.

[2] Levinthal, E.C., Carhart, R.E., Johnson, S.M., and Lederberg, J., _When Computers Talk to Computers_, Industrial Research, November 1975

[3] Wilcox, C. R., _MAINSAIL - A Machine-Independent Programming System,_ Proceedings of the DEC Users Society, Vol. 2, No. 4, Spring 1976.

[4] Wilcox, Clark R., _The MAINSAIL Project: Developing Tools for Software Portability,_ Proceedings, Computer Application in Medical Care, October, 1977, pp. 76-83.

[5] Lederberg, J. L., _Digital Communications and the Conduct of Science: The New Literacy,_ Proc. IEEE, Vol. 66, No. 11, Nov 1978.

[6] Wilcox, C. R., Jirak, G. A., and Dageforde, M. L., _MAINSAIL - Language Manual,_ Stanford University Computer Science Report STAN-CS-80-791 (1980).

[7] Wilcox, C. R., Jirak, G. A., and Dageforde, M. L., _MAINSAIL - Implementation Overview,_ Stanford University Computer Science Report STAN-CS-80-792 (1980).

Mr. Clark Wilcox also chaired the session on "Languages for Portability" at the DECUS DECsystem10 Spring '76 Symposium.

In addition, a substantial continuing effort has gone into developing, upgrading, and extending documentation about the SUMEX-AIM resource, the SUMEX-TENEX system, and the many subsystems available to users. These efforts include a number of major documents (such as SOS, PUB, TENEX-SAIL, and MAINSAIL manuals) as well as a much larger number of document upgrades, user information and introductory notes, an ARPANET Resource Handbook entry, and policy guidelines.

6 Methods of Procedure

This section details our approach to achieve the goals summarized in Section 3.3 on page 18 during the next five year period. As indicated earlier, objectives and plans for individual collaborating projects are discussed in Section 9 beginning on page 135.

Just as the tone of our renewal proposal derives from the continuing long-term research objectives of the SUMEX-AIM community, our approach derives from the methods and philosophy already established for the resource. We will continue to develop useful knowledge-based software tools for biomedical research based on innovative, yet accessible computing technologies.

For us it is important to make systems that work and are exportable. Hence, our approach is to integrate available state-of-the art hardware technology as a basis for the underlying software research and development necessary to support the AI work.

SUMEX-AIM will retain its broad community orientation in choosing and implementing its resources. We will draw upon the expertise of on-going research efforts where possible and build on these where extensions or innovations are necessary. This orientation has proved to be an effective way to build the current facility and community.

We have built ties to a broad computer science community; have brought the results of their work to the AIM users; and have exported results of our own work. This broader community is particularly active in developing technological tools in the form of new machine architectures, language support, and interactive modalities.

6.1 Resource Operations Plans

6.1.1 Resource Hardware

6.1.1.1 Rationale for Future Plans

As discussed in our progress report and supported by collaborating project reports, we have implemented an effective set of computing resources to support AI applications to biomedical research. At the resource core is the KI-TENEX/2020 facility, augmented by portions of the Rutgers 2050 and Stanford SCORE 2060 machines. These have provided an unsurpassed set of tools for the initial phases of SUMEX-AIM development in terms of operating system facilities, human engineering, language support for artificial intelligence program development, and community communications tools. As the size of our community and the complexity of knowledge-based programs have increased, several issues have become important for the continued development and practical dissemination of AI programs:

1) The community has a continuing need for more computing capacity. This arises from the growth of new applications projects, new core research ideas, and the need to disseminate mature systems within and outside of the AIM community. Nowhere is this felt more strongly than among the Stanford community where system access constraints have seriously impeded development progress. A picture of system congestion can be found in the summary of loading statistics in Appendix B on page 355 and in the statements from many of our user projects.

2) Many programs require a larger virtual address space. As AI systems become more expert and encompass larger and more complex domains, they require ever larger knowledge bases and data structures that must be traversed in the course of solving problems. The 256K word address limit of the PDP-10 has constrained program development as discussed in Appendix F on page 390. Increasing effort has gone into "overlays" resulting in higher machine overhead, more difficulty in making program changes, and lost programmer time. Simpler hardware solutions are needed.

3) AI programs are being tested and disseminated increasingly beyond their development communities. We cannot continue to provide all of the computing resources this implies through central systems like SUMEX. The capacity does not exist. Network communications facilities are not able to support facile human interactions (high speed, improved displays, graphics, and speech/touch modalities). And a grant-supported research environment cannot meet the technical and administrative needs of a "production" community. Thus, we need to explore better ways to package complex AI software and distribute the necessary computing tools cost effectively into the user communities.

An "obvious" solution to our capacity needs (but not the address space limitations) is to buy additional large machine resources that are software compatible with the existing community KI-10 and PDP-20 systems. By placing these nodes at user sites, an improvement in communication bandwidth would be possible to enhance the human interactive support. The addition of more DEC 2060 or larger machines to the SUMEX community is not cost-effective, however.

An alternative and more feasible approach to meet community needs is to explore the use of smaller, less expensive machines as satellites (some remote) to the main resource. A variety of technologies are now becoming available as machines that we can buy and use. These could have a number of advantages:

1) A relatively small investment in capital equipment is required for each incremental capacity augmentation.

2) New architectures directly support larger program address spaces.

3) Possible location close to individual research groups allows better human engineering of user interfaces by using higher speed

communication, improved display technology, and other modalities for human interaction such as speech and touch.

4) System capacity can be allocated more flexibly and efficiently by having to satisfy fewer simultaneous scheduling constraints and by being more easily dedicatable to operational demonstrations.

This approach poses a number of possible disadvantages stemming primarily from the distributed nature of the computing resources:

1) Each such machine would have a relatively small capacity. These may be sufficient for many computing tasks of a local user group. It would be difficult to aggregate such dispersed capacity, however, when needed for a single computing-intensive task except through multiprocessing. This would be made difficult by geographic remoteness. Such intensive computing needs will likely still be best handled by shared specialized central resources.

2) Decentralizing the computing resources places an increased centrifugal force on community interactions. Effective network communications must be maintained to allow continued collaborative interactions, software sharing, access to common knowledge and data bases, message exchange, etc.

3) Geographically distributed computing tends to encourage costly duplication of similar operations and maintenance functions for system hardware and software support. These added costs are lessened when distributed over clusters of systems near SUMEX-AIM community nodes.

These trade-offs, coupled with the developing new computer technology, suggest a continuing need for a spectrum of resource configurations and support functions over the next grant period including:

1) experimentation with new shared centralized systems

2) distributed single-user "professional workstations"

3) improved communications tools to integrate them together effectively.

In addition to continuing operation of the existing resources, we plan to direct SUMEX research efforts to explore the potential of such newly available systems as solutions to AIM community needs. Our approach will be to integrate a heterogeneous set of network-connected hardware tools, some of which will be distributed through the user community. We will emphasize the development of system and application level software tools to allow effective use of these resources and continue to provide community leadership to encourage scientific communications.

6.1.1.2 Summary of Proposed Hardware Acquisitions

As discussed in more detail in later sections, we plan to acquire the following additional hardware

yr 1 - Add 256K words of core to the existing KI-10 AMPEX memory to reduce page swapping overhead.

- Buy a VAX 11/780 with 2M bytes of memory and minimal disk and tape peripherals to provide large address space INTERLISP facilities, to experiment with AI program export, to support development of VAX system software for the community, and to alleviate congestion in the Stanford 40% of the SUMEX resource.

- Develop a file server coupled to SUMEX host machines via the high speed Ethernet. This will minimize the need for redundant large file systems on each host and alleviate the file storage limitations of the AIM community. The server will be based on a PDP-11 with 630M bytes of disk storage initially and tape facilities for backup and archives.

yr 2 - Add 2M bytes of memory to the VAX purchased in year 1.

- Add 630M bytes to the file server purchased in year 1.

- Buy 5 single-user "professional workstations" (PWS) based on the Zenith-MIT NU system (or equivalent) to develop and experiment with this means for AI program development, export, and human interface enhancements. These machines will be distributed within the Stanford community initially to facilitate development and will be coupled by Ethernet with the main resource.

yr 3 - Add a second VAX 11/780 for general community support with large address space INTERLISP. This machine will be managed for program testing in a way similar to the existing 2020.

- Add 2 PWS systems to be distributed within the AIM community under Executive Committee control.

yr 4 - Add 3 PWS systems to be distributed within the AIM community under Executive Committee control.

- Add 630M bytes to the central file server to meet expected growth in community file storage needs.

yr 5 - Add 3 PWS systems to be distributed within the AIM community under Executive Committee control.

6.1.1.3 Existing Hardware Operation

The current SUMEX-AIM facilities represent a large existing investment. The KI-10 facility has operated at capacity for more than three years, even with periodic augmentation. Significant augmentation to any of the present hardware configuration cannot be done without major upgrades to the mainframe and memory components. A factor of 5-10 increase in throughput could be achieved by replacing the KI-10's with a DEC 2060 or the projected new 2080 processor. This would maintain software compatibility in the same sense as the 2020 (TENEX vs TOPS-20) but would cost $500 - 1000K. We do not believe the funding for such an upgrade would be forthcoming. It also would not attack the INTERLISP addressing limitations or the needs for higher performance interactive support. Whereas this magnitude of capacity augmentation within the AIM community would indeed be welcome, we feel that SUMEX as a research resource should invest its efforts in exploring newer technologies that offer solutions to current needs with broader long range impact.

For these reasons, we do not propose any substantial changes to the existing KI-10 and 2020 hardware systems and we expect them to continue to provide effective community support and serve as a communication nucleus for more distributed resources. We do propose to augment the KI-10 AMPEX memory box purchased in 1977 in order to reduce page swapping overhead. During peak loads, an average of 15-20% of system capacity is lost to pager traps and a substantial additional load comes from drum service interrupt handling. The AMPEX will physically hold another 256K words or 512 pages of memory. Since our current configuration has a net of 852 pages available to users, this increment would provide 60% more physical user space at a cost of only $65,000. We feel this will measurably improve efficiency and smooth out interactive response at high loads.

It should be recognized that the KI-10 processors are now 6 years old and will be 12 years old at the end of the proposed grant term. We have already begun to feel maintenance problems from age such as poor electrical contacts from oxidization and dirt, backplane insulation flowing on "tight wraps", and brittle cables. These problems are quite manageable still and we expect to be able to continue reliable operation over the next grant term.

We plan no upgrades to the 2020 configuration. The current file shortage will be remedied in conjunction with that of the rest of the facility by implementing a community file server sharable and accessible via the Ethernet.

For both systems, we are actively working to complete efficient interfaces to the Ethernet to allow flexible, high speed terminal connections, file transfers, and effective sharing of network, printing, plotting, remote links, and other resources. This system will form the backbone for smooth integration of future hardware additions to the resource.

6.1.1.4 Large Address Space Machines

As indicated in Appendix F on page 390, the user address space limitations imposed by the architecture of the PDP-10/20 systems have been increasingly felt in building large knowledge-based systems for biomedicine. After considerable study, the ARPANET INTERLISP community has started active projects to convert INTERLISP to run on the DEC VAX and to extend the UNIX operating system for VAX to support demand paging and to take advantage of the 31-bit address space. VAX was also the preferred choice as an export machine for the DENDRAL project to support the biomolecular characterization community. Their choice of VAX was made to provide the best match with machines increasingly available in the biochemistry laboratory environment and able to run the programs being developed by DENDRAL (including CONGEN recently converted from INTERLISP to BCPL). Whereas other machines (e.g., PRIME) offer a comparable address space capability and are cost competitive, a comparable software community does not exist on which to base not only AI program development but also the extensive utility software packages for interactive user support necessary to the AIM community.

For these reasons we feel VAX is an ideal candidate for augmenting the SUMEX resource to experiment with large address space LISP systems, to provide added capacity to support software export efforts like DENDRAL, and to alleviate the congestion of the Stanford aliquot of the current system. We propose a modest configuration initially to support developmental efforts to integrate the VAX into the SUMEX resource during the first year of the continuation grant (see Figure 4 for a configuration diagram). This machine can be expected to support 8-10 users initially. In year 2 we plan to increase the memory size by 2 Mbytes to allow more efficient use of the VAX capacity, increasing the users supported to 15-20. In year 3, we plan to add a second VAX to make large address space LISP available more broadly in the community to support future program testing akin to the purpose of the 2020 system. We tentatively plan for another 11/780 system although by then newer models may be available.

6.1.1.5 Single-User Professional Workstations

Motivated by the development of AI programs that are truly useful to their target communities, another major thrust of our research plans for the coming term is the investigation of single-user "professional workstations" (PSW) as a vehicle for exporting AI programs and providing computing power local to the user so that high bandwidth human interactions can be supported (e.g., bit mapped displays for high quality video and graphics, touch, and speech). Emerging VLSI technology promises increasingly capable and cost-effective computing tools through denser packing of microelectronic circuits and reduced development costs to produce relatively specialized systems. Packing density increases by four orders of magnitude may be expected over the next five to ten years [16]. Such hardware advances make the cost-effective marketing of complex AI systems a coming reality.

Prototype single-user professional workstation systems based on current technology such as the Motorola MC-68000 or other special microprocessors are being developed and will begin to be delivered within the year. We must begin now to develop our software systems to take advantage of the improved computing environments these provide for biomedical AI programs. We propose an active role in integrating these systems into the SUMEX-AIM community so that user projects can exploit them for developing, testing, and disseminating their programs.

Current candidates as experimental single-user PWS's include the "PERQ" by Three Rivers Computer Corporation [17], the "D-Class" machines by Xerox Corporation [18, 19], the MIT-developed "CADR" LISP machine by Symbolics, Inc. [20], the MIT-developed "NU" system by Zenith [21], and the "Jericho" system by BBN. Details of the design of most of these systems are still proprietary but deliveries of PERQ, CADR, and NU are expected within a year with continued active development based on user community needs. Characteristically, these systems are intended to be high performance, single user computers with local disk storage, bit-mapped display, and connection to a contention network such as Ethernet or MIT's Chaosnet.

Considerable hardware and system software development work remains on these machines, but by year 2 (1982), we expect them to be relatively well established and we plan to purchase 5 for integration into the Stanford community. We budget $30,000 per machine based on projected pricing of the NU system. The NU will be produced by Zenith from a design by S. Ward at MIT around the MC-68000 microprocessor. This machine supports 23-bit addressing, 32-bit internal data and address registers, 16-bit asynchronous bus, and will soon have facilities for virtual memory management. These will be allocated with 2 machines for Heuristic Programming Project development work, 1 for the experimental ONCOCIN system, 1 for Prof. Shortliffe's research work in MYCIN, and 1 for development work within the SUMEX staff. Our efforts will be to tailor AI performance programs to these systems to provide improved and cost effective expert assistance to biomedical professionals. This first batch of machines will be limited to the Stanford community to allow close access for developing software and tailoring network connection facilities as well as easy maintenance. We will work during that year to tune the software systems on these machines for AIM community use.

In years 3-5, we play to acquire an additional 2-3 machines per year to be allocated among the user community based on Executive Committee advice. We will establish necessary communication links to couple these machines to other AIM resources using leased telephone lines, dial-up services, or commercial network links as appropriate.

6.1.1.6 File Server

An equally important resource to SUMEX-AIM community development is file storage. We have reported frequently in the past on the effects of file storage limitations for our existing resource. As AI programs develop larger knowledge and data bases, as the community of application projects grows, and as more and more external users gain access to test working programs, significantly increased file storage capacity will be needed to support interactive work. It makes little sense to duplicate expensive file storage facilities for each of the machines contemplated in the SUMEX-AIM resource and community.

We expect users to work between several machines in the course of their research and many of the files will be common. Similarly there are many system and documentation files common between the KI-10 and 2020 systems as will be the case between other clusters of similar machines (VAX's and professional workstations).

Thus, a more efficient approach is to implement for each machine only the amount of storage needed to support the currently active users together with a community file service coupled to each machine through a high speed local network (Ethernet). Such a "file server" has worked effectively in the Xerox Alto/Ethernet environment and is a natural approach for the evolving SUMEX-AIM environment. By centralizing file storage, we can minimize equipment costs and file backup, archiving, and operations costs. Such a system even makes selective redundancy for reliability possible and thereby makes users more immune to failures in individual machines.

We plan to implement a basic file server for the SUMEX-AIM community in the first year. It will be based initially on a PDP-11/34 computer with two 317M byte disk drives and two tape drives for backup. The choice of the PDP-11 is based on the ready availability of disk/tape systems for these machines. In years 2 and 4, we plan to add an additional 2 drives each year to bring the total capacity to 2000M bytes.

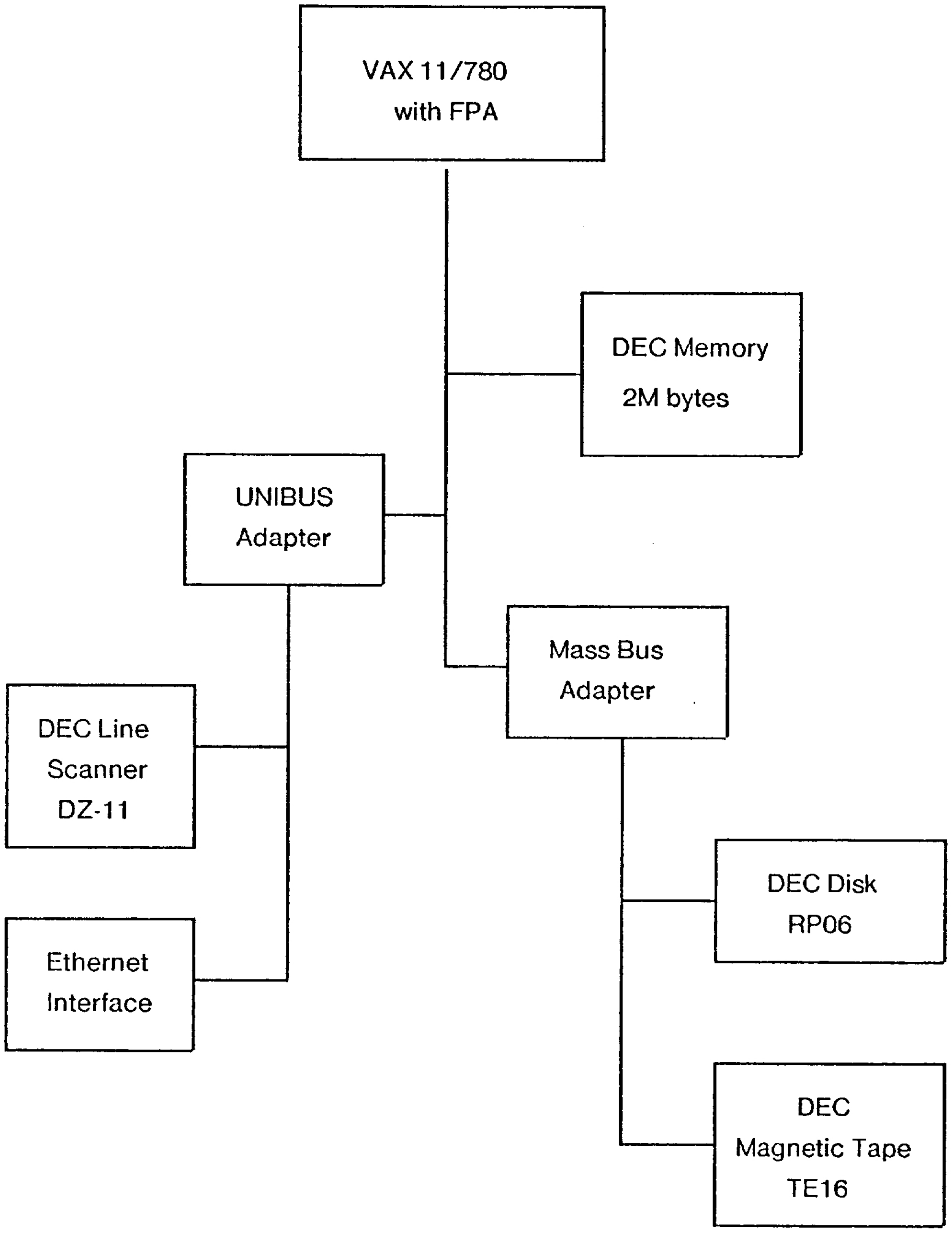

Figure 4. Proposed VAX configuration

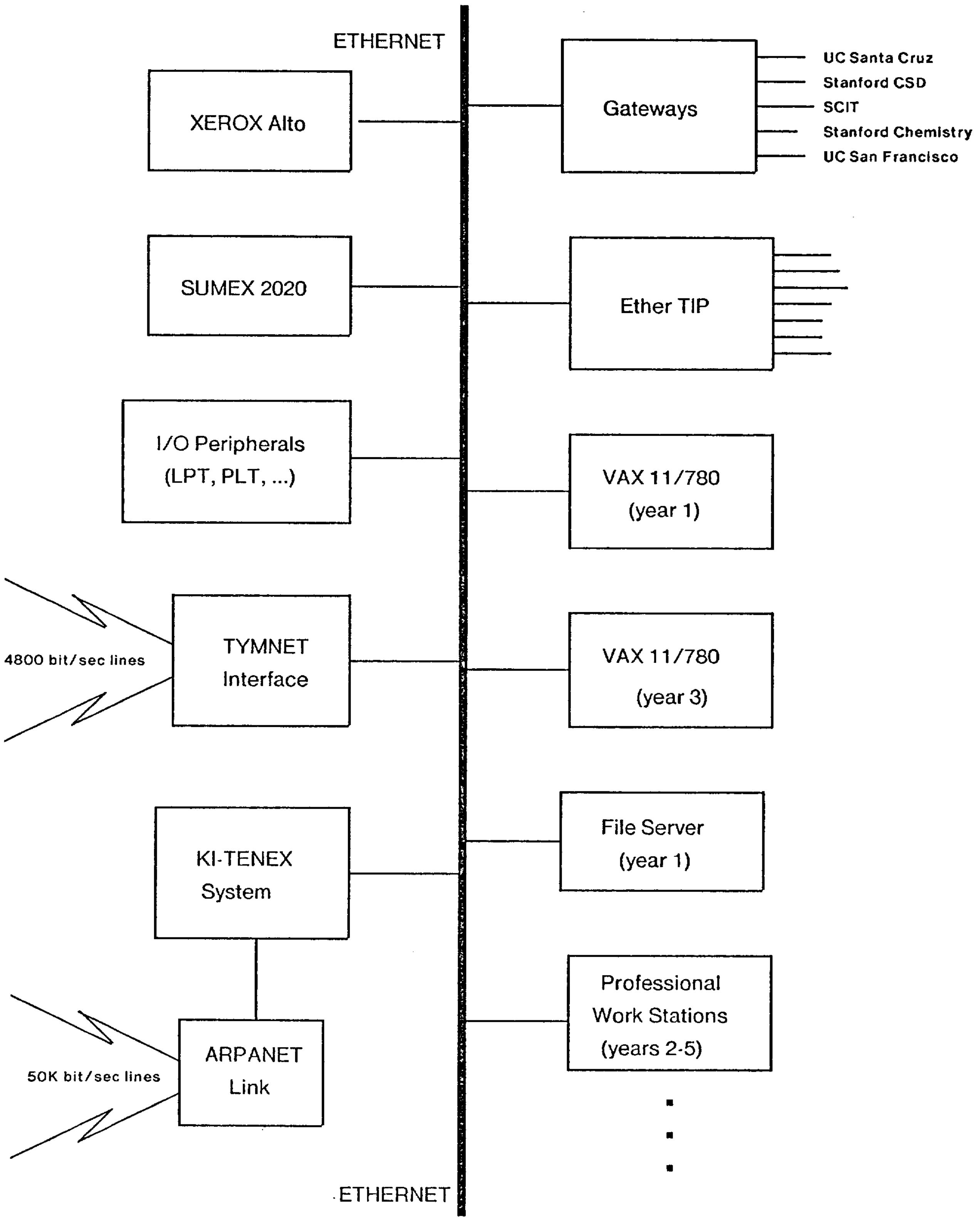

Figure 5. Planned Ethernet System to Integrate System Hardware

6.1.2 Communication Networks

Networks have been centrally important to the research goals of SUMEX-AIM and will become more so in the context of increasingly distributed computing. Communication will be crucial to maintain community scientific contacts, to facilitate shared system and software maintenance based on regional expertise, to allow necessary information flow and access at all levels, and to meet the technical requirements of shared equipment.

6.1.2.1 Long-Distance Connections

We have had reasonable success at meeting the geographical needs of the community during the early phases of SUMEX-AIM through our ARPANET and TYMNET connections. These have allowed users from many locations within the United States and abroad to gain terminal access to the AIM resources (SUMEX, Rutgers, and SCORE) and through ARPANET links to communicate much more voluminous file information. Since many of our users do not have ARPANET access privileges for technical or administrative reasons, a key problem impeding remote use has been the limited communications facilities (speed, file transfer, and terminal handling) offered currently by commercial networks. Commercial improvements are slow in coming but may be expected to solve the file transfer problem in the next few years. A number of vendors (AT&T, IBM, Xerox, etc.) have yet to announce commercially available facilities but TELENET is actively working in this direction. We plan to continue experimenting with improved facilities as offered by commercial or government sources in the next grant term. We have budgeted for continued TYMNET service and an additional amount annually for experimental network connections.

High-speed interactive terminal support will continue to be a problem since one cannot expect to serve 1200-9600 baud terminals effectively over shared long-distance trunk lines with gross capacities of only 9600-19200 baud. We feel this is a problem that is best solved by distributed machines able to effectively support terminal interactions locally and coupled to other AIM machines and facilities through network or telephonic links. As new machine resources are introduced into the community, we will allocate budgeted funds with Executive Committee advice to assure effective communication links.

6.1.2.2 Local Intermachine Connections

A key feature of our plans for future computing facilities is the support of a heterogeneous processing environment that takes advantage of newly available technology and shared equipment resources between these machines. The "glue" that links these systems together is a high speed local network. We have chosen Ethernet and the Xerox PUP [10, 13] protocols for these interconnections. This choice was based on the availability of that technology now and the economics of using already developed TENEX and other server software. We expect the Ethernet system to continue to meet our technical needs for the coming grant term and we plan to continue to use it. We are working closely with other groups here at Stanford and elsewhere to share hardware interface and software designs wherever possible.

Our goals are to complete integration of the 2020 system with the KI-10 system, including making selected KI-10 peripherals available as Ethernet nodes, creating links to nearby campus resources, and establishing needed remote links to other groups not on the ARPANET such as Wipke at the University of California at Santa Cruz. A diagram of our Ethernet system is shown in Figure 5 on page 61 and includes the following major elements:

1) KI-10 direct memory access interface. We currently have an inefficient I/O bus connection.

2) 2020 interface. Complete the hardware and software connection of the 2020 using the UNIBUS adapter.

3) Stanford campus gateway. Establish links to other Ethernets on campus to allow access to special resources (Dover printer, plotters, typesetting equipment, etc.) and to allow users to easily access various computing resources.

4) Ethertip. We need additional terminal ports into the system and the Ethernet provides a natural mechanism to do this supporting high speed terminals and connections to various resources (KI-10, 2020, VAX's, etc.).

5) TYMNET connection. This connection currently comes through the KI-10's and will be moved to a separate Ethernet node. This will free the KI-10's from handling the special TYMNET protocol and will allow TYMNET users to access any of the SUMEX-AIM resources. Similar facilities for the ARPANET may also be implemented depending on administrative constraints.

6) Printer/plotter service. We plan to make these local resources accessible from any of the SUMEX-AIM machines instead of being centered on the KI-10's. This will also free up the KI-10's from routine spooler tasks.

7) Connections for other machines (VAX's, Professional Workstations, file server, etc.)

6.1.3 Resource Software

We will continue to maintain the existing system, language, and utility support software on our systems at the most current release levels, including up-to-date documentation. We will also be extending the facilities available to users where appropriate, drawing upon other community developments where possible. We rely heavily on the needs of the user community to direct system software development efforts. Specific development areas for existing systems include:

1) completion of the Ethernet connections and necessary host software. This will include basic packet handling, PUP protocols at all levels, and relocation of shared existing resources to become Ethernet nodes.

2) bug fixes in the current monitors. We have 6 bugs partially characterized that cause infrequent crashes and that are hard to isolate because they cause system problems long after the fact. We will continue to work to repair these problems as time permits.

3) continued evaluation of system efficiency to improve performance.

4) compatibility issues. Our current compatibility package for TOPS-20 requires additional work to extend its features. We will also keep it up-to-date as DEC make new changes to their system.

5) continued work to create similar working and programming environments between our TENEX and TOPS-20 systems. This will include moving TENEX features like the SUMEX GTJFN enhancements and scheduling controls as needed to TOPS-20 and vice versa

6) continued work to improve system information and help facilities for users.

Our plans for augmenting the SUMEX-AIM resources will entail substantial new system and subsystem programming. Our goals will be to derive as much software as possible from the user communities of the new VAX and Professional Workstation machines but we expect to have to do considerable work to adapt them to our biomedical AI needs. Many features of these systems are designed for a computer science environment and lack some of the human engineering and "friendliness" capabilities we have found needed to allow non-computer scientists to effectively use them. We are beginning to experiment with physician needs for interfaces to our AI programs to be better able to adapt the new machines as professional aids. Also many of the utility tools that we take for granted in the well-developed TENEX and TOPS-20 environment (communications, text manipulation, file management, accounting, etc.) will have to be reproduced. We expect to set up many of the common information services as network nodes.

Within the AIM community we expect to serve as a center for software sharing between various distributed computing nodes. This will include contributing locally developed programs, distributing those derived from elsewhere in the community, maintaining up-to-date information on subsystems available, and assisting in software maintenance.

6.1.4 Community Management

We plan to retain the current management structure that has worked so well. We will continue to work closely with the management committees to recruit the additional high quality projects which can be accommodated and to evolve resource allocation policies which appropriately reflect assigned priorities and project needs. We expect the Executive and Advisory Committees to play an increasingly important role in advising on priorities for facility evolution and on-going community development planning in addition to their recruitment efforts. The composition of the Executive committee will grow as needed to assure representation of major user groups and medical and computer science applications areas. The Advisory Group membership rotates regularly and spans both medical and computer science research expertise. We expect to maintain this policy.

We will continue to make information available about the various projects both inside and outside of the community and thereby promote the kinds of exchanges exemplified earlier and made possible by network facilities.

The AIM workshops under the Rutgers resource have served a valuable function in bringing community members and prospective users together. We will continue to support this effort. This summer the AIM workshop will be held at Stanford and we are actively helping to organize the meeting. We will continue to assist community participation and provide a computing base for workshop demonstrations and communications. We will also assist individual projects in organizing more specialized workshops as we have done for the DENDRAL and AGE projects.

Fee-for-Service?

We have pondered the possibilities of a fee-for-service approach for allocation of the resource in the coming period. We believe that this would be inappropriate for an experimental research resource of national scope like SUMEX for several reasons:

1) We have based the development of the national SUMEX-AIM resource entirely on experimentation with tools for new AI research and inter-community scientific collaborations. If obliged to recover some portion of the overall facility cost, these goals may become diluted with administrative and financial impediments, and commitments to paying users, that are tangential to our main research efforts. There is little doubt that a facility of the quality of SUMEX could be tailored to attract paying users (we have turned down numerous such potential users already because they were not aligned with our AI research goals). However, there is little point in demonstrating once again that a computing resource can pay for itself. Rather we should judiciously allocate the available resources to encouraging new medical AI research efforts and stimulating scientific collaborations that cannot always be financially justified at these early stages.

2) A key element in our management plan for SUMEX is to encourage mature projects to acquire computing resources of their own, as soon as justified, and to couple them through communications tethers to SUMEX. This preserves the limited capacity of the central resource for new research efforts and applications. Maturing projects (those able to pay a fee) have every incentive to obtain separate facilities since they cannot obtain sufficient resources from the heavily loaded central resource. In this way such projects effectively pay a "fee" in securing their own facilities and freeing up part of the central facility.

3) A fee structure would impose substantial additional administrative overhead on the project, compounded by its national character. We would face problems of accountability for the transfer of funds from one institution to another. Also SUMEX is a evolving research resource based on changing experimental facilities. Any fee schedule would need to change frequently to fairly respond to developments in the system. Put simply, it would be an administrative nightmare.

For these reasons, we plan to continue indefinitely our present policy of non-monetary allocation control. We recognize, of course, that this accentuates our responsibility for the careful selection of projects with high scientific and community merit.

6.2 Training and Education Plans

We have an on-going commitment, within the constraints of our staff size, to provide effective user assistance, to maintain high quality documentation of the evolving software support on the SUMEX-AIM system, and to provide software help facilities such as the HELP and Bulletin Board systems. These latter aids are an effective way to assist resource users in staying informed about system and community developments and solving access problems. We plan to take an active role in encouraging the development and dissemination of community databases such as the AI Handbook, up-to-date bibliographic sources, and developing knowledge bases. Since much of our community is geographically remote from our machine, these on-line aids are indispensable for self help. We will continue to provide on-line personal assistance to users within the capacity of available staff through the SNDMSG and LINK facilities.

We budget funds to continue the "collaborative linkage" support initiated during the first term of the SUMEX-AIM grant. These funds are allocated under Executive Committee authorization for terminal and communications support to help get new users and pilot projects started.

Finally, we will continue to actively support the AIM workshop series in terms of planning assistance, participation in program presentations and discussions, and providing a computing base for AI program demonstrations and experimentation.

6.3 Core Research Plans

Motivation

SUMEX core research includes both basic AI research and development of community tools useful for building expert systems. Expert systems are symbolic problem solving programs capable of expert-level performance, in which domain-specific knowledge is represented and used in an understandable line of reasoning. The programs can be used as problem solving assistants or tutors, but also serve as excellent vehicles for research on representation and control of diverse forms of knowledge. MYCIN is one of the best examples.

Because the main issues of building expert systems are coincident with general issues in AI, we appreciate the difficulty of proposing to "solve" basic problems. However, we do propose to build working programs that demonstrate the feasibility of our ideas within well defined limits. By investigating the nature of expert reasoning within computer programs, the process is "demystified". Ultimately, the construction of such programs becomes itself a well-understood technical craft.

The foundation of each of the projects described in the proposal is expert knowledge: its acquisition from practitioners, its accommodation into the existing knowledge bases, its explanation, and its use to solve problems. Continued work on these topics provides new techniques and mechanisms for the design and construction of knowledge-based programs; experience gained from the actual construction of these systems then feeds back both (a) evaluative information on the ideas' utility and (b) reports of quite specific problems and the ways in which they have been overcome, which may suggest some more general method to be tried in other programs.

One of our long-range goals is to isolate AI techniques that are general, to determine the conditions for their use and to build up a knowledge base about AI techniques themselves. SUMEX resources are coordinated for this purpose with the multidisciplinary efforts of the Stanford Heuristic Programming Project (HPP). Under support from ARPA, NIH/NLM, ONR, NSF, and private funding, the HPP conducts research on five key scientific problems of the area, as well as a host of subsidiary issues [1]:

1) Knowledge Representation - How shall the knowledge necessary for expert-level performance be represented for computer use? How can one achieve flexibility in adding and changing knowledge in the continuous development of a knowledge base? Are there uniform representations for the diverse kinds of specialized knowledge needed in all domains?

2) Knowledge Utilization - What designs are available for the inference procedure to be used by an expert system? How can the control structure be simple enough to be understandable and yet sophisticated enough for high performance? How can strategy knowledge be used effectively?

3) Knowledge Acquisition - How can the model of expertise in a field of work be systematically acquired for computer use? If it is true that the power of an expert system is primarily a function of the quality and completeness of the knowledge base, then this is the critical "bottleneck" problem of expert systems research.

4) Explanation - How can the knowledge base and the line of reasoning used in solving a particular problem be explained to users? What constitutes an acceptable explanation for each class of users?

5) Tool Construction - What kinds of software packages can be constructed that will facilitate the implementation of expert systems, not only by the research community but also by various user communities?

Artificial Intelligence is largely an empirical science. We explore questions such as these by designing and building programs that incorporate plausible answers. Then we try to determine the strengths and weaknesses of the answers by experimenting with perturbations of the systems and extrapolations of them into new problem areas. The test of success in this endeavor is whether the next generation of system builders finds the questions relevant and the answers applicable to reduce the effort of building complex reasoning programs.

Research Plan

In the following descriptions, planned core research efforts are grouped under the five major headings listed above, although it should be clear that the boundaries are frequently crossed. Knowledge utilization and tool construction are grouped together.

6.3.1 Knowledge Representation

6.3.1.1 RLL -- The Representation Language Language

A framework for constructing new representation languages, called RLL (for "Representation Language Language"), is under development within the HPP. RLL explicitly represents (i.e., contains schemas for) the components of representation languages, including itself. The primitive building blocks of representation languages are larger and more abstract than the primitives of general programming languages in order to make them easier and more natural to use. Building blocks of a representation language include such things as control regimes (agendas, backward chaining, etc.), methods of associating procedures with relevant knowledge (footnotes, demons, etc.), fundamental access functions (put/get, assert/match, etc.), automatic inference mechanisms (inheritance of various kinds), and specifications of intended semantics of the components (consistency constraints, etc.).

RLL is designed to help manage these complexities by providing (1) an organized library of such representation language components and (2) tools for manipulating, modifying, and combining them. Rather than produce a new representation language as the "output" of a session with RLL, it is rather the RLL language itself, the environment the user sees, which changes gradually in accord with his commands.

The RLL system needs to be developed into a usable package, and experimented with. Only through multiple usages will directions for future research be revealed. Several systems are already planned for (some layer of) RLL, including: a new version of the system for diagnosis of pulmonary function disorders; a program for guiding a physician in constructing a new expert system automatically; and a few non-medical applications.

Already, we have isolated several core research issues, which will govern the direction of our research during the next five years. This agenda of issues includes:

(1) Incorporating the representational schemes of other researchers into RLL. For instance, the user should be able to specify that he or she wants a KRL-like environment, or a MYCIN-like environment, and the bundle of "organ-stops" which must be adjusted should change immediately.

(2) Codifying knowledge about representation. This includes refining our taxonomies of inheritance modes, control structures, etc.

(3) Building up our stock of ideas about fundamental representation issues: dealing with nested quantification, mass nouns, time, intensional objects, counterfactual conditionals, etc.

(4) Easier knowledge acquisition. One approach to this is to improve the interface to an expert user, who must transfer his knowledge into a program. For example, the knowledge acquisition program mentioned

above can direct its knowledge acquisition process because it possess a detailed model of what comprises such a session. A second, and currently underexplored, approach is to have the program automatically discover the knowledge for itself. This may appear much more costly, but recall that "expert knowledge" breaks down into facts and heuristics. The latter are almost never articulated by experts; it is easier to induce them from examples. This leads us to study:

(5) Automatically discovering new domain-dependent heuristics. This was the critical lack in an earlier discovery system, AM [22], which had some success in automatically discovering new (albeit elementary) concepts, by combining old ones. Our work in the past two years has indicated that powerful heuristics can be found as simple patterns in the values of slots, provided the system has very useful domain-specific slots. Thus this is pointing us to the problem which follows:

(6) Automatically discovering new domain-dependent slots which prove useful. Our approach, as usual, is to explicate and codify. We are building a taxonomy of slots; i.e., of useful relations between concepts (units). Already the number of slots is in the hundreds, and over the next five years we expect this number of different kinds of slots worth distinguishing to increase by an order of magnitude. This in itself will raise several new issues.

(7) Ultimately, tackling the problem of automatically discovering new representations of knowledge. Currently, our only plan to attack this problem is to represent each type of representation (e.g., graphical, schematized, linguistic,...) as a unit, organize these into a hierarchy, and see if the domain-independent heuristics are adequate to guide the search for new and better representation schemes.

6.3.1.2 Research on Planning

In many situations, solving problems by trial and error can be prohibitively expensive or impossible. One of the characteristics of an intelligent problem solver is the ability to formulate a "plan" before acting. Consider, for example, a physician ordering costly or risky tests or a chemist designing an experiment or a businessman trying to get to the airport. Much of the research in Artificial Intelligence has been concerned with formalizing this idea of planning in the form of intelligent computer programs. One approach has been to concentrate on techniques applicable in all task domains. Another approach has emphasized the importance of techniques specific to particular task domains. The experience in building high performance programs like DENDRAL, MYCIN, and MACSYMA has shown us the value of the latter, "knowledge-based" approach to system construction. However, even within this approach there are many domain independent questions yet to be answered. Consequently, we propose to explore some fundamental issues of planning that promise to increase performance and facilitate the construction of expert systems.

Research using SUMEX has demonstrated the value of the knowledge-based approach to planning program construction in the MOLGEN program. By extending the technology, we expect to enable similar success in other types of experimental design. The basic planning research we are proposing here will mesh nicely with a collateral effort to create an "intelligent agent" that has mastered the facts and lore of using complex computer systems and can use this knowledge to facilitate a user's interactions with the system (ARPA-funded research).

In our research on planning, we view problem solving as a three step process. Given a goal to satisfy, the problem solver uses information about the actions it can perform in synthesizing a plan to solve the problem. Then it executes the plan, possibly monitoring its performance to confirm success or detect failures. In the event of a failure (perhaps due to unforeseen circumstances), the problem solver must rectify any undesirable consequences and create a new plan.

A. Plan Formulation

The outstanding problems in plan formulation involve the use of strategical (or meta-level) control of planning, the development of a good representation for plans and planning methods, and the encoding of powerful planning techniques.

A.1 Meta-Planning

The operation of many planning programs can be described in terms of the "queue and process" model. The program maintains a data structure representing a partially designed plan and a queue of operations to perform on this data structure. An interpreter selects an operation from the queue and executes it, thereby expanding or refining the plan and possibly adding new operations to the queue. The problem of deciding the order in which to select members from the queue is a strategical one, and a variety of techniques have been proposed to solve it. One approach is to annotate each operation with a number reflecting its cost and probability of success. A more powerful approach is to view the selection task as a problem in its own right, on which the full power of the problem solver can be brought to bear. This latter approach is usually termed "meta-planning".

Stefik [23] has recently shown the power of this approach in the design of laboratory experiments. The MOLGEN program uses a level of strategical planning to direct the operation of the basic plan generator in designing gene-cloning experiments. His meta-planning techniques allow the program to choose between constraint propagation or a guess and backup approach.

We propose to consolidate Stefik's success by extracting the domain independent skeleton of MOLGEN and by developing additional techniques. We are interested particularly in the following questions.

(1) What is the appropriate structure for a system with meta-planning capabilities? Also, how many meta-levels are ideal?

(2) What techniques are there in addition to those used by Stefik?

A.2 Representations for Plans and Planning Methods

Sacerdoti's work on the NOAH system [25] has pointed out the importance of a flexible representation for plans that does not force one to make premature decisions about the order of actions or the identity of essentially arbitrary objects. Nevertheless, his "procedural net" formalism has several limitations, e.g. there is no way of representing conditionals, loops, or actions with parameters. We propose to develop an extension of his formalism that remedies many of these deficiencies.

Sacerdoti describes a procedural net as "a network of actions at varying levels of detail structured into a hierarchy of partially ordered time sequences. An action at a particular level of detail is represented by a single node in the network." [25] Each node may represent a "primitive action" or may point to a subnet of more detailed subactions. When executed in proper order, the subactions achieve the effects of their parent.

A "parameterized procedural net", or PPN, is a procedural net in which each node has associated with sets of input and output objects and sets of "prerequisites" and "postrequisites" (conditions that must be true for the action to succeed and those that become true after its execution). In Sacerdoti's formulation, each action node is described in terms of specific objects in the task domain. In a PPN the dataflow into and out of an action node is described in terms of other actions without naming any specific domain objects. Thus, a PPN is like a partial program. We believe the PPN formalism is an adequate representation for plans that allows complete flexibility in the specification of action type, control flow, dataflow, etc. However, we would like to study its formal properties and expressiveness, and we need to develop an appropriate interpreter to execute plans in this representation.

The extreme flexibility of the PPN notation makes it an excellent choice for encoding planning techniques as well as the plans they produce. The only difficulty is that its two-dimensional character makes it more difficult to use than the strings of characters used in conventional programming languages. Consequently, we propose to develop a one-dimensional version, similar to LISP except with multiple return values, nondeterministic function calls, and explicit representation of prerequisites and postrequisites.

A.3 Basic Planning Methods

The AI literature describes many domain independent planning techniques. Consider, for example, Newell and Simon's means-end analysis, prerequisite achievement, and Sussman's and Stefik's constraint propagation techniques. We would like to assemble a library of such planning techniques all encoded within our planning formalism, to be put at the disposal of a system builder. The library should also include domain specific techniques. Friedland [24] has already made some progress in this direction in the domain of molecular genetics. We would like to continue this work and strive to provide convenient methods for users to develop and edit these libraries.

B. Execution Monitoring

Once a plan is formulated, it can be executed. In many domains there is the possibility of failure due to unforeseen circumstances, e.g. a chemical synthesis has an unacceptably low yield or a computer system runs out of disk space. In other cases, a failure may occur due to partial or incorrect knowledge about the domain (as described in the last subsection). To trap such problems, the problem solver must monitor the execution of the plan. Execution monitoring is an important problem that has not yet received adequate attention. The key questions that must be answered include the following.

(1) How does one monitor the execution? In many cases, devising the test is as difficult a problem as solving the original problem. What special techniques are necessary in this regard?

(2) What aspects should be monitored? Which have the greatest diagnostic value? Which aspects are most likely to be violated? Given that testing is not free, how does one decide when to monitor?

The reason for monitoring each step of a plan rather than just the final outcome is that failures can propagate and that one can miss the opportunity to take corrective action.