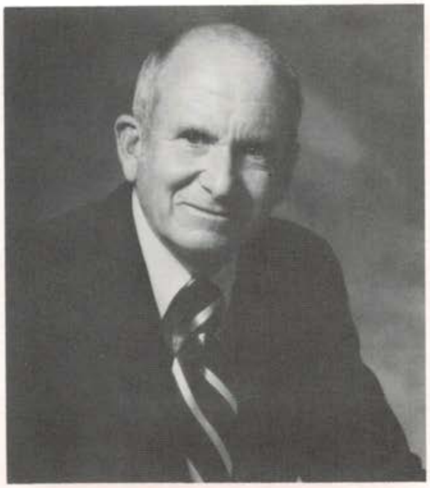
USC/INFORMATION SCIENCES INSTITUTE

USC/Information Sciences Institute
4676 Admiralty Way, Marina del Rey, California 90291
1981

INFORMATION SCIENCES INSTITUTE



Information Sciences Institute is approaching its tenth year of operation as a center for basic and applied computer science research. Though located 11 miles off campus, ISI is a major facility of the University of Southern California— one of three large research institutes administered by the university. ISI's programmatic autonomy within the university structure gives it the freedom to identify and engage in significant research programs of its own choice; its stability and continuity provide a supportive environment for those programs.

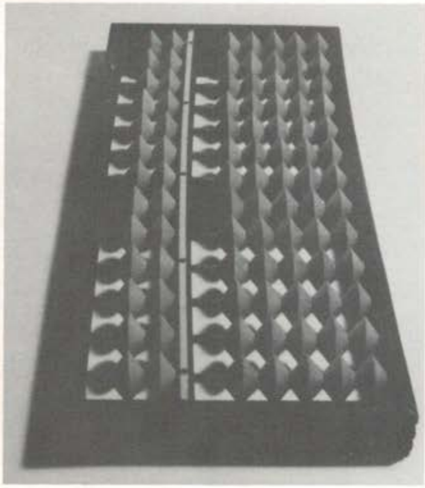
The Institute maintains a close relationship with USC's School of Engineering, particularly the Departments of Electrical Engineering and Computer Science. Some ISI staff members hold adjunct faculty appointments at USC; others teach at USC and neighboring universities. ISI benefits from interaction with local faculty and graduate students and provides opportunities for research and Ph.D. thesis supervision to graduate students from USC and other universities.

The Institute overlooks the Pacific Ocean from Marina del Rey, a suburban recreational community containing the largest man-made small boat harbor in the world. The Marina is located approximately 20 minutes from downtown Los Angeles, in the midst of some of the most desirable living areas in the Los Angeles basin.

ISI's 137 staff members (88 professionals) cooperate to provide a coherent yet diverse research program, with the Institute's interdisciplinary nature enhancing the fruitfulness of individual projects. A special democratic quality pervades the Institute at all levels. There are, for example, no second-class citizens as far as access to computing is concerned: each person has always been equipped with his or her own terminal.

Project leaders are given significant responsibility for the conduct of their projects and the direction of their research. Responsibility and professional reward are distributed among all research staff members; ISI provides all researchers visibility to program sponsors and to the information processing community at large, one aspect of which is encouragement to interact with others in the field and to present papers at professional conferences. Performance of professional staff members is gauged in part by their creative influence on the research community outside ISI and their success in identifying new areas of research that are of interest to others in the field.

ISI's computing facilities currently comprise five DEC KL-10s, one KI-10, a VAX 11/780, personal computers (Alto, Perq, and Apple), several PDP-11 minicomputers, and specialized devices for research in areas such as graphics, speech, and emulation. ISI's responsibility for providing a significant share of the computer cycles available to the ARPANET community serves the Institute's overall goal of high quality research, since available and reliable computing power is a basic necessity for performing research in this field; the inherent economy of scale is such that service to others serves ISI's interests as well. We are now planning a significant new computing environment to support future research needs. The goal is to provide each user with a substantial increase in address space and computer cycles (both personal and time-shared) and advanced terminals for office and home use. ISI also has a well-equipped electronics lab and machine shop for constructing any special hardware that projects require. Staffed by a full-time librarian,

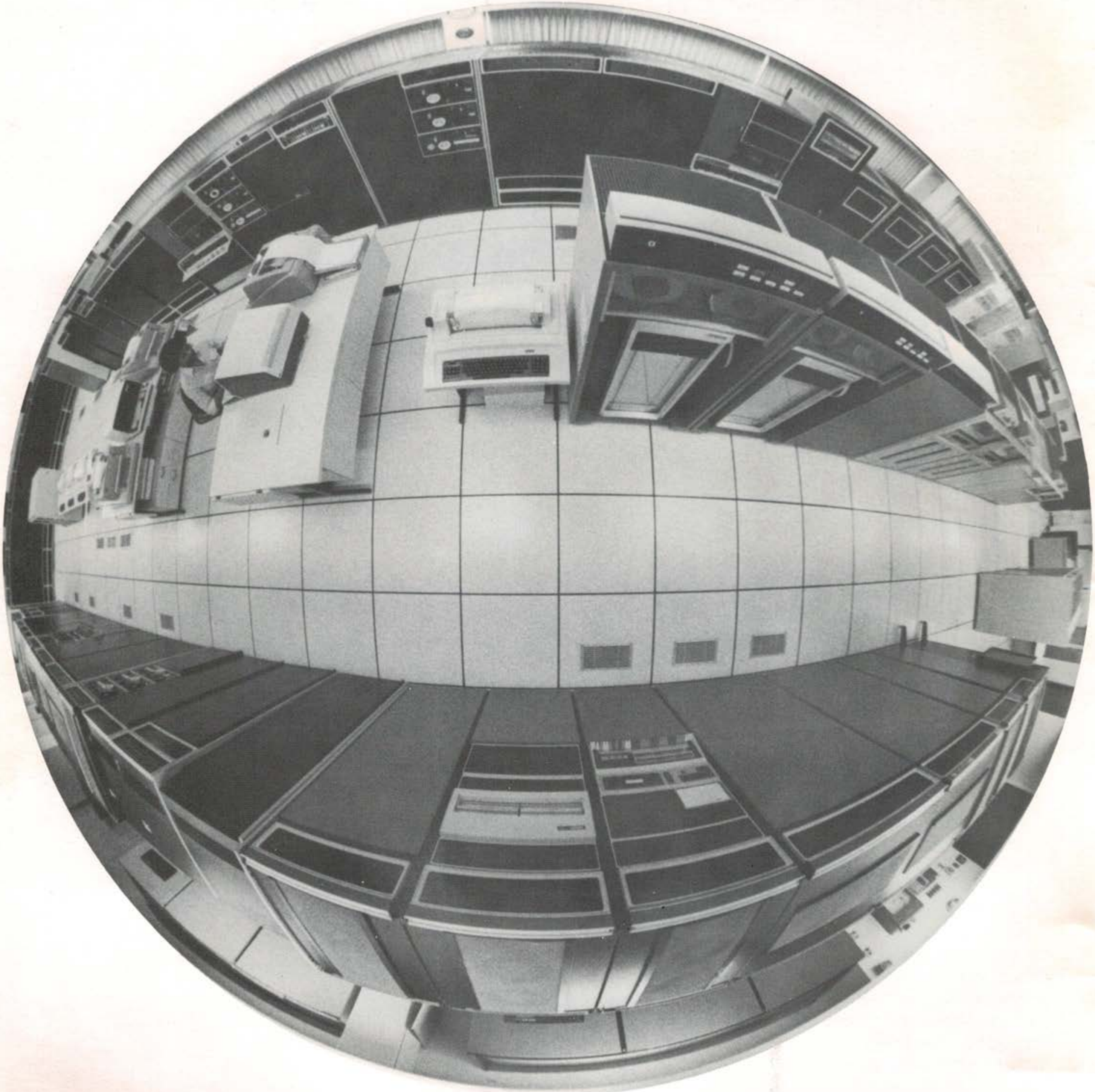


collection of current technical reports, subscriptions to more than 100 technical journals, and on-line access to several bibliographic databases.

ISI's major goal is the initiation, performance, and dissemination of significant computer science research. In some cases the design, development, and maintenance of complete systems follows (for example, the Sigma Message System, whose successful completion required—in addition to research into user needs and system design—terminals, communication lines, system maintenance, and user documentation). "Technology transfer" through reports, prototypes, or complete system installation is the end product of research at ISI.

A look to the future promises even more opportunity for research excellence. As information processing continues to dominate global society in the 1980's, there is a continuing need to remain flexible and shift emphasis as research opportunities unfold. ISI has made a commitment to discovering, tracking, and completing research of especial significance in the field of information processing for the world of the 1980's.

Keith Uncapher



CURRENT RESEARCH *Areas*

While ISI's individual projects pursue research in diverse application domains, four major areas currently dominate the research efforts: artificial intelligence, system specification, communication and networks, and computing environments.

Artificial Intelligence

The field of artificial intelligence is on the verge of having major impact on how computers are used to help solve problems. AI research at ISI includes both the investigation of fundamental problems in the representation and use of knowledge in the machine and the experimental application of AI technology to real-world problems. Current activity is focused in the following areas:

Inference

- Control of inference that must deal with large amounts of knowledge
- Tradeoffs between detailed knowledge representation and sophisticated control structure
- Plausible inference and the representation of uncertain knowledge

Knowledge base design

- Representation and control of large knowledge bases
- Acquisition of domain-dependent knowledge

Expert systems

- Architectures for expert systems
- Modeling of expert knowledge for delivery to nonexperts

Natural language

- Dialogue comprehension
- Generation of English text
- Adaptation of parsers to new domains of discourse
- Informal specifications

Projects working in several different application domains contribute their individual viewpoints to this ongoing reseach. The *Consul* project is examining the use of knowledge-based inference to provide a domain-independent cooperative interface to a set of interactive services. *Hearsay-III* is a domain-independent architecture for developing expert systems. *SAFE* investigates the use of built-in process-description knowledge and dynamically acquired domain knowledge to produce formal program specifications from informal descriptions. The *Knowledge Delivery* project is applying the results of basic linguistic research to the problem of constructing multiparagraph English text. The *Dialogue Comprehension* project is developing a model of human verbal dialogue and testing it by implementing a system that understands conversations.

System Specification

System specification occupies a central role in the attempt to make the software development process more tractable. At ISI, we are investigating the following aspects of specification:

Representation

What notations should be used for describing a formal specification? *(GIST, Formal Semantics, Consul)*

Creation

What can be done to make it easier to create a new formal specification? What are the criteria for a good specification? *(GIST, SAFE)*

Understanding

Are different or augmented presentations needed to make formal specifications understandable? Can the structuring mechanisms of informal communication help? Are special tools needed? *(GIST, Consul)*

Behavior

How will a specified system behave? How does that behavior compare with the specifier's intent? Analysis tools include verification, simulation, abstraction, and multiple views. *(GIST, Formal Semantics, AFFIRM, VMAS)*

Implementation

How is the implementation process broken into small steps and recorded? How are existing components incorporated into an implementation? *(TI, CAST, VLSI)*

Modification

Given a specification, how can its modification be facilitated and controlled? Which implementation decisions need to be revised and/or reconsidered? What flexibilities are required in the components so that they can be used in many situations and so that their behavior is itself adaptable? *(TI, CAST)*

One particularly interesting problem common to most of these aspects is determining the proper mix of human-machine cooperation. At one extreme, the computer can be a helpful clerk to a human making all the interesting decisions; at the other extreme, it can be expected to provide a significant function automatically.

REV B ROMS
2640A TERMINAL
HEWLETT · PACKARD
RESET TERMINAL
BREAK
DISPLAY FUNCTIONS
BLOCK MODE
REMOTE
CAPS LOCK
MEMORY LOCK
AUTO LF
TAB
ENTER
BAUD RATE

Communication and Networks

Communication among computers has become an ımportant part of computer science research, not only as a topic itself, but also in support of other areas. ISI has a major effort in bettering computer communications through the development of networks and network protocols ranging from office environments to international networks.

Applications

Advanced applications systems use integrated multimedia data in interactive conferencing, briefing aids, and computer mail. The user should be given choices among media including text, voice, graphics, facsimile, and video.

Environments

The communication environment is an interconnected system of networks of many different types, including long-haul (ARPANET), satellite (SATNET, WBNET), radio (PRNETs), and local (LCSNET, ETHERNET). A user and his computation processes should be able to access facilities across this entire environment.

The *Internet Concepts* project is working in the areas of protocol design, application, and verification. The *Wideband Communication* project is working in the areas of speech and video digital communication using a broadcast packet satellite network. The *Command and Control Graphics* project is working on device-independent color graphics to support command and control systems. The *Personal Communicator* project is developing a handheld multimedia terminal to enable the user to carry much of his computing and communication environment with him. The ISI *Computation Resource* supports a variety of computer communication facilities and programs.

Computing Environments

ISI's interest in computing environments spans the full range of interactive facilities, from the maintenance of a large service-oriented ARPANET node to experiments with a coherent environment for providing natural language interaction with the user. The collective goal of ISI projects pursuing research in this area is a deeper understanding of three aspects of cooperative interaction in computing environments:

Cooperation among services

Unification of diverse hardware (including a variety of personal machines) and system software facilities into a single computing environment that is coherent from a user software viewpoint.

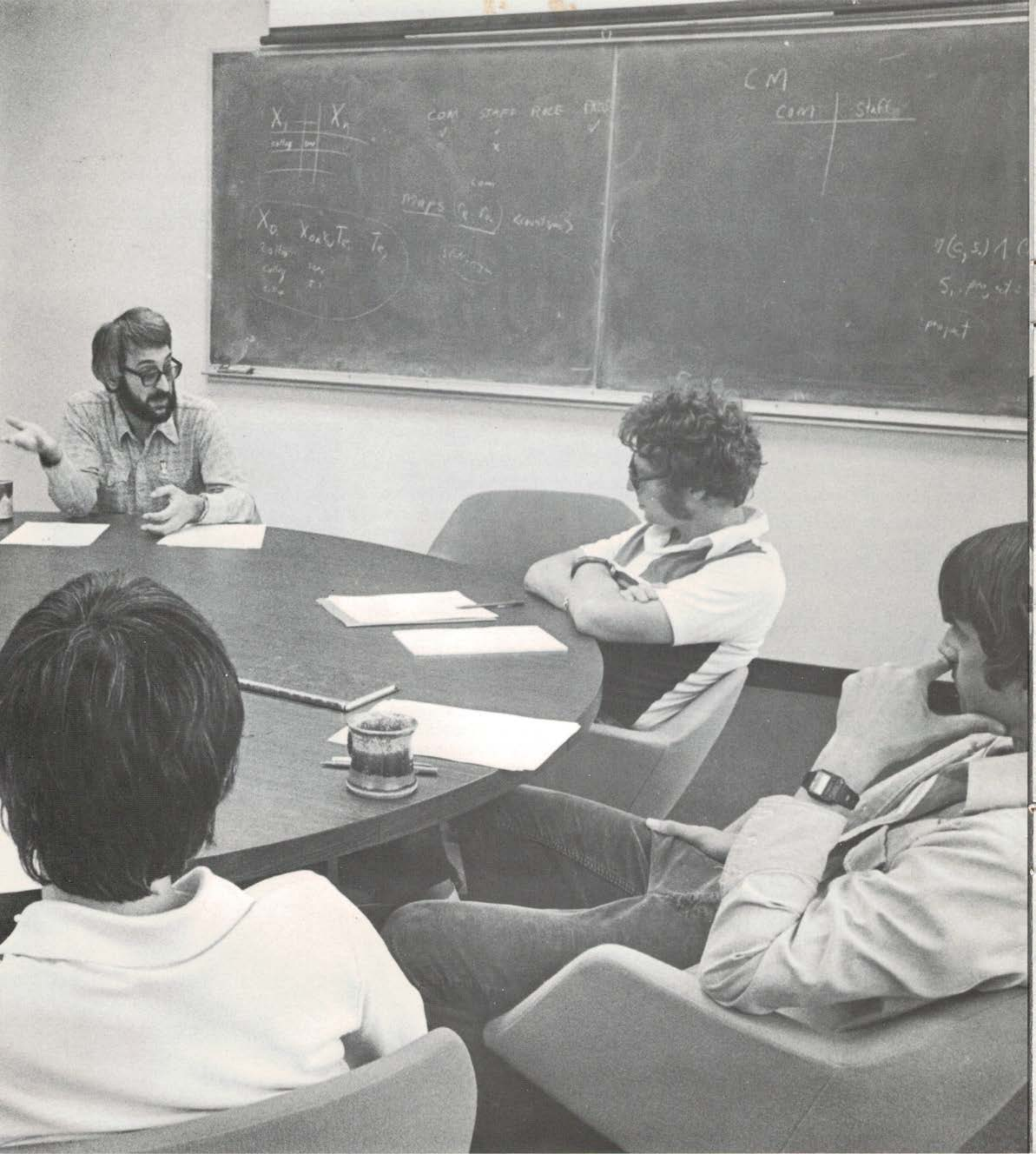
Cooperation among users and services

Creation of a single interface to all services in the environment that interacts naturally with the users (understanding requests and providing help).

Cooperation among users

Support for group efforts in program development, document preparation, and other office and administrative interactions, i.e., a systemwide information management facility.

The *Information Processing Center* provides stable, continuous facilities. The *New Computing Facility* project is planning for near- and far-term advances to support the needs of ISI's research community. The *CAST* project addresses the problem of software reconfiguration and reuse. *Consul* is attempting to provide a uniform, friendly user interface, while *Command and Control Graphics* concentrates particularly on graphical interfaces. The *Personal Communicator* explores how to provide integrated support to a user remote from the system. The *Wideband Communication* and *Internetwork Concepts* efforts deal with how to link computers effectively. The *INTERLISP* project is developing an easily portable version of the language and its support system.


CM
COM
Staff
COM STAFF RACE PROG
Maps
project

Cooperative Interactive Systems (Consul)

William Mark, David Wilczynski, Robert Lingard, Thomas Lipkis, and Bill Swartout

Current interactive systems must be made more habitable for a wide variety of users, especially those having little experience with computing. We are currently building the Consul system to explore methods for allowing natural interaction—natural-language requests, explanation of system facilities, user-understandable error handling—between a user and a set of online services (for text manipulation, message handling, etc.). Consul works by mapping user requests into appropriate system actions as dictated by a detailed model of system capabilities and requirements. This model is based on built-in service-independent knowledge, but is tailored to individual services as part of the service-building process. Our research therefore includes not only modelling and mapping methods that generalize across services, but also a programming environment for service builders that integrates their services into Consul's knowledge base.

Expert Systems Architecture (Hearsay-III)

Bob Balzer, Jeff Barnett, Lee Erman, Phil London

Hearsay-III is a system for building and experimenting with knowledge-based expert systems. Based on the Hearsay-II architecture, it provides a structure for defining diverse knowledge sources *(KSs)* and a global database, called the *blackboard,* for their communication. A KS can react to blackboard changes it cares about and, when executed, make changes of its own, reflecting its knowledge of the situation. Hearsay-III also provides the same facilities (KSs and blackboard) for dealing with the *scheduling problem*—the decision at each point of which KS to execute. Hearsay-III is implemented in INTERLISP and is currently being used to develop the *SAFE* system and a portion of the TI system.

Specification Formulation (SAFE), Testing (GIST Validation), and Implementation (TI)

Bob Balzer, Wellington Chiu, Don Cohen, Lee Erman, Martin Feather, Steve Fickas, Neil Goldman, Phil London, Bill Swartout, and Dave Wile

This project comprises three interrelated efforts working to facilitate the creation, testing, and optimized implementation of program specifications. The first effort, called *Specification Acquisition From Experts (SAFE),* is directed at helping people create unambiguous, consistent, and complete formal program specifications through informal description. While end users are quite capable of providing informal process-oriented descriptions of the task being automated, formalisms of any kind provide major impediments. The informal descriptions are characterized by partial, rather than complete, constiucts. The system uses a knowledge base of program well-formedness rules to disambiguate and complete the informal natural language descriptions. An early version of the system has successfully converted several small informal specifications into formal specifications. Attention is now focused on handling large specifications through incremental formalization; this new implementation uses *Hearsay-III* as its base.

The second effort seeks to ensure that a formal specification matches the user's intent, by systematically investigating its behavior through symbolic execution. The tasks here include the design of a formal abstract specification language, called *GIST*, the development of techniques for symbolic execution (some of which have already been developed within the SAFE context), and the explanation of the simulated behavior.

The third effort, called *Transformational Implementation,* seeks to provide a methodology guaranteed to produce only valid implementations of the formal specification by involving the computer in the implementation process to ensure that the evolving implementation remains consistent with the original formal specification. This is accomplished by limiting the activity of programmers to a choice of optimizations. These optimizations are embodied in a catalog of transformations, applied by the system in response to the programmer's choices, after verifying that they are applicable. A prototype grammar-based system has been implemented; current efforts are focused on structuring the steps of the development process and on facilitating the highest level optimization decisions—those that implement the nonalgorithmic aspects of specifications.

Knowledge Delivery in Multiparagraph Text

Bill Mann and Jim Moore

This project is developing new methods for autonomous text composition by machine, with the focus at the larger-than-sentence level. The project is producing a series of knowledge delivery systems, the first of which, called *KDS* (Knowledge Delivery System), was developed as a testbed for methods of text composition from computer-internal knowledge representations such as semantic nets. Innovations in KDS included a new expressive paradigm that does not rely on dividing the semantic net into sentence-sized pieces.

A second system, called *PENMAN,* is currently being developed to explore the problems of creating a portable knowledge delivery facility useful in multiple knowledge domains. PENMAN will seek to deliver knowledge (in English) from inside a system that was not designed to have a knowledge delivery component. The linguistic components of PENMAN are being based on Halliday's Systemic Grammar. A large systemic grammar of English has been implemented and is being fitted with semantic parts.

Dialogue Comprehension

Bill Mann and Jim Moore

Despite its great social significance, human communication remains a complex, poorly understood process. Human-computer interaction could be made easier and more effective if computers could comprehend strings of symbols in more human-like ways. Our prior research has created a model of human communication as a kind of goal-pursuit activity. The model represents four principal levels of knowledge in communication: goals being pursued, speech acts used to pursue goals, propositions

exchanged while performing speech acts, and linguistic symbols used to convey all of these.

One of the major prerequisites for such a model of communication is a strong parser of English. As part of an effort to implement the model, we have developed a parser that is an extension of the DIAMOND parser developed at SRI; the grammar is by Jane Robinson. Formal evaluations indicate that this parser is much more robust than its predecessors. The project is also developing a new action representation for dialogue.

Program Verification (AFFIRM)

Susan Gerhart, Rod Erickson, Stan Lee, and Dave Thompson

Program Verification means mathematically proving (as opposed to testing) the consistency of programs with their specifications. Its achievement requires formal methods and tools for programming and specification languages and for theorem-proving support. The Program Verification project has developed an experimental system emphasizing data abstraction and interactive theorem proving. This system, called *Affirm* (fully documented and available over the ARPANET), has been used for specifications and proofs of several examples: numerous (now standardized) data structures, a large file-updating module, portions of a security kernel, and high-level protocols. Allied areas of contribution are programming language design (EUCLID, ALPHARD, and Ada) and programming methodology.

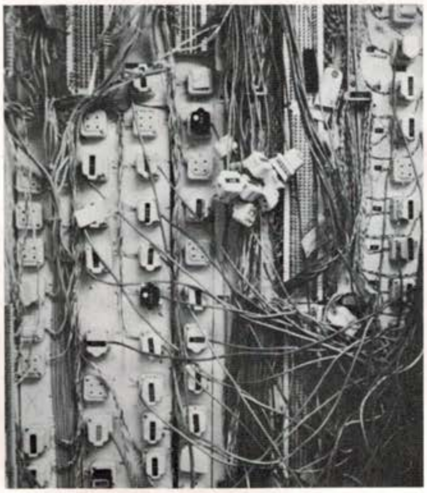
Adaptable Software Components (CAST)

Susan Gerhart, Rod Erickson, Stan Lee, and Dave Thompson

This project is concerned with the scientific principles and engineering practice of making software reusable by composing it from adaptable components: basic building blocks of software which can be reshaped and assembled. The research falls into four areas: identification, specification, verification, and assembly.

Specification and verification will use the techniques developed in AFFIRM. Both are absolutely necessary for software which is expected to be reused. In addition, they play a significant role in identification and assembly by providing the underlying analytic and synthetic methods for manipulating descriptions, assumptions, and actual program structures. Identification and assembly require a blend of programmer intuition and formal methods, addressing the questions of what makes good components and assemblies.

Still in its initial stage, this project will analyze an existing piece of software (probably a spelling corrector), identifying components within it, specifying and verifying these, and then assemblying the original and some variations of it. Besides a library of components and a newly structured piece of software, the experiment should produce various informal rules for generalizing and specializing components and extended AFFIRM methods for supporting these.



Verification Models, Algorithms, and Systems (VMAS)

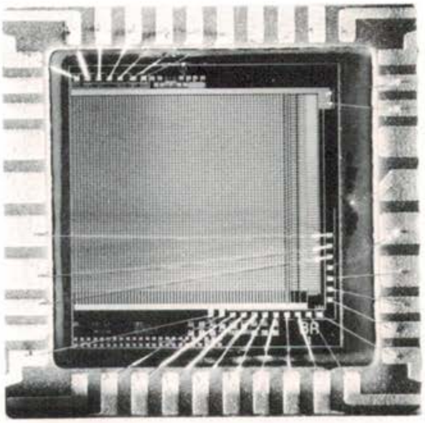
Steve Crocker, Jeff Cook, Vittal Kini, Leo Marcus, Dave Martin, Bill Overman, Sarma Sastry, and Craig Taylor

The goal of this project is to solve some of the engineering and mathematical problems impeding development of practical verification systems. We are presently focusing on three issues.

At the theoretical level, the critical open problem is how to represent concurrent actions in a multiprocessing system. We are experimenting with extensions to state deltas to model multiple interacting processors, including measures of time.

At the practical level, we are concerned with the invention of efficient decision and semidecision procedures for important mathematical domains; our work builds on the cooperating decision procedure model. Domains of interest are integer arithmetic, bit-string arithmetic, bags, sets, and sequences. Other domains are likely to be selected in the future.

A corollary concern is how the user should control a proof system. The proof system should be algorithmic, and the user should have an extremely clear view of the system's capabilities. In addition to the specification of *what* is to be proven, the user must also supply the guidance to the system showing *how* to carry out the proof. This guidance will be written in a proof language, and development of a suitable proof language is the third focus of this work.



Formal Semantics

Steve Crocker, Vittal Kini, and Dave Martin

The Formal Semantics project is carrying out research into methods for the validation and use of formal semantic definitions of programming languages. The Ada Formal Semantic Definition (FSD), written by the French group at INRIA, is being used as a vehicle for this research. A program which interprets the Ada FSD will be used as a tool to validate the FSD. The interpreter with the FSD installed within it will comprise a semantic-checker which takes Ada programs as input and either indicates where and why a program does not conform to the language semantics or else provides as output the denotations of the program. In the course of constructing the semantic-checker, the FSD will need to be parsed and type-checked, thus ensuring that syntactic and other minor errors in FSD are revealed. It is expected that the semantic-checker will be employed in two phases. In the first phase, Ada programs, with known and well-understood semantics, will be used as test case inputs. This phase will exercise the FSD, help to uncover deeper and less obvious errors in it, and instill confidence in its correctness. In the second phase, the checker may be used in answering detailed questions of interpretation of the language semantics. In addition to benefitting the definition of Ada directly, the project will provide insights into improving the state of the art in defining the formal semantics of practical programming languages.

Personal Communicator

Tom Ellis, Alan Kay, Steve Saunders, and Rick Shiffman

This effort explores the design and application of personal, portable computer communication terminals that provide text, graphic, and voice modes and synchronized combinations of them. The existence of full-time, location-independent, handheld access to communications will allow full exploitation of the area coverage and dynamic flexibility of packet radio. The goal of the exploratory design and experimental operation of model communicators within a multimode message system is to expose the functional and architectural parameters of the terminal and its user interface behavior in order to guide and support future designs of operational portable communicators and their associated systems.

New Computing Facility

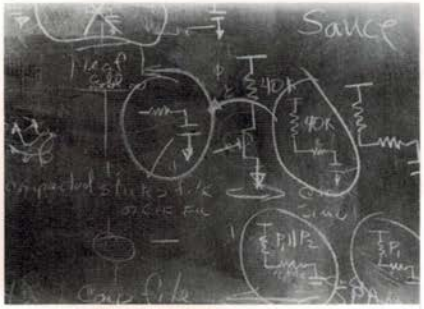
Danny Cohen and Dan Lynch

The purpose of this work is to significantly improve computation facilities for researchers at ISI and for remote and mobile users as well. The key concept is the design of a computing and information delivery system to match user needs. The major objectives are to design and implement a computing facility to support research appropriately and to initiate a set of experimental and development steps to be taken in the interim (or, "how to get there from here"). This facility will consist of many computers, including several large time-sharing systems, smaller ones for dedicated purposes, such as voice and graphics, and many personal computers. It will use high-speed communication among these computers and for communication with the outside as well (ARPANET, WBNET, etc.).

Internetwork Concepts

Jon Postel, Danny Cohen, Greg Finn, Alan Katz, Paul Mockapetris, Suzanne Sluizer, and Carl Sunshine

Many packet-switched computer communication networks now exist; this project is investigating ways to interconnect them usefully. The project has three task areas: the formal analysis of protocols, the design and prototype implementation of internetwork applications, and the design of protocols and development of new communication concepts.



The protocol analysis area is focusing on the correctness (in the program verification sense) of the Transmission Control Protocol (TCP). TCP is the host-to-host reliable data stream protocol developed for the internet environment. This task is exploring several program and protocol analysis tools.

The protocol applications area is focusing on computer mail. Two distinct systems are being developed: a text multinetwork mail system for use in the near term, and a multimedia mail system for use in the long term. Other applications will be explored in later stages of the project.

The protocol design and concepts area is focusing on the host-to-host and gateway level protocols, particularly the Internet Protocol (IP), a datagram protocol which is universal in the internet environment. There are many design issues to be resolved—for example, very mobile hosts, controlled routing, access control, fault isolation, and resource allocation.

Wideband Communication

Steve Casner, Bill Brackenridge, Danny Cohen, Randy Cole, and Ian Merritt

After several years of research with relatively low bandwidth packet networks, ISI is participating in a *Wideband Communication Program* (initiated by ARPA and the Defense Communication Agency) to investigate packet switching on a high-bandwidth satellite network. ISI has two goals. First, we want to develop the technology required to support future packet speech systems with thousands of voice channels, beginning with an experimental facility that can be accessed by a much broader user community than that of the first demonstration systems; this will help to determine the feasibility of realizing the economies of integrated voice/data packet transmission. Second, we want to explore new modes of packet communication made possible by the increase in bandwidth of the satellite network over that of previous packet networks and to investigate how the added bandwidth could benefit old modes of communication. ISI is working toward these goals through system design, development, and experimentation. Included are systems for transmission of narrow-band packet video, multiple channels of packet speech, graphics, text, and bulk data.

Command and Control Graphics

Richard Bisbey, Ben Britt, and Dennis Hollingworth

The Command and Control Graphics project is developing a distributable, display-device-independent vector graphics system for use in a command and control environment. The user communicates with the system via a device-independent graphics language whose primitives are mapped on internal graphics protocols from which device-specific graphics orders are generated. The project is also developing graphic command and control application programs, including a Situation Display (naval force information displayed on a geographic background) and a Briefing Aid system that demonstrate the use of the graphics system. The software is currently in use at the Naval Ocean Systems Center as part of the Navy's Advanced Command and Control Architectural Testbed, as well as at sites connected by the ARPANET and the secure subnet.

VLSI Implementation System

Danny Cohen, Yehuda Afek, Ron Ayres, Joel Goldberg, Gideon Hess, Dave Johannsen, Lee Richardson, and Vance Tyree

The VLSI project is focused on research in the design methodology of VLSI circuits at various levels and on the development of the tools needed for providing fast turnaround fabrication service to the ARPA VLSI research community.

Research is progressing in the automatic generation of chips from purely functional (non-geometric/algorithmic) specification. Two distinct approaches and their integration are being pursued. First, we are working on a silicon compiler, called *Bristle Blocks,* which produces a limited class of chips, namely microprocessors, with remarkably efficient designs, competitive with manual designs. Second, we are pursuing a silicon compiler, *RELAY*, which accepts the specification of arbitrary synchronous systems in terms of hierarchically described synchronous logic equations. We note that the choice of appropriate specification language is of fundamental importance and that the target medium (silicon) has a character quite different from that of software.

We are also developing the concepts involved in managing useful libraries for popular subdesigns ("procedures"). Unlike software libraries, useful silicon libraries require many implementations of the same function. The geometric alternatives presented in multiple implementations provide for the possibility that people will find a design that fits nicely within the geometric constraints imposed by other parts of their design.

ISI is also developing and maintaining the *MOSIS system,* which supports the fast turnaround fabrication requirements of the ARPA VLSI research community. MOSIS handles many of the issues of information processing, geometric manipulations, quality control (by wafer probing and device testing), and general management. In addition to the management-oriented tasks, the MOSIS team is working on the development of standard acceptance tests and other facilities required for developing the interfaces needed to support the separation of the design from the fabrication process.

INTERLISP—VAX Implementation

Bob Balzer, Dan Lynch, Dave Dyer, and Hans Koomen

This project is implementing and will maintain a fully compatible, portable, large-address-space INTERLISP. The first version is being implemented in C under UNIX for the VAX computer and will be released in March, 1982.

STAFF

Keith Uncapher, *Executive Director*
Thomas O. Ellis, *Deputy Director*
Robert Blechen, *Institute Administrator*

Research Staff

Ronald Ayres
B.A., Mathematics, California Institute of Technology, 1974
Ph.D., Computer Science, California Institute of Technology, 1979
Language processing, software and hardware compilers, recognizing fundamental properties of information as they relate formal and informal beings

Robert Balzer
B.S., Electrical Engineering, Carnegie-Mellon University, 1964
M.S., Computer Science, Carnegie-Mellon University, 1965
Ph.D., Computer Science, Carnegie-Mellon University, 1966
Transformation-based programming, formal specification, expert systems, design, informal language

Jeffrey A. Barnett
Methods of plausible reasoning, control structures for AI systems, representations of knowledge, theory and development of programming languages, theory of distributed evaluation, combinatorics

Richard L. Bisbey
A.B., Mathematics, University of California, Los Angeles, 1967
M.S., Information Systems, University of California, Los Angeles, 1968
Graphics

William A. Brackenridge
Wideband communication

Ben Britt
B.S., Electrical Engineering, University of Southern California, 1975
M.S., Computer Science, University of Southern California, 1976
Graphics

Stephen L. Casner
B.A., Mathematics, Occidental College, 1973
M.S., Computer Science, University of Southern California, 1976
Real-time multiprocess system design, network protocols, voice and video bandwidth compression, restricted domain programming

Danny Cohen
B.Sc., Mathematics, Technion, Israel Institute of Technology, 1963
Ph.D., Mathematics/Computer-Science, Harvard University, 1969
VLSI, machine architecture, real-time communication, computer communication protocols, computer image generation, cockpit information/flight simulation

Don Cohen
B.S., Mathematics, Carnegie-Mellon University, 1973
Ph.D., Computer Science, Carnegie-Mellon University, 1980
Reasoning, learning, specification languages

E.R. (Randy) Cole
Professional Engineer, Mathematics, Colorado School of Mines, 1967
Ph.D., Electrical Engineering (Computer Science), University of Utah, 1973
Speech processing, image processing, fast signal-processing computers, computer graphics

Stephen D. Crocker
B.S., Mathematics, University of California, Los Angeles, 1968
Ph.D., Computer Science, University of California, Los Angeles, 1977
Verification, formal semantics, software production technology

David Dyer
B.A., Mathematics, Immaculate Heart College, 1975
User interfaces, very large databases

Thomas O. Ellis
B.S., Engineering, University of California, Los Angeles, 1951
M.S., Engineering, University of California, Los Angeles, 1953
Man-machine communication, hardware systems architecture

Roddy W. Erickson
B.A., Electrical Engineering (Computer Science option); Biochemistry, Rice University, 1977
M.S., Computer Science, University of California, Los Angeles (in progress)
Mechanical theorem provers, axiomatic specification, computerized teleconferencing

Lee D. Erman
B.S., Mathematics, University of Michigan, 1966
M.S., Computer Science, Stanford University, 1968
Ph.D., Computer Science, Stanford University, 1974
Architectures for and control of knowledge-based systems, human/machine expert systems, informal specification, distributed artificial intelligence, speech-understanding systems

Martin S. Feather
B.A., Mathematics and Science, Cambridge University, 1975
M.A., Mathematics and Computer Science, Cambridge University, 1976
Ph.D., Computer Science and Artificial Intelligence, University of Edinburgh, 1979
Methodologies for program development and maintenance, computer-assisted program implementation, program specification, program transformation

Gregory G. Finn
B.S., Physics, Brandeis University, 1973
M.S., Computer Science, University of Southern California, 1977
Computer networking, multimedia message systems

Louis Gallenson
BSEE, Electronics, University of California, Berkeley, 1951
Local networks, system architecture, terminals, CPU I/O devices, emulations and microprogrammable CPUs

Susan L. Gerhart
B.A., Mathematics, Ohio Wesleyan University, 1965
M.S., Communication Sciences, University of Michigan, 1967
Ph.D., Computer Science, Carnegie-Mellon University, 1972
Program and specification verification methods, fault-tolerant computing, user interface design and evaluation, fallibility in programming

Joel Goldberg
B.S., Chemistry, California Institute of Technology, 1967
M.S., Computer Science, University of Wisconsin, 1969
Emulation and emulation support systems, graphics, distributed computations

Neil M. Goldman
B.S., Mathematics, Stanford University, 1969
M.S., Computer Science, Stanford University, 1971
Ph.D., Computer Science, Stanford University, 1973
Formal specification languages, computer aids for program and specification synthesis, computer generation of natural language

Gideon Hess
B.Sc., Electrical Engineering, Technion, Israel Institute of Technology, 1968
M.Sc., Electrical Engineering, Technion, Israel Institute of Technology, 1975
M.Sc., Computer Science, California Institute of Technology, 1977
Ph.D., Computer Science, California Institute of Technology, 1980
Modeling and simulation, real-time systems, programming methodology

Dennis Hollingworth
B.A., Mathematics, University of California, Riverside, 1967
Graphics

David L. Johannsen
B.S., Electrical Engineering, California Institute of Technology, 1978
M.S., Computer Science, California Institute of Technology, 1978
Ph.D., Computer Science, California Institute of Technology, 1981 (expected)
Computer-aided VLSI design, hardware description languages, software language systems

Alan C. Kay
(On sabbatical from Xerox Palo Alto Research Center)
B.A., Mathematics, University of Colorado, 1966
M.S., Computer Science and Physiology, University of Utah, 1968
Ph.D., Computer Science, University of Utah, 1969
Personal computing, novice programming, man-machine interaction

Vittal Kini
B. Tech., Electrical Engineering (Electronics), Indian Institute of Technology, Bombay, 1973
M.S., Electrical Engineering (Computer Engineering), Carnegie-Mellon University, 1975
Ph.D., Electrical Engineering, Carnegie-Mellon University, 1981
Program verification, programming language semantics, hardware reliability, hardware design automation

Johannes A.G.M. (Hans) Koomen
B.Sc., Computer Science, University of British Columbia, 1978
M.Sc., Computer Science, University of British Columbia, 1980
Cognitive science, knowledge representation, heuristics of control flow, problem solving, AI programming languages

Stanley Lee
S.B., Applied Mathematics, Harvard College, 1977
Ph.D., Computer Science, University of California, Berkeley (in progress)
Program verification, formal semantics of programming languages

Robert W. Lingard
B.A., Mathematics, University of California, Los Angeles, 1963
M.A., Mathematics, University of California, Los Angeles, 1965
M.S., Computer Science, University of Southern California, 1970
Ph.D., Computer Science, University of Southern California, 1975
Knowledge representation, software methodology, knowledge-based inference, knowledge acquisition

Thomas A. Lipkis

B.S., Information and Computer Science, University of Southern California, 1978
Ph.D., Computer Science, University of California, Irvine (in progress)
Automated explanation of expert systems, knowledge representation and modeling in expert systems

Philip E. London

B.S., Physics, University of Maryland, 1973
M.S., Computer Science, University of Maryland, 1975
Ph.D., Computer Science, University of Maryland, 1978
Informal program specification, computer-assisted program implementation, modeling and representation in problem solving, knowledge-based systems

Ralph L. London

B.A., Mathematics, Washington-Jefferson College, 1958
M.S., Mathematics, Carnegie-Mellon University, 1960
Ph.D., Mathematics (Systems and Communications Sciences), Carnegie-Mellon University, 1964
Specification languages, design and construction of modifiable programs, personal programming environments and work stations

William C. Mann

B.S., Electrical Engineering, Lehigh University, 1956
M.E.A., Engineering Administration, George Washington University (D.C.), 1963
Ph.D., Computer Science, Carnegie-Mellon University, 1973
Natural language processing, artificial intelligence, knowledge representation, linguistics, natural inference

Leo Marcus

B.S., Mathematics, Harvey Mudd College, 1966
M.Sc., Mathematics, Hebrew University, Jerusalem, 1968
Ph.D., Mathematics, Hebrew University, Jerusalem, 1975
Partial decision procedures for program verification, implementation of data types and axiomatic theories, semantics of specification languages, model theory

William S. Mark

S.B., Computer Science, Massachusetts Institute of Technology, 1974
S.M., Computer Science, Massachusetts Institute of Technology, 1974
E.E., Electrical Engineering, Massachusetts Institute of Technology, 1975
Ph.D., Computer Science, Massachusetts Institute of Technology, 1976
Representation of inferential knowledge, expert systems

David F. Martin
(Professor of Engineering and Applied Sciences, UCLA)
B.S., Engineering, University of California, Los Angeles, 1960
M.S., Engineering, University of California, Los Angeles, 1962
Ph.D., Engineering, University of California, Los Angeles, 1966
Semantics of computation, semantics of programming languages, correctness proofs of programming systems implementation

Ian H. Merritt
Communications, information systems, telephony and radiotelephony, human behavior

James Moore
S.B., Mathematics, Massachusetts Institute of Technology, 1964
Ph.D., Computer Science, Carnegie-Mellon University, 1971
Natural language, AI/knowledge engineering

William T. Overman
B.S., Chemistry, University of California, Los Angeles, 1973
M.S., Computer Science, University of California, Davis/Livermore, 1975
Ph.D., Computer Science, University of California, Los Angeles, 1981 (expected)
Practical verification systems, concurrency and verification of concurrent systems, machine description and hardware realization

Jon Postel
B.S., Engineering, University of California, Los Angeles, 1966
M.S., Engineering, University of California, Los Angeles, 1968
Ph.D., Computer Science, University of California, Los Angeles, 1974
Computer communication protocols, interprocess communication, distributed applications

Leroy C. Richardson
B.S., Electrical Engineering, Carnegie-Mellon University, 1965
M.S., Electrical Engineering, Carnegie-Mellon University, 1966
Ph.D., Electrical Engineering, Carnegie-Mellon University, 1972
Engineering, applications programming

Steven E. Saunders
S.B., Electrical Engineering, Massachusetts Institute of Technology, 1972
Ph.D., Computer Science, Carnegie-Mellon University, 1979
Design of pocket-size personal computers and their software, flat-panel displays, digital sound synthesis, customized compact interpretive code, authentication and encryption via public-key systems

Richard R. Shiffman

Vector graphics hardware, interactive graphics software, VLSI chip design, use of microprocessors in personal computers

Suzanne Sluizer

B.S., Mathematics, University of Illinois, 1975

Distributed databases, message systems, network and internetwork protocols

Carl Sunshine

B.A., Physics, University of Chicago, 1971

M.S., Computer Science, Stanford University, 1973

Ph.D., Computer Science, Stanford University, 1975

Computer networks, communication protocol design and analysis, network interconnection

William Swartout

B.S., Mathematical Sciences, Stanford University, 1974

S.M., Computer Science, Massachusetts Institute of Technology, 1977

Ph.D., Computer Science, Massachusetts Institute of Technology, 1981

Computer-generated explanations of programs. computer-aided program implementation, knowledge representation, knowledge-based systems

David H. Thompson

B.A., Mathematics, University of Colorado, 1974

M.S., Computer Science, University of Toronto, 1975

User interface issues, program verification, algorithm specification methodologies, system composition from pre-verified components, computers for the people

Vance C. Tyree

BSEE, Electrical Engineering, University of California, Berkeley, 1965

MSEE, Electrical Engineering, University of California, Berkeley, 1966

VLSI design automation systems, fault-tolerant VLSI circuitry, highly concurrent processors on silicon

David Wilczynski

B.A., Mathematics, University of California, Los Angeles, 1968

M.S., Computer Science, University of Southern California, 1972

Ph.D., Computer Science, University of Southern California, 1975

Intelligent user interfaces, knowledge-based systems, software design methodologies

David S. Wile

Sc.B., Applied Mathematics, Brown University, 1967

Ph.D., Computer Science, Carnegie-Mellon University, 1974

Programming language design, specification languages, programming methodologies, programming environments

Graduate Research Assistants

Yehuda Afek, UCLA
Wellington Chiu, USC
Jeff Cook, UCLA
Steve Fickas, UC Irvine
Alan Katz, UCLA
Steve Klein, UCLA
Christian Matthiessen, UCLA
Paul Mockapetris, UC Irvine
Sarma Sastry, USC
Daniel Schwabe, UCLA
Craig Taylor, UC Irvine

Administration

Administrative Support

Cindy Glasenapp
Chloe Holg
Lyvingston Holmes
Phyllis Keiser
Jayne Millar
Joyce K. Reynolds
Suzanne Robb

Research Project Support

Monique Benarrosh
Victor Brown
Lori Holzer
Manon Levenberg
Lisa Moses
Pamela Norton
Phyllis O'Neil
Linda Sato
Debby Shishkowsky
Audree Smith
Lisa Vail

Business Office

Bob Brown
Gail Geppert
Judy Gustafson
Maureen Jester
Julie Kcomt
Toni Leon
Gina Maschmeier

Technical Support

Development Laboratory

Bob Parker

Electronic

Vernon Dieter
Jeff LaCoss
Jerry Wills

Mechanical

Shorty Garza
Bob Hines
Herb Yallop

Graphics

Nelson Lucas

Library

Sally Hambridge

Publications

Jim Melancon
Sheila Gardner

Information Processing Centers

Dan Lynch, Director
Gary McGreal, Deputy Director
Dale Russell, Technical Assistant

Systems Software

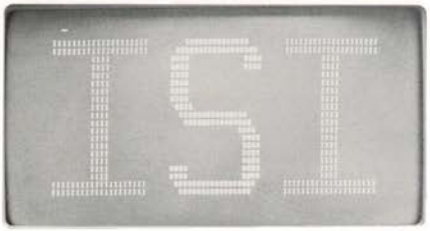
Dennis Smith
- Dale Chase
- Bill Moore

Applications

Lynne Sims
- Ray Bates
- Al Cooperband
- Joel Goldberger
- Ellen Lohneiss
- Pamela McDaniel
- Craig Rogers
- Barden Smith

Hardware

Dick Fiddler
- Charlie Brown
- Phil Crowe
- Glenn Gauthier
- Ray Mason
- Dan Pederson
- Leo Yamanaka



Operations

Jim Hurd

User Services

Jim Koda
- Vicki Gordon
- Debbie Williams

Operators

Ron Shestokes
- Steve Cinquemani
- Kathy Cousins
- Walter Edmison
- Jeryl Harper
- Andrea Ignatowski
- Norman Jalbert
- Joseph Kemp
- William Lee
- David McMinn
- Keith Miles
- Craig Ward
- Mike Zonfrillo

San Diego Facility

Serge Polevitzky
- John Metzger
- Scott Smith
- Tom Wisniewski