NASA Technical Memorandum 85631

NASA Computer Science Research Program Plan

NASA Intercenter Planning Committee for Computer Science

MARCH 1983

25th Anniversary
1958-1983

NASA

NASA Technical Memorandum 85631

NASA Computer Science Research Program Plan

Prepared by NASA Intercenter Planning Committee for Computer Science

National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1983

EXECUTIVE SUMMARY

The overall goal of the Computer Science research program is to provide the technical foundation within NASA to exploit advancing computing technology in aerospace applications. This goal will be realized through a program of generic research and experimentation which focuses on developing core skills within the Agency in disciplines critical to NASA, and on maintaining a strong university base of fundamental research in aerospace computer science.

Principal objectives are:

1. To provide the theoretical and technology base needed to develop advanced aerospace computing concepts and to evolve advanced system architectures in response to unique aerospace requirements,
2. To increase the Agency's knowledge and ability to develop high quality aerospace-related systems and software, and
3. To provide advanced theory, concepts, techniques, and capabilities for the effective use and management of aerospace information.

The authors of this plan examined the computing requirements and challenges facing Agency programs, and the state of the art of relevant computer science. Care was taken to construct a research program which will contribute substantially to the nation's technological base, focusing on broad issues which are either critical or unique to NASA's mission. The result is a program constructed around the following three "themes":

Concurrent Processing

Highly Reliable Cost-Effective Computing

Scientific and Engineering Information Management

Concurrent Processing addresses system architectures, languages, and algorithms for computationally intensive problems in aeronautics and space research, such as computational fluid dynamics and image processing. The research focuses on concurrency issues for tightly coupled multiprocessors and loosely coupled cooperative distributed systems.

The proposed research on Highly Reliable, Cost-Effective Computing addresses the technology underlying the construction of systems appropriate for long-duration unattended space missions as well as man-rated aeronautic and space flight vehicles. It includes investigations of fault-tolerant hardware architectures as well as cost-effective tools and techniques for developing verifiably correct software.

NASA is presently accumulating approximately ten billion words of scientific space-derived data per day, with expectations that this will grow a hundred-fold over the next decade. The third theme, Scientific and Engineering Information Management, focuses on the fundamental questions underlying the effective management and distribution of this data to an active scientific community.

The rapid rate of change in computer technology presents an enormous challenge to the Agency in utilizing the best developments to accomplish the NASA missions. It is the intent of this plan to provide the directions for building the required knowledge base in computer science for the Agency.

Concurrence:

Lee B. Holcomb
Head, Computer Science
and Electronics Office

Leonard A. Harris
Director, Aerospace
Research Division

September 1982

TABLE OF CONTENTS

PREFACE

This program plan was developed by a committee of representatives from eight NASA Centers and the Office of Aeronautics and Space Technology, NASA Headquarters. The committee had an inaugural meeting December 1, 1981, and two other meetings: February 10-11, 1982 and June 9-11, 1982. All other business was conducted by teleconference and electronic mail. The plan itself was developed using the AUGMENT system, of the TYMSHARE Office Automation Division.

The plan includes a taxonomy of Computer Science (similar to the new ACM Computing Reviews Classification System; Comm. ACM, Vol.25, No.1, Jan. 1982) and a brief technical summary of the state of the art of each of the major computer science categories of this taxonomy, written by members of the committee and others from their Centers.

The plan also includes a functional breakdown of NASA programs under the three broad categories: Aeronautics Research and Development, Space R & T, and Institutional Support. These programmatic areas were assessed against the computer science categories to identify the critical areas of the computer science research which were most needed, could contribute to NASA programs, and would not be advanced outside the Agency. From this assessment came the identification of the three themes of this research program: Concurrent Processing, Highly Reliable Computing, with an underlying concern for cost-effectiveness, and Scientific and Engineering Information Management.

Members of the committee, listed below, are to be commended for their efforts.

Susan J. Voigt, Chairman

NASA Intercenter Planning Committee for Computer Science

Edward S. Chevers
Johnson Space Center

Carl I. Delaune
Kennedy Space Center

Theodore E. Fessler
Lewis Research Center

Ronald L. Larsen
Headquarters, Aerospace Research Division

Guy M. Lohman
(Dec. 1981 to March 1982)
Jet Propulsion Laboratory

J. Thomas Renfrow
(March 1982 to October 1982)
Jet Propulsion Laboratory

Paul B. Schneck
Goddard Space Flight Center

Kenneth G. Stevens, Jr.
Ames Research Center

Susan J. Voigt, chairman
Langley Research Center

J. B. White
Marshall Space Flight Center

ACRONYMS

Adabas — Database management system commercially available from Software AG

ADP — Automatic Data Processing

ASC — Advanced Scientific Computer; a parallel, vector-processing computer built by Texas Instruments

ARC —NASA Ames Research Center

ASEE — American Society for Engineering Education

ATC — Air Traffic Control

CAD/CAM — Computer Aided Design & Computer Aided Manufacturing

CDC — Control Data Corporation

CFD — Computational Fluid Dynamics

CIG — Computer Image Generation

CMOS — Complementary Metal-Oxide Semiconductor

CODASYL — Committee on Data Systems Languages (group that developed COBOL and the Data Base Task Group database model)

CODMAC — Committee on Data Management and Computation, National Academy of Sciences

CP — Concurrent Processing

CRT — Cathode Ray Tube, Display screen for interactive computer terminal

CS — Computer Science

DBMS — Database Management System; DBM may also stand for database machine

DFRC — NASA Dryden Flight Research Center

FAA — Federal Aviation Administration

FFT — Fast Fourier Transform

Galileo — Deep Space planetary mission

GFLOPS — gigaflops; billion floating point operations per second

GP-B — Gravity Probe - B

GRO — Gamma Ray Observatory

HAL/S — Flight computer programming language used on the Space Shuttle and some other NASA flight projects.

HOL — Higher Order Language

ICASE — Institute for Computer Applications in Science and Engineering, located at NASA Langley Research Center

IEEE — The Institute of Electrical and Electronics Engineers, Inc.

ILLIAC — A parallel processing computer built at the University of Illinois; the ILLIAC IV was in use on the ARPAnet from NASA Ames Research Center for several years.

ISO — International Standards Organization

JPL — Jet Propulsion Laboratory

LANDSAT — Earth Resources Technology series of satellites

LaRC — NASA Langley Research Center

LPS — Launch Processing System

MACSYMA — Symbolic Manipulation Language and System, developed at MIT under Project MAC

MIMD — Multiple Instruction, Multiple Data Stream

MISD — Multiple Instruction, Single Data Stream

NAS — Numerical Aerodynamic Simulation; A high speed, large capacity computer complex for scientific computation, to be established at NASA Ames Research Center

NASTRAN — NASA Structural Analysis computer program

NMOS — N-type Metal-Oxide Semiconductor

OI — Operational Instrumentation

RIACS — Research Institute for Applications of Computer Science, located at NASA Ames Research Center

ROSS — Remote Orbital Servicing System

RTOP — Research and Technology Objectives and Plans, the formal proposal for a research activity within NASA; funding and progress are tracked by RTOPs.

SAR — Synthetic Aperture Radar

SECDED — Single Error Correction, Double Error Detection

SIMD — Single Instruction Stream, Multiple Data Stream; Architecture of a class of parallel processors like the ILLIAC IV.

SISD — Single Instruction Stream, Single Data Stream; Architecture of classical single CPU computers.

Smalltalk — An object oriented computer language, developed primarily at Xerox Palo Alto Research Center

SOC — Space Operations Center

SOS — Silicon on Sapphire

ST — Space Telescope

STS2 — Space Transportation System 2, the second Space Shuttle mission.

UNIX — Operating system written at Bell Laboratories, now available for DEC VAX and many other computers.

VAX — A class of Digital Equipment computers

VHSL — Very High Speed Logic

Viking — NASA Mars exploration mission

VLSI — Very Large Scale Integration

WYE — Work Year Equivalent, NASA unit of manpower on RTOPs; also work years of effort.

I. INTRODUCTION

This plan is the result of a seven month cooperative effort by representatives of the NASA Centers to define a fundamental research program in computer science based upon an assessment of NASA program areas. The impetus for the plan stems from several external studies done for the Agency in the past few years.

SAGAN Committee

The Study Group on Machine Intelligence and Robotics was established in June 1977 to assist NASA technology program planners to determine the potential in these areas. The study group, chaired by Carl Sagan of Cornell University, published their findings and recommendations in March 1980 [Sagan]. The conclusions begin with the following statement:

> "We believe that NASA should institute a vigorous and long-range program to incorporate and keep pace with state-of the art developments in computer technology, both in its space-borne and its ground-based computer systems; and to ensure that advances, tailored to NASA's mission, continue to be made in machine intelligence and robotics. Such advances will not occur of their own accord. Many NASA requirements in computer architecture and subsystem design will in turn have a stimulating effect on the American computer and microprocessor industry, which now faces an extremely strong challenge by foreign competition."

This 1980 report goes on to make several recommendations, the first of which is: "NASA should adopt a policy of vigorous and imaginative research in computer science, machine intelligence, and robotics in support of broad NASA objectives." This plan addresses the research program in computer science called for by this recommendation.

CODMAC

The Space Science Board of the National Academy of Sciences formed the Committee on Data Management and Computation (CODMAC) in the summer of 1978. The Board requested the scientist members to examine the management of existing and future data acquired from spacecraft and associated computations in the areas of the space and earth sciences and to make recommendations for improvements from the point of view of the scientific user.

The CODMAC report, published in 1982 [CODMAC], presents several recommendations, including the following:

> "We recommend that NASA have an ongoing technology management activity encompassing all areas of data systems.... The program should formulate research and development efforts in those areas where NASA and science or applications users would benefit from new developments."

NASA/ASEE 1981 Summer Study

A third study, which was the primary motivator for the development of this plan, was conducted by a 1981 NASA/ASEE faculty summer study group and was called Computer Science: Key to the Space Program Renaissance [Freitas].

This study concluded that failure to recognize the importance of computer science as one of the major contributing disciplines to NASA's efforts means that NASA cannot successfully meet the challenges of the future.

Three recommendations were made by the study group, and these are summarized below.

1. NASA establish a program office of computer science and technology charged with providing technological support to NASA's missions by establishing, promoting, and coordinating relevant computer science and technology at the NASA Centers and Laboratories.

2. NASA create coordinated information systems by developing a program for the establishment, use, and growth of general resource sharing networks, integrated scientific data/information bases, and an automated work environment to include executive and professional workstations.

3. NASA develop personnel programs for aggressive recruiting and effective retention of computer scientists and professional programs for better interaction with the external computer science community.

The authors of this plan considered the recommendations of each of these studies in constructing the proposed program of computer science research. In addition, an independent assessment was made of the computing requirements and challenges facing Agency programs, and a state-of-the-art assessment of relevant computer science disciplines was performed. Details of these can be found in the appendices. Care was taken to construct a generic program of research which will contribute substantially to the nation's technological base, focussing on broad issues which are either critical or unique to NASA's mission. The result is a program constructed around the following three "themes":

Concurrent processing

Highly-reliable cost-effective computing

Scientific and engineering information management

Concurrent processing addresses system architectures and algorithms for computationally intensive problems in aeronautics and space research, such as computational fluid dynamics and image processing. It includes research on highly parallel Single Instruction Multiple Data (SIMD) stream computer architectures for vectorizable problems, as well as the more complex but also more promising Multiple Instruction Multiple Data (MIMD) and Data Flow architectures.

The proposed research on highly reliable, cost-effective computing addresses the technology underlying the construction of systems appropriate for long-duration unattended space missions as well as man-rated aeronautic and space flight vehicles. It includes investigations of fault-tolerant hardware architectures as well as cost-effective tools and techniques for developing verifiably correct software.

NASA is presently accumulating approximately ten billion words of scientific space-derived data per day, with expectations that this will grow a hundred-fold over the next decade. The third theme, scientific and engineering information management, focuses on the fundamental questions underlying the effective management and distribution of this data to an active scientific community.

Specific research directions are presented in the chapter entitled Approach. These are organized around the general themes described above.

The rate of change in computer technology presents an enormous challenge to the Agency in utilizing the best technology to accomplish the NASA missions. It is the intent of this plan to provide the directions for building the required knowledge base in computer science for the Agency .

II. GOALS AND OBJECTIVES

The overall goal of the Computer Science research program is to provide the technical foundation within NASA to exploit advancing computing technology in aerospace applications. This goal will be realized through a program of generic research and experimentation which focuses on developing core skills within the Agency in disciplines critical to NASA, and on maintaining a strong university base of fundamental research in aerospace computer science.

Principal objectives are:

1. To provide the theoretical and technology base needed to develop advanced aerospace computing concepts and to evolve advanced system architectures in response to unique aerospace requirements,

2. To increase the Agency's knowledge and ability to develop high quality aerospace-related systems and software, and

3. To provide advanced theory, concepts, techniques, and capabilities for the effective use and management of aerospace information.

Aerospace computing requirements push the state of the art in highly reliable and high performance systems. A goal of the computer science research program is to develop an understanding of the fundamental principles underlying this type of highly specialized computing, to develop an understanding of the relationship and tradeoffs between algorithms and computing architectures, and to apply this fundamental insight to the development of advanced computational concepts and optimal system architectures for aerospace applications such as computational modelling of physical processes, flight crucial systems, and autonomous systems.

NASA's annual software expenditures continue to rise, currently running approximately $400 million, yet effective techniques for designing, developing, operating, and maintaining aerospace software continue to be elusive. Cost over-runs, late deliveries, and compromised requirements are all too commonplace. Industrial organizations specializing in software development have set a goal to make software productivity improvement rates commensurate with those experienced in hardware. TRW, for example, believes it may be possible to quadruple software development productivity by 1987 through the use of advanced software development environments. A goal of the NASA computer science program is to provide NASA the theoretical and technology base to realize these productivity improvements for aerospace-related software.

The computer science research program is a fundamental program of generic research emphasizing theoretical and experimental investigation of issues of critical (and typically unique) importance to aerospace applications. Its major goal is to provide generalizable knowledge and insight which can be applied to substantially advance the capability of aerospace-related computing systems and improve the effectiveness and productivity of NASA.

III. APPROACH

INTRODUCTION

This section describes the proposed approach to be taken to institute and build a computer science research program within NASA. This approach resulted from an examination of the requirements of Agency programs and activities for improved computational capabilities. The analysis detailed the high degree of dependence the Agency has on computing and the importance of improvements in computational capabilities to make significant advances in most programs. Examples can be found throughout the spectrum of the Agency's activities, including high-speed computing for computational physics and experimental data analysis, highly reliable computing for ground-based systems, civil aircraft, and space flight systems, and specialized systems for image generation and information management. Details of this examination are presented in Appendix A.

The present state of the art of computer science and technology was examined to determine its ability to fulfill the Agency's requirements. In many cases the state of the art is moving in a direction which would permit the Agency to meet its computational needs by being a smart consumer. In other cases the Agency was found to be in a position where it could adapt the state of the art to meet its requirements. In a few cases it was found that the Agency's requirements would not be met and the Agency would have to actively engage in research and development efforts in computer science in order to meet key computational requirements of its programs. A brief technology assessment and details of this examination can be found in Appendix B.

An assessment of NASA requirements versus the state of the art in computing revealed three computer science research areas which address the major computer science requirements of the Agency. These three areas, which comprise the themes of this program, are: *Concurrent Processing, Highly Reliable Cost-Effective Computing, and Scientific and Engineering Information Management.*

AGENCY REQUIREMENTS

The first theme, *Concurrent Processing,* addresses the following four pressing Agency requirements:

1. Computational Physics, including computational fluid dynamics, weather/climate prediction, etc., requires computational speed and memory size beyond the capabilities of current technology. This issue is elaborated in **Future Computer Requirements for Computational Aerodynamics** (NASA CP2032), but the following two figures from that report indicate the need. Figure III-1 presents the stages of approximation to three-dimensional aerodynamic simulations, with stages III and IV being the critical ones in the next decade. Figure III-2 displays the trend in effective speed of general purpose computers, clearly demonstrating the computational aerodynamics requirement will not be satisfied unless concurrent processing can be applied to the high-speed computing problem.

2. Flight Simulation and other related aerospace man-in-the-loop simulations must become more realistic and more complex. The simulation of a helicopter, for example, is a significantly more computationally intense problem than simulation of fixed wing aircraft. It will require the development of specialized subsystems employing concurrency for applications such as computer image generation, command and control of the cockpit, and high-speed calculation of flight equations. Such applications do not provide the commercial market required to draw the necessary industrial attention to this critical problem. Only NASA sponsored R&D in concurrent processing aimed at these types of simulations will ensure that the Agency's requirements are met.

STAGE		APPROXIMATION	COMPUTER CLASS FOR PRACTICAL 3D ENGINEERING COMPUTATIONS
I PAST	INVISCID LINEARIZED (1960's)	VISCOUS AND NONLINEAR INVISCID TERMS NEGLECTED	IBM 360 CDC 6600
II PRESENT	INVISCID NONLINEAR (1977)	VISCOUS TERMS NEGLECTED	CDC 7600 STAR CRAY ILLIAC IV
III NEXT STEP	REYNOLDS TIME-AVERAGED NAVIER-STOKES (EARLY 1980's)	NO TERMS NEGLECTED: TURBULENT MOMENTUM AND HEAT TRANSPORT TERMS MODELED	AT LEAST 40 TIMES CURRENT SUPERCOMPUTERS
IV FAR FUTURE	FULL TIME- DEPENDENT NAVIER-STOKES (CIRCA 1990)	SUB-GRID SCALE TURBULENCE MODELED	AT LEAST 100 TIMES REQUIREMENT FOR STAGE III

Figure III-1. Stages of approximation to three-dimensional numerical aerodynamic simulations.

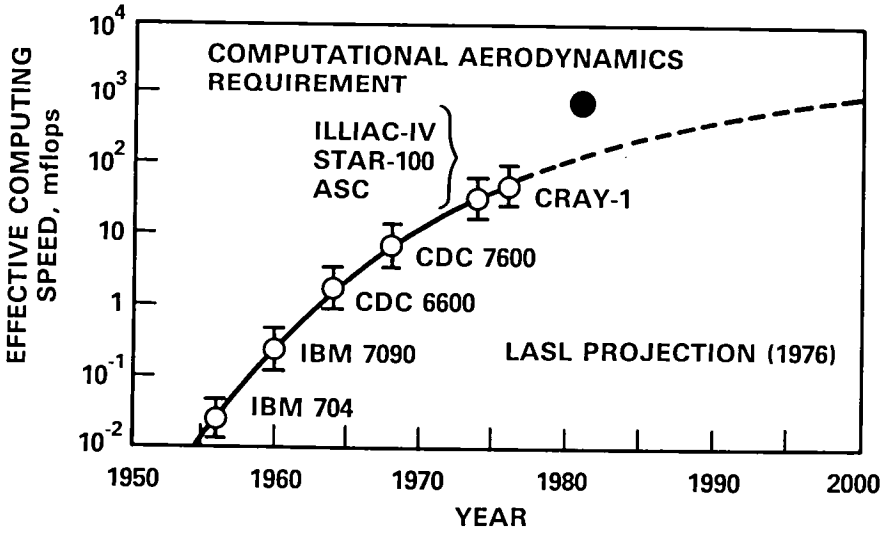

Figure III-2. Trend of effective speed of general-purpose computers.

3. Space systems, ranging from satellites with imaging sensors like SAR and Thematic Mapper to an orbiting space station or Remote Orbital Servicing System (ROSS), require on-board computing. Such specialized computer systems have computational speed, size and power constraints which can only be met through customized concurrent processing. In an orbiting space station, all major subsystems, including guidance and control, attitude control, environmental control, electrical power, data and communications will be controlled by autonomous subsystems made up of special and general purpose computers. These computers will be interconnected by highly reliable multi-path networks, permitting reconfiguration in the event of component failures as well as communication of information. Subsystems having real-time requirements may employ concurrent processing directly. The computer systems technology needed exceeds that which is being developed commercially. It is necessary, therefore, for NASA to sponsor the research and development of tools, methods, and approaches in order to develop the hardware/software system designs to meet the requirements of these programs.

4. Structural Analysis has been a major, focused program within NASA, resulting in widely used tools such as NASTRAN. Current technology permits the analysis of very large and complex structures using the method of substructuring. Problems with over one million unknowns have been solved using this technique. However, use of this method on a conventional computer requires the sequential solution of one substructure after another. Concurrent processing systems will permit several design teams to work simultaneously on the substructures, thus compressing the design calendar time and enabling exploration of more design alternatives. The capability to solve engineering problems by partitioning and concurrent processing of coordinated subproblems can be generalized from structures to multidisciplinary systems. A spacecraft design, for example, can be viewed as including structural dynamics, controls, and propulsion subproblems.

The second theme, *Highly Reliable, Cost-Effective Computing,* focuses on Agency requirements for flight-crucial systems, such as the engineering of unique computing systems to control spacecraft experiments or manned flight systems. Deep space missions such as Voyager and Galileo, for example, must be largely autonomous due to the large communication delays. A future Space Station will have a number of autonomous computer systems providing control for stability, environment, power generation, and a variety of experimental/industrial activities. Such systems must have high reliability and tolerance to faults, and yet be developed within a constrained budget.

There is a range of high reliability requirements of concern to NASA. These include: (1) extremely high reliability for relatively short bursts of time, such as automated shuttle reentry and touchdown, (2) extremely high reliability for moderate intervals such as flight legs of civil air transports, and (3) long term reliability for deep space missions where there is no opportunity for parts replacement or refurbishment. Reliability is a key issue because failure of these systems can lead to disastrous results (e.g., loss of life, loss of expensive equipment, or severe damage to same), or may lead to an inoperable system which can not be fixed (e.g., the object containing the system may be inaccessible in space). Automation of aircraft and shuttle systems to increase efficiency, reduce weight, lower operating costs, and improve performance capability hinges upon fully integrated, highly reliable digital avionics systems.

The progress toward increased reliability in digital flight control systems is shown in Figure III-3. Although device improvements have contributed to improved reliability, major advances in the 1970's came from systems architectures with triple and quadruple redundancy. Note that the reliability of commercial systems lags that achievable with state-of-the-art technology. The level of reliability for fully computerized flight control systems in the late 1980's requires fault-tolerant systems as indicated in the ellipse.

Computer system reliability is influenced by system complexity, design, implementation, maintenance, hardware components, and software. The two problems of complexity and reliability are interrelated. As the complexity of information systems increases, system reliability becomes much more difficult to en-

sure. The techniques presently employed to provide sufficiently high reliability for NASA missions are very expensive, labor-intensive and basically brute force. Present reliability modeling techniques are inadequate to handle system fault coverage parameters (e.g. latent, intermittent/transient, software, and hardware faults). New analytic techniques are needed for the design and evaluation of highly reliable fault-tolerant systems.

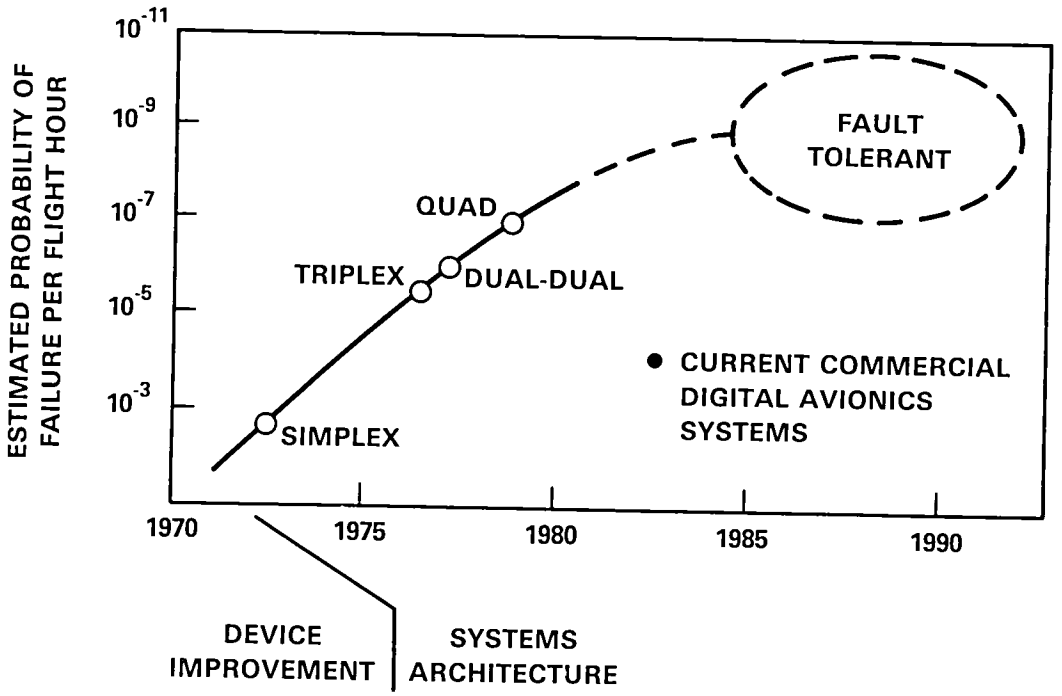

Figure III-3. Digital flight control systems reliability progress.

The third theme is *Scientific and Engineering Information Management.* NASA has accumulated an enormous amount of data which is stored primarily on magnetic tape at data centers or at the home institutions of principal investigators. The rate of data accumulation is increasing. In particular, earth-orbiting satellites have been returning ever larger amounts of data at an alarming rate. Earth-orbiting satellites are projected to increase yearly data volumes a thousandfold, from 10E13 to 10E16 bits/year, by 1995. The data rates of new sensors will increase from 15 megabits per second for the current multispectral scanner instrument, to 85 megabits per second for the Thematic Mapper, and to 200 to 500 megabits per second for the Multiple Linear Array and the SAR instruments of the late 1980's. Figure III-4 displays the increasing data rates for some future space missions. Magnetic tape is impractical for the storage of this large amount of data. A user of the data must have ways of determining what data exists, must be able to browse through the data, and must be able to retrieve desired data in a usable form. This is currently a very difficult and largely unsolved problem which will only increase in severity during the 80's and 90's.

The major use of data collected from experimentation, from aircraft, and from space exploration is analysis, either independently or correlated with data gathered by other sensors, missions, or means. In both cases it is essential that the researcher be able to tell the "pedigree" of the data, that is, the steps that it has gone through in the data translation and reduction process. When one wishes to do correlative research, running comparisons among multiple data sets, the data must be retrievable and translatable into comparable forms. Even if one knows the pedigree of the data, this translation process may be very diffi-

cult and time consuming. To generate, locate, and/or use algorithms to render this data in comparable forms may be very difficult. The user may not be familiar with the algorithms that are used or may want to use or develop other algorithms. Inadequate care and attention to the management of data can result in this effort becoming a major portion of the research activity, consuming scarce resources.

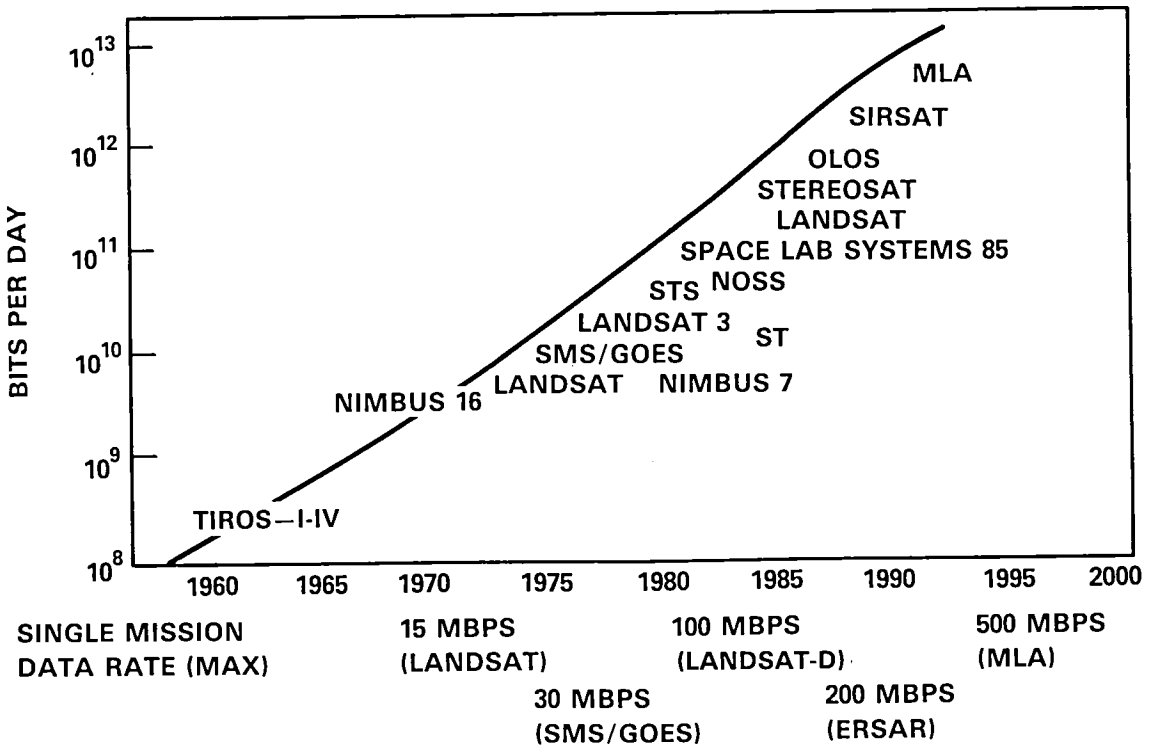

Figure III-4. Mission data rates.

To get some idea of productivity gains achievable in correlative research through automation, one may look at the difference between manual and online bibliographic searching done by librarians and researchers. Searches that once were performed manually can now be conducted much more easily and rapidly using online search mechanisms. It has become clear to users of these systems that the capability of automated systems far exceeds manual techniques. The cost effectiveness of online searching versus manual searching was quantitatively assessed in a 1978 study [Elcheson].

The study was conducted at Lawrence Livermore Labs, dealing primarily with scientific subjects. The results verify that online searching is generally faster, less expensive, and more effective than equivalent manual searching. Approximately five online searches could be conducted completely for every one manual search, assuming the existence of manual catalogues or indices in a central location (not always the case for NASA). The actual time spent in searching was between 3.5 and 22 times greater for manual searching than for online searching. The cost of online searches has decreased from $88 in 1968 to approximately $32 in 1977. Linear extrapolation applied to this data implies a cost of less than $10 in 1982. While linear extrapolation cannot continue indefinitely, the data clearly indicate the benefits NASA and NASA researchers can obtain from online access to data and related services.

Once the data is in a usable form for the scientist, tools must be available to facilitate its analysis. Current research indicates the nature of the interaction between man and machine is critical to effective and productive analysis. The system interface should be "friendly" and human-engineered in order to enhance rather than inhibit the scientist's productivity. The data should be presented in such a way that

the analyst can use the highly developed visual processing capabilities of the brain in conjunction with functions more efficiently performed by the system. The presentation of vector or scalar fields over two or three dimensions is a difficult process. Powerful, local user work stations augmented with computational resources such as data management systems and large, high-speed computers are needed to provide the scientific community with the required capabilities.

PROGRAM THEMES

CONCURRENT PROCESSING

This theme is concerned with the development of new computer architectures, operating systems, programming languages, and algorithms to exploit multiple processors in meeting NASA's computing needs. Tightly coupled multiprocessors (e.g. parallel processors) through cooperative distributed processing systems are addressed by this theme. Within NASA, concurrent processing has many diverse applications, from modular onboard spacecraft computing systems to supercomputers for computational modeling of physical processes.

Research Directions

Architectures

In the area of computer architecture, research shall be directed toward the development of tightly coupled multiprocessors for NASA applications which levy requirements not satisfiable by extant computer architectures. This research shall address Multiple Instruction Stream Multiple Data Stream (MIMD) architectures, Data Flow architectures, Systolic Arrays, and other novel approaches to configuring and controlling multiple processing elements on a single task. These architectures will be systematically characterized and their design parameters specified, including instruction sets, memory hierarchies, interconnection schemes, synchronization methods, and processing speeds. Simulation, emulation, and analysis methods will be developed for predicting performance of these novel architectures with respect to NASA computing requirements. In some cases actual prototypes will be constructed to verify the architectural definition procedure and the performance prediction methods. This ability to specify architectures tailored to particular classes of problems coupled with CAD/CAM capabilities for designing and constructing computers will permit the Agency to economically develop highly efficient computing systems to meet its diverse computational requirements.

In addition to tightly coupled multiprocessors, loosely coupled cooperative distributed systems will also be studied. The profusion of personal computers, database machines, information systems, high-speed processors, graphics systems, and control processors available and in use today present architectural challenges and opportunities heretofore unencountered. The interconnection of much of the computing capability of the Agency into a unified distributed system with automatic resource management determining the best facility for each task is one such opportunity which poses immense challenges. Research should be undertaken to coordinate sets of specialized processors on a single computational activity. As with the tightly coupled systems, this research will require the systematic characterization of system interconnection options, data communication and control techniques, and coordination mechanisms to bring the power of a distributed system to bear on a single task. In addition to this characterization, performance evaluation and prediction methods will have to be developed.

Given the characterizations and performance assessments of both tightly and loosely coupled concurrent system architectures, subsequent research will focus on the development of the knowledge, approaches, techniques, and tools necessary to design optimal concurrent systems from a statement of the computational requirements.

Operating Systems

Operating systems schedule and manage the diverse resources of a computing system. Research will address operating system structures and algorithms for concurrent systems which maintain efficiency and uniformity as perceived by the user. Dynamic resource allocation will be a major focus, enabling the user to view the system as homogeneous and maximizing the systems efficiency by allocating the most appropriate parts of the system to a particular task. This will require research into the implications of distributing an operating system over a set of heterogeneous processors, characterization of the system status and task requirements, and scheduling techniques. This research will build upon operating system theory developed for nonconcurrent systems and concurrent systems with homogeneous processors.

Languages

Programming languages provide the means by which the user expresses a problem to a computing system, but, more importantly, they also provide the framework for thought in problem-solving. A great deal of research has addressed programming languages and compilers for Single Instruction Stream Single Data Stream (SISD) and Single Instruction Stream Multiple Data Stream (SIMD) architectures. Building upon this research will be necessary to develop the programming languages and compilers for concurrent processors (MIMD). The goal is to develop languages which provide an effective problem-solving environment, facilitate the expression of a problem quickly and correctly, and which compile into efficiently executing code. To do this, research must be undertaken on techniques for the expression of inherent problem concurrency and on compilers which recognize when concurrent execution is possible. This will require the analysis of many of the Agency's computational problems jointly with users to determine the adequacy and usability of programming language features.

Algorithms

Historically, Numerical Analysis has focused on algorithms for efficient, stable, and accurate computation on conventional Single Instruction Stream Single Data Stream (SISD) architectures. With the potential for non-SISD architectures, special algorithms and alternative problem formulations are needed. During the 1970s new algorithms were developed for Single Instruction Stream Multiple Data Stream (SIMD) architectures like the ILLIAC IV, CDC STAR 100, and TI ASC. Considering the potential of MIMD architectures, further progress is required to develop new and more efficient methods for solving sets of partial differential equations, driving flight simulations, or controlling spacecraft systems.

In order to develop these new algorithms, new analysis techniques must be developed which enable the detection of inherent concurrency in existing algorithms and programs. Analysis techniques for decomposing problems into concurrently executable parts must also be developed. Research is then required on techniques for the conversion of nonconcurrently executable programs into concurrently executable ones.

Just as advances in systems analysis and engineering techniques are required in Computer Architecture and Numerical Analysis, so must there be advances in the Theory of Computation. Theory of Computation includes analysis of algorithms, graph theory, and formal logic. The analysis of algorithms addresses solvability and existence issues, including the study of deadlock, complexity measures, and time/space trade-offs.

Ten Year Plan

Near Term Program Objectives (three years)

Develop analysis methods for the decomposition of problems into concurrently executable parts. Develop alternative MIMD architectures for the solution of problems of interest to NASA. Develop languages, compilers, and operating systems for MIMD architectures tailored to aerospace applications. Determine the applicability of data flow concepts to the problems of interest to NASA via simulation, analysis, and prototype evaluation. Develop the systems methodology for unifying a set of disjoint computational resources into a uniform distributed processing system. Develop hybrid simulation and analysis methods incorporating interchangeable hardware and software modules to evaluate new computer architectures for spacecraft and ground applications.

Long Range Objectives (ten years)

Develop discipline-oriented high-level languages for concurrent systems operating at rates of at least ten billion floating point operations per second. Develop the techniques for assembling many specialized computing capabilities into a unified distributed processing system with automatic load-leveling and resource selection. This will include the development of software such as distributed operating systems and communications protocols and the development of hardware such as high-speed (greater than 100 megabits/second) communication networks. Develop highly automated design methodologies which permit rapid system architectural design after automated analysis and requirements-generation for the problems in question. Continue to develop increasingly faster tightly-coupled concurrent systems (e.g., greater than 10 GFLOPS).

Strategy

The roadmap (Figure III-5) for the concurrent processing theme contains four major paths, architecture, languages, operating systems, and algorithms. The architecture path begins with the development of an evaluation capability and the evaluation of the various concurrent architectures with respect to NASA's computational needs. It continues with the development of architectural classification and specification rules which will be used to design concurrent systems to meet requirements derived from algorithm specifications and concludes with a set of tools, strategies, and approaches for quickly and accurately designing concurrent systems which will efficiently execute the programs for which they were designed. The language path parallels the architecture path by examining existing languages aimed at concurrent systems and by extending other languages to efficiently address concurrent systems. Compilation techniques will be developed to efficiently recognize concurrency in conventional programs and generate efficient code for concurrent execution. This path concludes with the recommendation and implementation of a higher order language for concurrent systems.

The operating system path begins with the understanding of the problem of scheduling resources on a possibly heterogeneous concurrent system. This will require the ability to capture the state of the system as well as the requirements of the task to be scheduled on the system. In addition to the efficient scheduling problem there are implications for scheduling around failed portions of the system. The path will conclude with the development of a distributed operating system with dynamic resource management for heterogeneous concurrent systems.

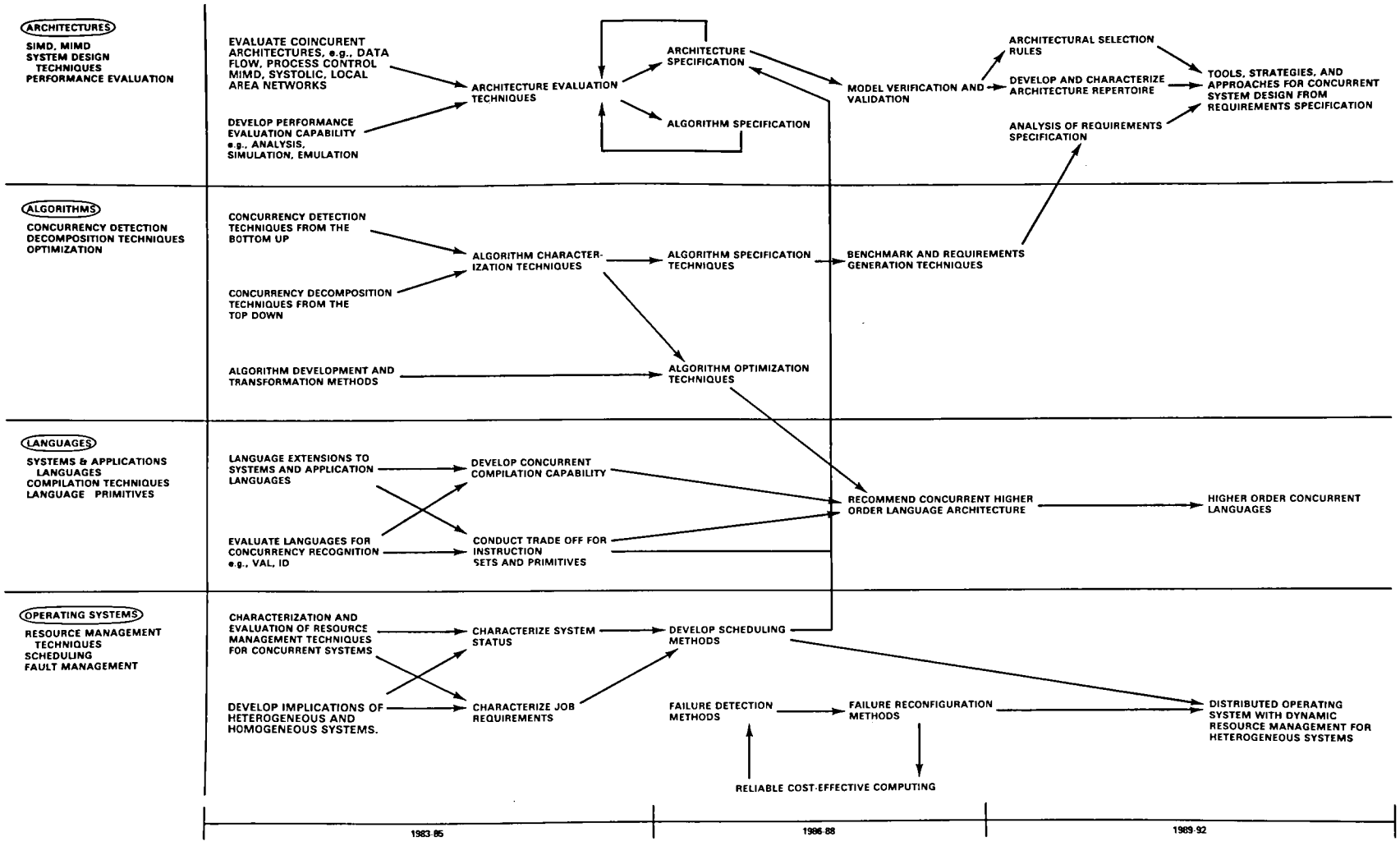

Figure III-5. Roadmap for the concurrent processing theme.

These three paths will lead to the tools necessary to design and implement concurrent hardware/software systems given a specific set of requirements derived from one of the many NASA problem areas which require concurrent processing systems. The fourth path, algorithms, will provide the capability of begining with the NASA problem area and developing efficient concurrent algorithms and system specifications and benchmarks to which the outputs of the other three paths can be applied to design the necessary hardware/software system.

Potential Applications

The potential applications of basic research in the area of concurrent processing to the programmatic areas of interest within the Agency are extensive. They range from improving ground-based systems for computational modelling of physical processes such as weather and climate simulations, computational fluid dynamics, computational chemistry, and structural analysis, through command and control of ground-based, flight, and space experiments or systems, to data reduction of experimental and management data.

The major contributions in the area of computational modeling of physical processes will be improvements in the execution speed and resolution of these computationally intensive problems as well as the development of user friendly interfaces to systems, thereby increasing the productivity of individual researchers. These improvements will be realized through the development of concurrent architectures and the mapping of the computational models onto these architectures. Distributed systems employing functional decomposition and distribution of tasks to specialized computational resources will also contribute to providing user-oriented, high performance computational modeling.

Parallel processing is needed onboard spacecraft to process Synthetic Aperture Radar (SAR) data. One SAR image requires an enormous amount of computing, requiring up to 10 hours of ground computing time presently. The goal is to reduce this time to minutes using special purpose computers. Placing this computing power onboard a spacecraft would enable much more effective use of SAR image data and of space-to-ground communications bandwidth.

Concurrent processing research has high potential payback in display, command, and control systems for ground-based flight simulation. Commercial digital computers augmented with many analog devices have served the special requirements of flight simulation until today, but the computational load imposed by impending requirements such as helicopter flight simulation exceeds conventional computing approaches. Advanced system concepts employ totally digital techniques, requiring concurrent processors to satisfy the computational load of flight simulation. This approach will require careful analysis of the basic problem, development of new concurrent algorithms and special processors, advanced computer image generation capabilities, and performance prediction and evaluation techniques in order to arrive at a successful implementation.

HIGHLY RELIABLE COST-EFFECTIVE COMPUTING

This theme is concerned with the engineering of computing systems, including hardware and software, which exhibit a high degree of reliability and fault tolerance, yet which can be developed and maintained in a cost-effective manner.

System reliability is closely related to complexity; the reliability of a complex system is much more difficult to ensure, yet may be even more necessary. Thus in order to design highly reliable complex systems, one must develop design and evaluation methodologies for dealing with architectural complexity.

Research Directions

Systems Engineering

Systems engineering is used here to refer to all the activities required to accomplish a mission, including design, modeling, development, testing, operations, maintenance, and management. Complex systems typically interface to many other systems, to people, and to the environment. Research in interface design, particularly the human-machine interface, is needed to develop improved tools for the design and development of systems, as well as to enhance operational effectiveness. Computer-aided tools for both engineering and management will help to control escalating costs and to maximize the use of limited manpower.

Particular attention must be given to system-level techniques for ensuring high reliability, including areas such as requirements, specification and design techniques, operating systems, communication protocols, languages, tools, reliability measures, and assessment methods.

Fault-Tolerant Computing

One of the major concepts in highly reliable computing is to use redundant hardware resources (processors, memories, input/output, buses) that can be reconfigured to perform different tasks if a failure occurs. The degree of fault tolerance and the reliablilty of such a computing system depends upon the amount of redundancy and the ability of the computing system to recognize (detect) and isolate a failure and reconfigure its hardware resources in sufficient time to provide error-free operation.

The criticality of a function determines the level of functional redundancy. For example, aircraft stability augmentation with relaxed static stability would require more redundancy than a function that is nonessential for safe flight. Typically a fault-tolerant system will contain spare hardware resources that can be switched into operation upon fault detection. When failure reconfiguration consumes all of the spare resources, then further failures will result in reduced functionality. Remaining resources will be used to perform the most critical functions, and processing of functions which are not critical will cease.

Another concept for obtaining reliable computing is the use of error detection and correction. Much attention has been paid to data storage and transmission error detection and correction, including parity checking of core memories and single error correction and double error detection (SECDED) codes in semiconductor memories. With the changing technologies and the ever increasing size of memories, more powerful error correcting and detecting codes must be developed. In addition, more efficient hardware implementations for these codes must be found. Complementing the work on error correction and detection for random access memory has been work in the area of burst error correction and detection commonly associated with data transmission and data storage on rotating magnetic devices such as disks.

With the ever increasing data rates of NASA space experiments, the currently used error correction and detection schemes for data transmission are becoming inadequate and improved methods must be developed.

When distributed systems or data sources are introduced into an environment with high reliability requirements, current fault-tolerant techniques become inadequate. Dynamic resource management and error detection, isolation and correction become far more complex. The concepts and techniques currently used, however, provide a solid base for extending fault-tolerant concepts and philosophy to more fully physically distributed systems. To develop fault-tolerant distributed system designs, the various systems and subsystems functional responsibilities must be defined. Analyses and tradeoffs must consider

the number of redundant modules, the data distribution network, operating system, hardware resource management software, failure recovery strategies, cross-strapping, manual/automatic manipulation, software complexity, and transient conditions. Architectural studies addressing the fault-tolerant, distributed environment must be undertaken to meet the Agency requirements of the future.

Software Engineering

Building highly reliable systems implies writing highly reliable software. System design errors and initial coding errors give rise to failure rates in system operation which can be characterized in a similar way to hardware failure rates. Software that is initially correct must therefore be a goal of a highly reliable system.

The development of correct software depends upon the approach used in its construction. The concept of software engineering is to expand and modify conventional engineering practice so that it is applicable to the software development process. NASA has a legacy of using good engineering practice to successfully attack engineering problems, but is just beginning to use similar practices in software development. It is crucial that NASA become involved in the development of software engineering methodologies. NASA is spending an increasing amount of its budget on software and there is every indication that this percentage will continue to increase.

Software engineering methodologies encompass such areas as requirements definition, design methodologies, validation and verification, life-cycle planning and costing, prototyping, and computing and programming environments. NASA must bring into use those software engineering methodologies which are extant and applicable, adapt methodologies and develop tailored tools as necessary, and do basic research in specific areas which are important to the Agency but are not receiving sufficient attention outside of the Agency.

Progress has been made in the formal proof of correctness of software. For example, a proof of correctness for certain critical portions of the SIFT fault-tolerant computer software using the techniques of mathematical logic has been produced [Melliar-Smith]. The proof was done with a combination of manual and automated logic provers including a software theorem prover. This initial success should be extended to cover all software applications of reliable systems.

In addition, software itself should be able to tolerate faults caused by software bugs, as well as hardware faults. Fault-tolerant software techniques efficiently provide redundancy at the algorithm level (somewhat analogous to hardware redundancy techniques). The current concepts for software fault tolerance work best where there is some reasonableness check for a computation, such as a physical condition which cannot be violated. In areas such as operating systems, there have been difficulties in formulating satisfactory concepts for fault tolerance. If such concepts can be found, they would permit the development of powerful aids to achieving the goal of reliable software. In the past some parameters were calculated via two different methods and the results compared to ensure that an error had not occurred in either calculation. Methodologies such as this and others based upon recovery blocks hold some promise for improving the reliability of systems [Anderson].

The state of the art in software engineering is addressed in the technology assessment of software given in Appendix B.

Formal Logic

Activities in this area will involve the application of existing theory to the pressing problems in the Agency related to fault-tolerant computing rather than extensive enlargement of the underlying theory. The formal proof of consistency between requirements, design, and computer code would provide a high degree of confidence in the reliability of computer software.

The most significant research contributions come from the following:

Theorem proving which is a critical element for most of the verification and validation work.

Multivalued Logic.

Abstract algebra which serves as the basis for the derivation of error correcting and detecting codes for memory and logic.

Ten Year Plan

Near Term Program Objectives (three years)

Develop improved concepts for fault tolerance and system reliablilty measures. Develop burst error correction and detection methods which permit at least an order of magnitude more data to be transmitted without an uncorrectable error. Develop more powerful random access memory error correction and detection codes (e.g. double error correction and triple error detection). Extend current software engineering research toward the development of highly reliable software. Develop planning and costing models for reliable software. Develop better computing and programming environments, including languages, verification and validation systems, and requirements definition and traceability systems.

Long Range Objectives (ten years)

Develop technology base, tools, strategies, and metrics for validating complex highly reliable computer systems. Develop support systems which include tools and methods for the design and evaluation of reliable systems. Develop techniques for verification of modified programs. Establish life-cycle disciplines for reliable cost-effective systems. Develop advanced fault-tolerant architectures and software for distributed systems.

Strategy

The roadmap for work in highly reliable cost-effective computing (Figure III-6) follows three major areas: systems engineering of reliable systems; hardware architecture focused on fault tolerance and error detection and correction; and software engineering tools, techniques, and validation procedures for both reliability and a cost-effective software life cycle.

Initial work in systems engineering will be to determine NASA's requirements over the next ten years for autonomous and complex systems, and from that define and develop a set of support tools and the systems methodolgy to design and evaluate highly reliable, fault-tolerant systems. A companion goal in ten years is to have the tools, strategies and the metrics for validating that a computing system is, in fact, highly reliable. Activities during the evolution of such a support system include development of system reliability measures and models for assessing the reliability of a system, investigation of techniques to manage and control system complexity, and definition of system design languages for specifying fault-tolerant systems. Also, alternate design approaches for fault-tolerant systems should be characterized and improved methods for failure detection and system reconfiguration developed.

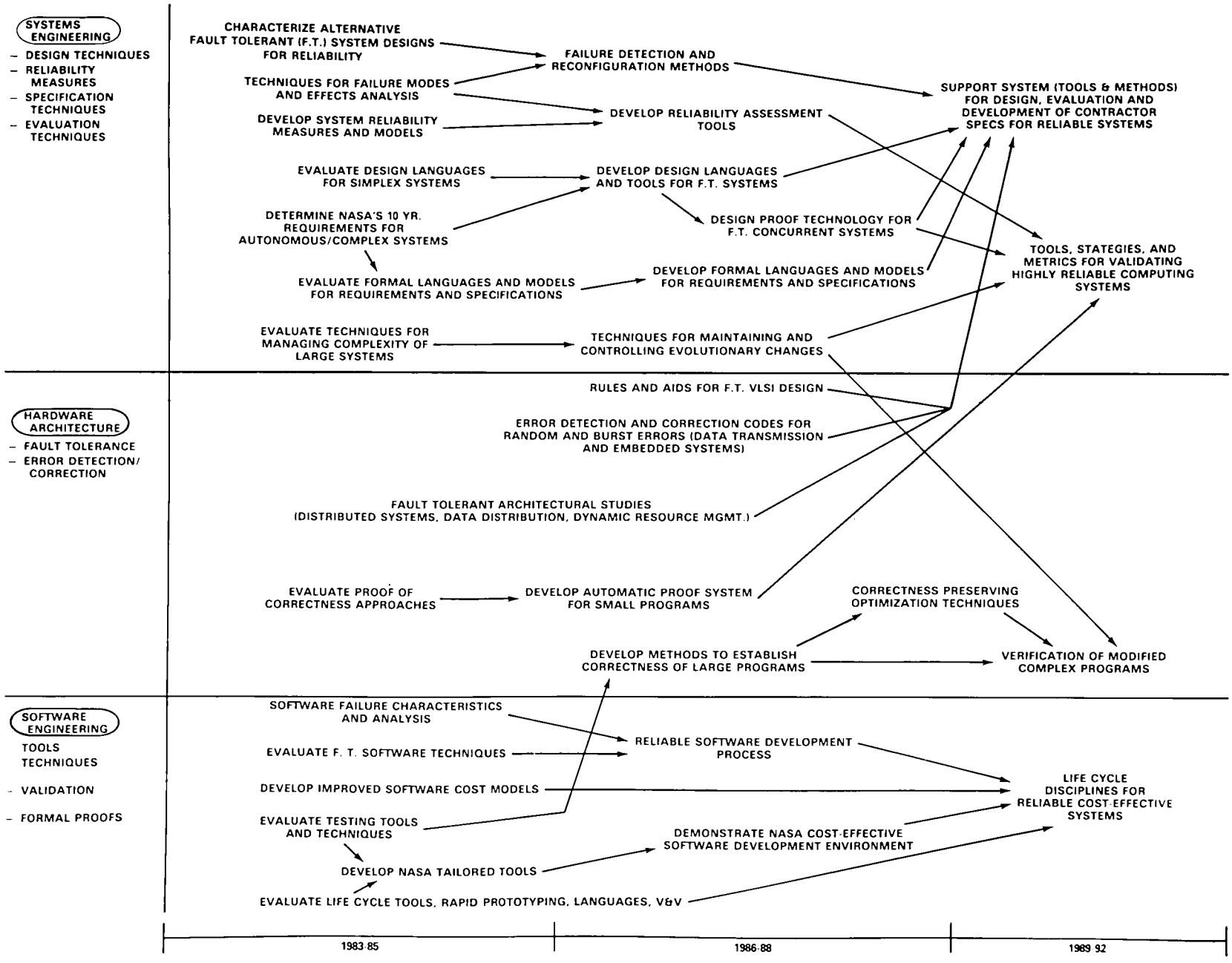

Figure III-6. Roadmap for the highly reliable cost-effective computing theme.

The hardware architecture studies appropriate to fault-tolerant systems will be coordinated with the concurrent processing research also sponsored under this program. Some of the reliability drivers will cause different emphasis, but many of the issues are common, particularly those derived from distribution of resources and data.

A continuing evaluation of software engineering tools, particularly those related to software life-cycle management, requirements specification and analysis, and rapid prototyping, is necessary to keep the Agency apprised of important advances which could be applied to effectively improve the software development process in both reliability and cost.

Specific assessments of techniques for software failure analysis and fault tolerance techniques shall be performed. These assessments will contribute to the definition and development of a reliable software development process. Proof of correctness approaches must be evaluated and automatic proof systems developed to provide validated highly reliable software for reliable computing systems. Techniques must be developed for testing and establishing the correctness of large, complex programs, including correctness preserving optimization techniques and change control.

Work in software engineering will result in the definition of life-cycle disciplines for reliable, cost-effective systems.

Potential Applications

Given NASA's many high technology activities there is an ever expanding need for highly reliable computing coupled with decreased expense. Examples include flight crucial systems for the shuttle, space station, planetary probes, and earth-orbitting free flyers; man-rated systems for aerospace vehicles; and ground-based systems, including super computers for computational modeling of physical processes. To decrease the expense of these systems, methodologies must be developed to improve programmer and hardware designer productivity, to decrease development time, and to lower the life-cycle cost by reducing maintenance costs and by improving the ability to reuse or share systems and subsystems.

SCIENTIFIC AND ENGINEERING INFORMATION MANAGEMENT

This theme deals with the management of data from the time of generation through the completion of its analysis by the end user. The theme focuses primarily on the data delivery mechanism (computer networks), the data management systems, the mechanisms for relating data sets, and the mechanisms for the user to interact with the data. These concepts, devices and technologies will be used by equipment and systems designers, and by scientists who analyze results of experiments or observations.

Research Directions

Computer Networks

The research in this area should be focused on developing the capability of various data centers and data users to intercommunicate easily and effectively. Protocols for the exchange of data should be developed which are applicable to the special needs of users of NASA's data. In particular it should address the engineering data, remotely sensed low rate data, and image data that NASA produces. In addition to defining the protocols, protocol implementation and network architecture should be addressed. At the conclusion of this research effort a demonstration network connecting NASA data stores, other related data stores, and the community of data users should be operational. New users should be able to join the network and use communications and applications software packages developed elsewhere with little additional effort.

Data Base Management Systems

The majority of the research conducted in the area of data base management has been for the commercial sector. Relatively little has been done on scientific and engineering data management systems. A high priority is to evaluate alternative data models for scientific and engineering data and establish how well the existing data base management systems can handle the data models. Special purpose data management hardware is becoming increasingly available. It is important to know how well this technology can presently handle scientific and engineering data and how the technology needs to be adapted to handle these types of data. Eventually data base management capabilities distributed over many data stores and user locations, all networked together in a logically integrated mode, will be required.

Information Exchange

The data gathered from one mission or experiment usually is processed in an experiment-unique or mission-unique way, exacerbating the data exchange and data comparability problems. Research is required in this area to promote and facilitate greater data sharing. Data needs to be expressed in a format understood by the individuals sharing the data. They must be able to understand the "pedigree" of the data. In addition, the ability to learn of the existence of required data and to be able to access it easily for preliminary examination is needed. This find and browse capability must be coupled with an intelligence on the part of the system so that the user does not have to be an expert at using the complex mechanics of the system, but can be guided by the system itself.

Man-Machine Interaction

The research in this area is focused on helping the user conduct his data analysis and interpretation. The system should present the data to the user in such a way that his brain is able to process the data in ways that it is particularly suited to do. Very often this means that the data should be presented in graphical, not tabular, formats since the visual processing system of the human brain is so highly developed. Systems should be flexible enough so the user can structure the presentation of the data in alternative ways to maximize the amount of information extracted. The physical facilities that the user interacts with should enhance his analysis capability. With the advent of inexpensive microprocessor technology the user should have at his disposal a very capable local microcomputer system which will allow him to pursue the bulk of his research locally, under his total control, and interact with other computational resources only when needed.

Ten Year Plan

Near Term Program Objectives (three years)

Develop data models for the description and manipulation of image data, graphical data, remotely sensed satellite data, engineering design and test data. Develop data base management and data storage architectures capable of handling these particular data types efficiently. Develop automated searching and browsing capabilities for users of a scientific and/or engineering data base. Develop networking capabilities for connecting users with the data stores. Develop common or uniform graphics formats and/or techniques for the display of sensor and engineering data.

Long Range Objectives (ten years)

Develop a complete data delivery architecture and capability for the rapid delivery of data from current and past missions. Develop a storage capability sufficient for 10E15 bits of data. Develop a distributed data base management system architecture which facilitates and enhances the user's ability to locate and

browse through data among geographically dispersed data stores. Develop protocols which will allow the transfer and reformatting of data to allow comparability of data. Develop sophisticated data presentation techniques and mechanisms for scientists and engineers. Develop intelligent work stations which can be tailored to individual users requirements.

Strategy

The strategy for accomplishing the goals stated above calls for work to proceed along four pathways (see Figure III-7) which converge in ten years to realize a system the scientist and the engineer will find tailored to locating, analyzing, sharing, and using NASA's data and other scientific and engineering data. A description of the four pathways follows.

The work in computer networks is initially divided into two areas. A seven layer Open System Interconnection Standard developed by the International Standards Organization defines the various layers of protocol to be developed to allow applications programs to communicate. The first three layers of the ISO Standard have been fairly well studied and have been implemented in several ways. The X.25 protocol is an example of the first three layers. How this standard and its implementation apply to NASA in establishing both local and global networks for the exchange of scientific and engineering data is a research issue. The development of the higher 4 levels of the standard so they reflect the capabilities needed to transfer and manage the type of data and applications that NASA has also needs study. As these standards are developed, they should be tested at NASA installations. This should lead in ten years to a fully functioning network with all the capabilities needed for the effective communication between NASA application level computing activities.

Data base management systems research will initially focus on the development of data models for scientific and engineering data. Assessment and evaluations will be made of the current technology in DBMS software packages, data base machine technology, and data storage technology. Alternative architectures will be created out of these building blocks that can be particularly focused on the efficient handling of scientific and engineering information. These architectures will then be implemented in a prototype single user DBMS system. After the development of a single user system is completed, work will be done on interconnecting data bases in a distributed fashion. This will involve using the networking technology developed under the activities described in the previous paragraph. The goal is to have a capability for managing scientific and engineering data among distributed sites with perhaps heterogeneous systems at each site.

Initially the work on information exchange will focus on ways to specify the "pedigree" of the data in a common format. In this way the people who access the data at least know what has been done to the data. The next step will be to develop mechanisms for translating the data into formats that the individual users can use. Not only must the data itself be shared, but information about the data must also be sharable. Automated cataloging and browsing systems must be created. These browsing and locating capabilities will be enhanced by the application of computer graphics and artificial intelligence to make the job require less knowledge of system specific idiosyncrasies. This will lead ultimately to the point where the user has the ability to locate desired data and to retrieve multiple data sets in a format that will allow intercomparison of the data.

The work on man-machine interaction will focus on techniques for the user to conduct his analysis conveniently and efficiently. He will not have to conform to the way the machine must work but the machine will conform to the way that he works. Better graphical techniques for the display and interpretation of the data will be developed. This may involve developing new algorithms or rethinking the way that data is presented. A workstation will be developed tailored to handle the problems that the scientist or engineer must solve. It may contain special VLSI chips devoted to performing some specialized, but often used, functions that the individual scientist needs. The system may be modular to the extent

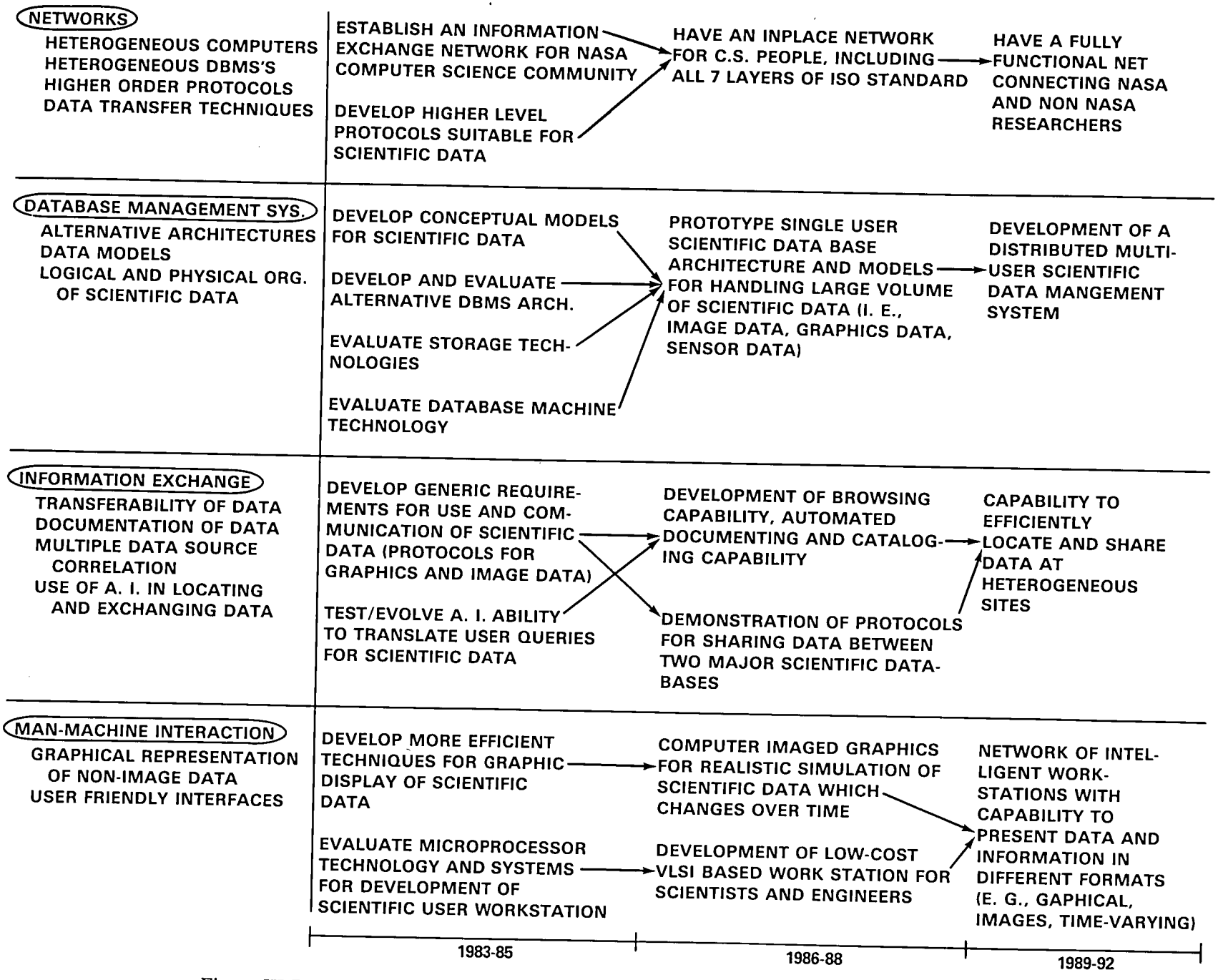

Figure III-7. Roadmap for the scientific and engineering information management theme.

that the workstations can be easily tailored to different classes of users by the addition or deletion of certain chips or computer boards. Eventually this will lead to a network of communicating workstations, each meeting the particular needs of the individuals using them and each being able to communicate with the others.

Potential Applications

The potential applications or benefits of this work include improving the productivity and capability of the managers, scientists, and engineers who will need to use NASA's data. This covers a wide range of people. There are managers who must manage very complicated development projects and need information about all aspects of the system development. There are planners who need data from previous activities and missions so that they can develop reasonable estimates on the future missions and activities. There are system architects and system designers who will need to control the design process for the system they are now creating and will have to apply relevant data from past systems. There are the people involved in the operations aspects of missions who will need to keep track of the progress of the mission and be able to analyze unpredicted or anomalous behavior. Finally there are the scientists and investigators who will be examining the data that has been gathered from experiments or missions. These represent the classes of people who will be beneficiaries of the new technology generated out of the basic computer science research conducted here.

The potential for the ways this development will help people in managing their information is currently inestimable. When online bibliographic searching became available in libraries the librarians found many new ways to use the system. The same is expected to be true for the scientific and engineering community. People will be able to ask questions and find answers that they never would have considered before due to the difficulty of locating and correlating diverse sources of information.

There are some applications which we can appreciate now. The community of users will increase since access to data will be easier. The analysis of the data should be more significant since users will be able to use more of their time in thinking about scientific analysis rather than in data translation and manipulation. The ability to display the data in various formats will allow a greater understanding of the meaning of the data. The accuracy and credibility of planners and designers will be enhanced since they will have more precise historical data upon which to base their designs and plans. The ability to cost projects and missions should be improved.

RESEARCH SHARING MECHANISMS

The objective of research sharing is to increase the visibility of the work being done by all computer science researchers within NASA. This visibility should extend among various groups or categories of people: the research communities within and outside of NASA, the researchers and their sponsors, the researchers and their managers, the researchers in different areas of computer science, and the researchers at different NASA Centers.

There should also be the objective of sharing information about research done in the past, research presently ongoing, and work planned for the future.

In order to accomplish this information exchange four mechanisms are recommended - publications and information banks, conferences, courses or lectures, and reviews.

Publications and Information Banks

A centralized information bank should be created at a NASA facility to store information about the work that NASA does or sponsors in computer science. This data bank should contain information about on-going work, as well as results and accomplishments. Reports, tutorials, and research programs and data should be stored in this data bank, accessible via computer network to all NASA Centers. The creation of such an information bank would in itself be a worthwhile project for computer scientists.

Research results should be published. Several options are available to the NASA researchers: Center technical reports, professional journals, conference proceedings, and books.

Conferences

Attendance at conferences where results can be presented and colleagues can exchange information is to be encouraged. Conferences organized by professional societies are particularly important for active researchers. Conferences within and among NASA Centers facilitate the exchange of information among the NASA community. Conferences between NASA and industry, academicians, and other government researchers also are beneficial.

Courses or Lectures

Material relevant to ongoing work can be presented by outside experts from industry or universities to those interested in the research. Also research results can be shared by giving courses or lectures. These courses and lectures can be videotaped and arrangements can be made for sharing these videotapes among the Centers. People within NASA can travel to other Centers to present or participate in courses, lectures or workshops. These activities should be well publicized and held regularly.

Reviews

Regular reviews of work accomplished and work planned should be held among members of the academic, industrial, and NASA communities. In this way research can be coordinated and focused on work that has been validated by others as having use and meaning. This type of forum is a critique of the work and hence differs from a lecture or a conference.

UNIVERSITY RELATIONS

Much of the research performed under this program will be done at universities. This will provide NASA with the benefit of excellent academic research, student involvement in NASA problems, and colleagues to stimulate the NASA researchers. The university involvement can take the form of grants and cooperative agreements under individual RTOPs at the Centers, block grants administered from a Center or from Headquarters (such as the MIT and Stanford block grants planned for FY 83), and research institutes affiliated with NASA Centers where academic researchers work on-site at a NASA Center (ICASE at Langley and RIACS at Ames). Two secondary benefits are also expected from such close ties with the University research community: the NASA monies will enhance and encourage experimental research in computer science, which is sorely needed [Feldman]; and students, exposed to NASA, will be prime candidates for recruitment upon graduation.

INSTITUTIONAL CONCERNS

Personnel

Recruitment of computer science personnel is exceedingly difficult at the present time because competition is fierce in the commercial sector and fewer PhDs are being granted in Computer Science. The Agency cannot compete with the salaries offered by industry and therefore must attract computer scientists with challenging problems and excellent research facilities. Currently the Agency lacks the research facilities specifically available to the computer science researcher. Any hope of recruiting high-level researchers depends on providing a conducive environment with state-of-the-art computing equipment for research purposes and a research atmosphere.

Administrative Support Systems

Several activities within the Agency are addressing the need to improve the use of computer technology in support of administrative and management personnel. Several common databases have been developed (e.g., legal, procurement), with Intercenter cooperation and access. The Electronic Information Services and Action Information Management System projects are focussing attention on improved productivity for Centers and Headquarters offices. Such activities could benefit from the technical knowledge base built within the computer science research program. Headquarters offices should coordinate between the research program and the application of the technology.

Software Development Guidelines & Policies

At present there is no set of software development standards for the Agency. The Intercenter ADP Committee provides coordination among the computer facilities, but does not set standards for software development or life-cycle maintenance. Individual projects must develop their own standards, often different from others in the Agency and without the benefit of reusing previously developed policies, guidelines, or software. The HAL/S Language Control Committee serves to provide some guidance for a limited community. The lead may need to come from the Chief Engineer, but again the technology base from the computer science research program should contribute.

COMPUTER SCIENCE RESEARCH FACILITIES

NASA is one of the largest users of computers within the federal government. Nonetheless there is almost no capability for doing computer science research. The funding approach employed for computer systems assures that they will be associated with mission requirements and not available for computer science research. Computer science research frequently raises questions about computer systems, so such systems must be available for experimentation. In addition to support which may be made available on general purpose systems, there is also a requirement to provide systems which are available solely for computer science "experimentation". On these systems the operating systems may be modified, the compilers may be experimental, and the entire system is available as a research tool. Clearly, generally available systems cannot be permitted to be experimented upon in this fashion.

One of the reasons NASA is woefully short of computer science personnel is that the computers available within the Agency are not of the latest technology. While that problem is partially a result of budgetary constraints and cannot be eliminated in a short time span, the availability of computer systems dedicated to experimental computer science research would be a solution for the computer scientists. Fortunately the solution is not expensive and could be accommodated without the severe impact of replacing or upgrading the Centers' very large computing complexes.

TECHNOLOGY ADAPTATION/UTILIZATION

In addition to fundamental research to provide the basic knowledge base for long range computer applications within the Agency, there is an urgent need to introduce current computer science technology into the operational aspects of NASA. This includes the effective use of Office Automation techniques, use of commercial networks for improved information flow and exchange, and application of current software tools, techniques, and workstations to improve software quality and programmer productivity. The Agency must develop a broad plan for infusing the latest computer technology into its ongoing programs and into the institution itself.

This research program plan has not addressed this immediate problem, but rather is concerned with the longer range view so that in ten years there hopefully will be a cadre of computer scientists within the Agency who will be a corporate conscience to influence the continued use of the latest technology.

IV. RESOURCES

The intent of the computer science research program is to develop a repertoire of capabilities from which the Agency can draw to develop advanced aerospace computing and information management systems. The program plan and the accompanying resource requirements are designed to build up the necessary capabilities in a time-phased manner, typically beginning with a thorough understanding of the aerospace requirements and an analytic ability to evaluate alternative solutions, progressing through focused research on generic problems, and converging on systems-level capabilities to conceive effective and innovative computing-based solutions to aerospace problems. The resources identified herein reflect a growth period, primarily in the first three years, to reach a nominal steady state level of support. Resources are identified in terms of NASA work years of effort (WYE's) and money. For purposes of these estimates, JPL WYE's are considered to be NASA WYE's. The money identified is required to equip NASA researchers and fund university grants and industrial contracts. Approximately 50% of the program resources are expected to be devoted to university research.

CONCURRENT PROCESSING

This part of the program should grow from about 11 work years of effort to about 24 work years and about 4 million dollars.

Table 1. - Resource Estimates for Concurrent Processing

ACTIVITIES	FY 83 WYE	FY 83 $K	FY 84 WYE	FY 84 $K	FY 85 WYE	FY 85 $K
Advanced Architectures	5	300	7	1,200	8	1,200
Algorithms	3	350	7	850	9	1,400
Languages	1	150	2	500	3	600
Operating Systems	2	200	3	650	3	800
Total for Concurrent Process	11	1,000	19	3,200	23	4,000

HIGHLY RELIABLE COST-EFFECTIVE COMPUTING

This part of the program should grow from about 16 work years of effort to about 36 and about 7 million dollars.

Table 2. - Resource Estimates for Highly Reliable Computing

ACTIVITIES	FY 83 WYE	FY 83 $K	FY 84 WYE	FY 84 $K	FY 85 WYE	FY 85 $K
Engineering of Complex Sys	4	400	8	1,600	11	2,200
Fault-Tolerant Architecture	4	600	6	1,500	8	2,300
Software Engineering	8	1,000	14	1,600	14	1,800
Total for Highly Reli. Comp	16	2,000	28	4,700	33	6,300

SCIENTIFIC AND ENGINEERING INFORMATION MANAGEMENT

This part of the program should grow from about 14 work years of effort to about 30 work years and about 5 million dollars.

Table 3. - Resource Estimates for Sci. and Engr. Information Management

ACTIVITIES	FY 83 WYE	FY 83 $K	FY 84 WYE	FY 84 $K	FY 85 WYE	FY 85 $K
Information Networks	3	600	6	1,100	5	800
Database Management System	4	650	5	1,050	9	1,400
Info. Access and Exchange	2	200	6	500	7	1,200
Man-Machine Interaction	5	550	6	750	7	1,200
Total for Information Mgmt.	14	2,000	23	3,400	28	4,600

TOTAL PROGRAM

Total program resources required to provide sufficient support to build a viable computer science knowledge base within the Agency is estimated to be the following, with a growth period of 4 years:

Table 4. - Total Computer Science Research Program Resource Estimate

FISCAL YEAR	WYE	R & D $M
1983	41	5.0
1984	70	11.3
1985	84	14.9
1986 & beyond	90	16.0

REFERENCES

[Anderson]

Anderson, T.; and Kerr, R.: "Recovery Blocks in Action: A System Supporting High Reliability," Proc. Second International Conference Software Engineering, 1976, pp. 447-457.

[Backus]

Backus, John: "Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs," Communications of the ACM, Vol. 21, No. 8, August 1978, pp. 613-641.

[CODMAC]

National Academy of Sciences, Committee on Data Management and Computation, Space Science Board: *Data Management and Computation Volume 1: Issues and Recommendations* , National Academy Press, 1982.

[Elcheson]

Elcheson, D. R.: "Cost-Effective Comparison of Manual and On-Line Retrospective Bibliographic Searching," Journal of the American Society for Information Science, March 1978, pp. 56-66.

[Feldman]

Feldman, J. A.; and Sutherland, W. R. (eds): "Rejuvenating Experimental Computer Science - A Report to the National Science Foundation and Others," Comm. ACM, Vol. 22, No. 9, Sept. 1979, pp.497-502.

[Freitas]

Freitas, R. A.,Jr.; and Carlson, P. A. (eds): *Computer Science: Key to the Space Program Renaissance, Final Report of the 1981 NASA/ASEE Summer Study On The Use of Computer Science and Technology in NASA* , University of Maryland Technical Report 1168, Vols. I & II, 1982

[Melliar-Smith]

Melliar-Smith, P. M.; and Schwartz, R. L.: "Formal Specification and Mechanical Verification of SIFT: A Fault-Tolerant Flight Control System," IEEE Trans. on Computers, Vol.C-31, No. 7, July 1982, pp.616-630.

[Sagan]

Sagan, C.; Reddy, R.; Heer, E.; et al.: *Machine Intelligence and Robotics: Report of the NASA Study Group, Final Report* , March 1980.

[Sci Amer]

Special issue on productivity in Scientific American, Vol. 247, No. 3, September 1982.

Ginzberg, Eli: "The Mechanization of Work", pp. 66-75.

Gunn, Thomas G.: "The Mechanization of Design and Manufacturing", pp. 114-130.

Giuliano, Vincent E.: "The Mechanization of Office Work", pp. 148-164.

[Wegner]

Wegner, Peter (editor): *Research Directions in Software Technology* , The MIT Press, 1979.

APPENDIX A. NASA PROGRAM OVERVIEW

INTRODUCTION

A description of NASA program activities is given here to provide some rationale for the research directions recommended in this plan. For each program area a general description of the activity is followed by some of the key problems currently identified in this area and a discussion of how computer science research activities could address these problems and potentially improve NASA's ability to perform its mission more effectively.

In its earliest deliberations, the Intercenter committee determined that a computer science research program within the Agency must clearly relate to NASA activities and have potential application to ongoing NASA programs. Therefore, a systematic breakdown of the Agency's program areas was created, from the computer science perspective. This Programmatic area breakdown then became the definition of requirements against which this research program plan has been written. The management organization of NASA was considered for the basis of this breakdown, but since many computer related functions overlap, a more discipline oriented breakdown was devised (See Appendix C). In the final analysis, even the Aeronautics and Space programmatic activities have many overlapping requirements for computer science research support. Hence, the aeronautics and space program requirements are combined in the following discussion.

DESIGN AND ANALYSIS METHODS

DESCRIPTION

Much of the aeronautical research conducted by NASA involves the design and analysis of aircraft systems. This includes vehicle configuration, structural analysis, material properties of airframe, propulsion, avionics, and a variety of subsystems related to the aircraft.

The design and analysis techniques associated with this research are highly dependent on advanced computer science and technology, both hardware and software. Computations involve large-scale predictive computer modeling, analysis and simulation of the behavior of aerospace vehicles subject to static, dynamic, thermal or acoustic loads.

Space vehicle design and analysis are somewhat different than that of aircraft since many space systems are unique and production often is limited to 1 to 5 units. Integrated vehicle analysis is a necessity since there is little historical background of engineering and design data available from which to proceed in a new development. In the total design process, synergistic effects are very important, so each engineering discipline must be coupled.

KEY PROBLEMS

The accuracy of the results of the analytical simulations is significantly degraded by the lack of adequate computer speed and memory during the computations. "Short cuts" around these problems involve such techniques as neglecting nonlinear contributions of the mathematical model, reducing the model complexity, and reducing the generality of the software by tailoring it to one specific application.

Large amounts of data, describing the model characteristics, the environment, and the types of analysis to be performed , present a difficult data management problem.

Design has many variables that need to be traded to develop satisfactory product. Computer-aided design systems are of limited use for unique system designs since the analysis programs that are coupled together may change significantly as the system design matures. Program integration requires the ability to store and retrieve design data, provide data dictionary and a program library while preserving integrity of engineering data. The basic components of computer-aided design software apply broadly, for example, a DBMS for data communication, a data dictionary, and a data and program library; an executive to provide the man-machine interface and system control; a geometry system to represent the physical model for visual interpretation; a data analysis system to interpret output data and provide graphical display of results; and the integration software to couple engineering analysis programs. These systems are awkward to use and inadequately support the aerospace systems designer. The coupling of analysis and design programs to user requirements is a major problem of integrated analysis.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

Many areas of computer science can contribute to the effective support of design and analysis of aerospace vehicles, ranging from improved computational methods to electronic mail for better communication between design engineers. Particularly important contributions are needed in database management (for engineering data from the analyses as well as management and control of design parameters), graphics for displaying analytical results and for design visualization, and software tools and languages with engineering user interfaces.

Examples of some of the contributions which can be made are:

Improved hardware and software for Database management and Mass storage devices.

Simplified graphics input and output-for geometric models, and interactive graphics techniques for rapid interpretation of results.

Effective use of color, holographic, and motion graphics for the display of physical phenomenon analyzed/modeled.

Network architectures for increased computational power (Including distributed processing and data communication).

Mathematics of computation for faster and more accurate analysis techniques.

Algorithms for special computer architectures, such as arrays of processors solving a partitioned problem.

Software tools and techniques, design technology, high-level languages, common software components, user interface techniques.

Computer system modeling, and performance measurement techniques.

SIMULATOR SYSTEMS

DESCRIPTION

The Agency has actively developed and used a wide variety of simulators in its aerospace research activities. In the development of aerospace vehicles man-in-the-loop type simulations in real time are essential for the evaluation of handling quality, quality of the instruments, design parameter verification, etc. These simulations must be as realistic as possible to the pilot, since the purpose of the simulation is to

make the pilot believe that he is flying the actual aerospace vehicle and hence provide an accurate evaluation. Ground-based simulators are also crucial to flight crew training because of safety factors in training crews to respond to malfunctions and because of the cost of using actual vehicles. Future space activities will require teleoperator and robotic devices to enhance man's capabilities for servicing, maintenance and repair, structural assembly, and space manufacturing. The simulation of a remotely controlled space system which includes manipulators, sensors, communications, environmental models and various methods of man-machine interaction is needed to understand and develop techniques for space operations.

Again the simulations must be as realistic as possible in order to evaluate the space system and to train the operators of the space system.

The requirements for high-speed real-time simulation have always acted as a stimulus for the development of new hardware and innovative software techniques. The choice of a simulator for a specific application invariably involves judicious compromises and careful consideration of tradeoffs among such factors as cost, accuracy, flexibility and speed. New developments in technology and computer science necessitate the reexamination of these tradeoffs. Likewise, the examination of these tradeoffs will indicate areas where new development in technology and computer science are required.

KEY PROBLEMS

In the art of visual simulation and scene generation a high degree of realism is required. Historically, this simulation has utilized a video presentation of a scale model in the case of space system simulation and a moving camera over a terrain map system in the case of flight simulation. Both of these approaches have several limitations, such as optical distortion and limited field of view. It also requires considerable lead time and expense to create the detailed scale models and terrain maps. More recently, computer generated images are being used to fulfill the requirements of these simulations. However, this approach is extremely costly and has the limitations of not appearing realistic due to a lack of detail. Also, to provide real-time simulation, the computational requirements are enormous.

Many problems arise in the real-time simulation of space systems. For example, simulations of the Space Telescope Pointing System as currently implemented on sequential digital computers, are characterized by responses many times slower than real-time. Thus, to determine the effectiveness of the momentum management control algorithms for just one orbit requires many hours of computation. A similarly complex simulation, that of a helicopter rotor system, has been analyzed in detail by J. A. Houck of LaRC and R. M. Howe of the University of Michigan. This analysis determined that to maintain reasonable accuracy for real-time simulation, the digital frame time can be no longer than five milliseconds. At LaRC, a FORTRAN program implemented on a CYBER 175 requires a minimum of 10 milliseconds, while a Sigma 7 implementation at ARC had a frame time approximately 40 milliseconds. The need for faster computing is clear.

Current flight simulation systems and those which will be constructed in the future will require the use of several computer systems. Coordinating this complex, distributed flight simulation task over several computer systems is a continuing problem.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

To improve computer image generation to the point where simulations will appear more realistic will require several improvements in the state of the art of computer science. Algorithms must be improved to store and display the images in a more realistic and less cartoonish style. This will require the development of fast algorithms which depict realistic texture, shadowing, shading, and toning of the images. Better hardware must be developed including higher resolution display devices, larger and faster memory systems for storing and displaying the database, faster systems to provide realistic dynamic motion.

The faster cost-effective computing requirements can best be met by the application of concurrent processing systems, e.g. several systems executing different parts of the simulation problem at the same time. To do this the simulation must be decomposed into concurrently executable cooperating tasks. The subsystems for these tasks can then be tailored to their execution, e.g. the computer image generation could be done on a subsystem tailored to that activity. To decompose the problem and design the system will require research and development in concurrent processor architecture, distributed operating systems, programming languages for concurrent systems, and algorithm analysis and decomposition techniques.

EXPERIMENTAL DATA HANDLING

DESCRIPTION

NASA experimental data come from a variety of sources including wind tunnel experiments, flight research tests, and space system payloads.

Experimental data handling in NASA includes the following major categories:

1. Creation and delivery of a data stream for control of a vehicle or model and its operational systems and for control of individual payload instruments;

2. Acquisition, delivery, and processing of status information from instrumentation and operational systems;

3. Acquisition, delivery, and processing of payload telemetry;

4. Acquisition, delivery, and processing of ground data supporting the operation and control of the vehicle.

The composite set of data handling functions is accomplished by an inherently distributed system of human, processing, instrumentation, and communication elements. Ground instruments and computers, vehicle instruments and computers, payload instrumentation and computers, and human monitors and controllers are nodes in a distributed system interchanging varying combinations of data across several communication links.

Critical and severe time and performance requirements are levied against the system as a whole and against individual elements. The safety of instruments, vehicles, or even human life depend on meeting failure detection and correction criteria which in turn depend upon management and delivery of appropriate sets of status data to human, automated or hybrid man/machine control points and the creation and delivery of resulting control data to other control points in a time critical mode. The scientific goals of a mission typically levy different, but just as critical, demands for the effective management and delivery of combinations of control, status, and payload data tailored to a particular investigation from distributed sources across network interfaces to a science processor and the return of control information through the network to a payload instrument.

At the very beginning of the space programs in NASA, primary emphasis was placed on the flight system and only token recognition given to the operational instrumentation. This is as it should be for programs which are success oriented and designed for relatively limited lifetimes. All of the manned spacecraft projects through Apollo were based on one flight per vehicle with approximately 2 weeks maximum flight duration. Deep space probes require much longer flight times and provide a much slower ground response to problems. These have forced the development of extensive operational monitoring and control functions using both hardware and software onboard the spacecraft and much more sensitive ground monitor and support functions. Increasing distances and limited bandwidth indicate a need for more

advances in onboard control and processing. These conditions have also resulted in very extensive ground test facilities, years of labor intensive flight system testing and the expenditure of a major portion of each projects budget on preflight data collection and analysis. Often, as a program came to an end it was discovered that much of the highly sophisticated ground test equipment and software programs were uniquely designed for the particular project and could not be used or modified to be applicable on the next project.

The space shuttle program was approached in exactly the same way as every manned project before it, but now as the orbital flight test (OFT) phase comes to a close and the orbiter is classified as an operational vehicle, NASA finds itself in the situation of completing a 10 year development phase on a project and facing a 10 year operational phase with the same vehicle. Deep space probes typically face similar problems but with different options for managing the health of the spacecraft. The avionics systems which do such a good job detecting hard failures in the infant mortality period do not perform nearly as well in detecting the slow degradation failures caused by wearout of subsystem components. Operational Instrumentation (OI) subsystems could detect the slow degradation failures, if they are designed for it in the early years of the project. Likewise, the massive ground test facilities which verify interfaces and flight control systems during the test phase are not set up to collect massive amounts of operational data and produce automatic trend predictions.

Early decisions decreed that people, not computers, would analyze data. Bandwidth is limited. Real-time avionics flight monitoring must compete with increasing data traffic from experimental data requirements. Both limited bandwidth and physically unavoidable communication delays indicate an increased need for onboard processing. Major advances have been made but more are needed in processing appropriate data to enhance the analysis on the ground.

At the present time conceptual design studies are in progress on a space station to determine the feasibility of making it the Agency's next major project. A typical scenario for the space station calls for a 4-6 year design and development phase, a 3-5 year evolutionary buildup phase in space and then a 15-20 year operational lifetime. Assuming the space station follows the same general pattern as the shuttle, a relatively high confidence level will be established in the ability of the onboard avionics to perform properly during the first two years of operation. At that time emphasis will shift to long term monitoring of slow degradation failures with the OI subsystem. At the same time a major effort will be placed on support for experiments. Note that the OI and experiment operational time period is 18-20 years on the space station compared to 10 years on the orbiter and two weeks on Apollo.

KEY PROBLEMS

One immediate and obvious problem concerns the collection, storage and retrieval of data. This includes data base management, statistical analysis, graphical display, pattern recognition, and advanced high capacity storage devices. For space systems, since downlink bandwidth is a limited resource, a major portion of the experimental data should be processed onboard. Quite often the problem resolution for slow degradation failures depends on comparison of preflight test data with inflight data, which implies correlation of multiple data bases and predictive modeling of failure modes in addition to trend analysis.

During system definition and design, more emphasis is required to determine exactly what data should be collected for long term monitoring of vehicle avionics systems, how should the data be stored, compressed, preprocessed and massaged to ease the load on a limited bandwidth downlink, and if the experimental data should be stored and preprocessed onboard a flight vehicle or completely transmitted to ground-based processing facilities.

The scientific results from experimental data depend heavily on efficient management of varying collections of data from the same mission (or test) and cross-correlation of data from different but related activities. The complexity and variability of the correlations in a research situation point to a key need for integrating data description, data manipulation, and presentation functions in an efficient user-oriented support function. One underlying element of efficiency is a careful structuring of data to the demands of the system just as the system is structured to the demands of the user community. Another area likely to produce major increases in performance is the distributed data base, i. e., the treatment of physically isolated data collections as an integrated whole.

Fault detection and correction on computers in a hostile environment opens new vistas for hardware/software recursive monitoring and control beyond the ground-based verification and validation fields.

Wind tunnel test data acquisition, reduction, analysis, archiving, and reporting has always presented a challenge for experimenters, who often have developed innovative ad hoc solutions to limitations presented by the particular computer system they must use. Judicious selection of raw data format can simplify data reduction and analysis. The data reduction process includes facility decoding, conversion of data to engineering units and then reducing data to forces, moments, pressures and temperatures in both dimensional and nondimensional form. Efficient storage and retrieval is important as it is for other types of experimental data.

Some of the problems facing all aspects of experimental data handling includes management of data; storing data never to be reused; faster, smaller, more sensitive smart sensors; pattern recognition and interpretation; and real-time acquisition systems.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

Database management

Graphics

Pattern recognition

VLSI

Expert systems for analysis of data

Workstations with graphics

Computer architectures

Real time programming languages

Advanced data recording media (e.g. laser disks, bubble memories)

FLIGHT CRUCIAL SYSTEMS

DESCRIPTION

In order to enhance the safety and utility of the nations aeronautical fleet (both commercial transport and general aviation), accurate navigation and guidance and in particular cost-effective advanced navigational systems and techniques must be provided. Advanced navigational systems must be installed at

non-instrumented, infrequently used runways. Better navigation systems with wide area coverage capability must be used to improve service at commercial installations. In addition, improved performance and safety goals are expected of future aircraft operations. Hence avionics and control technology must become an integrated part of aircraft design. Fundamental research on guidance and control concepts and integrated design techniques must be performed to complement navigation, cockpit avionics, and interface and integration research. An Agency objective is to stimulate improvements in aircraft performance.

Spacecraft designers are placing increasing reliance on sophisticated computer systems. More complex tasks are performed more economically by on-board computers. Most spacecraft systems: environmental, attitude, guidance, data acquisition, etc. can be monitored or controlled by computer. Some spacecraft requirements and relevant computer science disciplines are discussed below.

KEY PROBLEMS

Simulation

Digital computer simulation is used extensively as a tool in airframe and engine design. Digital simulation is easy to perform, cost effective, and accurate. Its major problem is that it is computationally intensive. Effective cycle times of nanoseconds are needed to perform large scale simulations in reasonable times.

Reliability

Future high reliability, high performance integrated avionics require validated fault-tolerant systems. The development of advanced avionics systems depends on fundamental research in fault-tolerant computing. Areas of critical importance are:

Theory of analysis and design of fault-tolerant systems

Development of a theoretical base and practical techniques for validation of fault-tolerant systems

Analysis of current experience and requirements of future fault-tolerant systems

Software which executes on flight crucial systems must be extremely reliable. This will require development of innovative techniques of software engineering - especially in system validation and verification.

Hardware

On-board computers have unique characteristics, many of which can benefit from the results of computer science research. On-board systems must be radiation hardened. They must be small, light, and cannot consume much power. At the same time they must be increasingly powerful. Thus, on-board systems in the immediate future must capitalize on VLSI research. Similarly, research in reliable, fault-tolerant hardware will have immediate applications for on-board systems.

Analysis Of Algorithms

Future flight crucial systems will be required to perform tasks which will depend on improved algorithms for computationally intensive problems. Solutions of navigation equations, optimization of flight paths, collision avoidance, and even complex displays for flight crews will depend on improvements in mathematical software and analysis of algorithms.

Artificial Intelligence

Any spacecraft, whether manned or unmanned, is expensive to control from the ground. Control of deep space missions is further complicated by the effects of signal transit time. These problems are being solved by assigning more operations, maintenance, troubleshooting, and exception handling functions to on-board computers. These on-board systems can be improved by applying the results of artificial intelligence research. Smart spacecraft will be more reliable and cost-effective.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

Computer Science research may contribute to the solution of this problem in several areas. New types of computer architecture may yield the necessary effective cycle times. One approach to the achievement of high performance simulators is to fashion them from arrays of processing elements where each element can be a complete microprocessor. While there appear to be no significant hardware obstacles to the design of such a parallel system, formidable modelling and software difficulties remain to effective overcome. Before the advantages of highly parallel architectures can be fully exploited, a base of knowledge must be constructed. Contention problems and asynchronous iteration methods must be better understood, and these results must be incorporated in applications programming languages.

As computers assume greater responsibilities on aircraft and spacecraft, they are applied in autonomous, single purpose modules. The next step will be to integrate these stand alone computers using local network bus concepts. Research is needed in the design of distributed systems for optimum sharing of information and resources. Improved combinations of autonomy and cooperation between systems will result in improvements in reliability, flexibility, and capability.

Flight crucial computing systems, both ground and aircraft based, depend on the reliability and efficiency of real-time operating systems. Advances in operating system design, both for general and special purpose applications are significant to the Agency.

OPERATIONS

DESCRIPTION

Research in the area of operations is intended to improve ground, airborne, and space system operational support. This research applies to all types of aircraft and spacecraft, from general aviation to advanced rotorcraft, and from manned spacecraft in low earth orbit to deep space explorers. Clearly, specific areas of emphasis differ for different types of vehicles and modes of operation.

Included in the area of ground-based research are traffic control, airspace management, runway occupancy, spacecraft command & control, operational data delivery, and maintenance diagnostics, including automation of maintenance procedures and information gathering to increase efficiency.

Research into airborne operations deals with navigation, flight path determination, communications, and air traffic control system considerations. Navigation involves determining where to go and how to get there, e.g., what route to take. Flight path determination is the generation of a desired trajectory that an aircraft should follow based on existing flight conditions. The trajectory usually is generated to optimize a particular objective function, such as fuel minimization. Flight communications research is concerned with the development and use of the data link capability used to send and receive information from ground-based sources and from other aircraft. Information transferred in such communication includes air traffic control requests, positional information, and weather conditions.

For space systems, a major step for the manned space flight program will be the transition of the shuttle from developmental mode to its operational era. This transition will be characterized by a drastic increase in launch frequency which will require all checkout, launch, and flight activities to be performed to the same degree of precision, but at a rate an order of magnitude faster than at present. It will be necessary to make better, more efficient use of computer resources if this goal is to be met.

KEY PROBLEMS

Ground-Based

Traffic control for runway occupancy, etc., must accommodate the changes in the air traffic control (ATC) systems. Models of the ATC are used to study the effects of proposed changes and accurately reflect the environment.

The maintenance diagnostic process is labor intensive, time consuming, and subject to human error. Hence, quality assurance is a continuing problem.

Airborne

Navigation depends on accurate, up-to-date information. Determination of the best route must consider the current ATC environment and weather conditions.

Generation of a flight path is based on minimizing fuel, a specific time of arrival, or some other such constraint. Calculation of the trajectory requires a large amount of information; for real-time trajectory determination, this data must also be timely.

Communication between the aircraft and ground-based sources as well as other aircraft is important and has great potential. However, problems in communicating through data links presents problems, including communications processing and quality assurance.

Air traffic control considerations from the point of view of the individual aircraft include collision avoidance, cockpit display of information, merging, spacing, etc. Flexibility in response to changes in ATC requests is also important.

Space

Variations in missions, vehicles, cargoes, and ground support equipment will require modification of launch procedures and, in some cases, system software for each launch. These modifications must be made quickly and correctly, and then they must be integrated and validated. The validation procedure is the greatest area of concern in this process. Better techniques of software engineering, especially validation are required in the near term.

Data from major tests are captured and archived by the Launch Processing System (LPS). Included are measurement values from vehicle and ground support equipment and all LPS command-response traffic. Data are recorded during critical tests, countdown, and launch. The volume of data accumulated during STS2 was 200 billion bits.

The purpose of this effort is to provide historical or near real time data to systems engineers for review and analysis. Access times range from a few seconds to nearly an hour, depending on the age of the data.

The retrieval function is normally used either for real time troubleshooting or for post test analysis. In either case, selected groups of measurements are made available for display or processing by online analysis routines. The complete data storage and retrieval system is a tool for performance monitoring and post test data reduction.

Although the system has worked well for the first three shuttle launches, the scheduled increases in launch rate threaten to degrade system performance characteristics.

A goal of the shuttle operational era is to reduce the number of firing room personnel during checkout and launch to less than 50. This goal may be achieved by creating expert systems within the Launch Processing System. Expert systems will also be needed for the Space Operations Center (SOC) - an intermediate term project. In both cases, cognizance of results of artificial intelligence research is necessary.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

Data communications devices

Better modeling techniques for process management

Problem solving & expert systems for maintenance diagnostics automation and discrepancy isolation

Increased data storage on-board, especially for flight path determination calculations

Computer communication networks and protocols

Expert systems for control center monitoring and diagnostic functions

Real-time systems for maintenance diagnostics, navigation, flight path determination

Systems analysis and engineering

Software engineering

Operating systems, process management

Optimization techniques for flight path calculations

Man-machine systems modeling

Database management

COMPUTATIONAL MODELING OF PHYSICAL PROCESSES

DESCRIPTION

Computational modeling of physical processes is a vital tool in NASA's research and technology programs. Major uses of computational modeling are to be found in the areas of:

Computational Fluid Dynamics (CFD)

Sound Generation and Propagation

Computational Chemistry

Aeroelasticity

Aerothermodynamics

Structures and Materials

Although these areas encompass a wide variety of physical processes, they can all benefit from the same advances in computer science. This is because many of the physical processes of concern in aeronautical and space research and technology may be described mathematically by the solutions to systems of partial differential or integro-differential equations defined on geometric domains with a variety of imposed initial/boundary conditions. Since analytical solutions to the mathematical equations describing these processes are rarely possible, the general approach is to use computers to discretize the mathematical model, solve the resulting algebraic systems, and to display the approximate numerical results both numerically and graphically.

The solution methods include finite difference, finite element, spectral, and combination methods. Further discussion of these methods can be found in the section Mathematics and the Theory of Computation (Appendix B). Computational Fluid Dynamics (CFD) uses all these solution methods and thus will be considered the generic type of all the various physical modelling problems throughout the remainder of this section.

CFD has developed in the last twenty years from an area of basic research to a commonly used design tool in the aerospace community. However, its entry into the design cycle is still severely limited. This is not for lack of adequate equations or physical understanding, but rather for inadequate numerical methods and limited computer resources.

KEY PROBLEMS

The efficiency of a numerical method for solving the governing flow equations depends on the number of mathematical operations required to obtain a solution. Early methods required thousands of iterations to obtain a converged solution for a high Reynolds number problem. New methods are being found to drastically reduce the number of iterations required as well as to reduce the number of operations per iteration. There still is much room for improvement and these activities are discussed in more detail in the section on Mathematics and Theory of Computation.

The time required to compute the flow about a wing-body combination using the Reynolds-averaged Navier-Stokes equations is such that it would take ten minutes or less if the computation rate was one billion floating point operations per second or greater. This is the minimum required speed and it is arrived at by assuming that future numerical methods will have 4 times the efficiency of those available today. It is interesting to note that the solution of the same problem would take about a month on an IBM 360/67, a day on a CDC 7600 and hours on a current supercomputer. These existing machines clearly are not adequate for the task at hand.

In addition to the speed of performing arithmetic operations, the other aspect of computational power that must be considered is memory or working storage. There are about 31 variables associated with each grid point in the 3-dimensional case since some of the quantities must be carried for two time steps. This number could be somewhat larger if complex turbulence models having more than two variables are required. It is estimated that a minimum of one million grid points are needed to resolve a 3-dimensional

flow field. This number is adequate for optimizing aircraft components but might not be enough to resolve the flow about complete aircraft having complex shapes. Of course, problems requiring more grid points still can be solved but the time needed for solution will be greater than 10 minutes at a billion floating point operations per second. Multiplying the number of variables per grid point by the number of grid points gives the amount of memory required. This totals to slightly over 30 million words for the 3-dimensional problem - an amount almost 100 times larger than that currently being used to solve 2-dimensional problems with the Reynolds-averaged Navier-Stokes equations.

Execution of these large computational models of physical processes presents the largest problem associated with computational modeling. At present there does not exist a computer system with sufficient speed and memory size to allow the solution of these problems in a timely manner. Other activities associated with computational modeling also present problems which could be overcome by advances in computer science and technology:

Code Preparation

Surface Geometry Preparation

Grid Generation

Development and Analysis of Numerical Methods

Result Presentation

Result Analysis

Code preparation is hampered by the necessity of using a heterogeneous set of hardware/software systems. For example at Ames Research Center a user is faced with the use of DEC, CDC, and Cray systems and at LaRC the choice is between CDC Cyber 175, Cyber 203 and Prime 750. This requires the user to know several operating systems and hardware architectures and slows his work. Also, there is an overhead of making files useable as they pass from system to system. This hierarchy of systems is necessary because we do not yet have a single system which can efficiently handle the spectrum of tasks ranging from batch "number crunching" to interactive text editing.

There is also a need to manage very large scientific databases. (Very large in this case is more than several million words.) Although large, these databases are usually well structured and frequently consist of large matrices or other large blocks of structured data (e.g., graphics images), which are manipulated within the analysis programs using the data structure. Some commercial DBMS systems can manage the storage and access of the large blocks, but cannot handle the internal imbedded structure. Often the computer memory is not large enough to contain the data being accessed, so the problem must be partitioned, or techniques must be devised to permit work on only part of the data. In addition, the access patterns to this scientific data are quite different from those the commercial DBMS systems are set up to handle.

Surface geometry preparation and grid generation depend on the use of graphics to display 3-dimensional data for validation of the data which will be used as input to modeling codes. Graphics devices and methods for displaying 3-dimensional surfaces and vector fields are not yet commonly available.

Result analysis and result presentation activities are also hampered by inadequate display methods as well as the lack of good scientific database management systems which will allow the researcher to quickly analyze and extract data from massive raw databases generated by computational models. As yet, researchers are unable to effectively display 3-dimensional scalar or vector fields.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

The most pressing problems for computational modeling of physical processes are the lack of sufficient computational capacity (in terms of both speed and memory space) to permit the most advanced techniques to be used on a routine basis and the lack of capacity to permit the development of more advanced techniques which will better model the physics.

Current computer technology is getting to the point where it is limited by the speed of light. It is no longer possible to assume that component technology will improve sufficiently to get the required speeds from traditional computer architectures. New architectures which compute several things concurrently will have to be developed. In the 1970s two important Single Instruction Stream Multiple Data Stream (SIMD) architectures were developed. One was the ILLIAC IV which utilized 64 lock-step parallel processors to gain concurrent execution. The other was the STAR 100 which used vector pipelines (like assembly lines in the auto industry) to perform floating-point operations on streams of data at much higher rates than are possible for individual (scalar) values. Although these SIMD architectures were able to achieve concurrent execution on a significant class of problems, they proved to be somewhat restrictive and not applicable to all problems. Multiple Instruction Stream Multiple Data Stream (MIMD) architectures offer the potential of improving on the concurrency of the SIMD architectures. Two leading architectures which are currently under consideration are user controlled multiprocessors and data flow architectures which are controlled by the availablity of data.

In addition to the hardware architectures, there must be accompanying advances in software at the systems level to efficiently use these new architectures. These software advances must be in the area of compiler optimization techniques for these architectures as well as the development of efficient operating systems and utilities.

These problems can only be overcome by active participation in the advancement of the state of the art of computer architecture (hardware and software) and the mapping of these problems onto these architectures. The other problem areas such as lack of adequate display devices and database management systems can be overcome by following the state of the art closely and adapting it to the solution of the problems.

MANAGEMENT APPLICATIONS

DESCRIPTION

Management applications of computer science encompass two distinct roles. First, there is the use of computer science and technology in performing administrative operations. Second, management's understanding of computer science issues and opportunities will enable its use in new ways.

As the unit price of computing and data processing have been rapidly decreasing much of the American (indeed worldwide) industrial base has put in place systems to improve productivity in manufacturing, and research and development environments [Sci Amer]. At the present time NASA lags significantly in implementing these systems.

KEY PROBLEMS

Administrative Operations

Computer science gives us the ability to structure many of the business and institutional problems of the Agency in ways not yet attempted. For example, an integrated file system will allow access to data not only by name of location (the traditional specifiers) but also by value. Thus it is possible to locate and identify problem items by virtue of the fact that they are problem items and are therefore 'different'.

Opportunity for using computer tools is rife within the Agency. The challenge will be in not merely replacing manual systems by computer systems but in reexamining the functions which are being performed and, where appropriate, implementing new computer assisted approaches to effect those functions. There is much activity in the Agency at the present time as many individuals recognize that opportunities to improve their operations. We must caution that is *essential* that individuals with strong computer science and management/business backgrounds be involved in the implementations of administrative operations. With a merely technological approach the results are almost always judged unsatisfactory from either a business viewpoint (failing to meet criteria which were not stated explicitly nor recognized by the technologists) or from a personnel viewpoint (the system is viewed as threatening or difficult/impossible to use productively).

Many of the routine administrative procedures in use in the Agency were designed and implemented for paper based records. They require that all actions be documented and journalled on a paper copy that moves from station to station. In many cases it is possible to replace the physical movement of paper (with its attendant transportation delays and potential for loss) by an electronic file which is immediately available at the next work station. Where flow need not be sequential, but must pass through a number of 'gates', the information may be accessed in parallel and released when all approvals have been received. Where required by regulations a paper copy of the action may be created for documentation purposes.

Management

When administrative operations have been computerized so that data are routinely available electronically then managers have the opportunity to call up and display data which are of particular interest. Even more important is the ability of systems to notify management automatically when parameters exceed predetermined limits or when specified events occur (or fail to occur). The definition of what is of interest to each manager must, of course, be left to the manager.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

The contribution to be made by computer science and technology is twofold: first, the computer systems which implement the required management and administrative functions and, second, a formal systems-level context within which the information flow of NASA can be represented.

The major impediment to implementing such automated management support systems is the reputation such systems have of being unwieldy, unresponsive, and at worst, of operating out of control. To overcome this (somewhat deserved) reputation two actions are in order:

1. Offer educational opportunities to NASA management.

2. Implement a high-yield, low-risk first example to gain confidence in the approach.

The area of management applications is on the fringe of computer science and technology as we envision it in the NASA environment. We treat it here so that it does not appear to have been omitted due to lack of importance and so there will not be a vacuum which causes impromptu and unsatisfactory responses to be generated.

ENGINEERING APPLICATIONS

DESCRIPTION

There are a number of areas where advances in computer science can be used to assist the engineers and scientists in the performance of their work. The following topics and areas appear appropriate for this section:

CAD/CAM/Computer Aided Testing

Custom VLSI Design; VHSL; Software Design, analysis, documentation, coding, and test tools; systems engineering, design, and construction of facilities, models, and test vehicles.

Computer Aided Instruction

Operator training, software tool training, project standards, expert software design consultation system.

Standards

Software products, implementation process, management/development networking visibility and reporting system, software change management.

Quality Assurance

Automated inspection and reporting system; handling, preparation and analysis system; accountability and traceability system.

Scientific Computing

This area refers to the provision of computing facilities and services to support scientific and engineering research at the NASA Centers; such computations include analysis, design, data reduction, simulation, and software development.

A typical scientific computing center consists of a network of medium to very large scale computers with access to a common data storage facility and a wide variety of input and output subsystems for local job entry and printing, remote job entry, interactive terminals, plotting and graphics, computer-aided design, simulation, data reduction, and possibly other specialized uses. Operating software and procedures are standardized as much as possible to reduce and simplify the information required for effective user interface. Usually both special and general purpose applications software are provided, often in the form of "libraries" of programs and subroutines. The computer center staff includes specialists who manage and operate the facility and provide system analysis and design, consulting, and applications support.

Facility Automation

Many test facilities at the NASA Centers are being upgraded to include microprocessors for control of the environment as well as as real-time control of the test operation. New facilities, such as the National Transonic Facility, include computer control as a basic requirement of operation.

Scientific Literature Access

Recon, DIALOG, and other on-line reference services are proving very useful to the NASA research community. The potential of video and laser disk technology for future on-hand reference material is an attractive alternative to current library services.

KEY PROBLEMS

The key problems in the area of engineering applications are both technical and nontechnical. The technical problems center around constructing an environment for each engineer or scientist which will meet his needs. In the description given above we have noted several classes and types of tools that would be useful to an engineer or scientist. It would be nice if all these tools could be modularized and made highly portable so that all of them could be used by a broad class of users. What is really the key technical problem is how to connect all these tools with each other and with common sources of information so that they can function in an integrated fashion.

The nontechnical problem is being able to demonstrate the usefulness of these tools well enough so that the people who would fund this work would be able to see the cost effectiveness of these "tool kits". It is hard to obtain cost benefit figures for some of these tools. They are not focused on the solution of one large, clearly identified, and clearly felt problem which is of immediate concern.

Some of the major problems are in the areas of: Scheduling, operations and maintenance management; Performance measurement; High-speed computing; Massive file storage and management requirements; User-friendly software; Reliable and efficient systems software; Portability of applications software; Complexity of applications program requirements; Currency of information for system users; and Information storage and retrieval.

POTENTIAL COMPUTER SCIENCE CONTRIBUTIONS

We have given below detailed examples of potential computer science research contributions in the areas of project engineering, computer assisted tools, standards , and resource sharing. The other areas of computer science which are applicable to this topic are simply listed following these examples.

Project Engineering

In the engineering of a large project one is faced with a large measure of complexity. There are a large number of interrelated items involved in the engineering effort and this set of interrelationships must be monitored and managed. Computerized tools can be used to assist in this process. The research efforts needed for the development of these tools arise from work in the areas of highly reliable systems and engineering data management systems. The tools can be used in requirements generation and analysis and traceability analysis, resource and schedule estimation, life-cycle costing, life-cycle performance data archiving and analysis, change management, and resource utilization monitoring. These tools will relieve the project management and project staff of routine tasks which a machine can do.

Computer Assisted Tools

Tools can be designed which will assist the engineer in learning about a system, in designing a system, and in tracking information about a system. Computer aided design systems are useful for dealing with complex systems and nearly all the systems that NASA must design are complex. Computer aided education tools will allow engineers to learn about systems within NASA with which they are unfamiliar. They should be able to assimilate the information that they need more rapidly than if they had to search through volumes of technical manuals, or locate people who built the system. Work on scientific and engineering data management systems will be needed to develop these tools. Computer assisted bibliographical searches enable engineers to locate quickly all the information available on a particular subject in the published literature. This makes it possible or convenient for people to stay current in technical areas relevant to their work.

Standards

By developing and imposing standards one can help to ensure reliability and portability of information systems. Information systems' standards can be developed in the area of software development methodologies. These will help to ensure that appropriate methodologies are used in software design and will make it easier for other people to review software designs. If standards are developed for applications software then they tend to increase the portability of the software and also will aid other people as they examine the software. In addition, there should be standards for documentation of software systems. Finally standards for data and data formats would enhance the sharing of data.

Resource Sharing

Even in this time of decreasing hardware costs there are still advantages to the sharing of resources. For the NASA community this concurrent use of computing resources can be done on both an intra-center level and an inter-center level. This can be accomplished by local networks within a Center, and low or high bandwidth networks among Centers. It is also advantageous to have an interconnection with other centers of computer science research outside of NASA. By this means one can take advantage of special computers (e.g., Cray 1) or special application routines which are available only at certain installations.

In addition to the sharing of computer hardware there is the need to share other objects. If the software that NASA designs is portable then software can be shared among NASA Centers. Data, either from research or from missions, can be shared over these same networks. Finally documentation can be shared more effectively among Centers. As an example, the preparation of this program plan for computer science has been accomplished by all the NASA Centers having access to a common advanced office automation system through a national packet switching network. This has facilitated immeasurably the preparation of the document

Other areas of computer science research in which work done could lead to development and application of tools for engineering application include:

Computer system modeling

Performance measurement techniques

Distributed processing, networks, data communication

Mass storage devices, database management

Interactive and graphics techniques

Software tools and techniques

Software design technology

High-level languages

Computer-aided design

Electronic mail

Computer-aided education

APPENDIX B. TECHNOLOGY ASSESSMENT

INTRODUCTION

This portion of the program plan is a survey and assessment of the state of the art in various disciplines of the computer sciences with an indication of the future trends in research, particularly in the areas most related to NASA's needs. The survey material primarily addresses those areas of computer science that have a relevance to the programmatic goals of NASA.

This assessment also has been focused by the three themes within computer science identified as being especially relevant to NASA's computing needs. As research is done in these areas it is important that NASA use or advance from the current state of the art and not repeat or redo work already done or in progress. Thus a state-of-the-art assessment is an integral element in this plan.

The taxonomy for the material presented here is given in Appendix D and is based upon the one developed by the Association of Computing Machinery presented in the January, 1982 issue of the Communications of the ACM.

HARDWARE

INTRODUCTION

Computer hardware includes the technology, physical and logical structures, physical and logical elements and engineering aspects associated with control structures, firmware and microcode, arithmetic and logic structures, memory structures, data, command, and I/O processors, data storage, logic design, and integrated circuits. Insofar as NASA's needs, requirements, and applications are concerned hardware research is applicable to three primary categories of hardware: spacecraft, airborne, and ground-based computer systems. Spacecraft systems pertain to launch vehicles, shuttle and spacelab experiments and payloads, free flying payloads such as ST, GRO, GP-B, etc., all types of mapping, weather, communication, and tracking satellites, and interplanetary systems of the Viking, Voyager, Galileo class. Airborne systems include aircraft instrumentations, controllers, guidance equipment, and test equipment. Ground-based systems include scientific computers, automated launch vehicle processors, computers used in mission and operational control centers, numerous classes of computers used for modelling and simulation running the gamut from those used for small control systems to those used in weather forecasting and prediction, satellite data processors entailing data processing and manipulation, image enhancements, pattern recognition, etc.

NASA's needs, requirements, and constraints among each of these three classes can differ considerably. For example, spacecraft systems are constrained significantly in power, weight, volume, reliability, and other engineering and packaging aspects. Ground-based systems require large amounts of memory and must have high throughput rates. These needs, requirements, and constraints usually result in different architectural approaches in control structures, memory structures, data storage, and computer engineering. For example, parallelism such as multiprocessing, pipelining, array processing, etc., is used to obtain high throughput which may be the forcing function in ground-based systems while reliability requirements in airborne and spacecraft systems may dictate long life and fault-tolerant architectures. Even though considerable differences exist in these three classes of applications, a common interest exists in basic technologies and the areas of firmware and microcode, arithmetic and logic structures, logic design,

and the development of very large scale integrated circuit technology. By their very nature, ground-based architectures are more general purpose and generally depend more heavily on commercial developments than non-ground-based systems, because these latter systems are usually more specialized requiring special designs and engineering considerations.

Some basic technological considerations for these computational systems will be addressed and typical areas requiring further research and development will be considered. These areas are not intended to be exhaustive, but only representative of the type of research and development activities which NASA should pursue in the forthcoming years.

VERY LARGE SCALE INTEGRATED CIRCUITS

Tremendous strides have been made in integrated circuit technology within the last decade. For instance, device density (gates/mm) has been doubling every two years and speed power product has been decreasing by a factor of ten every five years. Figure B-1 presents the integrated circuit densities for both memory and logic chips beginning with the development during the 1960's. For semiconductor memories, device capacity has increased by a factor of approximately four every three years while the relative number of bits per dollar has increased by a factor of three every three years. These trends are expected to continue in the forseeable future. The Motorola 68000 can place all its transistors on 265 x 265 mils of real estate. Laboratory chips of the same physical size have been fabricated which contain approximately 500,000 transistors. Thousands of companies whose future depends on staying at the leading edge of this technology have invested billions in such developments. In fact, industry probably invests yearly more than 20 times NASA's total yearly budget in such developments.

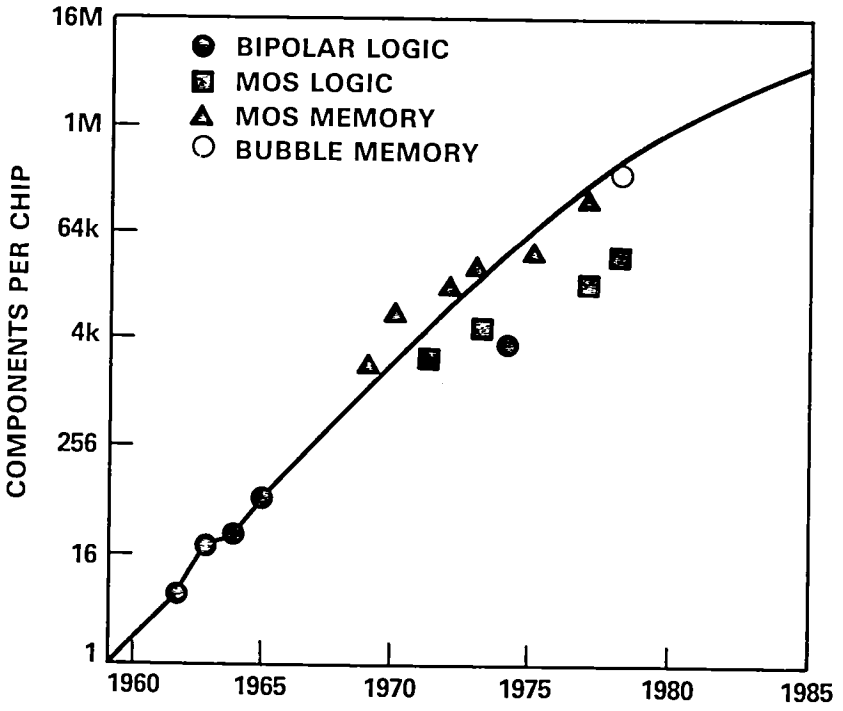

Figure B-1. Integrated circuit densities.*

*Reprinted from Bhandarkar, D.P., "The Impact of Semiconductor Technology on Computer Systems," *Computer*, September 1979, Copyright © 1979 IEEE.

Most of the industrial capability, however, is devoted to the production of general purpose components for consumer products that can be sold in large volume. Such components can be expected to meet only a fraction of NASA's needs for function capability, performance, long term reliability, and correction operation in the presence of large temperature changes, for example, or radiation. NASA's role in VLSI must be to ensure that it has the capability to develop the customized VLSI chips that will satisfy specialized but low-volume needs that cannot be met by standard commercial or military components. This capability must include several things:

An understanding of the methods of designing algorithms for VLSI implementation, and mapping complex logical circuits onto silicon

Computer aided design tools that will relieve the designer of the burden of managing all the details of complex circuit layout and specification, and thereby make custom design economically more feasible

Access to, and methods of working with, state-of-the-art industrial mask and wafer houses for VLSI fabrication, since NASA cannot afford the cost of building and maintaining such services for its own internal purposes

There are areas which are of special interest to NASA (and DoD) and to which industry is devoting very little attention. Two examples are radiation hardening of semiconductor devices and the need for standardization of widely used semiconductor chips. Caution should be exercised in the latter case because rapidly developing technology leads to obsolescence in a very short period of time.

A fundamental problem has been encountered in applying VLSI technology to spacecraft NMOS memory systems for earth orbital and interplanetary missions. In high density technologies, with small cell geometries and less electrical charge per gate, less ionization potential is required to discharge the gate capacitor in the memory cell, resulting in increased vulnerability to radiation induced random soft errors. These soft errors result either from internally emitted alpha particles or from impacts with cosmic ray particles. In addition to the soft errors, long term exposure to radiation can result in hard failures not only in memory devices, but in other support electronic circuits as well. Many of NASA and DoD programs are sensitive to this problem.

One approach to the solution of this problem is to develop or use a technology less sensitive to radiation. Radiation hardened silicon-on-sapphire (SOS) complementary metal oxide semiconductor (CMOS) is a candidate technology. The basic requirement is to develop a technology and a set of components which can be used to implement a radiation hardened onboard computing system.

PRIMARY MEMORY SYSTEMS

Another area which should be addressed falls in the categories of spacecraft memory structures and computer engineering. Electro-mechanical memories such as magnetic tapes and disks have served for years as satisfactory and low cost mass memories for ground-based systems. They have been pressed into spacecraft service simply because no other systems existed. They are unsatisfactory for spacecraft applications for several obvious reasons. One possible solution to the spacecraft auxiliary memory problem is through the application of magnetic bubble technology.

Magnetic bubble memory technology has been touted and pursued for a number of years as being the answer to onboard as well as commercial bulk storage because of its faster access, higher bit rates, lower power, non-volatility, high reliability, non-mechanical implementation, and ultimately lower cost. Production problems, notably bit sense and yield, surfaced which could not be readily overcome and has led

to the abandonment of the technology by such companies as Texas Instruments, Rockwell, and National Semiconductor. However, developments are continuing at Western Electric, Intel, Plessey, and several Japanese companies. Western Electric produces magnetic bubble memories for their commercial operations and have sold several large systems to the Air Force.

Considerable effort has been expended both at Langley and Wright Patterson on magnetic bubble memory device technology. This effort has focused primarily on the physics and engineering aspects of the technology as opposed to the development of a memory system for flight applications.

Although production problems have been encountered, the magnetic bubble memory technology is essentially here and should be used for airborne and space applications. To date, no satisfactory auxiliary or bulk storage memory system for utilization as an extension to a flight computer mainframe memory has been developed or flight packaged. The need for such a system is clearly illustrated in the Space Shuttle and Spacelab programs where various bulk storage techniques are being employed, but these at best serve as a stop gap measure only.

The magnetic bubble memory system offers several outstanding advantages and characteristics over existing systems. It has faster access time, higher bit rates and lower power. Parallel readout and error correcting codes can be readily incorporated. It is inherently more reliable because of the technology employed and because it has no moving parts. In addition, magnetic bubble storage can be mass produced for a fraction of the cost of current systems.

FIRMWARE, MICROCODE AND CONTROL STRUCTURES

Microprogramming, as originally conceived by Wilkes in 1951, is a systematic and orderly approach to designing the control section of a computer system. The microprogramming concept generated modest theoretical interests in the 1950's but was not widely accepted, one reason being that the cost of implementation was prohibitive. Microstore implementation advanced from magnetic cores to semiconductor technology in the 1960's with a major burst of interest when IBM announced the System 360 series in which all but the largest systems were microprogrammed. State-of-the-art technology has developed to the point where 32K bit chips of erasable programmable read only memory is more or less an industry standard and 128K bit chips of mask programmable read only memory is available. This technology can be expected to lead to new and innovative approaches to software/hardware system development.

Present airborne and spaceborne computer systems execute program code in machine language. Software may be written in assembly language or a higher order language (HOL). If written in HOL, several translation processes are required to obtain the object code. Translators, compilers, assemblers, etc., are required to yield the final object code and unless the airborne or spaceborne computer system architecture "emulates" a well known commercial system with well developed support software tools, this support software must be developed specifically for that particular flight system which is time consuming, costly, and error prone. It is possible through firmware and microcode to implement compilers, assemblers, etc., directly at the application computer level. This may be accomplished by incorporating the required assembler etc., in every machine, or designing the function directly into the machine's control structure.

ARITHMETIC AND LOGIC STRUCTURES

In the mid-1800's, George Boole developed a two value logic system which today is known as Boolean algebra. It was almost a hundred years later before Claude Shannon utilized it to simplify logical or switching circuits. Since the logical elements employed in the earliest computers were implemented with relays and later when vacuum tubes were developed with bistable multivibrator circuits (flip-flops), it has been natural for systems to evolve to the point where they are today utilizing two state logic almost ex-

clusively. However, under certain assumptions, i.e., that the complexity of a logical element is proportional to the number of states in which it can exist, it can be shown that the simplest overall system can be achieved utilizing logic elements with the natural number e states; i.e., 2.78... This implies that tri-state devices are more efficient than bi-state elements. It is suggested that further investigations into multi-value logic elements and systems are needed to determine whether or not possible gains in efficiencies are worth the increased complexities.

TESTABILITY

VLSI technology, which makes the implementation of hundreds of thousands of devices on a single chip possible, has also brought unique problems in testability. For economic reasons, it is desirable to place as many devices as possible on a chip, but low chip yield results when device density exceed certain limits. For this reason, along with power dissipation and pin limitations, device density and chip size have practical limitations. The problem is how does one test and verify that all the thousands of devices or logical networks on the chip are functional. Combinational analyses are time consuming and cost prohibitive. The situation can be further aggravated when redundant devices are placed on the chip which is often the case in memory chips. Both NASA and industry can benefit by exploiting this common area of interest.

MASS DATA STORAGE

Considerable potential and demand exists for mass data storage hardware. The primary mass storage device in use today for storing sensor data in ground-based computer systems is high density magnetic tape. The "tape approach" has been able to handle the recording of data in the 1-20 megabit range even though the tapes must be further processed in an off-line environment before the data is available to the user. With the increase in bandwidth required for near term and long range missions, the present tape systems will be unable to handle the high rates and large volumes. The development of a new technology for mass storage devices, such as optical disk, provides an alternative that can handle the near term requirements of 25-50 megabits and at the same time can be further expanded to record at rates of 50-3000 megabits. In addition to the high data rates, the optical disk provides a large volume capacity (10E11 bits) that can be used on-line with a data base management system which in turn can make the recorded sensor data available to users in near real time. Presently, an optical disk read/record system with rates up to 50 megabits/sec and on-line volumes of 10 bit is being pursued. The primary record media is a 12" disk containing 75 x 10" bits and has an archival life of 10-15 years. Continuing technology development is needed in the area of mass storage devices and data base management system that can respond to the high record rates of the storage.

COMPUTER SYSTEM ORGANIZATION

As hardware becomes less and less expensive, both in an absolute sense, and relative to software in particular, NASA will have to utilize new technologies and new system architectures to take advantages of the new programmatic opportunities that present themselves in a cost-effective manner. Until recently traditional mainframe computers and mini-computers have dominated the ground-based portion of NASA's computing environment. The space-based portion of the environment has consisted of specially built hardware utilizing custom designed equipment and developed on a case by case basis. The computer systems that NASA uses and/or designs in the future will be based on micro technology which is exploding on the electronics scene today. Architectures will be developed based on concurrent processing concepts to meet the high computational requirements of future programmatic thrusts with NASA. Thus this review of technology for computer systems organization will focus on the areas of microprocessor technology, large processor architectures, and computer networks.

MICROPROCESSOR TECHNOLOGY

The workhorse of the microprocessor industry at least for computer applications has been the 8-bit microprocessor. The sixteen-bit processors such as the Intel 8086, Zilog Z-8000 and the Motorola 68000 are coming into their own now. Software that was originally created for the 8-bit microprocessors is being moved to the sixteen-bit architectures. Operating systems such as CP/M are being upgraded to work in a distributed environment (CP/NET) and also in the multiuser sixteen-bit environment (CP/M-86).

Personal computers have been built around 8-bit chips. Usually the maximum amount of primary memory available to the user in these systems has been 64K bytes, based on the fact that the number of binary addresses which can be expressed in two bytes is 64K. These machines generally ran at 2 to 4MHz and had access to floppy disk drives each of which held approximately 240KB of information. These systems have served well as word processing systems, processors for running modest size programs, and as smart terminals for use in communicating with large mainframes or minis.

There is now appearing on the scene a new generation of microcomputer systems which is based on the 16-bit architecture. For example, the Wicat System S150 series ranges from a one user to a six user system and is based on the Motorola 68000 microprocessor and operates at 8MHz. Each member of the series comes equipped with a 10 MB Winchester hard disk, a 960 KB floppy disk, memory management capability, numerous communication ports and local network capabilities, operating systems, and computer languages for between $8,000 and $12,000 (March 82 prices) complete. Since these computers are based on a 16-bit architecture it is also possible to use other more sophisticated operating systems such as Bell Labs' UNIX. The research that must be done now is to develop an understanding of how to integrate these powerful systems with the larger mainframes and special purpose computers so that the load balancing of tasks is done effectively. A significant portion of the computing effort required for the total system can be done on the personal computer and the remainder, the part requiring large "number-crunchers", can be done on the larger machine.

Microprocessor technology is being applied to many special purpose applications also. In fact a great deal of research is going into the techniques to be used in designing custom VLSI chips. In this way the "end user" can eventually design the types of chips he needs and not depend on the large firms such as Intel and Texas Instruments to produce the chips he needs. Algorithms that are the central part of many computing methodologies are being implemented on chips. NASA is currently using this technology to improve its capability in image processing and computer graphics. Intel has acquired by merger a firm specializing in data base management systems and is considering implementing data base management systems using specialized chips. DEC, Xerox, and Intel have joined forces to work on local area network development. Intel is supposedly building micro components which will implement the appropriate network software. Thus the research in this area appears to be in the direction of making it easier to solve particular computing problems using specialized micro technology which can be developed in a systematic manner.

Microprocessor technology has had a considerable and rapid evolution since it first appeared on the scene in the early 1970's. The width of the data bus of the microprocessor has been used as a rough measure of the sophistication of the microprocessor chip. The data bus width has increased from 4 bits for the Intel 4004 to 8 bits in the Intel 8080 to 16 bits in the 8086 and ultimately to 32 bits in the iAPX-432 from Intel. The large hardware firms seem to be very well prepared to produce the standard chips that NASA and other organizations will need. Customized microprocessors, designed to accomplish specific tasks, may be an area of research which NASA can profitably pursue.

PROCESSOR ARCHITECTURES

The predominant architectures which have been developed to date have been based primarily on the Von Neumann style of architecture in which sequential processing of instructions is the manner in which instructions are executed. Today it is becoming clear that one must use a different style of computer architecture if one is going to be able to solve the large computational problems that NASA must solve. Invariably this technology will revolve around the concept of parallelism in computing - having more than one sequence of actions taking place at the same time, either synchronously or asynchronously. The form that this parallelism in computing can take is varied, as can be seen from the discussion below.

In order to achieve the computing rates needed to solve today's computational problems, technologies have been refined and improved so that individual processors have become faster and faster. Computers have advanced through several generations - the first generation incorporated vacuum tube technology, the second generation used transistors, the third used integrated circuits, and the current generation is using medium and large scale integrated circuits. In each case the speed of the individual processors was improved in the process. Concepts involving parallelism have been incorporated into the technologies, for example, in the form of separate I/O processors, floating point processors, timesharing and multitasking operating systems, and pipeline processors. Technologists are coming to believe that it will be hard to speed up processor speeds enough so that general single unit processors will operate fast enough to solve large scale scientific problems. Some computer architects have stated that they don't believe that the fifth generation of computers will be based on the classical Von Neumann concept. Undoubtedly the fifth generation of computers will involve parallelism significantly.

Where possible NASA should and probably will take advantage of any general purpose computer that industry develops which will help in meeting NASA's needs. NASA will, however, have to do research and development activities on the classes of "supercomputer" architectures that it, and a small community of other science centers, need for rather unique problems. Thus the discussion which follows will discuss the architecture of supercomputer processors or architectures particularly related to NASA's needs.

We have gone through at least two generations of supercomputers. The first generation included the CDC STAR-100 and the TI ASC, both of which were designed to handle arrays of data very efficiently. The second generation of supercomputers began with the introduction in 1975 of the Cray-1 followed by the CDC Cyber 203 and 205 and the Burroughs Scientific Processor. This second generation reflected the ability of the machine to handle scalar as well as vector operations very efficiently.

The concepts which are being investigated today for developing the next generation of supercomputers will be based on using high-speed special purpose processors, processors operating on multiple sets of data with single instructions (SIMD), and multiple processors operating on multiple sets of data (MIMD). Supercomputers can be designed to be general purpose or to be special purpose, solving only a very limited problem. The price one pays for generality in such systems is decreased speed, decreased efficiency of hardware utilization, and increased software requirements. The following architectural concepts represent the directions in which supercomputer technology is proceeding.

Multiple special purpose functional units - The concept of systolic architectures, developed quite extensively at Carnegie Mellon University, deals with the systematic (as opposed to an ad hoc approach) development of special purpose functional units designed to implement certain specific algorithms. Data flows from computer memory to the unit and back in a regular rhythmic cycle (hence the adjective systolic). Researchers have demonstrated the usefulness of this for solving

triangular linear systems of equations, convolutions, filters, and FFTs. The systematic approach to the design of such systems is necessary in order to ensure that it integrates well in the architecture of the system, not causing for example some memory bottleneck. These special purpose units must of necessity be incorporated into larger architectures.

Associate processors - These computers are built around associative memory technology, where information is manipulated based on the data content rather than the data location. This concept allows a large degree of parallelism. The processors that are used are in some sense attached to each cell of memory. Goodyear Aerospace Corporation developed the STARan computer using this technology, for application to automated cartography problems and image processing problems. This technology is being considered today for use in data base machines since information can be located by content and not by a complicated series of pointers or indices.

Array processors - These are the classical examples of Single Instruction Multiple Data (SIMD) machines. They are useful when arrays of data must be processed. In order to be used effectively they must be tailored to the specific application being solved. Great increases in speed are possible using only a large collection of moderately faster components. The network that interconnects these individual processors must also be tailored for the special application for maximum effectiveness. NASA has sponsored the development of the Massively Parallel Processor, which is based on the SIMD concept, and has intended it for use in image processing of remotely sensed data. The processor has 16,896 processors arranged in a 128 row x 132 column matrix. An implementation of the nearest neighbor interconnection network has been adopted. The design of the system has been tested out by numerous simulations.

Data flow computers - These are computers in which execution is controlled by the availability of data. An individual instruction is executed as soon as the data needed for that operation is available. A different style of languages is needed to describe the computational steps the machine should execute. This architecture has the potential to give good increase in speed (results of some analysis predict a 100 fold increase in a weather analysis model over that achieved by an IBM 360/91). As in other architectures the associated problems of memory conflict and processor interconnection must be solved. This type of technology may be used as an attachment to a main processor.

Functional programming language machines - These computer architectures would be based on the ideas of functional or applicative programming in which operators manipulate functions directly rather than data. Languages for such systems would be like those introduced by John Backus [Backus]. Actual systems have been developed on these concepts but its integration with regular machines is still a research topic.

Multiple processor architectures - These are the machines that are referred to as multiple instruction multiple data (MIMD) machines. The multiprocessor mainframes commercially available now use this conceptual model. The increase in speed that can be achieved by N of these units is at most N times that of a single processor. Full linear speed-up may be hampered by synchronization requirements, the nature of the algorithms implemented, the fact that parallel algorithms may take more steps per unit than a sequential algorithm, and the contention and scheduling of resources. On the positive side they are extremely flexible because control is effected by means of software. If designed appropriately the total number of nodes can be changed, and each node can have different performance characteristics. Reliability, survivability, and modularity are increased because of the loose coupling between systems. System upgrades can be made more easily than in other architectures. One does sacrifice speed for this flexibility.

The C.mmp and the Cm* are examples of systems developed at Carnegie Mellon University which have this type of architecture. The C.mmp was designed with 16 processors and 16 memories which were connected by a crossbar network. Linear speed-up on a number of algorithms was achieved. The Numerical Aerodynamic Simulator (NAS) being developed by NASA is an example of the MIMD architecture. It is projected to be capable of handling 1 billion floating point instructions per second. A baseline (cube) network connects the 512 independent processors with the 521 memory modules that each processor can access.

Fault-tolerant computing is an aspect of computer system organization for which NASA has a very critical need and NASA has stimulated much of the research in this area. Fault-tolerant computing has four aspects to it: fault detection, fault management, fault diagnosis, and fault recovery. Faults can occur in a system on a permanent basis, or, more dangerously, on an intermittent basis. As the architectures being developed today become more distributed the problems associated with fault tolerance change and grow more complex. There are more options available as to how one provides redundant systems. New techniques are being developed in self diagnosis of faults and how recovery is achieved. NASA has sponsored the development of fault-tolerant systems such as the Fault-Tolerant Microprocessor, FTMP, and the Software Implemented Fault-Tolerant System, SIFT. Fault-tolerant computing is maturing into something more than a collection of techniques and tools. We may soon be able to synthesize a few highly modular architectures, which could provide good fault coverage and allow cost-effective system configuration in a wide range of applications.

While this discussion has focused on hardware, it is absolutely critical to realize that in order to make this (in some cases, radically different) hardware useful in solving NASA's problems, a great deal of work must be done in constructing software and in constructing algorithms which will allow the translation of the problems NASA must solve into implementations which take advantage of these hardware advances. This will involve a considerable amount of research if full utilization of the hardware is to be made.

COMPUTER NETWORKS

Whenever processors are proceeding in parallel, some form of communication may be needed between them. The processing elements can be two processors in a tightly coupled multiprocessor computer, two people sharing computer resources within one local facility, or two cooperating computing activities which are geographically separate. Computer networking technology is evolving to handle these three types of situations - interconnection networks, local area networks, and long haul computer networks.

Interconnection networks are a very active area of research. There are various topologies which are possible for the networks - cross bar, star, cube, ring, bus, and fully connected. The implementation of each requires research in certain technologies but the most critical aspect of the research on these relates to how processors should be interconnected in order to efficiently perform their intended functions. If one uses a fully connected network (whose complexity must increase quadratically with the number of processors) one will have all the connections needed but this may introduce such complexity into the design that it is not practical. Too few connections may cause the routing of data through numerous processors before the data reaches its final destination. Tradeoff studies and network simulations must be done to determine the best interconnection strategies.

Local area networks are being developed now to meet the telecommunication needs present within one physical facility or within a geographically close area (e.g., 1 mile radius). A variety of commercial vendors are developing approaches to local network implementations. All these implementations are not the same and this can be the source of some serious problems. There are two primary techniques suggested

for the communication carrier baseband techniques and broadband techniques. Baseband techniques are less expensive to implement yet have more limited capability. The bandwidth of the networks vary from several hundred kilobits per second up to several hundred million bits per second in transfer rate. There is software being developed by the vendors which implements certain network protocols.

The research that must be done on local area networks is to develop complete sets of protocols and their implementation on heterogeneous machine architectures which will be useful for the particular applications for which the network is intended. Some very preliminary work is taking place in this area. The IEEE is trying develop IEEE 802 Local Area Network Reference Model and Standards. The work is ongoing and the committee working on this is trying to decide between baseband and broadband, token access versus contention schemes for gaining access to the network, and similar difficult issues. The research that will be most beneficial to NASA is the development of a set of protocols that it can use internally among and between all the Centers and to develop or acquire highly transportable software which can be used on a variety of computing systems to connect them to the network. This is a rapidly evolving technology and NASA should simply experiment with it to be aware of its full potentialities.

The development of networks to serve geographically dispersed locations is much more mature than the technology for local area networks. It nonetheless is a rapidly evolving field. As is the case with local area networks there are many vendors who are implementing long distance networks or who are establishing various (and mostly different) implementations of network protocols, and the major problem is the ability for the NASA community to be able to talk to all the computers and terminals that it needs to.

The two standard switching techniques which are used to establish global networks are packet switching and circuit switching. The most well known implementation of public networks have used the packet switching techniques. The ARPA network, funded by DARPA for the Department of Defense, was the first significant implementation of the packet network concept. It was begun in 1968 and has grown from an initial set of 4 nodes in late 1969 to over 60 nodes presently. The ARPA network is still an evolving system and many of the lessons learned in the building of the ARPA network have been used in the construction of other public packet switching networks, such as Tymnet and Telenet. Packet switching networks exist in other countries and standards are being developed to the point where these networks are being interconnected. In order to use these networks in a cost effective manner NASA must have implementations of the X.25 packet switching protocols which are compatible with these public networks. These implementations must be available for all the various machine architectures found in NASA. The development of transportable software for this purpose is a research effort that needs to be undertaken.

The network protocols which have been implemented for the public packet switching networks represent only the bottom "layer" of the set of protocols needed for the proper interconnection of processes at different computer sites. This bottom layer of protocols simply assures that a series of bits will be faithfully and correctly delivered from one computer to another computer. The meaning of the bit stream is not specified. The fact that the two computers may "speak different languages" is not recognized by this level of network protocol at all. This bottom "layer" is in fact the bottom three layers of the 7 layer Reference Model of Open Systems Interconnection established by the International Standards Organization. The public packet switching networks have implemented versions of the X.25 protocol for these bottom three layers. Work has been in process for some time to standardize the upper four layers of the model.

Research should be done on these higher level protocols so that NASA's needs in the transfer of scientific information and in the sharing of programs and processing resources are being addressed. NASA should be involved in the development effort and should be experimenting with various versions of the protocols

and their implementations to have some measure of which protocols and implementations it should aim toward. If NASA is able to come up with a common set of protocols which can be used among all Centers (and hopefully other members of the scientific community) then NASA will have gone a long way in solving its data and resource sharing problems.

SOFTWARE

Historically software and hardware have been the two components of all data and information systems. They are the two elements of an information system for which the cost of development and maintenance are most well known. In the early days of computing (circa 1950's) the cost of hardware far exceeded the cost of software. As a consequence software was designed around hardware. Today the situation is just the opposite. Hardware is becoming ever cheaper and software costs continue to escalate. Figure B-2 depicts the changing relationship between hardware and software costs over time. There are several reasons for this high cost of software and the work that is done in the area of software research is intended to help keep this cost to a minimum.

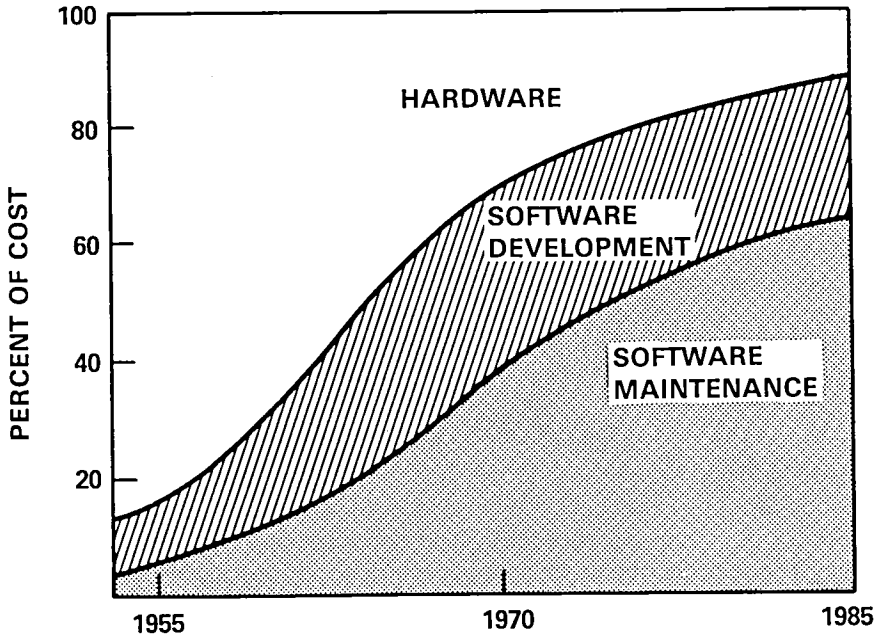

Figure B-2. Hardware-software cost trends.*

There are three ways that software is different from hardware and thus why the expense involved with software is now higher than the cost of hardware.

1. The logical complexity of software is greater than that of hardware.

2. Software accumulates over time. It is not a static quantity such as is hardware which essentially remains fixed in structure once it is built. It is also much easier to change than hardware, hence it changes over time as new facilities are added, etc.

*Reprinted from Boehm, Barry W., "Software Engineering: R&D Trends and Defense Needs," *Research Directions in Software Technology*, P. Wegner (ed.), Copyright © 1979 by The Massachusetts Institute of Technology.

3. Software has a heavy involvement with people. People must specify what they want the software to do, people must translate this into actual systems, and these systems must be used by people to meet their needs.

The research work that is reviewed here attempts to address at least these three problems.

SOFTWARE ENGINEERING

Software Management

Since today's software development projects are complex and are in fact part of very complex systems, it is essential that their development and control be managed carefully. The challenge is to combine technical methodologies and tools into a structure which facilitates adequate control and management of the development process.

Major advances in software management have come about because we understand the life cycle process of software development better than we used to, and also because we better understand the human organizations needed to bring about this development.

We now realize that the early phases of the life cycle of a software development project are very important and that improving the management of these stages can result in very cost effective improvements in the quality of the final product. If systems are to be operational for any period of time then they must be designed in such a way that system maintenance is facilitated. This can ensure easier management of the software during the latter phases of its life cycle.

There are organizational structures that enhance the production process for software development. The concept of chief programmer teams, structured walkthrough techniques, and program librarians, are examples of ways that the structure of the people assembled to do the task can affect the success of the task.

New management techniques are being developed and continue to need to be developed which recognize the nonsequential nature of software development. Software development does not proceed serially through requirements, design, implementation, test, and operation, but rather these components proceed simultaneously and repeatedly at various levels while the software is being developed. Because of this parallelism and concurrency the complexity of the management problem increases, but if done effectively, so does the gain in productivity.

Requirements and Design Methodologies

The main issue that programming methodologies are attempting to address is how to handle the complexity of software systems. In the early years of software systems, the systems could be designed by one individual and the execution could be monitored solely by that one individual. Today nearly all systems are so complex that they must be designed and implemented by a team of people.

Numerous results document the fact that more cost effective systems are built if careful attention is paid to the requirements and the design phases of the development. Some of the problems in the area of requirements and design are:

Stating requirements in a formal manner so that they can be analyzed automatically for consistency, completeness, and correctness.

Translating requirements for a system (i.e., what it is supposed to do) into designs for a system (i.e., how it is supposed to do the work), especially so that traceability of requirements is possible.

Evaluating designs to assess their performance characteristics and the costs of design implementation and system maintenance.

Devising test criteria to assess the work that is done during the different phases.

There are a number of different methodologies which have been developed to deal with requirements specification and analysis, system design, and system implementation. All recognize as their fundamental challenge the ability to handle complexity. These methodologies all suffer from the trouble that each is fine in its own particular area but they need to be engineered into an integrated system that has a smoothness of flow from requirements to design to implementation.

Validation and Verification

Validation and verification are the processes of assessing and assuring the quality of the software and its conformance to requirements and design specifications. It is absolutely critical to perform these functions for systems where the existence of an error can be catastrophic. It consumes a large amount of resources in any software development effort.

Historically testing has been an activity that has been done primarily during the integration phase of the project, after implementation has been accomplished. As systems have become larger, and more has been identified about the nature of the software development process, the verification and validation (V&V) activity has been enlarged and integrated with more of the components of the development cycle. Figure B-3 presents a comparison of the relative cost for correcting errors in the development of software versus the development phase when these software errors were detected. Because so much cost can be saved by the integration of testing activities through out the life cycle, V&V is being used more widely as can be seen by the fact that -

Validation and verification activities are being stressed during requirements and design phases. Requirements are being analyzed for completeness and consistency. Traceability of designs back to requirements is being done. Automated tools are being developed to accomplish this.

Software is being developed with testability in mind. If the implementor or designer is not going to be testing the software himself, at least he makes a conscious effort to make the efficient testing of the product one of the objectives of his design or implementation.

Modules are tested as they are developed and as they are built up. Testing is not placed after all development but is done as the hierarchy of modules is developed.

Automated testing tools are being used to relieve the human of the boring, error prone, repetitious tasks that are involved in testing.

The future trends in validation and verification work appear to be concentrated in three areas -

More extensive use will be made of computerized tools to achieve more complete system testing. This means more tools to analyze requirements and design (pre-implementation) and also more tools to evaluate the developed products (post-implementation).

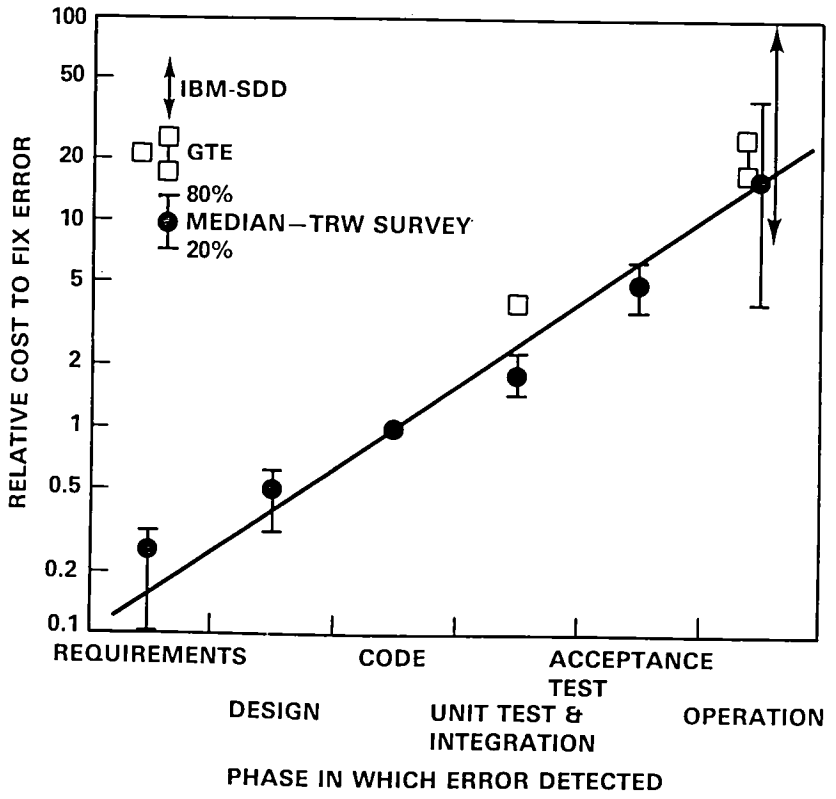

Figure B-3. Relative cost of correction of software error.*

Testing activities during the early phases of the life cycle of the development process will be enlarged.

New theoretical concepts will be employed which will tend to prove the correctness of the software. Presently proof of correctness techniques work only on small programs. If these techniques can be improved and the modularity of programs can be enforced then techniques may be developed which will allow proving the correctness of programs by proving assertions about individual modules and the ways the modules interact with each other.

Computing or Programming Environments

Programming environments are the methodologies, tools, and techniques which can be used by a developer of a software system in the performance of his task. The name environment is intended to imply some global or all encompassing nature to this collection of objects. The goal is to have everything the individual needs to do his task of software development. Pieces of these environments exist at present. The goal of the work going on at present is to system engineer these separate parts into a useful, smoothly functioning whole or "environment". There are well known examples of programming environments - several environments exist for UNIX or UNIX-like systems including the programmers workbench, and also the Interlisp Programming Environment.

There is no such thing as "the programming environment". Different classes of users have different needs in their environment and thus different systems need to be developed. To provide an environment means, a fortiori, to understand the functions that the environment must support. This is not an easy task for it means that one must have a good picture of the application area in which the environment is to be used. This, in general, is only poorly understood, and therefore the taking of the second level step, namely the building of a support environment, must be slow. One can give examples of areas of software development where a programming environment would be useful -

*Reprinted from Boehm, Barry W., "Software Engineering: R&D Trends and Defense Needs," *Research Directions in Software Technology*, P. Wegner (ed.), Copyright © 1979 by The Massachusetts Institute of Technology.

Software Maintenance Group

Large scale software production group

Software Management Group

Quality assurance or validation and verification group

While there is no such thing as "the programming environment", there are characteristics of good programming environments whatever the application. These are

The environment must support the entire range of activities that a group is charged with supporting.

The system must be user friendly.

The system must have a set of components that is flexible enough to adapt or to be configured to meet many different specific needs.

There must be a smooth integration of the tools so that it is easy for the user to combine the tools and not be hindered by interface problems between the tools.

There must be a central repository or data base of information available to all the tools and the users who must share the environment.

The direction of research in the area of programming environments will probably be a bottom up, integrating approach. Existing systems will be studied to identify the key features of these systems which make them successful. Collections of tool fragments will be assembled into small environments for testing. As these systems become better understood more integration will be possible. As has been indicated earlier, success in this effort will go hand in hand with an understanding of the systems which the environment will surround.

PROGRAMMING LANGUAGES AND TECHNIQUES

In the past there has been the tendency to devise languages that would serve a broad community of users at some level of support. Everyone would be able to solve his problem using computer languages such as FORTRAN or COBOL. The difficulty of solving the problem may have been great because of the generality of the language used. Modern programming languages allow applications programmers to define data structures which are tailored to the application the individual is working on.

There is tendency now to associate the logic of the programming languages with the particular applications they are intended to support rather than with a particular hardware. A productivity gain of a factor of 10 has been reported in some cases where a programming language was specialized to a particular class of application.

By making the system fit the user environment more closely it becomes easier for the end user to be the "programmer". For office workers the Xerox STAR system is easy to use. Much of the work is presented in pictorial form. Smalltalk is an example of an object oriented language which is easy for the nonprogrammer to pick up, understand, and use.

Applications programming can be made easier by the use of artificial intelligence techniques in the programming effort. Intelligent systems can interact with the programmer as he builds up his program. He can tell the system in some sense how he wants the program to work and the AI system figures out how to make the program work.

Programming languages developed from the style laid down by von Neumann and Turing. The languages were of a sequential form, performing one computation after the other. There was very little structure imposed upon the sequence of instructions that a programmer could write. Very little checking was done at run time to make sure the system was only doing "legal" or "proper" operations. Today's thrust in programming language development differs from this generation of languages in two ways. There is more effort to add structure and correctness to programming languages and there is the addition of parallel sets of operations to the class of programming constructs.

Pascal and Ada are examples of languages which allow a great deal of structuring. The structure of data can be specified. The set of admissible values that variables may assume can be specified and checked at run time. The structure of the program itself may be controlled. The program may be broken into modules which are small enough to be understood easily by one human being. The program can be constructed in some hierarchical fashion.

There are other languages being created which are designed to exploit parallelism in computing or else not be so wedded to a traditional computer architecture. Data flow languages, applicative or functional languages, and concurrent languages are examples of languages which are not necessarily sequential in nature.

An applicative language uses combining forms for creating programs. Combining forms can use high-level programs to build still higher level ones in a way not possible in conventional languages. Functional programs deal with structured data and usually are not repetitive or recursive. They are hierarchically constructed, do not name their arguments and do not require the complex machinery of procedure declarations to become generally applicable. There are indications that applicative programming concepts will lead to new kinds of computing architectures that will be able to fully utilize large scale integrated circuit technology. Such language based computer design can ensure the programmability of a radical architecture, since in a language-based design, the computer is a hardware interpreter for a specific base language.

In concurrent languages and data flow languages there is the need for coordination and control of the different processes which can be going on in parallel if one is to exploit parallelism to the maximum. If one can structure his problem in terms of concurrent processes then he can apply the technology of parallel processing immediately. In many cases the program has been written in a sequential language and a compiler or analyzer must break the system up into concurrent processes. In this latter case it may not be possible to exploit parallelism as effectively as it could be if the program had parallelism built in during the design and coding.

OPERATING SYSTEMS

Research in traditional operating systems has reached a mature level now. Work on distributed operating systems is still in the research stage. During the 1960's and 1970's a great deal of research was focused on developing the mechanisms which should be a part of operating systems and also on determining what was the proper set of roles and functions for an operating system. Figure B-4 presents a graph depicting the evolution of operating systems. Operating system research can be broken down into five areas of achievement.

The first area is that of process management. Processes are a generalization and abstraction of the notion of a computer program. They are units of a task which can run on a computer. Computer scientists noted that as computers became more powerful that not all of the resources of the computer hardware were being utilized as well as they might be. The CPU had to wait in an idle state while an I/O transfer was accomplished, for example. This led to the idea of parallelism and concurrent processes existing within a computer. The general question was how to manage concurrent processes safely and expeditiously. The problems that have been addressed include the controlled sharing of common memory and files, the effective sharing of limited resources among competing processes (e.g., avoiding the deadlock problem), and managing communications and coordination among asynchronous processes.

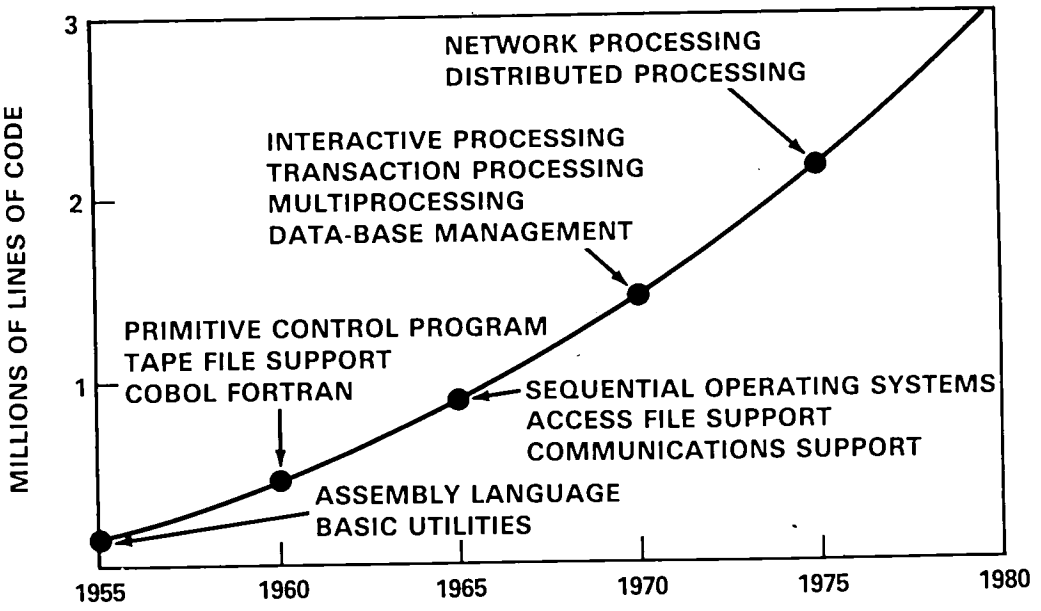

Figure B-4. Evolution of operating systems.*

Research in storage management is the second area of achievement. Storage management research has been dealing with how to make a large amount of storage appear to be available to each user and to insulate each user's memory from any other independent user's memory space. The technique of virtual memory was developed to give the user the ability to use a seemingly large address space even though the actual storage capacity of the machine would be considerably smaller. More efficient file system structures have been developed to allow users to organize their files in structures which suit their needs and which are easy to manipulate. As more users made use of computing resources it became necessary to ensure that each user did not interfere with the memory space of another user, either by accident or by design.

A new area of research that is growing in importance is information protection and security. The operating system must control the access rights of each user to the appropriate sets of files and data. Users may be able to access files for certain purposes but not view them. Users may be able to read files but not write to them. Users who have been granted access rights may be able to assign access rights. All this makes the problem of access control very complicated. Another issue of security is known as flow control. This is a very difficult problem to solve completely. It involves determining information by ob-

*Reprinted from Champine, G.A., "Perspectives on Business Data Processing," *Computer*, November 1980, Copyright © 1980 IEEE.

serving the flow of data in the system. Two examples may serve to illustrate the problem. If a user has access to the salary data for groups of employees but not for individuals how can the system assure that, by asking a series of legitimate questions, the user does not gain information about individual salaries. If a user is not allowed access to files containing "top-secret" information, and he now finds his access restricted to a file that had previously been available to him, then the flow of data has given him the knowledge that this file now contains top-secret information.

The fourth area is work in resource allocation. It has been directed at developing mathematically based techniques which allow assessment and allocation of computing resources among a set of users. This has led to maximizing resource utilization, improving throughput, and serving many customers fairly. The work in this field has drawn heavily from the disciplines of operations research.

The fifth area of research involves the generation of the operating system itself. An operating system is a very large piece of software itself. Techniques which have been used in building large software systems have been used in building operating systems also. Special programming and development techniques have been applied to building operating systems which are maintainable, reliable, and verifiable. The builders have tried to isolate functions into modules of the operating system which have minimal interaction with the other modules of the system. In this way the programmers using the operating system have to be familiar with only those aspects of the operating system that they need to use.

Research directions in operating system technology seem to be directed towards improving and simplifying existing operating systems or developing operating systems which function in a distributed or network environment. Operating systems such as UNIX and CP/M are being widely used because they can run on a number of different hardware systems. The interface between the user and the operating system is being made more user friendly. Operating systems are being developed which are fault tolerant, reliable, correct, and more secure. Operating system functions which have previously been implemented in software are now being implemented in special purpose hardware. This will lead to improvements in performance and reliability. Distributed operating systems will become necessary as networks of computers, either tightly or loosely coupled, become more prevalent.

An excellent summary of the state of the art in software technology and research directions was developed under the auspices of the Department of Defense and published in 1979 by MIT Press [Wegner].

DATA

There are many differences between issues related to scientific data and non-scientific data (which includes mainly commercial data). Much more attention has been focused on commercial data because there is a much larger market for techniques involving this class of data. NASA primarily has to deal with scientific data and thus cannot count on the academic and industrial communities to address the problems that it faces with scientific data.

Data that NASA deals with often is generated onboard a vehicle which is in space, either within the atmosphere of the earth, circling the earth, or in some other part of the solar system. In many cases this data is communicated back to the ground through an RF link. In this situation the signal carrying the data is subject to interference and in the process noise is added to the data component of the signal. In the past a great deal of research has been devoted to developing coding schemes which could be used to reconstruct the original data stream that was sent by the spacecraft.

The results of this coding research have been very successful. Two coding techniques, convolutional coding with Viterbi decoding, and Reed Solomon coding are very powerful techniques. When these two encoding techniques are used in series onboard the spacecraft and the reverse series of decoding processes

are applied on the ground very good recovery of the original data stream is obtained. In fact, this scheme is so successful that it is not felt there is a need to develop better coding schemes for data transmission from space in the foreseeable future. Efforts are underway within NASA to adopt these techniques as standards to be used in all space-to-ground transmission.

The work in coding theory that needs research is how to implement these coding techniques using VLSI technology. This could allow more data to be sent through the system more rapidly. VLSI technology could also be used to construct building blocks for decoding or encoding and then these could be connected with some degree of parallelism to construct efficient coding schemes for a variety of circumstances.

Once the data has been returned to ground, or if data originated on earth, it must be manipulated and stored on the computer. Again, techniques have been developed for storing information which is related to commercial operations - financial, managerial, and administrative data. Storage techniques for geolocated data and for image data have not been developed as well. The access methods for retrieving the data are also very primitive. Most of the research money has been devoted to techniques to acquire and bring the data data back to earth, and the data has simply been stored in some primitive order, usually the order in which it was acquired by the spacecraft, on some primitive storage medium such as magnetic tape. Optical disk storage devices are being developed now which work well with permanently archived data, which is typical of NASA's data. NASA should develop data structures which will be useful in the storing of this data on this new medium.

Data may at times have to be sent in a secure fashion over a network. It may also have to be stored in a computer in an encrypted form. This may arise as facilities and capabilities for data sharing are expanded. Only certain sets of users may be allowed access to the data (e.g., principal investigator teams during the first few months after the acquisition of the data). There are two encrypting methodologies which are receiving wide attention nowadays and are the techniques that would probably be used in encrypting data.

In January 1977 the U.S. Government adopted an encrypting technique, known as the Data Encryption Standard (DES), developed by IBM as its official standard for unclassified information. The encryption technique transforms a 64-bit block of data into another 64-bit block of data using a 56-bit key. Because the government has adopted this technique as a standard many companies are now implementing this technique in hardware so it may soon be in wide spread use at a reasonable cost.

Because the key has 56 (possibly random) bits in it and because of the nature of DES, cracking the code is very hard. There has been some argument concerning the security of the code however among cryptologists. It has been argued that a massive brute force attack on the code using a parallel collection of microprocessors could crack the code. Another problem with DES which is shared by many other coding schemes is that the key which allows people to decode the encrypted message must be shared by both the sender and the receiver. If the community of users is large or is initially unknown then this can be a problem if secrecy is to be maintained. R. Merkle has developed a solution to the key distribution problem which allows users who have not previously exchanged keys to interact over an insecure network and develop a key which they and they alone can use. The disadvantage of this technique is that a very large amount of computing power and transmission bandwidth must be used before a key has been established.

The other class of techniques which can be used for sharing data securely are collectively known as public key cryptosystems. They are very attractive in that keys for the use of these systems need never be distributed, and a large class of users who have never communicated before can instantly communicate securely with each other. To a message one can attach "digital signatures", the electronic equivalent of handwritten signatures, which can prove that the message attached to the signature came from the individual who "signed" it.

Public key cryptosystems were announced publicly by Diffie and Hellman in 1976 and have been implemented via several different techniques. The most famous one was developed at MIT and was based on the difficultly of factoring very large (200 decimal digits) numbers. In each implementation there is an encrypting technique and a corresponding decrypting technique. The encrypting technique is made widely known to all potential users. The key that enables the encrypting technique is also distributed publicly. The corresponding decryption technique may be known also but only the individual who should receive the message knows the key for this decryption. Knowing the encrypting key does not in any way help one to know the decrypting key. This novel and significant idea is only useful if there are encrypting techniques which have these properties. As has been indicated above there are such techniques. Organizations are working on ways to implement these techniques in hardware presently. The increasing need for exchange of financial and personal data securely over networks has stimulated this implementation effort.

If data is to be shared among users, whether in a secure environment or in a non-secure environment, the meaning of the data must be intelligible to all users if the data is to be useful. If there is not a common meaning or interpretation of the data then at least the meaning of the data should be able to be determined from the information that accompanies the data.

The current intensive activity within the national and international standardization organizations, which is aimed at developing a full repertoire of layered Open Systems Interconnection (OSI) standards that enable computer-to-computer dialog, may be expected to solve the basic problems of intercommunication. However, these standards will be oriented primarily towards solving the problems of commercial applications. NASA, while capitalizing on the commercial developments, will need to develop some standards which are customized to its own unique data communications applications so that automation and interoperability within the mission support elements may be achieved. These supplementary standards will focus in areas that are peculiar to NASA's mission environment, such as the acquisition, storage, cataloging, and distribution of huge quantities of space-derived information. Embryonic work has already begun within NASA in the advanced development of these special-purpose standards for use within the Agency. Examples of needed developments include:

Standard message formatting protocols, which permit many different categories of user application data associated with space missions to be encapsulated within globally-interpretable labelling structures which facilitate their identification and cataloging. It is foreseen that a very large family of globally-related format structures will be needed in order to permit exchange of mission data.

Standard spacecraft/ground data transmission protocols, embracing the exchange of both telemetered measurements and telecommanded control information.

Standard methods of correlating and relating measurement data sets, including protocol elements for timetagging data and describing the geographical coordinates of the remotely-acquired information.

MATHEMATICS AND THEORY OF COMPUTATION

INTRODUCTION

A basic grounding in the foundations of computer science is required within the Agency in order to attract and retain quality personnel and in order to be able to operate within the computer science domain, including those areas which have a direct impact on NASA programs. For example, NASA needs the capability in-house to create new algorithmic approaches to its unique computational mathematics requirements. Specific elements of mathematics and the theory of computation are discussed below.

COMPUTATIONAL MATHEMATICS

Much of NASA's research work involves modelling of physical systems. Systems of ordinary and partial differential equations are frequently employed in the formulation of a mathematical model representing a physical process. When the mathematical model is transformed into a computer model, the differential equation representation is transformed into a difference equation representation. There are several key issues associated with this type of transformation:

Representation. Is the problem to be solved by the difference representation the same as the one in the differential representation, and more importantly, is it the same as the physical process under investigation?

Complexity. What resources (time and space) will be required to solve the problem?

Convergence. Will the system converge, to the correct answer? What is the accuracy, the sensitivity to parameter error?

Representation

NASA problems in computational modelling are frequently much larger (by orders of magnitude) than otherwise similar problems worked on elsewhere. For example, at the present time with fewer than sixty supercomputers (Supercomputer is used here to mean a computer which processes over one hundred million - 10E8 calculations per second.) installed worldwide, NASA already has four such machines. The Numerical Aerodynamic Simulation system is proposed with speeds of approximately one billion (10E9) calculations per second and a memory in excess of 64 million words. This opens up new avenues of research which were previously not feasible due to limited computational resources.

As problem size increases by orders of magnitude, representation effects which were unimportant and unnoticeable may become significant. It is therefore extremely important that we have in-house the ability to recognize these issues and to deal with them appropriately. Some of the specific mathematical disciplines in which we must maintain a state-of-the-art awareness and practice include numerical analysis, matrix theory, functional analysis, and the theory of partial differential equations.

Complexity

Direct solutions to modelling problems may result in excessive demands for computational capability, or storage of data used while obtaining a solution. Traditional algorithms were developed to be computationally attractive for equipment with significantly different characteristics than the machines in use by NASA today. Many algorithms were first described for use on mechanical calculators (or in some cases, i.e. Richardson's PDE method for weather prediction, with human calculators). For example, as a result of the Courant-Friedrich-Loewy convergence criterion, the time step must be reduced proportionately with the space dimensions. This results in a significant increase in computing time as spatial resolution is improved. A two-fold increase in spatial resolution demands a sixteen-fold increase in computing time and an eight-fold increase in storage space.

Rather than simply continue to extend traditional algorithms indefinitely with ever larger and faster computing equipment it is necessary to examine alternative algorithms for solving the physical modelling problem. As an example, ca. 1970, Winograd demonstrated that matrix multiplication required only n exp log(sub 2) 7 multiplications rather than the n exp 3 previously thought to be required. For large

matrix problems this and related algorithmic advances have the potential for saving large amounts of machine time, storage space, and improving turnaround time in a problem solving environment. With matrices of order 1000 the improvement in computational requirements is an order of magnitude, the difference between 10E9 and (approximately) 10E8 operations required to obtain inverse matrices.

The availability of technology to stand in for complexity improvements can not be guaranteed. Flight software frequently becomes significantly more expensive to produce when it is constrained by available memory. Because of weight, power, size, and flight qualification limitations it may not be possible to increase the memory size to alleviate the situation. Improvements in time and space complexity of the algorithms may obviate the need for faster processors or greater capacity for onboard systems.

Convergence

After the mathematics is verified, computational methods decided upon, and programs coded there is frequently a question of convergence. Will the system converge? Will the system oscillate? If the system converges, will it converge to the desired value or is there a nearby spurious solution? In answering these questions we must examine the lower level computational mathematics on which our systems are built: number system, precision, roundoff algorithms, rules of arithmetics, etc.

These issues arise because the discrete mathematics used within computers is not isomorphic to the real numbers employed in the analytical formulations of the problem. When users switch computer systems they frequently find that the results are different than those produced on the previous system. (Not wrong, just different) There is a large psychological barrier in moving from one system to another because of these differences. The situation is immediately seen to be symmetric: If two users, each working on the same problem, switched computers then each user would be uncomfortable with the results produced by the new computer.

For example, while the IBM 360 computers are functionally identical, an exception was made for the 360/95, it utilizes a non-restoring divide while all other 360 models utilize a restoring divide. The effect is visible in the low order quotient bit one-half of the time (when the remainder is greater than one-half of the low order bit) and actually results in a closer (better?) answer. This difference proved to be significant in hindering the interchangeability of programs between models of the 360 series. Despite the slight improvement obtained with the 360/95 its initial use was disconcerting to the scientific community.

THEORY OF COMPUTATION

This section deals with abstract areas of computer science. Because NASA is a technology organization such areas are frequently overwhelmed by the technological components of the Agency. The basic understanding of the computational problems faced by NASA requires an appreciation for underlying theoretical foundations and system level implications of designs. For example, if data is considered one of NASA's most important products, than the Agency must be able to consider the structure and representation of data, to extract information from data, to define systems which operate on the data, and to create the tools needed to do all this. This requires an understanding and application of information theory, coding theory, representation of knowledge and information, hierarchies of complexity which help determine what solutions are viable (or, what problems are not viable), and principles of language theory as it applies to codifying requirements and defining solutions.

INFORMATION MANAGEMENT

INTRODUCTION

The primary technologies that will be addressed in this section relate to database management systems. These systems are the subject of the majority of the research efforts ongoing in information management systems. Current research and development in a number of DBMS areas will significantly increase DBMS capabilities over the next five years. The most significant areas, discussed below, are:

Novel applications

Relational DBMS

Physical storage schemas

Nonprocedural command languages

Data dictionaries

Database machines

Distributed databases

NOVEL APPLICATIONS

The majority of DBMS applications to date are management and administrative in nature. Other application areas will grow significantly in the next five years. These include:

Pictorial data

Geographical data

Graphical data, especially for computer-aided design and computer-aided manufacturing (CAD/CAM)

Textual data, especially for office automation

RELATIONAL DBMS

The reason relational systems are becoming popular is the simplicity of their interface to users. All data is represented as tables instead of the complex pointers and sets used by systems based upon the hierarchical or network (e.g. CODASYL) models. Users express queries in an nonprocedural language, i.e., in terms of what data is desired rather than how to find it. The absence of complex interconnections between tables in the relational model has hampered performance compared to CODASYL systems.

In the next five years, vendors will improve the algorithms needed to improve performance in relational DBMSs based upon practical experience. The simplicity of the relational model makes it the easiest DBMS model to be implemented in firmware or hardware, creating a special purpose "database machine".

PHYSICAL STORAGE SCHEMAS

A current shortcoming of virtually all DBMSs is a physical storage schema, which permits the data base administrator to map logical storage structures that are expressed in the logical data definition language (DDL) onto the physical storage devices available.

NONPROCEDURAL COMMAND LANGUAGES

By 1985 virtually all DBMSs will have a high-level, nonprocedural language by which users specify data manipulation. This type of language specifies only what is wanted, not the procedure for obtaining what is wanted. When the access route needed to retrieve the information is complicated then queries composed in nonprocedural languages are easier for the user to specify, and also more readable. Facilities are being built into DBMSs which will optimize these queries so that they are performed efficiently.

DATA DICTIONARIES

Data dictionaries retain information about the database, such as:

Data definitions

Data sources

Data products

Users

Processes

Transactions

The data dictionary also relates any of this information to each other.

The trend now is for the data dictionary to be integrated into the DBMS, treating the data dictionary as another (privileged) database to be accessed via the DBMS. However, many so-called data dictionaries in current DBMSs only retain data definitions, i.e., the schema. Growing concern for proper management of databases will force vendors to expand the role of data dictionaries in DBMSs of the future to include at least the functions provided by the stand-alone systems. For example DEC's new CODASYL-based DBMS for the VAX permits the data dictionary to reference data sets that have not been loaded into "the database".

DATABASE MACHINES

The most significant improvement in DBMS performance will come from the advent of database machines (DBM). Although many university prototypes exist, few commercial DBMs are on the market currently. The best know commercially available DBM is the Intelligent Database Machine (IDM) 500 built by Britton-Lee of Los Gatos, CA.

The performance improvement from DBMs that are available now is significant. Software AG has demonstrated a 25% average increase in throughtput for an off-loaded Adabas versus a host-resident Adabas running the same jobs.

DISTRIBUTED DATABASES

Many factors are driving vendors toward DBMSs that can manage distributed databases. Networks of cheap and powerful mini/micro processors are cost-effective alternatives to the monolithic mainframe dedicated to DBMS applications. Specialization of these processors into database machines has already begun. A growing number of DBMS applications deal with distributed users. The DBMS vendors as well as the academic community have recognized this trend. Perhaps the best known prototype system is SDD-1, under development by Computer Corporation of America.

Distributed databases pose many complex challenges, such as concurrency control, redundancy control, semantic integrity, and query parsing in a distributed database environment. However, it is clear that these problems are beginning to be solved, and that distributed DBMSs will be a major development in the DBMS market by 1985.

COMPUTING METHODOLOGIES

ALGEBRAIC MANIPULATION

Mathematicians and scientists frequently manipulate symbols in their work. For complex systems, keeping track of the symbolic manipulation can be a tedious process and prone to human error. It is in these cases that the computer can help. The computer can track the various symbols and how they are combined, so the user knows the exact form of an integral, a polynomial, a function, or an infinite series. By having an analytic expression for the solution, computing many values of the solution set is possible by evaluating the solution expression.

Computers normally are programmed to solve a scientific problem using the numerical techniques of approximation and iterative convergence. These answers are only approximate, sensitive to the instability of numerical processes, and each new answer sought requires another use of the algorithm to produce it. In many cases this is quite satisfactory since a closed analytic solution to the problem is not feasible and the approximation is sufficient.

The languages used for doing symbolic manipulation on computers are not the common ones used for numerical evaluation. FORTRAN is a language used for formula translation and is the standard one used for numerical evaluation of results. FORTRAN does not support dynamic allocation of memory space, and also does not support a wide range of data types. These two features are required in systems handling symbolic manipulations. Large intermediate forms can result as symbolic manipulations are performed and not yet simplified or reduced. Data structures are needed to handle polynomial and infinite series and exact arithmetic calculations. MIT has develop a very extensive capability in their MACSYMA system which can handle a wide range of symbolic computations.

The classical algorithms developed for various computations with functions and numbers are not readily adaptable to computers. For example, the Euclidean Algorithm can be used to calculate the greatest common divisor of two polynomials. This technique works well in theory but it can lead to very large numerical coefficients in the intermediate results. Alternative algorithms have been developed which may be harder for humans to track but which computers can handle easily and maintain reasonable growth in the size of intermediate results. Classical polynomial factorization algorithms are inherently exponential in nature. Factorization algorithms using modular arithmetic techniques have recently been discovered which can be used to solve the general cases that must be treated. A fertile area of research for computer scientists involves finding efficient algorithms for symbolic processes.

The successful applications of algebraic and symbolic computation to date arises primarily from problems which are solved by using perturbation theories. This includes the fields of celestial mechanics and quantum electrodynamics. These techniques have been used in computing the orbits of Saturn's moons.

ARTIFICIAL INTELLIGENCE

Artificial Intelligence is a field which in many respects resembles an engineering discipline such as Electrical Engineering. AI is constructive, trying to build useful software and is also dependent on a body of abstract theory. Some of the major sub-disciplines which can be distinguished are Natural Language Understanding, Speech Understanding, Vision and Scene Analysis, Problem-Solving, Theorem-Proving, Robotics, and Game Playing. This discussion will focus on Problem-Solving and will mention the potential for natural language understanding. This should illustrate the potential for the use of Artificial Intelligence within the NASA community.

Great advances have been made in the power of the automated problem solvers in the last few years due to the discovery of the technique known as "knowledge engineering". "Expert systems" have been constructed that plan and schedule, write computer code, analyze Nuclear Magnetic Resonance spectra and deduce molecular structure from the spectra. Others diagnose various diseases such as glaucoma or blood infections, trouble-shoot computers, configure computer systems, model military operations, etc. During the last 15 years, there has been a steady growth in both the power and generality of these systems, so that at last we can begin to be confident that they will be able to solve real-world problems in a quite general way.

Within NASA a planner/scheduler has already been constructed which assembles command sequences for a spacecraft computer. It has demonstrated its ability to generate realistic sequences and even resolve conflicts in the initial requests, automatically. It is the first planner to handle the time parameter. Work is being done currently to increase the power of the system to meet more sophisticated user demands, and to make the system more user friendly. The system is capable of planning for military logistics problems, robot action sequences, and any other domains that have well-defined actions, states, and state changes.

A diagnostic system for monitoring spacecraft telemetry is also being built. The system employs the reasoning mechanisms of the system described in the previous paragraph but uses a different control structure and knowledge base. It is already capable of diagnosing spacecraft failures at a first-level degree of difficulty. In principle, it should be able to go as deep as its knowledge base allows, and research is continuing to allow it to apply general laws such as Ohm's Law in predicting the results to be confirmed against test measurements, ruling for or against evidence on the basis of time, etc. Note that in this area of application, spacecraft monitoring, the human beings employed are the engineering equivalent of paramedics, so that the system will be able to duplicate their performance with a modest knowledge base. The system should become operational in two years.

These systems illustrate the potential usefulness to NASA of Artificial Intelligence. When both of these systems have been perfected, they have the potential of being combined into a larger system that plans actions, executes and monitors the actions, and diagnoses any failures in the plan. This would be a learning system that might plan and direct the actions of sophisticated robots. Such a system would have a multitude of applications since it begins to approach some of the power of biological brains. Such a system may be available in five years.

The usefulness of a Natural Language Understander is obvious as the front-end of a problem-solving system. Much progress has been made in understanding utterances in a restricted domain for which knowl-

edge can be supplied to the computer. Humans routinely make references in their utterances to knowledge they take for granted as being shared and for which there is no explicit mention in the stream of conversation. The only hope for a computer to fathom the meaning of such references is to "know" about them in advance. It is not possible to predict now when such understanders will become practical.

Robotics and teleoperator systems are in use in some manufacturing processes, particularly in environments adverse to human beings. Space operations are a natural extension of this activity, but will require far more sophisticated systems than are currently available. Related capabilities such as vision and sensor systems also must be developed for effective remote operations and repair activities in space.

COMPUTER GRAPHICS

Computer graphics is emerging as a critical discipline in the field of computer science. The need to absorb the tremendous amount of data being generated in many disciplines dictates the requirement to find new methods of data display. Computer graphics is lagging behind the development of inexpensive display devices and personal computers. Work is needed in the area of image synthesis including mathematical formulas and digital databases. Additional developments in mathematical software and database management also will benefit computer graphics.

Like image processing, in order to do high quality computer graphics a tremendous amount of computing power is needed. A large number of pixels must be stored, manipulated, and displayed, and sometimes this processing must take place in real-time. Many algorithms have been devised for working with the synthesis of images from data but implementing these on the computer so that they run effectively has been a major problem. Major improvements in computer graphics hardware and processors are needed.

The output devices for computer graphics are becoming cheaper and more sophisticated. Calligraphic displays are intended for line drawings, while raster scan devices are suitable for displaying solid shaded areas on the screen. Conventional storage CRT's are calligraphic display devices which retain images once they are drawn on the screen. Thus no memory is needed for the image. Vector refresh devices are calligraphic devices which do not present the image permanently on the screen but regenerate it 30 times a second (in order not to produce flicker) from a memory within the display device. This allows for portions of the screen to be dynamically changed and this ability makes the vector refresh CRT a popular choice for high-performance interactive applications. Raster scan devices are like TV screens, producing images by many adjacent horizontal lines. In vector and raster scan devices, the CRT screen is treated as a matrix and each element (called a pixel) of the matrix is brought to a specific gray level or to a specific color and specific intensity. Current state of the art in raster scan technology is a 1024 x 1024 matrix of pixels, and each pixel has 8 bits of data for use in specifying gray level or color. Vector graphics devices have much higher resolution, such as a 4096 x 4096 matrix, for interactive display. The inherent two-dimensionality of the display screen is a handicap of all CRT's. True three-dimensional display can be provided by stereo viewing or the use of a vibrating curved mirror whose changing focal length gives apparent depth to the reflection of an image on a CRT.

The low cost of semi-conductor memory makes it practical to build special purpose memories, called frame buffers, for the specific purpose of refreshing raster scan CRT's. Frame buffers provide an example of the impact that technological improvements can have on work in computer graphics. Presently for a frame buffer to support a 512 x 512 screen with 8-bit memory per pixel requires 128 16K memory chips. Japanese manufacturers are on the verge of producing 256K memory chips. The same frame buffer could be built with eight of these chips and a frame buffer for a screen with 1024 x 1024 pixels (with 24 bits per pixel) would only require 96 chips. Thus an improvement in computer memories can have a very significant impact on the quality and range of images one can create.

There is a renewed interest in the already developed basic algorithms used in computer graphics. These algorithms include techniques for scene transformation and clipping, line and curve drawing, two-dimensional polygon filling and similar operations. Until now these algorithms had been implemented primarily in software either in the main computer or an intelligent work station. Now there is the prospect of implementing these algorithms using VLSI technology. The concept of parallelism can be exploited to speed up scene generation by using these special purpose VLSI chips in some parallel architecture.

Research work is needed and is being conducted in the design of a graphical interaction language. This is a hard problem because it is difficult to design a language that is suitable for all applications and because it is hard to describe how a human relates with graphics software. There is also the critical need for the standardization of graphics software. Applications programs need to be able to issue standard calls to any graphics package and have the same action result. Efforts are now underway to define a graphics software standard.

Another area of research interest is the generation of realistic pictures of three-dimensional scenes, and in some cases, in real-time. There are the two problems: how to generate the images and how to do it fast enough so that it simulates real motion. In order to portray scenes realistically one needs hidden surface algorithms to allow one object to appear to be in front of another. Objects must also be given shading, texture, reflectivity, and gradations of color. The surface of the object must be described using mathematical expressions, and recent efforts have yielded algorithms for directly presenting complicated surfaces without approximating them by polygons. Two major research problems yet to be solved are the generation of images with fuzziness (e.g. trees, smoke, clouds) and the generation of images involving objects with a large number of interconnected parts. The number of computational cycles required for each pixel to take into account all these image qualities is very large. For example, one fairly sophisticated image required more than 1 hour of computer time on a VAX-like machine. Very sophisticated images of spacecraft encounters with planets have been produced but each has required a large amount of computer resources.

Real-time scene generation requires a large amount of computing power achieved in a very specialized and parallel architecture. For example, the Evans and Sutherland CT-5 terminal is used in many training simulators to create the image the trainee should see as he manipulates his controls. The image currently produced is not realistic in all its features, although it is realistic in position, view angles, and shadows.

Another important application of computer graphics is in the generation of images from non-imaging data. People are able to comprehend the meaning of a complex collection of data more rapidly and easily in some cases if it is presented to them visually. Changes in screen patterns may be the most effective way to indicate exceptional conditions for personnel monitoring data streams. Scientists doing simulations of complex processes which produce large amounts of data may be able to develop understanding of the processes involved if they can compact and display their data in color images on the screen. Recently, phenomena in the fields of heat transfer and fluid dynamics have been studied by using dynamic graphical techniques.

Computer aided design (CAD) relies upon computer graphics. The design of electronic circuit layouts and of geometric models of 3-dimensional objects are important CAD activities. Geometric design involves primarily line drawings but in the future, as more computational power is available, realistic three-dimensional visualizations of the objects will be possible.

IMAGE PROCESSING

NASA has motivated or has done much of the work in image processing to date. A great deal has been done in the development of algorithms for manipulating images. The algorithms handle image enhancement techniques such as filtering, smoothing, stretching, geometric representation techniques, and classification techniques and are fairly well understood.

These algorithms have been implemented on large mainframe computers, which operate usually in a batch mode with a serial stream of instructions. They have been implemented in a standard programming language but with a complicated user interface. Experts on the image processing system have usually had to work side by side with the science user. During the next decade, user friendly systems will be developed.

The capability of image processing is limited today by hardware and the interface between the user and the system. The algorithms have already been developed. It is now possible to place a large amount of computing power into individual work stations. VLSI technology is developing and new hardware architectures which utilize custom VLSI chips to perform specialized functions are being studied and developed.

Large primary memories are needed in order to speed up the manipulation of images. If an image is broken into pieces and stored on secondary memory then the image must be brought into main memory a piece at a time and the system becomes I/O limited very rapidly. Large directly addressable memory space will greatly improve the image processing time.

The processing time for image enhancements can be sped up by implementing algorithms in special purpose VLSI chips. For example, a common requirement is to reproject and register images taken from an oblique satellite view to a normal projection on the surface. This reprojection along with the need to correct for any systematic camera distortions requires that images be stretched like a "rubber sheet" to fit the desired reprojection. This shift which entails many rotations and magnifications within each image requires relocation of interpolated data to locations which may be far distant from some original position. For a typical LANDSAT Thematic Mapper image 50 to 70 operations are performed per pixel. If each edge of one such image has 6,000 components then this leads to 36,000,000 pixels. It should be obvious from this elementary set of calculations that significant improvements in performance are necessary before significant calculations can be done on images. VLSI operation can reduce the cost and time of processing by factors of 50 to 200 times.

Technology is needed to translate image processing algorithms into VLSI circuitry. Also required is the capability of simulating the performance of the algorithm in the architecture that has been used on the chip. By incorporating more image processing capability onboard a spacecraft the transmission bandwidth can be reduced, and the spacecraft itself can achieve more autonomy and intelligence.

Work must be done to improve the man-machine interface between the system and the user. Since hardware is becoming cheaper it is quite feasible for the end user to have direct access to his own image processing system. It is not practical to attempt to replace human thought processes of visual analysis by implementing them completely in computer hardware. Rather, the user interface should be very friendly. The system should help the user decide which algorithms he should use. Software portability among different systems is also needed.

PATTERN RECOGNITION

Pattern recognition that is of primary interest to NASA is optical pattern recognition. This methodology deals with the analysis of images. The five components to pattern recognition are image sensing, image segmentation, feature extraction, classification, and statistical analysis.

After an image has been sensed and converted to a matrix of pixel values, image segmentation can be accomplished so that one can identify the objects one is looking for in the image. This is the area which places the highest demand on computational resources. Feature extraction is the process by which an image is analyzed so that the features (color, shape, texture, etc.) can be quantified. Then the pattern which has been described in quantifiable terms can be compared against classes of objects to which it may belong. Once classification is accomplished, a statistical analysis of the patterns found can be made.

Most current pattern research is in the discovery of fundamental algorithms for use in identifying patterns. The primary emphasis is not in using very fast computers to implement algorithms already well understood.

Research in the area of pattern recognition has best been accomplished by attempting to solve one specific pattern recognition task, for example, to identify white blood cells, army tanks, or clouds. Often the techniques discovered will generalize to other types of image analysis. However it is very hard to attempt to discover "a new class of pattern recognition algorithms". Likewise there are not any general purpose pattern recognition machines. Most of the existing pattern recognition systems were built for very specific purposes, such as the machine built to perform blood cell differential counts. More examples will surely follow in the 1980's, especially in the medical and military areas. Microprocessor technology has made it feasible to build such special purpose machines. This is perhaps the key area where computer science research can contribute, namely in the development of special purpose hardware/software systems for specific pattern recognition tasks.

SIMULATION AND MODELING

Although computer modeling and simulation has none of the limitations of wind tunnel and flight testing, there are the limitations of computational speed and storage capacity. These limitations have been much more restrictive in the past, but now the computational power and storage capacity of state-of-the-art computers has reached the point where numerical modeling and simulation is emerging as an important aerospace design tool. While it is not yet possible to rely solely on computation to design a new aerospace vehicle, there are numerous examples of experimentally verified improvements to design that have evolved from the application of computational methods.

In the field of computational aerodynamics for example, the current state-of-the-art models are capable of solving the complete time-dependent Navier-Stokes equations of viscous fluid flow. In essence, all of the turbulent eddies of significant size could be computed for a sufficiently long time period to yield both the time-averaged characteristics of the flow as well as its unsteady components. Computers and numerical methods are now fast enough to permit the equations to be routinely solved at least for two-dimensional flows, but much faster machines with considerably larger memories are required to solve the three-dimensional equations in a short enough time to be practical for routine use in aircraft design. (The effective speed of general purpose computers has been projected to level off at 1 billion floating-point operations per second by the year 2000.)

Even though general purpose computers are not likely to satisfy the computational aerodynamics requirement in the foreseeable future, it appears to be technically feasible to construct special-purpose processors having the necessary capability. Micro-miniaturization is proceeding at a very rapid rate. This means that enhanced computer capability can be obtained by matching machine architecture to the prob-

A major question is how does one utilize a raw computer resource capable of 1 billion floating point operations per second, to solve problems whose solutions are dependent upon such a resource being used efficiently. The answer is that we must understand how these new supercomputers get their potential speedup, and use them accordingly. Since the gigaflop machine necessarily will have multiple parallel and/or segmented functional units to obtain such a speed, system programmers, software writers and users will be forced to deal with the non-trivial task of writing operating systems, compilers and application programs that utilize such a capability efficiently.

There are two aspects of a specialized computer which are independent of the particular architecture; these are reliability and programmability. A high performance computer using current technology will consist of many components. As the number of components approaches the mean time between failure of one component, the frequency of component failure increases to the point where individual users are aware of system failures. To prevent this requires a system design whereby the system can continue functioning correctly in the presence of failed components. For memory components, this implies error detection and error correction. For processing components, this means error detection capability in some form: residue arithmetic, selective monitoring and emulation, or duplicate arithmetic units.

As for the aspect of programmability, since the processor is specialized for a reason, the programmer will have to be cognizant of the nature of the specialization and will probably be required to deal with this specialization in the syntax of the programming language; the alternative is to defeat the purpose of the specialization. On the other hand, too arcane a programming facility runs the risk of being unmanageable by a programmer, again defeating the purpose of specialization, or even the purpose of the facility's existence to begin with.

Another aspect of simulation and modeling is the performance analysis of computational facilities themselves. In particular when NASA develops special-purpose systems, for flight or ground operation, the simulation and analysis of the performance of each system is an important step in the design of the system. Tools for system performance analysis exist for some types of systems; however, the high complexity of typical NASA systems tax these tools and additional research and development is needed in this area.

TEXT PROCESSING

There are currently many stand-alone text editors and word processing systems on the market today. Many of them run on microcomputers and are available in configurations that fit easily into the user's office environment. Each system has a somewhat standard set of capabilities (inserting and deleting text, moving text within a document, etc.), but some word processing systems do offer many sophisticated and useful functions not found on other models. These advanced functions can include the ability to check the document for spelling errors, to generate graphics material for inclusion in the text, to automatically generate tables of contents and indexes, and to provide special formatting for documents with many sections, subsections, etc.

The type of hardware systems that these word processors are implemented on varies widely. The sophistication and ease of use of the man-machine interface on these systems also varies considerably. On some systems there are no special function keys to implement the commonly used functions of the word processor. One must remember the special control characters which implement the functions. Some systems that use the standard keys for implementing special functions place multiple labels on the keys, often in different colors and at different places on the keys. The most helpful configuration is found on those systems which have a set of function keys which can be used exclusively for word processing functions.

A problem of some magnitude that exists today is that the files generated by different word processing systems do not conform to a standard format and thus are not readily sharable among different systems. Each system has a different convention for indicating indentation, paragraph endings, right and left justi-

fication rules, formatting conventions, etc. When a file produced on one system is sent to another system, it is usually sent without any of the special information produced by the word processing system. Special programs and machines have been built specifically to do this translation among systems, but each of these works only between two specific systems. Efforts are underway now to define some common conventions for word processing systems so that at least each system can translate its own format into a common format when files must be shared among several systems. However, this standards setting effort will take years to complete.

There are some word processing systems which allow more than one user to be on the system simultaneously. The hardware for these systems can consist of microprocessors, minicomputers, and even large mainframes. In this situation users have access to a common set of files and thus access to other people's documents is facilitated. The system may also have a very large file capability, implemented via hard disk drives as opposed to floppy disk drives found on the stand-alone systems. In addition a common printer may be available, to which files can be spooled or queued for printing.

The most advanced text processing systems, implemented on large mainframes, offer many more features than are found on the stand-alone systems. The key characteristic of these systems is that they have integrated word processing with the other systems on the computer. Users can access the system from anywhere in the country via a public packet switching network. Electronic mail is implemented among all users. A group of users can work on the same document simultaneously, watching as one edits and revises the document and then passes on control. These systems support sophisticated filing systems, which model the filing systems found in the office. Documents which are filed can be retrieved using key words or other useful query techniques.

Document production is another evolving feature of word processing systems. The simplest document production capability is the ability to print, using a dot matrix or typewriter quality printer. A more sophisticated capability allows for the generation of some limited set of type fonts and some graphics production. The most sophisticated systems allow users to define their own fonts and the products are produced on laser printers. These very sophisticated document production systems are only cost-effective when they are used by a large community of users each having a computer link to the system. This can be achieved when the data entry facility can be used both as a stand-alone system or as a programmable intelligent terminal attached to a main frame computer in which the word processing function resides.

APPENDIX C. NASA PROGRAMMATIC AREAS

The NASA program can be considered to be in three general areas: Aeronautics Research and Technology, Space Research and Technology and Support to the Institution itself. The following taxonomy was created to help define the NASA activities which may benefit from research in Computer Science.

A. AERONAUTICS R & T

A.1 Design and Analysis Methods

- A.1.a Vehicle Design, Technology & Systems Studies
- A.1.b Electronic Devices & Components
- A.1.c Materials, Structures & Dynamics
- A.1.d Airframe/Propulsion Integration
- A.1.e Flight Stability & Control

A.2 Simulator Systems

- A.2.a Flight Scene Generation
- A.2.b Solution of Flight Equations
- A.2.c Crew Controls & Display
- A.2.d Simulator Control and Run-time Environment

A.3 Experimental Data Handling

- A.3.a Data Acquisition and Correction/Calibration
  Wind tunnel testing, flight testing, instrumentation research
- A.3.b Data Analysis
- A.3.c Data Archiving
- A.3.d Data Dissemination/Delivery

A.4 Flight Crucial Systems

- A.4.a Flight Guidance and Control
- A.4.b Engine Control
- A.4.c Telemetry
- A.4.d Diagnostics

A.5 Operations

- A.5.a Ground-based
- A.5.b Airborne

A.6 Computational Modeling of Physical Processes

- A.6.a Aerodynamics and Fluid Mechanics
- A.6.b Sound Generation and Propagation
- A.6.c Computational Chemistry
- A.6.d Aeroelasticity
- A.6.e Aerothermodynamics

A.6.f Structures and Materials

B. SPACE R & T

B.1 Design and Analysis Methods

- B.1.a Integrated Vehicle Analyses
- B.1.b Advanced Mission Analyses
- B.1.c Electronic Devices and Components
- B.1.d Analysis for Large Space Structures
- B.1.e Testing Methods

B.2 Simulator Systems

- B.2.a Teleoperators
- B.2.b Spacecraft Simulators
  Includes Shuttle mission
- B.2.c Data Systems Simulators
- B.2.d Space Environment Simulators
- B.2.e Crew Controls and Displays

B.3 Data Handling

- B.3.a Data Acquisition and Sensors
- B.3.b Data Analysis
  Onboard and ground-based
- B.3.c Data Archiving
- B.3.d Data Delivery
  Quick-look and broad dissemination

B.4 Flight Crucial Systems

- B.4.a Control and Guidance
- B.4.b Navigation
- B.4.c Engineering Information
- B.4.d Payload Command Sequences

B.5 Operations

- B.5.a Launch Control
- B.5.b Science Operations
- B.5.c Spacecraft Management
- B.5.d Tracking and Network Control
- B.5.e Spacecraft Traffic Control

B.6 Computational Modeling of Physical Processes

- B.6.a Planetary & Geophysical Models
  Atmospheric, ocean, interior and surface
- B.6.b Astrophysical Models
- B.6.c Astronomy Models
  Planetary, planetary dynamics, cosmology
- B.6.d Structural Dynamics Models

C. INSTITUTIONAL SUPPORT

C.1 Management Applications

C.1.a Administrative Systems
C.1.b Decision Support Systems
C.1.c Office Automation
C.1.d Data Administration

C.2 Engineering Applications

C.2.a CAD/CAM/CAT
C.2.b Computer-Aided Instruction
C.2.c Scientific Computing
Modeling, optimization, mathematical computation
C.2.d Scientific Literature Access

APPENDIX D. COMPUTER SCIENCE TAXONOMY

This classification of the subdisciplines in Computer Science was defined by the NASA Intercenter Planning Committee and is based upon the Computing Reviews classification scheme published in the January 1982 Communications of the ACM.

1. Hardware

1.1 Hardware Structures

1.1.1 Control structures and Microprogramming
1.1.2 Arithmetic and Logic structures
1.1.3 Memory structures
cache, buffer, associative, virtual

1.2 Data Communications and I/O

1.2.1 Data communication devices
1.2.2 I/O devices
1.2.3 Interconnections
Fiber optics, physical structures, topology

1.3 Data Storage
1.4 Logic Design
1.5 Integrated Circuits
1.6 Computer Engineering

1.6.1 Design
1.6.2 Packaging
1.6.3 Power/cooling requirements
1.6.4 Performance analysis
1.6.5 Reliability, testing and fault tolerance

2. Computer Systems Organization

2.1 Computer Architecture

2.1.1 Processor Architectures

2.1.1.1 Single data stream
SISD, MISD
2.1.1.2 Multiple data stream (multiprocessors)
MIMD, SIMD, array
2.1.1.3 Other styles
Data-flow, stack-oriented, adaptable arch., analog, hybrid

2.1.2 Process control systems
2.1.3 Fault-tolerant avionics systems
2.1.4 Concurrent processor systems

2.2 Computer-Communications Networks

2.2.1 Network architecture and design
2.2.2 Network protocols
2.2.3 Network operations
2.2.4 Distributed systems
2.2.5 Local networks
Buses, Rings, Access schemes

2.3 Systems Analysis and Engineering

2.3.1 Design studies
2.3.2 Modeling techniques
2.3.3 Performance assessment
2.3.4 Reliability, availability, serviceability

3. Software

3.1 Programming Techniques

3.1.1 Functional (applicative) programming
3.1.2 Automatic programming
3.1.3 Concurrent programming
3.1.4 Sequential programming

3.2 Software Engineering

3.2.1 Requirements and Specifications
3.2.2 Tools and techniques
Structured prog, flow charts, programmer workbench, software libraries
3.2.3 Coding
Program editors, pretty printers, standards
3.2.4 Program testing and verification
3.2.5 Program environments
3.2.6 Distribution and Maintenance
3.2.7 Metrics
3.2.8 Management
Life cycle, cost estimation, programming teams, software configuration management, software quality assurance

3.3 Programming Languages

3.3.1 Formal Definitions
Semantics, syntax
3.3.2 Language Classifications
Applicative, data-flow, extensible, macro and assembly, parallel and pipeline, nonprocedural, very high level, real time
3.3.3 Language Constructs
Data types, control structures, concurrent program structures, procedures and coroutines, modules, packages
3.3.4 Processors
Code generation, compilers, assemblers, cross-compilers, interpreters, parsing, optimization, programming and run-time environments, translator writing systems and compiler generators

3.4 Operating Systems

- 3.4.1 Process management
  concurrency, deadlocks, scheduling, synchronization
- 3.4.2 Storage Management
  allocation/deallocation strategies, secondary storage, virtual memory
- 3.4.3 File Systems Management
  access methods, directory structures, file organization, maintenance
- 3.4.4 Communications Management
- 3.4.5 Reliability
  Backup procedures, checkpoint-restart, fault tolerance, verification
- 3.4.6 Security and Protection
  Access control, authentication, security kernels, error handling
- 3.4.7 Organization and Design
  Distributed, interactive, hierarchical, real-time systems
- 3.4.8 Performance
- 3.4.9 Systems Programs and Utilities
  Command language, linkers, loaders

4. Data

4.1 Structures and Representations

- 4.1.1 Logical structures
  Arrays, graphs, lists, trees
- 4.1.2 Physical storage representation
  Contiguous, primitive, hash-table, linked

4.2 Coding and Information Theory

- 4.2.1 Data compaction
- 4.2.2 Encoding schemes
- 4.2.3 Data Encryption

5. Mathematics and Theory of Computing

5.1 Numerical Analysis

- 5.1.1 Numerical Linear Algebra
  Matrix operations, linear equations, eigenvalues and eigenvectors, large sparse matrices
- 5.1.2 Quadrature and Numerical Differentiation
- 5.1.3 Roots of nonlinear equations
- 5.1.4 Optimization
  Constrained, linear programming, nonlinear programming, integer programming, least squares
- 5.1.5 Ordinary Differential Equations
  Single step, multistep, stiff
- 5.1.6 Partial Differential Equations
  Parabolic, hyperbolic, elliptic, boundary conditions, grid generation

5.1.7 Integral equations
5.1.8 Approximation
5.1.9 Interpolation

5.2 Discrete Mathematics

5.2.1 Computer arithmetic
5.2.2 Graph theory
5.2.3 Combinatorics

5.3 Statistics and Probability

5.3.1 Regression analysis
5.3.2 Estimation techniques
5.3.3 Time series analysis
5.3.4 Random number generation
5.3.5 Approximation of probability distributions
5.3.6 Multivariate analysis

5.4 Mathematical Software

5.4.1 Algorithm analysis
5.4.2 Reliability and robustness

5.5 Theory of Computation

5.4.1 Analysis of Algorithms and Problem Complexity
Numerical algorithms, nonnumerical algorithms, complexity measure tradeoffs, apparently and inherently hard problems
5.4.2 Logic and Meaning of Programs
Computability, recursive function theory, specification techniques, semantics

6. Information Management

6.1 Models and Principles

6.1.1 Systems and Information theory
6.1.2 User/machine systems

6.2 Database Management

6.2.1 Logical design
6.2.2 Physical design
6.2.3 Languages
DDL, DML, query languages, report writers
6.2.4 Systems
Distributed, query processing, transaction processing
6.2.5 Database Administration
6.2.6 Database Machines

6.3 Information Storage and Retrieval

6.3.1 Content analysis and indexing
6.3.2 Information storage
6.3.3 Information search and retrieval
6.3.4 Library automation

7. Computing Methodologies

7.1 Algebraic Manipulation

7.1.1 Expression representation
7.1.2 Algorithms
7.1.3 Languages and systems

7.2 Artificial Intelligence

7.2.1 Expert systems
7.2.2 Automatic programming
7.2.3 Deduction and theorem proving
7.2.4 Knowledge representation
7.2.5 Learning
7.2.6 Natural language processing
7.2.7 Problem solving, control methods and search
7.2.8 Robotics
7.2.9 Vision and scene understanding

7.3 Computer Graphics

7.3.1 Hardware architecture
7.3.2 Picture/Image generation
7.3.3 Graphics utilities
7.3.4 Computational geometry and object modeling
7.3.5 Realism and three-dimensions
Animation, hidden line/surface elimination, shading, color, texture

7.4 Image Processing

7.4.1 Digitization
7.4.2 Compression (coding)
7.4.3 Enhancement
7.4.4 Restoration
7.4.5 Reconstruction (from projections)
7.4.6 Segmentation
7.4.7 Feature measurement
7.4.8 Scene analysis

7.5 Pattern Recognition

7.5.1 Models
7.5.2 Design methodology
7.5.3 Clustering

7.6 Simulation and Modeling

- 7.6.1 Simulation theory
- 7.6.2 Discrete event simulation
- 7.6.3 Continuous system models
- 7.6.4 Model validation and analysis

7.7 Text Processing

- 7.7.1 Text editing
- 7.7.2 Document preparation
- 7.7.3 Index generation

8. Computer Applications

- 8.1 Administrative Data Processing
- 8.2 Physical Sciences and Engineering
- 8.3 Life and Medical Sciences
- 8.4 Computer-Aided Engineering
- 8.5 Information Systems
  - 8.5.1 Office Automation
  - 8.5.2 Decision Support Systems
  - 8.5.3 Communications
    Electronic mail, teleconferencing, videotex

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-85631		

4. Title and Subtitle	5. Report Date
NASA Computer Science Research Program Plan	March 1983
	6. Performing Organization Code 505-37, 506-54

7. Author(s)	8. Performing Organization Report No.
NASA Intercenter Planning Committee for Computer Science Susan J. Voigt, Chairman	
	10. Work Unit No.
9. Performing Organization Name and Address National Aeronautics and Space Administration Washington, DC 20546	11. Contract or Grant No.
	13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546	14. Sponsoring Agency Code RTC-6

15. Supplementary Notes

16. Abstract

The overall goal of the Computer Science research program is to provide the technical foundation within NASA to exploit advancing computing technology in aerospace applications. This goal will be realized through a program of generic research and experimentation which focuses on developing core skills within the Agency in disciplines critical to NASA, and on maintaining a strong university base of fundamental research in aerospace computer science.

Principal objectives are:

1. To provide the theoretical and technology base needed to develop advanced aerospace computing concepts and to evolve advanced system architectures in response to unique aerospace requirements.

2. To increase the Agency's knowledge and ability to develop high quality aerospace-related systems and software, and

3. To provide advanced theory, concepts, techniques, and capabilities for the effective use and management of aerospace information.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Computer Science, Concurrent Processing, Reliable Systems, Software Engineering, Information Management	Unclassified - Unlimited Subject Category 59

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
unclassified	unclassified	101	A06

For sale by the National Technical Information Service, Springfield, Virginia 22161

NASA-Langley, 1983