NASA Technical Memorandum 103860-Vol-2

Computational Needs Survey of NASA Automation and Robotics Missions
Volume II: Appendixes

Gloria J. Davis

May 1991

National Aeronautics and Space Administration

NASA Technical Memorandum 103860

Computational Needs Survey of NASA Automation and Robotics Missions
Volume II: Appendixes

Gloria J. Davis, Ames Research Center, Moffett Field, California

May 1991

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035-1000

SUMMARY

This report is the second volume to "Computational Needs Survey of NASA Automation and Robotics Missions - Volume1: Survey and Results ". It presents in appendix form the supplemental information to the results provided in Volume 1. This information is provided as follows:

List of Acronyms

Appendix A: The Assessment Method

- Questionnaire
- Interviews
- Workshop

Appendix B: Questionnaire Form

Appendix C: List of Contributors

Appendix D: Questionnaire Project Descriptions

Appendix E: Interview, Survey and Workshop Data

Reference List And Suggested Readings

LIST OF ACRONYMS

AAMP	Advanced Automation Methodology Project
AANMS	Advanced Automation Network Monitoring System
AI	Artifical Intelligence
ALF/HAL	Advanced Optical Glasses, Superconducting Glass/Ceramics
A&R	Automation and Robotics
ASAL	Automated Structures Assembly Laboratory
BATSE	Burst And Transient Source Experiment
CCMS	Checkout, Control and Monitor Subsystem
CLIPS	C Language Integrated Production System
CLOS	Common Lisp Object System
CRAF	Comet Rendezvous Asteroid Flyby
CRIMS/TIDE	Cometary Retarding Ion Mass Spectrometer/Thermal Ion Dynamics Experiment
CSI	Controls-Structures Interaction
C&T	Communications and Tracking
DBMS	Data Base Management System
DMS	Data Management System
DSP	Digital Signal Processor
DTA-GC	Differential Thermal Analyzer -- Gas Chromatograph
EASE	Engineering Analysis Subsystem Environment For Spacecraft Control
ECLSS	Environmental Control Life Support System
EVA	Extra Vehicular Activity
EVAR	Extra Vehicular Activity Retriever
FDIR	Fault Detection Isolation and Recovery
FEL	First Element Launch
FTS	Flight Telerobotic Servicer
GN&C	Guidance Navigation and Control
GPC	General Purpose Computer
HARV	High Altitude Research Vehicle
HST DADS	Hubble Space Telescope Data Archive and Distribution
IOC	Initial Operational Configuration
JPL	Jet Propulsion Laboratory
JSC	Johnson Space Center

KATE	Knowledge-based Autonomous Test Engineer
LEO	Low Earth Orbit
LOC	Lines Of Code
LPS	Launch Processing System
LSRA	Landing Systems Research Aircraft
MIPS	Million Instructions Per Second
MSE	Midcourse Space Experiment
MSFC	Marshall Space Flight Center
NASP	National Aero Space Plane
OPERA	Operations Analyst
PAB	Performance Analysis Branch
PEGACUS	Payload Operations Data file Electronic Generation And Control
Utilization	System
PMAD	Power Management And Distribution
PSC	Performance Seeking Control
RISC	Reduced Instruction Set Computer
RTAIS	Real-Time Artificial Intelligence System
SADP	Systems Autonomy Demonstration Project
SBIR	Small Business Innovative Research
SDP	Standard Data Processor
SEI	Space Exploration Initiative
SEU	Single Event Upset
SHOOT	Super fluid Helium On Orbit Transfer
SLOC	Source Lines of Code
SRFCS	Self-Repairing Flight Control System
SSE	Software Support Environment
SSF	Space Station Freedom
SSMEC	Space Shuttle Main Engine Controller
SVMS	Spaceborne VHSIC Multiprocessor System
TCS	Thermal Control System
TDRSS	Tracking Data Relay Satellite System
VHSIC	Very High Speed Integrated Circuit
V&V	Verification and Validation

APPENDIX A: THE ASSESSMENT METHOD

To assess the capabilities and limitations of NASA's A&R systems, the assessment was designed to be as comprehensive as possible, both in terms of projects polled and the information collected from them. Three methods were utilized to perform the assessment: an in-depth questionnaire was developed and distributed widely across NASA, personal interviews conducted with selected project engineers at each of the centers, a workshop was held at Ames Research Center specifically focused on computational processing requirements. Additional supporting information presented here is from various referenced reports.

Questionnaire

The assessment was designed and performed by the Advanced Processing Technology group of the Information Sciences Division at NASA Ames Research Center. The first part was the development and distribution of the questionnaire form, included at the end of this appendix. The questionnaire consists of 47 questions grouped into relevant sections. The questionnaire was designed to ask high level project questions, followed by system-oriented questions (complete and deployable), and finally specific to the internal pieces of the system. It begins with biographical information of the person responding and an overall **project description**, the purpose and goals of the project, time frame with relevant milestones and definition of success and any known technology limiting factors. The next section covers the **functional requirements** of the project in terms of operating environment and conditions, if it uses a standalone system, flight hardware, if there are automation aspects and relevant milestones for the automation capabilities. The **current or proposed system** is then characterized by describing the hardware and software being used, if the application is symbolic, numeric or a mix, and the size of the software. A tangential question regarding the systems use as a benchmark was also included. On characterizing the **processor and processing requirements** internal to the system, questions on description of both hardware and software were asked. Processor(s) selection, criteria used, quantifying limitations realized by selected processor(s), real time responsiveness, memory size, communications capacity, and physical size and power constraints. Space was provided to describe an 'optimal' system for the project barring any monetary or current capability limitations. The 'lessons learned' and 'would change if I could' questions closed the questionnaire.

The final form and content of the questionnaire was arrived at through much discussion and debate. Some questions have been criticized as being ambiguous. However, to be more explicit, either some answers would have been precluded or the survey would not have been answered at all. Also, the provision of multiple choice answers could have precluded answers. The intent was to have the questions interpreted from the perspective of the responder. The intended recipients of the questionnaire were project managers, scientists, system designers, hardware and software engineers. A distribution list of 180 names was generated by references in relevant A&R reports, word-of-mouth recommendations and, to ensure significant persons were not inadvertently excluded, each Center Director was contacted and asked for other pertinent contacts. Of these, 35 completed questionnaires were returned.

Interviews

Distribution of such a lengthy survey, although comprehensive in its quantification aspects, did not provide much qualification of answers. It was considered desirable to have a description of why things were designed the way they were. Also, it was recognized that the time required to complete the survey would preclude some from responding. To fill in the gaps, interviews were arranged at each center. A summary of these contacts is found in appendix B.

These interviews had the format of free-form discussions. With the freedom from a strict line of questionning, much insight was gained as to how different centers work, why languages are selected, perceived problems on projects, and orientations that hardware/ software/ systems for space should follow, etc.

Workshop

The third integral part of the assessment was the *Computational Requirements Assessment Workshop*, sponsored by the Information Sciences Division of NASA Ames Research Center with support from McDonnell Douglas and Recom Technologies, Inc. This workshop was held July 7-9, 1990 and was attended by 90 people, representing various NASA projects from each of the Centers, private industry and academia. The objective of the workshop was to provide a forum that would foster exchange of spaceborne A&R requirements and produce a consensus on the priorities of these requirements. Each invited speaker in the workshop was presented guidelines to follow in providing information on their project. These guidelines were derived from the questionnaire, focusing on current system design, technology used, limitations, if any, due to the design and/or technology, and sacrifices made in realizing the mission successes. The agenda, shown on the following page, was designed to provide a spectrum of requirements, in terms of time and orientation in space. Throughout the workshop, participants were encouraged to record issues they perceived during the presentations relative to computational processing requirements and the effect on NASA. The results and issues will be discussed in the Results section of this paper.

The three day workshop consisted of presentations the first two days and discussion sessions on the third day, ending with a summary of each discussion. A copy of the set of viewgraphs presented are available from NASA Ames Research Center, Intelligent Systems Technology Branch Chief.

Computational Requirements Assessment Workshop for NASA Automation and Robotics Missions, July 9-11,1990

Agenda

Assessment Preliminary Results : Gloria Davis, ARC
Planetary Spacecraft ... Requirements and Experiences: Bob Bunker, JPL
FTS DMPS Requirements: Stan White, Martin Marietta/Denver
EVA Retriever Architecture for Space Station Freedom: Lou McFadin, JSC
Earth Observing System Processing Requirements: Ed Chang, GSFC
SSF Data Management System Platform Definition: Don Woods, McDonnell-Douglas/Houston
Evolutionary SSF/SEI Accommodation: Gregg Swietek, NASA
SEI Planetary Surface Systems: Les Pieniazek, JSC
Computational Requirements of Aeronautics Applications: Lee Duke, ARC
Future Computing Needs, The Necessary Support Systems : John Muratore, JSC
Current Processor Technology Survey: Y. K. Liu, ARC
Advanced Processing Technology: Don Woods, McDonnell-Douglas/Houston
Flight System Computer Development: Bob Bunker, JPL
Code M Flight Computation Requirements: Ed Chevres, JSC
DARPA Computational Requirements/Architectures: Tice DeYoung, DARPA
SSF Hardware Upgrade Strategies: Ned Yelverton, IBM
Next Generation Computer Resources: Cmdr. Dave Hogen, U. S. Navy
Mips Component User Requirements: Jim Fleury, MIPS Corp.
Boeing Spaceborne Processing Array: Jerry Williams, Boeing
Spaceborne Computing: Research Issues and Perspectives: Prof. Ravi Iyer, Univ. of Illinois
Evolutionary Architectures for NASA: Bob Hedges, ARC
Intel Corporation Technology Overview: Herb Marks, Intel Corp.
Splinter Group Discussions:
- Hardware Requirements Specification
- Software Requirements Specification
- Dissemination of Computational Processing Requirements

APPENDIX B: QUESTIONNAIRE FORM

This questionnaire was distributed to each of the NASA centers following a contact list of greater than 300 people. It was introduced by the attached cover letter.

Dear Colleague,

The Advanced Processor Technology (APT) group at NASA/Ames Research Center is investigating data processing requirements for automation and robotics in future NASA space missions. We are trying to determine if NASA's needs in this area are being met by current and planned technology, or if new approaches to hardware and software development will be required by the increased levels of automation anticipated in the future.

We are currently performing an assessment of the data processing requirements for automation and robotics in current NASA projects, and you are invited to take part in this process. Attached is a questionnaire concerning the requirements of your current project. We would appreciate your taking the time to answer as many questions as you can. We are attempting to make this survey as broadly based as possible, so it is important to get information about a large cross-section of projects. Although our primary focus is on embedded processor requirements, some ground-based systems requirements will also be of use to our survey. If your project is primarily ground based, please indicate as such and extrapolate the information as best you can to reflect it in an embedded environment. If you are aware of others in your organization who could contribute useful information, please feel free to duplicate this questionnaire and distribute it to them.

We are also planning to interview in person many of the participants of this survey. The purpose of the interview will be to clarify the questionnaire responses, if necessary, and to get a more in-depth look at the automation aspects of the projects. You may be contacted by a member of the APT group concerning the questionnaire and ask if you would be available for an interview.

Thank you for your cooperation. If you have any questions please contact me at FTS 464-4858.

Gloria J. Davis
Information Sciences Division
m.s. 244-4
NASA/Ames Research Center
Moffett Field, CA 94035

Advanced Automation Computational Systems Requirements and Capabilities Survey

Gloria J. Davis
NASA Ames
MS 244-4
Moffett Field, CA 94035
FTS 464-4858

Survey Overview

Heading	Questions	Topic
Project Information	1-6	Schedule, Goals, Limitations
Functional Requirements	7-14	Automation System Environment
Current System	15-18	Global Quantification
Processor/Processing Requirements	19-42	HW/SW Requirements/Capabilities
Lessons Learned	43	Changes?
		Workshop Interest
		Further Contacts

Automation & Robotics
Computational Requirements Survey

This survey is designed to enable characterization of current and future NASA projects in terms of automation needs and capabilities, system architecture (both hardware and software), computation rates, application size and complexity, etc. We want to thank you in advance for your cooperation with this effort.

Please answer each of the following questions as best as you can. Any answers that you provide will be of benefit to our study.

Biographical Information :

name

phone number

title

organization

place

project name

project description

Project Information

1) What are the overall purpose and goals of the project?

2) When is this project scheduled to be completed?

3) What is the current phase of the project?
(concept, planning, design, development, production, maintenance)

4) What would constitute success for your project?

5) What do you see as the limiting factor for success of this project?

6) Is this an evolutionary project , allowing for system upgrades as they are made available?

Functional Requirements

7) (a) Is your system standalone or a subsystem of a larger system?

(b) Does/will it interface with other systems?

(c) Are there restricting requirements due to this interface? What are the restrictions?

8) Are there any automation aspects of this project? What are they (specific)?

9) What are the critical functions for the automation system?

10) Are the automation requirements necessary for success of the project or do they represent an enhancement?

11) What is the operating environment for the automation system (ground, space)?

12) Is flight hardware required?

13) If there were no constraints on resources (money, time, hardware, etc.), what further automation capabilities would be most desirable and useful?

14) Describe the project schedule in terms of the automation capabilities/milestones needed at various points in time.

Current System

15) What hardware and software are being used or considered for this project? (compilers, languages, applications, operating system, DBMS, network, etc)

16) (a) Is the application numeric, symbolic, or a mixture?

(b) What is approximate mixture ratio in terms of both code and execution time?

17) How large is your software (lines of code)? (predict size of finished product)

18) (a) Can your system be made into a usable benchmark? (benchmark = software to measure critical aspects of an architecture (hardware,compiler, mem. references, etc)

(b) What characteristics of this system would lend themselves to a good benchmark?

Processor/Processing Requirements

19) Are project objectives based on what could realistically be achieved given a specified processor or performance level?

20) What is the lead time between choosing a computational platform and delivery of the end product (your system)?

21) What requirements are used for processor selection?

22) Are currently available processors able to meet all of your computational requirements? If not, what are the current limitations?

23) Were project objectives scaled back because of a specified platform or performance level? How?

24) What processors have been considered, proposed, or selected for your project? (please state the number and kind of processor in the final configuration)

25) If more powerful processors were available, would they be considered?

26) How much more computing power would be desired if there were no constraints?

27) Should it be possible to easily upgrade your hardware system?

28) Does the system have any real-time response requirements? If yes, what is the response time required?

29) How critical are these real-time constraints to the success of the project?

30) What are your special radiation hardening requirements?

31) (a) What are your fault tolerance and reliability requirements?

(b) How were they determined?

32) What maintenance and replaceability constraints exist?

33) How much computational power (in mips or other metric) is required to meet project requirements? (compare to something known, if possible)

34) How much memory is required?

35) How much mass storage space (disk, tape, etc.) is needed?

36) What kinds of networks or communications are required?

37) Is data bandwidth an issue? If so, what is the rate?

38) (a) Is a multiprocessor/parallel processing system being used or considered for this application? If so, why (performance, distributivity, reliability, etc.)

(b) Please describe the system architecture.

39) Would the existence of either a heterogeneous or homogeneous multiprocessor be desirable or useful?

40) (a) What power budget is realistic for your performance requirements?

(b) Does a specified power budget affect objectives of the project?

41) What physical form factor (volume, weight) is realistic to provide your performance requirements?

42) Is it desirable or necessary for the development language and platform to be the same as the implementation language and platform?

43) LESSONS LEARNED (WHAT IS WRONG WITH THE CURRENT SYSTEM THAT WOULD BE DIFFERENT IF YOU COULD CHANGE IT NOW)

(A) PROCESSOR/SYSTEM

(B) LANGUAGES

(C) PACKAGES

BASED ON THE ABOVE QUESTIONS, DO YOU KNOW OF ANY ADDITIONAL PEOPLE OR PROJECTS FROM WHICH WE CAN SOLICIT VALUABLE INFORMATION ON PROCESSING REQUIREMENTS FOR ADVANCED AUTOMATION?

Further Contacts:

Thank You for your Help.

APPENDIX C: LIST OF CONTRIBUTORS

Computational Requirements Assessment

	Last Name	First	Company Name	Attended Workshp	Completed Survey	Interview	Other
1	Andary	Jim	Goddard Space Flight Center				yes
2	Austin	Robert	George C. Marshall Space		yes		
3	Bachtel	Frederick	George C. Marshall Space		yes		
4	Balker	Ed	Langley Research Center			yes	
5	Bartlett	Roger	Digital Equipment Corporation	yes			
6	Bedard	Roger	Jet Propulsion Laboratory			yes	yes
7	Bell	Stuart	Mitre Corporation	yes			
8	Bledsoe	Jim	Boeing (KSC)			yes	
9	Brown	Barbara	John F. Kennedy Space	yes	yes	yes	
10	Brown	Barbara	Jet Propulsion Laboratory			yes	
11	Brown	Robert	McDonnell Douglas Space	yes			
12	Brown	H.E.	George C. Marshall Space		yes		
13	Bull	George	Lyndon B. Johnson Space	yes			
14	Bunker	Bob	Jet Propulsion Laboratory	spk		yes	
15	Castellano	Tim	Ames Research Center	yes	yes		
16	Chacon	Vince	Dryden Flight Research		yes	yes	
17	Chambers	David	Lyndon B. Johnson Space		yes		
18	Chang	Ed	Goddard Space Flight Center	yes			
19	Chevers	Edward	Lyndon B. Johnson Space	spk		yes	yes
20	Citrin	Elizabeth	Goddard Space Flight Center		yes		
21	Coles	Stephen	Jet Propulsion Laboratory	yes	yes		
22	Compton	Michael	Sterling Software	yes			
23	Culbert	Chris	Lyndon B. Johnson Space			yes	
24	Dalton	Danny	Goddard Space Flight Center			yes	
25	Davis	Leon	John F. Kennedy Space			yes	
26	DeLaquil	Don	Rockwell International	yes			
27	DeMasie	Mike	Lyndon B. Johnson Space		yes	yes	
28	Dewberry	Brandon	George C. Marshall Space		yes	yes	
29	DeYoung	Tice	DARPA/ISTO	spk			
30	Doggett	William	Langley Research Center		yes	yes	
31	Dolce	Jim	Lewis Research Center			yes	
32	Dollman	Tom	George C. Marshall Space			yes	
33	Donaldson	James	Jet Propulsion Laboratory	yes	yes	yes	
34	Downie	John	Ames Research Center	yes			
35	Duke	Lee	Dryden Flight Research	spk		yes	
36	Erickson	Daniel	Jet Propulsion Laboratory	yes			
37	Ethridge	Edwin	George C. Marshall Space		yes		

	Last Name	First	Company Name	Attended Workshp	Completed Survey	Interview	Other
38	Fernquist	Alan	Ames Research Center	yes			
39	Fleury	Jim	MIPS Computer	spk			
40	Freydin	Boris	Lockheed Engineering &		yes	yes	
41	Friar	Mason	WRDC/AAAT				yes
42	Friedland	Peter	Ames Research Center	yes			yes
43	Froloff	Walt	Lockheed Engineering &	yes			
44	Fry	Chuck	Recom Technologies, Inc.	yes			
45	Gaines	R.Stockton	USC/Information Sciences	yes			
46	Galant	David	Ames Research Center	yes			
47	Galliher	Jack	John F. Kennedy Space		yes	yes	
48	Gibson	Jim	Recom Technologies, Inc.	yes			yes
49	Glass	Brian	Ames Research Center		yes	yes	
50	Goforth	Andy	Ames Research Center	yes			
51	Gogan	Ray	Harris Corporation	yes			
52	Graham	Robert	Boeing Aerospace &	yes			yes
53	Grant	Terry	Ames Research Center	yes			
54	Gregory	Tom	Ames Research Center	yes			
55	Grisson	Larry	Lyndon B. Johnson Space	yes			
56	Gudea	Denny	TRW, Inc.	yes			
57	Halterman	Karen	Goddard Space Flight Center	yes	yes	yes	yes
58	Hanselman	Phil	WRDC/AAAT	yes			
59	Healey	Kathy	Lyndon B. Johnson Space				yes
60	Hedges	Bob	Ames Research Center	spk			yes
61	Hogen	Dave	Space & Naval Warfare	spk			
62	Hubbard	Charlie	Integrated Inference Machines	yes			
63	Iyer	Ravi	University of Illinois	spk			yes
64	Johnson	Marjory	RIACS	yes			
65	Jorgensen	Chuck	RIACS	yes			yes
66	Katzberg	Steve	Langley Research Center			yes	yes
67	Kehoe	Mike	Dryden Flight Research Facility			yes	
68	Khan	Ashish	MIPS Computer	yes			
69	Kish	James	Lewis Research Center			yes	yes
70	Kissel	Ralph	George C. Marshall Space		yes		
71	Knackstedt	Rich	McDonnell Douglas				yes
72	Krog	Ralph	Lockheed Engineering &			yes	
73	Kulkarni	Deepak	Ames Research Center		yes		
74	Lago	Jose	Boeing (KSC)			yes	
75	Lambert	Ken	Jet Propulsion Laboratory			yes	yes
76	Lawler	Dennis	Lyndon B. Johnson Space		yes	yes	
77	Lee	Junji	Jet Propulsion Laboratory	yes			

	Last Name	First	Company Name	Attended Workshp	Completed Survey	Interview	Other
78	Levinson	Rich	Ames Research Center		yes		
79	Liu	Y. K.	Ames Research Center	spk			yes
80	Loder	Bill	Intel Corporation	yes			
81	Lollar	Louis	George C. Marshall Space		yes	yes	
82	Lum	Henry	Ames Research Center	yes			
83	Mah	Bob	Ames Research Center			yes	yes
84	Maine	Trindle	Dryden Flight Research Facility		yes		
85	Mangieri	Mark	Lyndon B. Johnson Space		yes		
86	Manner	Dave	Sverdrup Tech. (LeRC)			yes	yes
87	Marks	Herb	Intel Corporation	spk			
88	Marquardt	Matt	Recom Technologies, Inc.	yes			
89	Matijevic	Jake	Jet Propulsion Laboratory			yes	
90	Matthies	Larry	Jet Propulsion Laboratory	yes			
91	McFadin	Louis	Lyndon B. Johnson Space	spk			yes
92	McGevna	Vince	Lockheed Engineering &	yes			
93	Meyer	Donald	Jet Propulsion Laboratory	yes			
94	Moeller	Michael	Harris Corporation	yes			
95	Montgomery	Terry	Dryden Flight Research Facility		yes	yes	
96	Morf	Martin	Stanford University	yes			
97	Morrison	Scot	ISI	yes			
98	Muratore	John	Lyndon B. Johnson Space	yes	yes	yes	yes
99	Murry	Nick	Langley Research Center			yes	
100	New	Edwin	John F. Kennedy Space		yes	yes	
101	Noneman	Steven	George C. Marshall Space		yes		
102	Nornholm	Rick	McDonnel Douglas				yes
103	Obenschain	Rick	Goddard Space Flight Center			yes	
104	Obenschain	Rick	Goddard Space Flight Center			yes	
105	Patterson-Hi	Ann	Ames Research Center	yes			yes
106	Pelnik	Tammy	Mitre Corporation		yes		
107	Petrik	Edward	Lewis Research Center			yes	
108	Pieniazek	Lester	Lockheed Engineering &	spk			yes
109	Pinkowski	Patrick	John F. Kennedy Space		yes		
110	Pivirotto	Donna	Jet Propulsion Laboratory				yes
111	Pocock	Gerry	University of Lowell, Computer				yes
112	Purvey	Michael	George C. Marshall Space		yes		
113	Raftery	Michael	Boeing Aerospace &	yes			
114	Raghavan	Bharathi	FCCD	yes			
115	Raugh	Mike	RIACS	yes			
116	Regenie	Vicki	Dryden Flight Research Facility			yes	
117	Reid	Max	Ames Research Center	yes			

	Last Name	First	Company Name	Attended Workshp	Completed Survey	Interview	Other
118	Ringer	Mark	Lewis Research Center	yes		yes	yes
119	Robinson	Claude	Lockheed Engineering &	yes			
120	Robinson	Peter	Ames Research Center		yes		
121	Rohr	John	Jet Propulsion Laboratory	yes			
122	Sarchet	Mark	ISI	yes			
123	Scarpelli	Al	WRDC/AAAT-2	yes	yes	yes	
124	Schulbach	Catherine	Ames Research Center	yes			
125	Schunk	Greg	George C. Marshall Space		yes		
126	Scolese	Chris	Goddard Space Flight Center			yes	
127	Selig	W.J.	George C. Marshall Space		yes		
128	Sitz	Joel	Dryden Flight Research Facility		yes	yes	yes
129	Sliwa	Nancy	Ames Research Center	yes			yes
130	Smith-Taylor	Rudeen	Langley Research Center		yes		
131	Sowziral	Henry	RIACS	yes			
132	Spencer	James	Boeing (KSC)			yes	
133	Statler	Irving	Ames Research Center	yes			
134	Stevens	Ken	Ames Research Center				yes
135	Swab	Rodney	Ames Research Center	yes			
136	Swietek	Gregg	National Aeronautics & Space	spk			yes
137	Thomas	Dale	George C. Marshall Space		yes		
138	Thompson	Dave	Ames Research Center		yes		
139	Utterback	Harry	Space Dept./Johns Hopkins		yes		
140	Valrand	Carlos	IBM Systems Integretion				yes
141	Varnavas	Kosta	George C. Marshall Space	yes			
142	Wall	James	Jet Propulsion Laboratory	yes			
143	Walls	Bryan	George C. Marshall Space		yes	yes	
144	Weeks	Dave	George C. Marshall Space			yes	yes
145	White	Stan	Martin Marietta	spk			
146	Will	Ralph	Langley Research Center		yes	yes	
147	Williams	Jerry	Boeing Aerospace &	spk			
148	Woods	Don	McDonnell Douglas Space	spk		yes	yes
149	Wright	Belinda	George C. Marshall Space		yes		
150	Yan	Jerry	Sterling Software	yes			
151	Yelverton	Ned	International Business	spk			
152	Zimmerman	Wayne	Jet Propulsion Laboratory			yes	

APPENDIX D: QUESTIONNAIRE PROJECT DESCRIPTIONS

PAB : Performance Analysis Branch; Propulsion Laboratory

This multi-program laboratory supports propulsion systems and propulsion design and development activities in the area of performance analysis including: trajectory analysis and reconstruction, SRM performance, liquid engine performance, test evaluation (SSME ground test), flight evaluation (shuttle), real time flight support (shuttle), and engineering photographic and video analysis. The requirements stipulated represents ground based data analyses and reduction systems. Focus primarily on increasing high productivity, efficient use of manpower and efficient processing of flight/test data. Computational power required for the ground based data analysis is: for image processing requires at least 25 MIPS; workstation engineering performance at least 10Mflops and real-time flight/test needs a distributed system with elements at least 25 MIPS for accurate, efficient real-time analysis. To support the various projects, 25 to 30 GB of online memory is required along with >100GB of offline memory. A multi-processing environment would be considered to alleviate the performance constraints currently realized. Language issues could be contained if the development and target platforms housed the same language programs. A lesson learned within the laboratory is that consistency in the environment (Unix) and network protocol (TCP/IP) would enable more efficient use and development.

ASAL : Automated Structures Assembly Laboratory

ASAL develops scenarios/ sequences for the assembly and repair of tetrahedral structures with constant or variable strut length. It also develops human interfaces to facilitate the display of critical information. Currently in the development phase of the project, the use of the algorithms and techniques developed in orbital construction would constitute success of the project. The system will employ robotic/tele-operated techniques to assemble, utilizing onboard fault handling, with a human operator able to advise or override the system. The hardware is intended to be space qualified. The planner will be capable of modeling the entire sequence. The hardware shall be space qualified, modular and self diagnosing to facilitate quick repair. Because it is still in the concept development phase, the lab system utilizes many systems, languages and packages: Unix, C, FORTRAN, Pascal , Ada, VxWorks, VAXELN, VMS, DOS, 386, 68000, VAX. Although the processing requirements are not yet firmly identified, it is clear that current technology is not sufficient. The most intensive requirements come from the real-time 3-D graphics capability. Though Silicon Graphics is approaching the required capability, it is a ground based system.

Robotic Control by Neural Networks

An in-house and SBIR project, the purpose of this project is to demonstrate closed loop control of multi-joint robots using Neural Networks. Upon the completion of two SBIR's, this project is in the planning phase and is seen as an ongoing activity. Ultimate uses would be semi-autonomously for supervised operations (locally) assisting shuttle operations, remote operation with communication

delay (on the moon), and autonomous control (on Mars) in rover-type operations by the year 2019. Critical functions within the system are maintaining stability and verifiability. With no resource constraints, it would be desirable to automate the system to the point where a robot could fully replace an astronaut.

The current system is based on "C" and Brainmaker using Silicon Graphics workstations, PC's, and digital signal processor accelerators. The application is a mix of vision processing and controls. This system would lend itself to a good benchmark of comparing conventionally operated robot arm control versus a neural network application of the same. Current serial processors are not fast enough (throughput wise) to adequately address the system performance requirements (unstated) but it is held that analog neural hardware processors can. The AT&T DSP32C has been selected as processing platform. Also the i860 is being considered for use. Response times of at least 10MHz have been identified as an adequate operational level. With no specific requirements for fault tolerance and reliability operations, it should at least provide graceful degradation to permit safe operation far beyond standard fault tolerant systems. Eventually, this system must operate without maintenance or replacement.

In terms of computational throughput, small systems can now operate with DSP's (25 MFLOP) however more practical system may be limited by the amount of learning required while operating. Also, the mass storage space needed is all that is available. Video training is one example. However small systems easily operate under standard availability.

The current architecture is as described: camera into network generating proper commands to drive motors which make the robot grasp or dock. The power budget and physical form factor are currently unknown, but should present no problems.

HST DADS : Hubble Space Telescope Data Archive and Distribution Service

This project will provide an archive for 30TB of Space Telescope data. Also, making information about the data available on-line and distributing data electronically and on physical media. The purpose of this project is to provide for the integrity of HST data in the archive, make data easily accessibly to a worldwide user community and promote use of this data for research and analysis. Scheduled to be completed in August of 1993 is currently in the design phase. The ability to archive data at rates sufficient to keep up with HST data production and to faithfully reproduce data for users and provide easy access to data descriptions would be success. An example is that 3 TB of data are required to be online (accessibly) within 60 seconds without human intervention. This translates into juke boxes for the on-line optical disk archive. Current requirements identified to meet project goals are 25 Vax MIPS, 20 MFlops capacity in array processor, 656MB memory, 30 TB optical disk storage, 25 GB magnetic disk storage, all with 95% system availability. The currently defined 9 processor system does not support parallel operation.

KATE : Knowledge-based Autonomous Test Engineer

The goal of this project is to provide autonomous monitoring, control and diagnosis of space related ground support equipment in conjunction with reducing software development and maintenance costs, reducing human error and decreasing the test team size. This project focuses on the production system as a reusable piece of software, to be used with various implemented models (the knowledge-base).

V&V is seen as the primary limiting factor for success of this project. A diagnosis and control system of which knowledge capture has been added as an enhancement. Automation capabilities are being increased in steps across all areas needed at the same time. These are control, monitor and diagnosis of multiple systems simultaneously training, checkout and simulation capability, machine learning, explanation of all phases of operation, integration of video, natural language, speech synthesis and recognition.

This is a lisp system on a lisp machine. It is currently being ported to other language (C, Ada). It has 10K loc and is statically 80% symbolic, dynamically 50% symbolic. The requirements used for processor selection were lisp operating system, path to real-time and lisp execution speed. Study is in progress to evaluate lisp and Ada on the 386 processor. The capabilities of the end system have been limited due to the processor performances. Definitely need more performance "all of it", but also needs to support common lisp. Real time constraint is 10mil sec context switching. To meet project requirements, depending on the implementation, need processor performing from 5 MIPS to 100 MIPS with memory size of 8MB and up, mass storage space of 100 to 300 MB. Focusing primarily on performance of a system, would definitely consider using a multiprocessor system.

OPERA: Operations Analyst

OPERA is a suite of expert systems designed to enhance the "operations and maintenance support" of the checkout control and monitor subsystem; a distributed computer network within the Space Shuttle Launch Processing System (LPS). This system integrates CCMS support staff (test conductors, analysts, system engineers, h/w maintenance engineers/techs) functions such as 1) test configuration requirements, 2) error evaluation, 3) error detection, 4) cpu status/history, 5) recovery action required, and 6) error log/tracking. Scheduled to be completed in Spring 91, it is currently in the development/maintenance phase. This project will be successful when used in a real-time operational environment. The current Checkout, Control and Monitor Subsystem (CCMS) architecture is a prime limiting factor.

There are three automation aspects to this system: Automated Problem Report (PR) Generation which reformats the OPERA fault analysis reports for entry into the PR database, the Automated System Message Knowledge acquisition, and the Automated Validation Scenario Generation. Using a TI Explorer II, Sun SparcStation, Sun 3/160, MacII with MicroExplorer with Common Lisp, CLOS, C, Unix, Dos Intellicorp KEE the application is symbolic with more than 50Kloc plus seven knowledge bases approximately 1.3MB in total size. Requirements used for processor selection was to have symbolic processing, user interface efficiency, reliability, and the ability to conform to industry standards. However, have found that symbolic processing performance is a limitation.

Would consider more powerful, if possible, but this is not quantified. The memory size required is >16MB on Lisp cpu for the OPERA expert system, >16MB on SparcStation for the user interface, and >8MB on sun 3/160 for the CCMS system message interface.

The system architecture is described as follows: The opera expert system is based on a distributed blackboard architecture. The OPERA Controller provides the distributed blackboard control structure which receives system messages (SM) from the data sources (predefined scenarios, tape, or real-time) and queues the SMs for analysis by the RTSEMC (real-time system error management expert system). During the process at fault report creation the RTSEMES may ask the assistance of the problem impact analysis (PIA) expert system, Remote Control Video Switch (RCVS) Configuration Expert System (RCES) or the PRACA data base, as well as update the graphical user interface. This assistance is mediated by the opera controller, and replies to this assistance are posted on the blackboard.

This system would benefit from the availability of a multiprocessor system Would change to a Unix CPU with a Lisp coprocessor. It was nice for development to have a tool to quickly prototype an expert system, but deployment and software maintenance costs are prohibitive. Also conforming to standards other than proprietary packages lessens procurement hassles due to the need for single source procurements.

System messages are received real-time and placed in a queue. The OPERA expert system must run fast enough to make sure that the system message queue does not overflow.

STS/SSF Flight Software Building and Verification Facility

Projects reported on here are the expansion and automation of the current facility used for building and testing space shuttle flight software, to include capabilities for the Space Station Freedom. The overall purpose is to incorporate automation technology to improve the flexibility and repeatability of flight software for manned flight programs. Special emphasis is on the isolation of development and production activities. Executed as an upgrade of the current facility, this project is in all phases, from concept through maintenance. Success would be the reduction of STS flight software test time from 77 to 42 days with no increased risk due to test deletion. Also, increased ability to perform automated evaluation of test results. A closed interface that restricts the access to flight computers for data extraction and evaluation is seen as the limiting factor. Automation is the critical function of this entire program: automation in setup of tests, analysis of test results, code and data audits prior to system builds, build sequences and generation of as-built documentation. This program is necessary to increase the STS flight rate.

The current operating environment for the automation system is IBM 3090, Sun, Vax and R6000 workstations, all targeting flight systems (STS GPC AP1015 and SSF SDP 386). If there were no constraints, the user workstations would interface directly to the flight hardware so that tests could be setup, run and analyzed directly by the workstation base automation. Initial automation tasks are targeted to be operational late '91 with advanced capabilities in use for the shuttle program in '93 and then for the SSF program first element launch in 1995.

The processing capabilities are realistically based and will eventually require 120 MIPS. (mainframe). Also identified about 200MIPS capability in 10 to 15 workstations and 4 MIPS in the flight systems. The reliability of the ground system is specified at 99.95% with actual ground availability at 98%. The ground system will require 157 GB disk and the flight system requiring 500MB.

A strong suggestion from this community is not to use proprietary languages. e.g.use Ada rather than HAL/S for shuttle flight software. Also use Ada or C rather than PL/1 in the ground system and SQL based database rather than IBM IMS.

Research Display Computer

Development of computers, 1553 interfaces, and software necessary to create customized displays on the existing F18 HARV. It is expected that the hardware will ultimately interface with helmet-mounted displays.

The F18 provides many weapons-delivery displays for the pilot. These displays cannot be modified to be more useful for flight research without major software changes to a flight critical computer. The RDC provides an alternative approach. This effort provides a tool to meet requirements that are very poorly defined.

The first flight is scheduled for 1991. Software will be continually upgraded to meet evolving project requirements. Prototype hardware is available now. Software development underway. There is no formal plan for hardware upgrades. Hardware used is T800 (transputer) using C and Occam. The application is numeric and the graphics code is being developed on the Sun workstation using object oriented language Eiffel.

A 3 processor system, there are a "few thousand lines of C code" per processor for early applications. Software may grow by a factor of 10 for later applications. However, not enough memory for 100Kloc. The 3 32-bit processors provide much more capability than the traditionally used flight computers. The memory available is too small for AI applications, however for what we've been doing, this is fine.

Memory size does impose some software limits. The total number of possible messages on a 1553 bus is huge. We must efficiently store a subset of these. For the F18 HARV this si not a problem. Real-time update rates reach up to 80 Hz. This is not a flight critical system. It is at most mission critical. The T800 is (maybe) 10 MIPS ~= 1 MFlop. Each T800 has 256KBytes of battery-backed RAM and 256KB. of PROM. Memory is constrained by board area and choice of technology. Static Ram is in DIPs. A modest redesign could increase the memory by a lot. Using a 1553 mux bus communications and T800 links at 10 MBps among the processors. The data bandwidth is not really an issue here. Would like to have a multiprocessor system available.

Power is relatively unconstrained, however heat dissipation is the bigger concern. Hardware will consume <60W in the worst case. The form factor is 4.5" x 7". Is much larger than the project

would like. The circuitry will eventually be put on PCB the size of playing cards (2.5" x 3.5"). The weight is unconstrained.

For ground development, we are using transputer cards that fit in a PC compatible. The T800 is suitable for embedded applications because it is highly integrated. T800 + power + oscillator = working computer since software can be loaded through the serial links after power-up. The processor is unique: small instruction set, 3 register stack, microcoded multi-tasking kernel, 1 interrupt, 4 link processors, timers. View: ok processor.

The languages in use are not specifically suited for embedded applications. The documentations does not address rom-ability of the code, etc. We wrote a dis-assembler to get more information. The software creation process seems to be generating large files containing only small amounts of executable code - which we do not understand.We could maybe find a better environment. Ada = $22K while C is <$1K. We'll stick with C for awhile. Parallel processing requires an attuned mind set. The links are not automatically shared. Deadlocks are a real concern.

We are doing this work with no assembler, no hardware emulator, and no logic analyzer. Amazingly, the work is going well. The PC is an acceptable development host. I wish that we had a second development system.

RECOMMENDATION: for management: nothing that couldn't be solved by a tall tree and a short rope. Well, lots of tall trees and short ropes (assuming parallel processing).

X-29A

F-18 HARV : High Altitude Research Vehicle
CV-990 LSRA : Landing Systems Research Aircraft

These aircraft (systems) provide flight research. Interfaces with the Air Data computers, Sensors and pilot interfaces, the X-29 has analog, discrete and customized serial bus interface with limited time available to input values from other systems. The automation of this project is testing of onboard software. The systems used for AUTOTEST are FORTRAN, C, SparcStations, Unix, Universal Memory Network (40MB/s, no protocols, dissimilar computers). The X-29 uses a 5301 assembly, 4MHz. The F-18 uses 1750's , Ada, 701E's and assembly language. The X-29 uses 32Kloc for the flight control software, F18 has 32Kloc and the CV990 has 10Kloc. Basic requirements for flight computer is that it be flight qualified and have a real-time performance.

Project objectives are always based on what could realistically be achieved, given a specified processor or performance level. Time and memory available always limit performance level. The lead time between choosing a computational platform and delivery could be as long as 5 years. The requirements used are typically the "fastest, flight qualified". X-29 had funds a major driver in selection, and off the shelf 5301 was the cheapest.

If faster processors were available and flight qualified and the aircraft systems were designed to swap in new computers, we would. The X-29 memory is limited, speed limited - is 99% full. A 1000% increase in speed would allow discrete systems to approach analog performance.

Limitations cause project sacrifices. X-29 flight control modes were removed to reduce flight test requirements. Also, variable gain selectable by the pilot is limited and testing hooks were not added to make software easier to test. However, if more powerful processors were available, the nature of flight test does not lend itself to swapping in new processors as available (yet). For the F-15 SRFCS (Self-Repairing Flight Control System), some applications desired would swamp a CrayII. These include real-time modelling of thermodynamics, stress (NASP) engines (F-15), aero.The real-time constraints of 40Hz (8.25 to 25 msec) in the X-29 are life critical, as the vehicle is fly-by-wire, thus having no mechanical backup. It is very critical for simulation to predict proper responses and for aircraft control systems to provide the desired capability.

The rad. hard. requirements and fault tolerance are mil. std. Need "as much processing power as possible" - CrayII level at least, along with reasonable amounts of memory (4 - 8 MBytes) with 10 meg. mass storage- again as much as possible. Compilers are inefficient. Programs that start with higher order languages always do portions of software in assembly.

PSC : Performance Seeking Control

It is an engine control system designed to optimize the engine's steady state operation. It is going to be implemented on the F-15 research airplane. The algorithm attempts to determine optimal engine trim settings real-time, taking into account engine degradation and variability. Scheduled to be completed April of 91, it will be flight tested in "a couple of months".

Success would be measurable improvements in fuel flow for a constant thrust and 5 to 15% improvements in thrust, though not simultaneously. The algorithm relies very heavily on the accuracy of an onboard engine model. If the model proves to be significantly less accurate than expected, the algorithm will have serious problems. This is a control system that requires only that the pilot turn it on and select the operating mode. The interface with the basic engine control systems severely limits the PSC algorithm. It controls the type, frequency and magnitude of all engine inputs.

If there were no constraints, greater memory capacity and speed would be helpful. The biggest help would be to rework the interface with the engine control system. This whole project was predicated on the assumption that the existing hardware would be acceptable.

Using F77 on a Rolm Hawk computer, the application is 100% numeric in about 13K loc in about 1 MByte. Greater computational capability could have lead to more complex models or a different algorithm but from the start, this program was designed around existing hardware. Timing issues are not critical because of the steady state assumptions made for the algorithm. This system is not flight safety critical so all that is required is the ability to recognize a fault and turn the system off. For this the basic engine control is well established.

Being able to use FORTRAN is a real luxury. Would like to have been better able to force the contractor to rigidly stick to F77 ANSI standard.

SSMEC: Space Shuttle Main Engine Controller

SSMEC is a second generation engine control computer which monitors and controls SSME operation. The computer is a dual/ dual architecture with Motorola 68000 CPUs. The system is designed for high reliability through extensive self-test and the use of class "S" or equivalent piece parts. This is to provide a system that will reliably control an SSME and provide red-line protection in the event an out-of-normal engine condition occurs. Unit production scheduled to complete in '93 and remain operational for <= 10 years. Success of this program would be no in flight failures coupled with minimum ground failures, because ground failures erode confidence. This is not an evolutionary project due to cost.

This system interfaces with the GPC's and sensors and actuators of the shuttle. Designed to monitor red-line temperatures and pressures and control valve positions, it has full control over the engine valve actuators to implement commands received from the GPC. Also has authority to shutdown the engine in response to anomalous situations. A two-part system, requires both ground and space parts. Substantial effort is underway in the area of condition monitoring/failure prediction. It might be desirable to incorporate this feature into this system, given limitless resources.

Current hardware is MC68000 at 8MHz running C and assembly language, a numeric application. Software must fit in less than 128 KBytes. Requirements used for processor selection were memory access efficiency, availability of information necessary to certify chip to class "S" and throughput of >= 0.5MIPS. Most commercial processors could meet the computational needs, but difficulty lies in obtaining the certification level necessary to fly. For upgrade to hardware, there is not limited consideration being given to the feasibility of a processor upgrade as an enhancement to the current system. Some projections for a health monitoring system estimates approx. 10MIPS would be needed. Our current system is capable of 0.5 MIPS. Memory size is 64K words (16bit) per processor. As an enhancement to the system a pre-processor is being considered and will be used to reduce the load on the CPUs. A multiprocessor would be useful, only if the existence of a highly certified support for the device, i.e. compilers, disassemblers, etc.

Current system is composed of 2 channels, each with a pair of processors used in a self-checking pair configuration, each channel having dual watch dog timers in addition to extensive cross-strapping between channels with 1 channel active and the other a hot backup. Rated at about 700 watts, the size is 15"-x-18"-x-24" @ 215lbs.

Lessons learned: Assuming the availability of the proper class of parts (which is a major problem) I would obviously shift from the current 16bit 8MHz 68000 cpu with 2K-x-8 static RAM to a higher speed 32bit cpu with non-volatile EEPROM or similar.

The current language used is C, but would return to assembly for superior speed/efficiency.

SHOOT : Super Fluid Helium On Orbit Transfer

Ground and shuttle based software to operate 1992 (STS-59) cryogenics experiment. The purpose and goals of the project are twofold: to support Goddard and provide ground based control and monitoring software, and to develop autonomous capability to transfer helium from shuttle Aft Flight Deck (AFDex). Success would be no software problems, good scientific return and in flight have autonomous transfer successfully completed. A limiting factor to the success would be adequacy of ground testing.

Because this is an experiment, the commercial platform provides no special fault tolerance/reliability capability. Safety is not a prime concern and loss of experiment is the only risk. The system will be deployed for 1 week in low earth orbit.

Since the project is a technology demonstration, there would be no advantage to automate the entire operation of the payload, but, the planned system is a minimal autonomous system because of hardware and resource constraints. A much more sophisticated autonomous system could be imagined for error detection and diagnosis, but it does not make sense economically to automate a one time operation.

This system, written in C and CLIPS on a grid 386 (a system specified by the shuttle program) is a mixture of 80/20 numeric/symbolic code.Memory size has been constrained to 2 MB for flight and 8MB for ground.

Space Station Freedom, in general

The purpose is to establish permanent manned presence in low earth orbit, scheduled to be completed in 1999. The SSF is to permanently support live crew, house zero-G science and operate as a staging arena for planetary missions. The on-orbit resources (power, crew time, etc) are seen as the limiting factors for success.Automation is currently seen as an enhancement to the program but not as a critical factor for success of the basic station. Functions targeted for automation are dextrous manipulation, fault diagnosis of subsystems, etc: used to alleviate resource problems regarding crewtime for EVA and maintenance. One such automation system specified could be a house-cleaning robot to scrub walls. This would free a substantial block of crewtime.

Although current project objective are based on what could realistically be achieved given a specified processor, it is apparent that the present DMS cannot meet the station requirements as written. Requirements are mostly unknown

SSE : Software Support Environment

This project involves the development of tools, procedures, and standards to support the developers of Space Station Freedom computer software systems. It will provide a consistent set of tools and standards to be used for the development of Space Station Freedom Program (SSFP) software.

Scheduled for completion in 1993, there will be sustaining engineering extensions indefinitely. The project is being developed incrementally and is currently in the operational increment number 5 phase. Success of the project would be acceptance and use by the community of users. However, it is recognized that it is difficult to impose software development tools and rules on developers, given different tastes and techniques.

Each instance of the SSE is a Software Production Facility (SPF). Each SSFP developer owns a SPF and all are networked together into one large system. There are interface control descriptions that govern the characteristics of the interfaces.

Many parts of a software production process are subject of automation, such as Automated code generation and automated customized software release construction.

Although this is a ground based system, flight hardware is supported as a target platform from the SSE. The project automation milestones are driven by the users and the next are to be developed are automatic code generator and software build automation.

Primarily a collection of COTS systems, the environment will also include approximately 1 million source lines of Ada code to support configuration management.

Requirements for selection of environment is primarily large support from commercial software development tool vendors. Looking ahead for enhancements, the software is being developed with portability in mind. A primary lesson learned is that Ada has limited commercial support. However, this is becoming less of a problem with time.

SSF PMAD: Space Station Freedom Power Management And Distribution Automation Evolution

This project consists of several tasks which are unified toward experimentally demonstrating the operation of a highly autonomous, user-supportive PMAD system for the Space Station Freedom HAB/ LAB modules. Success of this project would be to demonstrate a fully automated PMAD system with a fully integrated user override capability. Also, to identify the resources and requirements for deploying such a system in space. A goal is to impact the requirements for SSF. Automated aspects of the system include immediate power system safing, short term load shedding, limit checking and reporting, redundant load switching, schedule implementation, fault detection, fault isolation, fault diagnosis, scheduling load prioritization, and dynamic scheduling.

The critical functions of this system are implementation of normal operation, FDIR, scheduling and load prioritization. This system is initially targeted to support the Station from the ground but eventually be installed onto the station itself. The current lab system uses common Lisp and CLOS and porting from Symbolics machines and motorolla 68010's to Solbourne 5/501 and q-max 80386's. Project objectives can be met with hardware currently used, but this is optimistic as the system is redefined and trimmed to more closely resemble resources available for use with SSF. Would like more computing power at the Symbolics level. With more than 50K LOC of which 45%

is symbolic.To meet the processing requirements, a total of 20 Vax MIPS is required for the Solbourne, 2 MIPS for the 80386's and about 5 MIPS for the Symbolics. Need 0.5 MB memory at the controllers and 2 to 3 MB at higher levels.

A lesson learned in this project is that use of proprietary systems is a problem, using open systems from the beginning is a good move.

ECLSS: Environmental Control and Life Support System for the SSF program.

It is our goal to utilize previously unused closed loop life support technology to minimize resupply/return logistic penalties. Success of this project is a system that meets all program requirements including those for respirable atmosphere, potable and hygiene water, and emergency operation. System weight, power and volume constraints influence the ECLSS design. On-orbit computational resources will impact the overall level of automation, including the number of sensors and effectors. Funding levels could defer complete closure of the ECLSS until later in the SSFP lifetime.

Functionally, the ECLSS is standalone, although it requires services from supporting systems, such as electrical power, DMS, thermal control and man systems. Supporting systems have a finite resource available (such as kw, MIPS, etc) in which the ECLSS must coexist with other space station systems. Man systems is different in that it serves as the interface between ECLS subsystems and the crew.

The ECLSS is highly automated. Aside from corrective/preventative maintenance, crew established manual overrides and non-critical fault isolation and redundancy management, the ECLSS can be fully automated. Onboard computers perform most process control functions, interactions with other systems, fault detection functions, criticality 1 ORU fault isolation and, if the proper manual overrides are inhibited by the crew, the ECLSS can perform emergency reconfiguration or take appropriate response to an emergency, such as suppress a fire. Considering the life critical nature of the ECLSS, most all automated functions that support it are critical.

The ECLSS is to be primarily automated on-orbit, although the breakdown between ground and space processing has not been completely defined. Some flight hardware will be required. It is currently unknown how large the application software will be or if it will be symbolic, numeric or a mixture. Answers to the processor/processing requirements were deferred to SSF DMS personnel.

ECLSS : Environmental Control and Life Support System Advanced Automation Program for Space Station Freedom

Development of rule-based fault-detection algorithms for baseline on orbit ECLSS regenerative systems, with initial operating configuration (IOC) model-based diagnosing on the ground. The model based diagnosis systems will move to flight software when computer resources on board permit. The overall goal of this program is to increase the autonomy of the SSF ECLSS.

Success of this program would be actually increasing the autonomy of the ECLSS with actual use of rule-based fault detection algorithms on board, complemented with model based diagnosis autonomy in the ECLSS ground support center. A limiting factor of success would be a lack of computational resources on orbit and on the ground which cause baseline software development managers to be extremely conservative with their development methods. We are counting on the computational resources of the SSF, especially those dedicated to ECLSS processing, to be upgraded to allow migration of high fidelity automatic fault diagnosis software on-board. Also, our development is based on a cyclic model in which we return to the requirements phase after each demo. The on-board fault detection algorithm will meet the Ada task interface. It will be called as an Ada task. The diagnosis software must use a round support and display software. Detection interfaces with the element ECLSS supervisor software diagnosis interfaces with display software and may one day recommend recovery action or command tests automatically. On the ground the interface is to the network.

The major software hooks and hardware scars necessary for evolution to a more autonomous ECLSS have been identified. The advanced subsystem FDIR requires component sensors to be available from the RODB within 1 second, allowing subsystem's control loop latency of 5 to 10 seconds. This provides real-time fault detection and fault preventive reconfiguration to use 3 to 8 seconds with communication to ECLSS subassembly monitoring process taking 2 seconds. Also called for is software process location transparency (dynamic memory allocation). This was explicitly removed from baseline. The automation efficiency would be increased by the use of model-based reasoning tools, like KATE and ART/Ada for early design knowledge capture. It is recommended that these tools be added to the SSF SSE. This model-based reasoning approach to subsystem FDIR would allow minimal use of explicit leak detection sensors by inferring leaks using the baseline process control sensors.

Processor power pulled down so unfortunately nothing extra, like advanced automation, can be onboard. Basically anything that can be achieved on ground must be done so. Only real-time critical functions are left for space-based system. Reasons are for power, weight, cost.

Onboard ECLSS supervisory system, using soft real-time, utilizes total of 1 SDP worth of processing/memory size (4 MIPS/4 Meg). The hard real-time system is distributed among about 20 MDM's and the total size is 10 to 12 MDMs. The fault sensor here are directly connected to the MDMs. It is only the supervisor section that interfaces with the DMS. The reason for automation is that we can do more with less. Make sure sensors necessary are in place. We have enough for what is critical. We don't have leak detection sensors. Compensate for this by knowing the characteristics of failures downstream, analyze in the model base and then detect the leak, or the general area, from this system.

On getting information from the DMS, are assured "yes, you'll get your information on time", but performance requirements are sketchy at best. The software developers for the supervisory section have no strictly defined requirements.

Automation of water quality monitor output, requiring fast processing of real-time chemical and/or microbial analysis in the life support control system would ensure quick determination of drinkability

of water. Onboard processing may require fast symbolic and/or neural net processing. Recommend more research on this topic.

The current lab. system for the Advanced automation ECLSS is using Sun RISC, porting to sun 386i and SunOS using KATE, ART/Ada, Lisp and TAE+. The software is a mix of 80% symbolic and 20% numeric code of which 500KB is the size for the detection and 4MB for the diagnosis. On-orbit applications have been limited to fault detection because the model-based fault diagnosis requires more processing capabilities than are available. Thus fault detection only is onboard with no deep diagnosis.

Because now is ground-based system, programming in C. We know this language. Leaving Clips and ART/Ada because neither supports objects. They are more oriented to associative reasoning or production systems.

For our current efforts, 4 MIPS with 4 Meg are good. We have a slow system and it works ok. Our turnaround time on sensor sample analysis is at worst case minutes.

On fault tolerance, most systems utilize TMR. For each SDP on SSF there is a cold spare backup SDP that would require up to 30 minutes to boot up and enable to a working state. Unclear whether the information during transition is lost or what.

FTS : Flight Telerobotic Servicer

FTS is a Space Station Freedom element for use in assembly and maintenance. Beginning as a tele-operated machine, it is planned to evolve into an autonomous robot during its 30 year lifetime. This is to assist astronauts in assembly and maintenance of Space Station Freedom, reduce need for crew EVA time and provide technology transfer to U.S. industry. The FTS will be ready for SSF First Element Launch March 1995. However, there are planned upgrades tot he FTS throughout its lifetime.

The limiting factor for success of the project is that there are many technical challenges to be overcome in a short development time. There are many restricting requirements on the system affecting power, weight, safety , size, human factors considerations of the operator, structured environment needed for autonomous sequences, knowledge of workspace, thermal dissipation.

At FEL, preplanned sequences will run in a supervised autonomous mode (under astronaut supervision) and later evolve to include more autonomy by including better sensors and vision systems which would enable onboard path planning and perception.

The system will use DDC-I Ada running bare Ada in the distributed 80386/387 environment. With 40 MIPS processing capacity at FEL, there will be more than 20 computers in the flight system. The program code will be strictly numeric, with flight software the size of 60K SLOC in Ada and the man-machine interface coming from the DMS. Memory requirement is listed at 20+ MB RAM and 600MB mass store. A parallel processing capability along a strong backplane is desirable. 200 Watts is the stated power budget for this project.

Requirements used for selection of the processor(s) were cost, performance, availability for flight qualification, support of Ada, compatibility with SSF, reliability and modularity. So far, the initial operational capabilities of the FTS will be supported by the proposed system, however if problems arise the intent is to upgrade the platform and not scale back the project. Other processors being considered for upgrades are RISC technology. The real time response requirement is 10 millisecond around the loop between hand controller to manipulator and force reflection back to the hand controller. This is essential for stability, performance (dexterity) and human response requirements necessary for teleoperations. When evolved into a robot with the human out of the loop, the timing will be less critical.

Within the program, a lesson learned is that the upgrade to fiber optic networks within the telerobot would yield more centralized processing, parallelism and faster performance among the cpu's. Ada is heralded as a good programming language choice for this project.

SSF User Operation Integration

The User Operation Integration project for the SSF is the function of integrating and coordinating the operations of SSF payloads by users located around the world. Success of this project would be successful operation of payloads onboard the SSF such that science data is collected and delivered to the scientists and investigators. A primary limiting factor for this is ensuring timely communications capability, using voice, video and data. Currently in the planning and design phase, this is scheduled for completion in 1997. Incorporated into the system must be protection against unauthorized access.

This system must be highly reliable. Critical functions for the automation system are providing mission planning support, electronic mail, electronic forms processing and database management. If no constraints were realized, would use expert systems and AI in general for the data collection applications. Operating in both ground and space environments, some flight-qualified hardware will be required. Although the computational processing requirements have not yet been defined, the development environment currently consists of Vax/VMS architecture, DECNET running FORTRAN and Ada. It is anticipated that existing hardware should meet all computational processing needs. The real-time response requirement to be satisfied is within 5 seconds. This system is not safety critical, it must be at least 1 fault tolerant.

PEGACUS : Payload Operations Data File Electronic Generation And Control Utilization System

This project will generate and control the development of payload procedures, ranging from manual procedures to semi-automated to fully automated procedures. The overall goal is to allow users to develop, verify, and uplink payload procedures. This system is a subsystem of Payload Operations Integration Center (POIC). There are automated command sequences that the computer would step off and execute at the appropriate time. The PEGACUS system itself is not automated.

The critical functions for the automated procedures are timing and the ability to have correct orbital information, e.g. sunrise/sunset times, moonrise/moonset times, etc. This automated procedure is an enhancement in that it frees up a crew member or a ground controller from having to perform certain steps or send certain commands. The system has two parts: ground based for training and checkout purposes and space based for actual experiment operations. The automated procedure will run on flight computers, but will not be generated on flight computers. The ground system is written in C on a Vax. This program is still in the design phase and hence the processing requirements have not yet been determined.

AANMS : Advanced Automation Network Monitoring System

This involves the development of a network monitoring system capable of intelligent fault monitoring of FDDI-based networks. The critical functions are to provide in real-time predictable, hard-deadline scheduling within SSF constraints, e.g. total time to respond to any fault is under 1.0 second. This system shall automatically detect, isolate and diagnose network faults based on data passing on the network. Also provide a flexible user interface for semi-automated systems. We would like to have, instead of 10MIPS platform 16*25=400MIPS, for data capture and extraction functions only. This number is based on Silicon Graphics experiment with NetVisualizer - a conventional network monitoring software, running on their 16 MIPS personal IRIS 4-D/25. This software was able to process only 4% of FDDI traffic. 4% is 25 times less than would be required computing power for all of FDDI traffic. The FDDI will generate (worst case) 12.5 MB/sec. For data capture we would like to have 40MB RAM (3 seconds worth of storage). Other functions may require around 24 MB. So, minimum is 40+24=64 MB. In order to reach our evolutionary objectives, we may need from 128 to 256 MB and up. Each hour worth of raw data will require 12.5 MB/sec times 3600 sec/hour = 45 GB/hour. FDDI is already too fast for our platform. Addition of other OSI layers should be supported by an additional computing power.

AAMP : Advanced Automation Methodology Project

AAMP is the investigation of the development of advanced automation systems within Space Station Freedom guidelines. To do this it will develop two independent systems, each being integrated into existing SSFP testbeds upon completion. The ultimate goal is production of an engineering methodology. The two systems are the Advanced Automation Network Monitoring System (AANMS) and the Recovery Procedure Selection Application (RPSA).Success for this project would be development of an engineering methodology for advanced automation systems for use in the SSF program and successful integration of AANMS into the C&T testbed, and RPSA into the DMS. The RPSA will automatically select plans for altering the KU-band (space to ground communication) configuration to an operational mode whenever it learns that ORUs have failed.

Critical functions for the projects are that the AANMS will collect data, detect faults, isolate fault causes and determine trends. The RPSA will select plans to alter configuration of the space-to-ground communication subsystem. The only constraints that exist for this project are that each system be "easily maintainable" by virtue of the structure, documentation, and readability of the source code. In terms of processing performance, the RPSA should require less than 25 MIPS and

the AANMS at least 100 MIPS. The memory constraint is unknown for the AANMS, and RPSA is limited to 0.5 MB. Both systems are ground-based in the testbed environment. An AANMS-like system would be space-based when the SSF is operational.

Ada is the primary programming language used with C for certain driver interfaces. The 40K SLOC for RPSA runs currently on Sun workstations and the AANMS (5K SLOC) uses VMS-based workstations. The computational platforms were defined prior to initiation of the AAMP (i.e. the testbeds provide the platform).

Without constraints, each system might attempt to cover all aspects of fault detection, isolation and recovery, as well as deal with uncertainty in the reasoning process (e.g. the potential falsehood of facts). However, the project objectives will not be scaled back because the operations will be determined by prototyping results. AANMS would consider/use more powerful processors if available, but for RPSA the memory size is the major limitation.

SADP : Systems Autonomy Demonstration Project

The SADP highlighted AI-based control of the Space Station Freedom prototype thermal bus. This project was designed to demonstrate AI based real-time control of a representative space subsystem. Success of this project would be safely control the thermal bus during nominal operations and recognize all required faults, and act upon their recognition.

Current system is/was the Symbolics 3650, implemented in Lisp for efficiency and ease of development, and FORTRAN for compatibility with existing software. The software is 2/3 symbolic and 1/3 numeric, size is 24 MB memory (on each of 2 Symbolics machines) and 8 MB on the microVax, with 450MB hard disk space plus 600 MB for virtual memory, and utilizing >=32bit paths. Processing required is roughly 3xSymbolics 3650 or 9-10 MIPS. This was found to require an order of magnitude faster computational processing performance, radiation-hardened SSF-compatible Lisp processor that runs KEE.

Because it was a research project, the system selected for the demonstration was to determine what was realistic to achieve. Limitations were realized in performance causing the hypothetical reasoning to be scaled back, an order of magnitude improvement was needed to retain it. Real-time response requirements were set at cycles <30 seconds. A multiprocessor/parallel processing capability would have enhanced the performance and been considered for use. The major lesson learned is to stay away from proprietary packages (no KEE-dependence).

CRAF/Cassini: Comet Rendezvous Asteroid Flyby/Interplanetary Mission to Saturn

The purpose of CRAF/ Cassini is to study planetary bodies and to try and understand the origin of life on our own planet. "All interfaces are restricting simply because an interface requires definition, and definitions restrict." Critical functions of the automation system are doing fault detection, recovery and spacecraft 'safing'.

Current computational power is 0.4 MIPS with ~12 MB memory and 3.6x10^9 bits storage. The system architecture resembles a group of intelligent nodes communicating over a network of 1553B buses. The data bandwidth rate is 1Mb/sec. Throughput on the microprocessor internal parallel bus is the real bottleneck. Power limitation is 20Watts. If we could have more resources, they would be higher bandwidth, more cpu throughput but requiring less total power. Really would/could use a 32 bit microprocessor instead of a 16 bit. Rad hardening requirement of SEU LET >= 37, latchup LET >= 100 and total dose is <= 40Krads. On fault tolerance, "Success critical single failure points are not permitted if avoidable by functional or block redundancy" (JPL d-1489). This was based on mission plan and science objectives. This system, because of the end environment, cannot be maintained or replaced.

EASE : Engineering Analysis Subsystem Environment for spacecraft control

This will permit multi-mission control team analysis of uplink and downlink telemetry data for Galileo, Magellan, Mars Observer, CRAF, Cassini, and Ulysses. A fault diagnosis expert system, EASE would monitor and analyze downlink telemetry data for anomalous events. The real-time responsiveness to anomalies is the systems critical function.

The Sun workstation environment running Unix, C and X-windows, supporting AI tools is necessary. A minimum of 12 MIPS per station (for 6 workstations) with 16 MB RAM and 10GB optical disk are required for project support. However if 50 MIPS per processor/workstation were available, it could be utilized to good advantage. The system must execute round the clock, so high reliability and fault tolerance are necessary.

DTA-GC : Differential Thermal Analyzer-- Gas Chromatograph

It is an autonomous soil analysis instrument, developing autonomous controller for the coupled geochemical instrument DTA-GC, develop data analysis modules, develop real-time (0 min.) response system to alter experimental run to maximize data return, develop planning system to select between alternative hypothetical futures based on currently available data during an experimental run. Learning and discovery of relations in data by various techniques. Prepare for the Mars Rover Science Analysis Instrument. The overall purpose of the project is to control experimental runs and provide data analysis of unknown soil samples, assisting remotely located scientists. Success of this project would be achieved if geochemists on the team deem the system useful as an assistant for remote analysis of unknown soils.

With space as the final operational environment for this system, the ground-based prototype system uses Lisp and KEE. The application is ~5000 LOC and is strictly symbolic. Being a research environment prototype, the system requirements are still to be determined.

MSE : Midcourse Space Experiment

Space observation platform that has infrared, ultraviolet, and visible sensors. It involves tracking and attitude control computers. Success of this project would be successful launch and observation of dedicated targets. Factor limiting this success would be the ability of the sensor to "see" as well as advertised.

The system is primarily numeric, consisting of ~50K LOC written in Ada. The 1750A processor was selected (2 single cpu's and 1 dual), providing 0.18 MIPS, but the capability of a 25MHz 80486 system would provide all desired computing power. The system must be qualified to sustain 15Krad at 880 Km polar positioning. The power is limited to 50 Watts. The fault tolerance requirement is not yet quantified, although the system must operate 100% of the time for each of the 30minutes mission events.

The surveyed engineer indicated that a MIPS-type processor (performance wise) in a rad-hard version at about 30 Watts for general space applications would be "useful" .

SEI : Space Exploration Initiative

The overall purpose of this project is to establish permanent Lunar and Mars bases. To be completed in the 2000 to 2020 year time frame, this project is in the concept and planning phase. Because of this, much of the processing requirements are not yet determined. It was indicated that automation applications would entail guidance, navigation and control of transportation systems, and manned and unmanned facilities/ experiment operation and maintenance, such as ECLS. The processing systems shall be designed to tolerate/operate in space, the lunar environment and the Mars environment.

CSI : Controls-Structures Interaction Program

The interaction and integration of Controls and Structures for large space systems. The purpose of this program is to investigate and integrate the design and control of large space structures in support of NASA flight missions. This project is scheduled for first flight experiment in FY 95. This experiment is to demonstrate a match of laboratory and space data. Budget was stated as a limiting factor for success of the project.

Much of the system is still being defined, however the 1750A running Ada has been selected as the target system, based on science requirements. Parallel processing would be considered as an enhancement to performance.

CRIMS/TIDE: Cometary Retarding Ion Mass Spectrometer/ Thermal Ion Dynamics Experiment

No description supplied

This could be an automation system. Due to limited telemetry bandwidth, all available data cannot be transmitted. A heuristic method for selection of data to be transmitted would be useful if it could be shown to be effective. However, no humans can state this heuristic, the processing power onboard (2 each NS 32016 using C) is not sufficient for this function, and the project scientists do not trust this function to symbolic processing. If, however, a heuristic capability was implemented, it would be seen as an enhancement.

The selection of the onboard processors was based on rad hardness, speed and power consumption. It is admitted that most processors are not available in rad-hard versions and hence the selection is limited. Because of real-time constraints, some floating point calculations will possibly be given up. More processing power (stated "as much as possible") would be desired if no constraints were realized.

The system consists of 2 microprocessors splitting the duties of data collection / experiment control and formatting/ onboard analysis /telemetry compression

BATSE : Burst And Transient Source Experiment

A set of eight gamma ray detectors monitor the entire sky (not obscured by the earth) for sudden changes in the gamma ray flux. An onboard processor based on the Texas Instruments SBP9900 processes the data to detect these bursts and switches all processing to those detectors that best see the bursts. Calibration and housekeeping chores are also handled by the SBP9900. The limiting factor for success is the ability to completely analyze the large amount of data produces by BATSE.

Automation aspects of the project are to gain calibration of the detectors and provide rapid burst location to allow switching to the detectors that best see the event. Accuracy is the critical function of this automation. The calibration could be handled from the ground but the burst location could not because the data is transmitted once a day and is therefore 12 hours old (on average).

This system uses the TI SBP9900 programmed with its own assembly language (8K words of machine code). With as little "computing" onboard as "practical", most is performed in data reduction. The onboard phase of burst detection and location is very critical: essentially a triggered transient analysis of each detected burst to SGE both the event and some data prior to the event are performed. This is accomplished by saving far more data than could continuously telemeter. When an event is detected, the current block of data is saved and downlinked (requiring 1 hour). False triggers or failures to trigger are serious problems.

With eight detectors interfaced to the spacecraft, performance would be enhanced and improved if one processor was within each detector (rather than one central processor for the spacecraft). It would also be good if higher level language and a cross compiler to the flight processor were usable.

ALF/HAL : Advanced Optical Glasses, Superconducting glass/ ceramics

Involves the development of a high temperature containerless processing facility for processing of glasses and ceramics. Success would be flight of a high temperature levitation apparatus for non-electrical conductors; demonstration of high index of refraction/low dispersion glass; and development of useful glass/ceramic superconductors.

The automation aspect of the project is the multiple sample exchanger and programmable processing cycles for telescience/teleoperation. The data transfer rates for the video is the most critical part of the automation. This automation is absolutely required in order to achieve the highest throughput of samples and data collection during the extremely valuable low-g processing time. Flight hardware is required for this project and a 1 MIP processor with 4 MB memory would probably be adequate.The rapid, efficient transmission of video data is the major concern and more than the current data transmission bandwidth between shuttle and ground would be useful.

RTAIS : Real-Time Artificial Intelligence System

RTAIS will develop and demonstrate a high-performance, JAIWG-compatible AI processor design that efficiently executes both data and AI real-time avionics applications, such as the DARPA/WRDC Pilot's Associate. This system is scheduled to be completed in March of 1993.

The RTAIS is to host Ada, C++ and a contractor-developed real-time operating system. The hardware processor is based on RISC technology. The throughput of the multiprocessor system will be at least 200 MIPS and have 40 to 100 MB memory. Each SEM-E module (JAIWG standard) must consume less than 50 watts. Fault tolerance and reliability are designed into the system.

APPENDIX E: INTERVIEW, SURVEY AND WORKSHOP DATA

The next sections summarily present the requirements identified in this survey. Some of the project descriptions are expanded on to provide insight into the reason for the requirements, how they were derived, and an idea of the state of program development. These projects are presented specifically relative to their targeted environment, as presented in Volume 1 of this report.

Ground Systems

The ground-based computational arena is described in a manner similar to that for off-ground. The top priority for selection of the ground-based systems assessed was typically cpu performance. With an unlimited amount of resources available, the fastest processors with the greatest available support are used for development, mission operations and analytical applications. When necessary, reliability can be designed without regard to power and mass considerations. The software design is easily focused on without great concern for limitations in memory size. This is only a crude description of the ground environment, the results of which are next presented.

Development Environment

It was found that many research systems are developed without adhering to textbook software engineering methods. Proper methods, necessarily followed by the operations programs of shuttle, research aircraft flight systems,etc. include functional requirements documents, premeditated software structure design and systems tradeoff analysis. Adhering to these methods ensure the result of a well designed, modular, efficient and well documented system. However, the typical scenario presented in the predominantly research arena was thus: an idea evolves into a quick, working prototype, using whatever machines and systems are available. After a feasibility study, the prototype is renamed a development system and is iterated upon, often bypassing those techniques designed to ensure a quality software product. Many development systems, especially for automation, use special purpose symbolic machines. Using Lisp in a prototyping mode promises rapid results on which to decide the concept feasibility. However, after establishing feasibility of the prototype system and readying it for operational use, the same hardware system is typically not available for end, deployable system support. In one interview it was pointed out that "the 1750A is just fine for us, we don't need any more space-qualified computational capability". This same lab was working on several different "automation" prototypes using Symbolics workstations and Sun SparcStations using Lisp. These applications are intended to be deployed in space, but there was "no plan yet" as to how they would be transitioned from this laboratory environment to the end space environment. It is not necessary that all prototyping be performed using only space-qualified systems, but rather there must be more of a consideration as to how a transition would/could take place from the lab to the target environment, and beyond through maintenance.

Independent NASA Laboratories

The following sections present the concerns expressed at four different research laboratories visited. All of these used equipment designed for ground, however not targeted expressly for ground-based systems. The first laboratory presents the functionality of an automation system that is in the initial stages of development. Their requirements have been analyzed and the hardware is currently being selected. The second and third labs have the end system defined and are in the middle of functional development. Both are for similar types of robotic-systems, that of the second lab is ground based and the third is for space. This is the most complex of the four presented, an automated robotic system for space. The last of the labs presents a fully functioning system and the concerns at the "end" of a development effort.

Lab1: Automation Project Initiation One responsibility of the Software Testing Laboratory at JSC is performance testing of flight software. A sequentially followed matrix of more than 20 tests has been established for all software testing. Therefore when a test fails, the entire process is re-executed because it is difficult to know a priori to the discovery of failures which other sections of a system are affected and how. After an analysis of how this testing process would or could benefit from automation, one of the tests, the performance testing of loads, was selected as the first candidate. This currently requires 40 engineers to accomplish: 30 GNC and 10 Vehicle Cargo systems. They test 2 cycles per flight and 4 to 6 flights are processed at a time. The functional requirements for the automation system were to reliably perform rapid plotting and analysis of data. A typically mundane job, it is described as 98% boring and 2% terror. Because the Shuttle flight rate is increasing, the time frame for performing this task has decreased significantly. Based on these requirements, it has been determined that *high powered workstations* with windows, plotfilters and a table filter will be selected. This system is currently being designed and developed. As can be seen, the requirements used for processor and system selection are not always explicitly quantified. It is rather more commonly found that "fast" or "the fastest available" is indicated as the requirement for system performance. In this environment, this is not only acceptable but most often all that is warranted.

Lab2: Hardware Satisfaction At another laboratory, the focus is on Shuttle tile and orbiter radiator inspection. The engineers are working on robotics that will dock and mate with millimeter tolerance. The movements are at high speed and hence the error margin is not much. Having a machine able to handle this limited margin is a problem. They explicitly stated that they have no need for further processing capabilities. "When we run out of speed, we can always buy faster". Currently using a Symbolics 3640, they have achieved the real-time control they need. The system on the Symbolics is executing shared control, path planning and reasoning applications. The system is currently based on force feedback control because it is most feasible to achieve success in this implementation. The system is physically stiff but mechanically fast. Although additional computational capabilities are not required, it was stated that "we want a real-time , 3D,Image processing, pattern recognition system". They pointed out that "AI and vision processing are not necessary to accomplish high speed processing. Rather just use neural networks and high parallel distributed processing.

Concern was expressed here relative to the software philosophy. It has been consistently followed that "whatever works is best" will be sufficient. Yet those writing the code want and prefer to do it right, (i.e.follow strict software engineering principles) thus reducing future time for maintainability. Simply stated, they need software Quality Assurance. These engineers would like to see standards across NASA established and a degree of configuration control. This could reduce software costs significantly, especially relative to the maintenance costs.

Lab3: Hardware Limitations A third laboratory is focused primarily on Automated Robot Assembly. The team is currently writing an expert system that would enable a robot to assemble large structures in space, drawing on a knowledge base of the structures. This project is designed with flexibility so as to be used on SSF, Aerobrake activity, Mars Rover/ habitat, etc. It uses planners, a robot, implementor, end effectors, and sensors, all dependently integrated. The system is designed to be functionally layered with a planner for each layer. This would therefore enable multiple plans to be generated for same end effect. Ultimately the (or an) optimal plan would be selected and executed.

A relatively young project that's been operational for 1 year, the current efforts focus on software and robot interface. Everything is serial. The Expert system is in FORTRAN using a MicroVax and the robot end effectors are based on the M68000 with a PC XT. All communication is serial via RS-232. With this kludge of hardware, it was admitted that "we are using whatever we can get our hands on". "Currently focusing on getting functional, later get realistic." They went on further indicating they are using FORTRAN because "one unknown is enough". Good software structure of the system has been established. With the rules known, the inference engine will be incorporated at a later time, when more of the hardware system is together. However, the software has mushroomed, and they don't yet have the experience in this area.

Like all of the other NASA centers, a problem of generational gaps was expressed here. The new technology using planning and expert systems is difficult to establish acceptance. No one worries about computer architecture as hardware is the longest lead time item. Yet, because of the software structure already defined, a speed-up in system performance is already known to be required. This speed-up will be achieved through concurrency, networking, and hopefully more processing capabilities. For the robot to operate in supervised autonomy mode, it requires lots of cpu just updating the user - let alone planning/replanning itself. The processing requirements would be more stringent in a non-supervisory mode because the user updates would be replaced with more replanning capability and added reliability. The operation of total autonomy is the ultimate goal, however the hardware requirements are for this end system have not yet been determined.

It has been the experience with this team that, when collaborating on work with other organizations - SEI servicing/proc, assembly/construction, A&R, Mars Rover and Habitat Construction - all of have extensive plans, but none looks at computational processing architectures. We are in a catch 22 where we must design expert systems/planning systems that work. The systems people say the software is good so use X architecture. After the architecture is selected, though, we must cut and fit the software application that is already designed, thus sacrificing some criticalness of the software. The mission sacrifice here is not easily pointed out because "every part of the system is critical, otherwise it would not be in the system in the first place".

If funding were consistent, it was stated that "we'd have been into this architecture thing and have models developed". It was acknowledged that hardware selection dictates software design. On requirements for computational processors, it was readily stated that "most people don't know what they want, they don't realize what they need or could use until it's built and run in front of them."

Lab4: A Working System At the AI laboratory of KSC, the *Knowledge-base Autonomous Test Engineer (KATE)* project is well underway. A functional system has been developed. The intent of KATE is to provide autonomous monitoring, control and diagnosis of space related ground support equipment. Its nominal use is designed to reduce software development and maintenance costs, reduce human error and test

team size. This project focuses on the production system as a reusable piece of software, to be used with various implemented models that are supplied in the knowledge-base.

Verification and validation (V&V) is seen as the primary limiting factor for success of this project. It would be optimal to perform reliable software validation of KATE independent of the knowledge base used, therefore making it easier to be used in a wide variety of applications without revalidating core software. One such is a diagnosis and control system to which knowledge capture has been added as an enhancement. Automation capabilities in this project are being increased in steps simultaneously across all areas needed. These are control, monitor and diagnosis of multiple systems simultaneously training, checkout and simulation capability, machine learning, explanation of all phases of operation, integration of video, natural language, speech synthesis and recognition.

This is a Lisp system on a Lisp machine. It is currently being ported to Ada and a C version may also follow. After it was developed the usefulness was realized in demonstrations. Now, rather than perform a redesign of the entire system, it will be adapted to application areas "as best as can be achieved". It has 10K SLOC of which 80% is symbolic, however dynamically is 50% symbolic. The initial requirements used for processor selection were to provide a Lisp operating system with a path to real-time , and fast Lisp execution speed. A study is in progress to evaluate Lisp and Ada on the 386 processor for the port. The capabilities of the end system have been limited due to the processor performances. Definitely need more computational processing performance, quantified as "all of it", but also needs common Lisp. Because of this, speed is not of primary importance. The real time constraint is 10msec context switching. To meet project requirements, depending on the implementation, a processor is needed performing from 5 MIPS to 100 MIPS with memory size of 8MB and up, mass storage space of 100 to 300 MB. Focusing primarily on performance of a system, engineers would definitely consider using a multiprocessor system.

This is an example of a system that was developed, the usefulness realized in the demonstrations and, rather than face a redesign of the whole system, adapted to application areas as best as can be achieved. The processor speed, however important, is of secondary concern. The focus is on system performance and this system is currently mandated to process Lisp efficiently.

Laboratories: Common Concerns

Although the four laboratories reviewed had different focuses and were in different phases, they nevertheless present a spectrum of the concerns all have or will deal with. As in lab1, the initial concern is determination of the overall functionality of the system and selecting a usable system that will provide the maximum performance and flexibility at the outset. Because the use environment is ground, this is not selected by an indepth analysis of data throughput and timing. The selection criteria are relaxed. However, once the system is under development, and a certain "level of commitment" is established with the hardware at hand, the realization of the limitations begin. At this point in time, the limitations are not identified as hardware or software but rather "system". Because the hardware is selected and at hand, the only true variable is the software. Hence, subsequent major changes in the system are typically made in the software.

The second and third labs admitted using readily available hardware. This was due in part to lack of funding but was accepted because the idea of hardware upgradability was a given. To focus on the unknown of creating a software system, the outcome is more in the engineers' control, thus leaving the "fastest

hardware available" to be added in at the last moment. The last lab with several demonstrations of a working system is past the feasibility design stages. The concern at this point is in making the system more efficient. However, in doing so, the system is very much tailored to the hardware and hence compensations are realized in the software application. More specifically, rather than redesigning the entire application to all new hardware, the scope of the current application is reduced to executing more efficiently on what is readily available.

Ground Mission Operations

A working suite of expert systems yields requirements that are not met by the currently available processors. This is primarily due to the selection of symbolic processing. OPERA is, by design, an automation system to be used to enhance Space Shuttle operations and maintenance. The description is provided below.

Case Study: Operations Analyst (OPERA) OPERA is a suite of expert systems to enhance "operations and maintenance support" of the checkout control and monitor subsystem; a distributed computer network within the Space Shuttle Launch Processing System (LPS). This system integrates Checkout, Control and Monitor Subsystem (CCMS) support staff (test conductors, analysts, system engineers, h/w maintenance engineers/techs) functions such as 1) test configuration requirements, 2) error detection, 3) error evaluation, 4) cpu status/history, 5) recovery action required, and 6) error log/tracking. Scheduled to be completed in Spring 91, it is currently in the development and maintenance phase. This project will be considered successful when it is used in a real-time operational environment. The current CCMS architecture was cited as a prime limiting factor.

There are three automation aspects to this system: automated Problem Report (PR) generation which reformats the OPERA fault analysis reports for entry into the PR database, the automated system message knowledge acquisition, and the automated validation scenario generation. Using a TI Explorer II, Sun SparcStation, Sun 3/160, MacII with MicroExplorer with Common Lisp, CLOS, C, Unix, DOS, Intellicorp KEE, the application is symbolic with more than 50Kloc plus seven knowledge bases approximately 1.3MB in total software application size. Requirements used for processor selection were to have symbolic processing, user interface efficiency, reliability, and the ability to conform to industry standards. However, it was found that symbolic processing performance is a limitation. More powerful processors would be considered, if possible, but this has not been quantified. The memory size required is >16MB on Lisp cpu for the OPERA expert system, >16MB on SparcStation for the user interface, and >8MB on Sun 3/160 for the CCMS system message interface.

The OPERA expert system architecture is based on a distributed blackboard architecture. The system Controller provides the control structure which receives system messages (SM) from the data sources (predefined scenarios, tape, or real-time) and queues the SMs for analysis by the RTSEMES (real-time system error management expert system). During the process at fault report creation, the RTSEMES may ask the assistance of the problem impact analysis (PIA) expert system, remote control video switch (RCVS) configuration expert system (RCES) or the PRACA data base, as well as update the graphical user interface. This assistance is mediated by the opera controller, and replies to this assistance are posted on the blackboard.

It was reported that this system would definitely be more efficient based on a multiprocessor system. Also, the use of a Unix supported CPU with a Lisp coprocessor is desired. It was nice for development to have a tool to quickly prototype an expert system, but deployment and software maintenance costs are prohibitive.

Aeronautic Systems

It can be argued that aeronautic applications are likely the most demanding on computational processing systems, particularly due to real-time interrupt and task-switching performance constraints. Reliability is of utmost importance and must also be designed in from the outset. These systems must be extremely fault tolerant, but for different reasons than space systems. The sacrifice in these systems for problems could be the life of the pilot. Yet, because of the space limitations and the research programs flying additional systems, the flight control computer is as small as is allowed. The memory is so small and the processor performance not state-of-the-art that the programs are in assembly language and are bare bones. That is, much testing that could be onboard, for efficiency and productivity, is deferred to the ground. Extremely little margin for error exists in these two areas for these applications. Both commercial and military/NASA aeronautic designers are also now realizing (as they transition to "fly-by-wire" systems) that radiation and SEU (cosmic ray) hardness is also becoming an increasing factor in the reliability domain. Processor performance is a consideration, but with few processors to choose from, the primary focus is on real-time performance and establishing reliability for the applications designed. Although these two different issues are typically not dependent upon each other, they receive equal consideration in system design.

Four completed questionnaires were received from Ames-Dryden Flight Research Facility, discussions were held with 7 engineers and a presentation on aeronautics application was made at the workshop. Information on the X-29, F18 and CV-990 aircraft, regarding displays, automated flight qualification and landing systems and flight controls was presented. The results are as follows.

Aeronautics General Requirements

At DFRF it is recognized that there is a small market for flight computers. Because of the nature of the environment and the military standards imposed, qualification of processors requires a sufficient amount of time and extensive tests. No matter what the project, there is inevitably a lot less computing power available than is needed. Although several computers are available, they are not computationally powerful.

Although computational processing throughput is of concern, the primary requirements are physical size and cooling. It is now, when the operating environment for computing is in the air, that the processing throughput cannot be the initial consideration. The limited size of the aircraft and the lack of temperature control place great restrictions on system design. Airborne processors must be able to take power hits and be able to handle "glitches". PROMs are typically used in applications for this reason.

Because of limitations in processing capability, there is no onboard processing of research data. Rather the data are telemetered down to the ground, processed on high powered workstations, and sent back to the

aircraft. For efficiency in operations and research projects, it would be optimal to put this onboard for researchers. In the AFTI F-16, the pilot has over 100 buttons and lights. He is the integrator of this whole complex system. This is now becoming a tradeoff of data versus information. There is a definite need for onboard automation to integrate the increasing amounts of information for the pilot.

Critical Technologies

The requirement of efficient, capable computers must be met. They must provide real time performance in flight critical or even mission critical situations. It is in mission critical areas where a lot of technology can come to bear. The aeronautic environment by nature pushes the limits on operations and hence the enabling technology must do the same.

For aeronautic applications, it is imperative to determine a better way to transfer from development environment to working system. With the current development environment that utilizes design tools, it is not feasible to include an inference engine on the flight computer. Our target environment requires systems that are "fast and clean", therefore after the systems are designed in research, they must be cut and trimmed in order to execute correctly on an operational system.

In this environment, applications are scaled back on a continual basis, even though the limitations of processing power are always recognized from the start. To enable maximum throughput from the systems, software is adapted on a regular basis. This is best described by an example, as such: Have a do loop in the code; rewrite it iteratively n times to save the tests at the outside of the loop. This is a trade off of memory size to processing speed. Here the speed is a critical limitation.

The flight control computers are typically selected 5 to 6 years in advance of the flown applications. Systems flown are at least 95% full at flight and when sacrifices are made, "extraneous software" is removed. This also is done to ease the testing of software. Also, with the software tailored to the hardware, oftentimes tricky code is used. This is difficult to maintain, as time goes on. The risk of accident greatly increases as the original designers of the code leave.

It was explicitly stated that the software either limits or enables the system performance. The available system restricts the size of the task. The major sacrifice here is in system reliability. The F-15 project gave up a high fault detection model for the damage assessment and compensation system onboard, thus scaling down the task of self repairing.

The software is designed to the hardware. Using relatively old hardware the flown systems are not as capable as ground technology renders. And, once written, the software is set into the computer. Even if a good or better architecture is designed, the switch to it would be so formidable that it would not be done. So only new applications would incorporate new hardware.

If more compute power was available with more memory, then the size of the software would increase, which requires more testing, then it won't fly. Additional V&V is necessary to handle this. Simply stated, if a processor on next program that was ten times faster it would allow for:

1) increased onboard testing
2) incorporation of vehicle system health monitoring.

3) more ground testing onboard
4) do all in higher order language.

Research Aircraft

Three of the primary research aircraft at Ames-Dryden Flight Research Facility are the X-29A, F-18 HARV (High Altitude Research Vehicle) and the CV-990 LSRA (Landing Systems Research Aircraft). These aircraft (systems) provide flight research through interfaces with the Air Data computers, sensors and pilot interfaces. The X-29 has analog, discrete and customized serial bus interface with limited time available to input values from other systems. Basic requirements for the flight computer is that it be flight qualified and have a real-time performance.

Project objectives are always based on what could realistically be achieved, given a specified processor or performance level. Time and memory available always limit performance level. The lead time between choosing a computational platform and delivery could be as long as 5 years. The requirements used are typically the "fastest, flight qualified", unless funding limits this selection.

If faster, flight qualified processors were available then the aircraft systems would be designed for upgrades in new computers. This is not typically the case, however, and therefore whatever flight computer a system is designed on is the one to stay. The X-29 memory and processor speed are limited, flying at 99% full. A 1000% increase in speed would allow discrete systems to approach the analog performance. This capability is not projected in the near future so we continue to design to these older systems.

The hardware limitations definitely cause project sacrifices. The X-29 flight control modes were removed to reduce flight test requirements. Also, variable gain selectable by the pilot is limited and testing hooks were not added to make software easier to test. However, if more powerful processors were available, the nature of flight test does not lend itself to swapping in new processors as available (yet). For the F-15 SRFCS (Self-Repairing Flight Control System), some applications desired would swamp a CrayII. These include real-time modelling of thermodynamics and stress (NASP) engines (F-15).The real-time constraints of 40Hz (8.25 to 25 msec) in the X-29 are life critical, as the vehicle is fly-by-wire, that is having no mechanical backup. It is very critical for simulation to predict proper responses and for aircraft control systems to provide the desired capability.

The radiation hardness and fault tolerance requirements are military standard. Need "as much processing power as possible" - CrayII level at least, along with reasonable amounts of memory (4 - 8 MBytes) with 10 MB. mass storage- again as much as possible. The currently available compilers are inefficient. Programs that start with higher order languages always end up with portions of software rewritten in assembly language due to memory size and execution efficiency.

Research Display Computer The F18 HARV aircraft provides many weapons-delivery displays for the pilot. These displays cannot be modified to be more useful for flight research without major software changes to a flight critical computer. The Research Display Computer (RDC) provides an alternative approach in providing customized displays on the existing hardware. It is expected that the new system will ultimately interface with helmet-mounted displays. This effort provides a tool to meet requirements that are very poorly defined.

The first flight is scheduled for 1991. With software development underway it will be continually upgraded to meet evolving project requirements. The prototype hardware is available now with no formal plan for upgrades. The 3-processor system is based on the T800 (transputer) using C and Occam. So far, there are a "few thousand lines of C code" per processor. This is expected to grow by a factor of 10 for later, more complex applications. The application is numeric and the graphics code is being developed on the Sun workstation using object oriented language Eiffel.

The 3 32-bit processors provide much more capability than the traditionally used flight computers. The memory available is too small for AI applications, however for what we've been doing, this is fine.

Memory size does impose some software limits. The total number of possible messages on a 1553 bus is huge. We must efficiently store a subset of these. For the F18 HARV this is not a problem. The real-time update rates reach up to 80 Hz. This is not a flight critical system. It is at most mission critical. The T800 is (maybe) 10 MIPS ~= 1 MFlop. Each T800 has 256KBytes of battery-backed RAM and 256KB. of PROM. Memory is constrained by board area and choice of technology. Static Ram is in DIPs. A modest redesign could increase the memory quite a bit, using a 1553 mux bus communications and T800 links at 10 MBps among the processors. The data bandwidth is not really an issue here. We would like to have a multiprocessor system available.

Power is relatively unconstrained, however heat dissipation is the bigger concern. Hardware will consume <60W in the worst case. The form factor is 4.5" x 7" , which is much larger than the project would like. The circuitry will eventually be put on PCBs the size of playing cards (2.5" x 3.5"). The weight is unconstrained.

For ground development, we are using transputer cards that fit in a PC compatible. It has been found in this project that the T800 is suitable for embedded applications because it is highly integrated. T800 + power + oscillator = working computer since software can be loaded through the serial links after power-up. The processor is unique: small instruction set, 3 register stack, microcoded multi-tasking kernel, 1 interrupt, 4 link processors, timers.

The languages in use are not specifically suited for embedded applications. The documentation does not address rom-ability of the code, etc. A disassembler was written to enable more information to be obtained. The software creation process has generated large files containing only small amounts of executable code. Parallel processing requires an attuned mind set. The links are not automatically shared and deadlocks are a real concern.

PSC: Performance Seeking Control Control systems are always a concern for NASA. Aircraft systems are among the most critical to be controlled, because they carry human life onboard. One such system is the Performance Seeking Control project.

The PSC is an engine control system designed to optimize the steady state operation of the F-15 engine. The algorithm is designed to determine optimal engine trim settings in real-time while compensating for engine degradation and variability. It controls the type, frequency and magnitude of all engine inputs. This project is scheduled to be completed in April of 1991.

Success would be measurable improvements in fuel flow for a constant thrust and 5 to 15% improvements in thrust, though not simultaneously. This is a control system that requires only that the pilot turn it on and select the operating mode, however the interface with the basic engine control systems severely limits the PSC algorithm.

Using F77 on a Rolm Hawk computer, the application is 100% numeric in about 13K loc in 1 MByte of memory. Timing issues are not critical because of the steady state assumptions made for the algorithm. This system is not flight safety critical so all that is required is the ability to recognize a fault and turn the system off. For this the basic engine control is well established.

If there were no constraints, greater memory capacity and speed would be helpful. The biggest help would be to rework the interface with the engine control system. This project was designed around the existing hardware and predicated on the assumption that its performance was acceptable. Greater computational capability could have lead to more complex models or a different algorithm.

Aeronautic Systems: Launch, Shuttle,NASP

Launch vehicles present a mix of processing requirements different from the ground-systems, involving the real-time performance of aeronautic systems and the environmental and extended mission considerations of space systems. System reliability is of utmost importance and changes in software are not realized as quickly as for research aircraft. However, because they are maintained on a regular basis, the upgrades are readily achieved, more so than with deployed space systems.

The information presented in this section is extracted from the Space Shuttle program. Launch vehicles must consider the most critical issues from both the aeronautics environment and the space environment.

Space Launch Vehicles

A Boeing study was conducted to identify technology programs needed to demonstrate flight critical avionics architecture for next generation space launch vehicles. [see bibliography] The Multi-path Redundant Avionics Suite (MPRAS) System/Subsystem presents requirements for advanced space launch vehicles, stressing that "autonomous flight and ground operations are key features". An analysis of three candidate configurations were defined and presented summarily in Vol.1. Each configuration presents increasing degrees of autonomy, operating in the same mission scenario. As can be seen, the performance requirements of the total system vary from 17 to 20 MIPS. The upgrade to the new general purpose computer (GPC), a three times increase in processor speed over the previous GPC, yields a 40% increase in system performance. It is unclear whether this capability will be sufficient for the studied configurations.

SSMEC: Space Shuttle Main Engine Controller Information was provided by MSFC on the Space Shuttle Main Engine Controller (SSMEC) in a completed survey. The SSMEC is a second generation engine control computer which monitors and controls the main engine operation of the Space Shuttle. The system is designed for high reliability through extensive self-test and the use of class "S" or equivalent piece parts. This is to provide a system that will reliably control an SSME and provide redline protection in the

event an out-of-normal engine condition occurs. Unit production scheduled to complete in 1993 and remain operational for <= 10 years. Success of this program would be no in-flight failures coupled with minimum ground failures, because ground failures erode confidence. This is not an evolutionary project due to cost.

This system interfaces with the GPCs and sensors and actuators of the shuttle. Designed to monitor redline temperatures and pressures and control valve positions, it has full control over the engine valve actuators to implement commands received from the GPC. Also has authority to shut down the engine in response to anomalous situations. A two-part system, it requires both ground and space parts. Substantial effort is underway in the area of condition monitoring/failure prediction. It might be desirable to incorporate this feature into this system, given limitless resources.

The currently utilized hardware is based on a dual/dual architecture with MC68000 cpus at 8MHz running C and assembly language, a numeric application. A firm requirement is that the software must fit in less than 128 KBytes. Requirements used for processor selection were memory access efficiency, availability of information necessary to certify chip to class "S" and throughput of >= 0.5 MIPs. With these requirements, most commercial processors could meet the computational needs, but the difficulty lies in obtaining the certification level necessary to fly. For upgrade to hardware, there is now limited consideration being given to the feasibility of a processor upgrade as an enhancement to the current system. Some projections for a health monitoring system estimates approximately 10MIPs would be needed to provide sufficient throughput capability. The current system is capable of 0.5 MIPs. Memory size is 64K words (16bit) per processor. As an enhancement to the system a pre-processor is being considered and will be used to reduce the load on the CPUs. A multiprocessor would be useful, only if the existence of a highly certified support for the device, i.e.compilers, disassemblers, etc.

The current system is composed of 2 channels, each with a pair of processors used in a self-checking pair configuration, each channel having dual watch dog timers in addition to extensive cross-strapping between channels with 1 channel active and the other a hot backup. Rated at about 700 watts, the size is 15"-x-18"-x-24" @ 215lbs.

The major problem facing computational systems design for the shuttle is the lack of availability of the proper class of parts. A shift from the current 16bit 8MHz 68000 cpu with 2K-by-8 static RAM to a higher speed 32bit cpu with non-volatile EEPROM or similar add on is necessary. Also, the current language used is C, but assembly language should be used for its superior execution speed and efficiency.

SHOOT: Super Fluid Helium On-Orbit Transfer A project that is to be flown on the Space Shuttle (STS-59) in 1992 as an experiment is the Super Fluid Helium On Orbit Transfer system (SHOOT). The purpose and goals of the project are twofold: to develop an autonomous capability to transfer helium from shuttle Aft Flight Deck (AFDex) and to support engineers at GSFC in providing ground based control and monitoring software. Completing the experiment with no software problems, yielding a good scientific return and having autonomous transfer successfully completed in flight would mean success of the project. Inadequacy of ground testing of the system poses the greatest limiting factor to achieving success.

Since the project is a technology demonstration, there would be no advantage to automating the entire operation of the payload, but the planned system is a minimal autonomous system because of hardware and resource constraints. A much more sophisticated autonomous system could be imagined for error detection and diagnosis, but it does not make sense economically to automate a one time operation.

This system, written in C and CLIPS on a grid 386 computer, is a mixture of code type with 80% numeric and 20% symbolic. It uses ground and shuttle based software to operate the cryogenics experiment. The memory size has been constrained to 2 MB for flight and 8MB for ground. The system has been designed to these hardware limitations; more onboard testing could be added to the system if more memory and faster processors were available. Because the system was designed to the selected hardware, no mission sacrifices were realized.

The Software Dilemma

John Muratore, Division Chief of the Shuttle and Space Station Freedom Software Testing Division of JSC not only completed a survey but also spoke at the workshop. He expressed concerns specifically associated with software, both systems and applications.

Software Costs The Space Shuttle computational processing system enforces heavy memory constraints. The cost of software is currently $10,000 per line of code. In the design, software cost = hardware cost. However, due to testing and maintenance, the software cost is 80% of the overall system price. It is nowhere more apparent that the largest consumer of funds is software testing and validation. The less code there is to test, the less the overall cost. Therefore it is often a design decision to limit the amount of memory available - primarily to limit the overall cost. Therefore software growth is controlled by limiting hardware availability.

The systems used to fly the Shuttle currently have some verification problems, primarily due to the limitations in computational capabilities. The system uses 90% of the processor duty cycle, allowing 10% for errors. However, when the change to the new general purpose cpu is made this will be reduced to a comfortable 50%. This projection is based on keeping the current tasks as is and providing for no further technological enhancements.

However a hardware upgrade is imminent. The first flight of the new GPC is scheduled for summer in 1991. This will provide a larger memory space of 128K ram from the currently available 104K RAM.

Support tools are highly integrated, in expectation of V&V for both Shuttle and SSF software. It was emphasized that systems must be built on standards to retain the large investment, in time and money, in software. To facilitate this, standard support on any and all hardware for these applications must be included.

Further capabilities that should be added to the shuttle system are autonomous ascent and guidance, autonomous rendezvous, further FDIR and autonomous monitoring systems. This can only be accomplished with increased performance over the available processors and the increased size of memory allocated for onboard use.

Software Testing Given that this facility is oriented specifically to testing flight software, it is readily acknowledged that there is no way to exhaustively test the effect of all software changes. Because of this, a 'comfort factor' is used for acceptance. Given that the system is required to fly, the risk areas and those areas that other critical systems depend upon are identified and tested. Upgrades of computer systems would provide additional capabilities, but would also require new tests of the resident application to ensure that it

works the same as before. This would require identification of new test hardware to help with v&v, and isolation to divorce from underlying hardware issues.

Problems have been found in hardware microcode but this is not typical. The primary changes and problems are realized in the system and application software. Although a systems approach to overall design is warranted, isolation between software and hardware must be retained because "not all code is created equal". Because of the nature of the shuttle launch schedule, some requirements change all right up to just before flight. Other critical design functions that change very slowly must be completed long in advance. These concerns must be isolated to minimize the impact of changes on them. A small carry on computer is recommended for the rapidly changing applications. This would facilitate the isolation requirement and enable reliable last minute testing.

Most of the critical applications for the Shuttle rely on real-time execution. Yet there is an inherent maintenance problem for real-time and there is little to no support for development and reliable debugging, except in Ada. Even with the intended incorporation of real time expert systems into the shuttle program, the real-time attribute is more important than the expert system part. Further support in development and testing tools must be provided to enable continued deployment of reliable systems in the increasingly aggressive launch schedules.

Software solutions are described in a way relative to the orientation of the designer: mathematicians solve toy problems, engineers cheat to solve real problems and proofers cheat to have success at toy problems. This is so, partly because of the lack of tools available, partly because of a lack of standard development procedures to follow, and partly because of the nature of the discipline. Establishment of a uniform process would enable solid solutions to real problems.

Performance Not all low earth orbit computers are to be qualified to class S specifications. For experiments to be conducted in flight, they need to be able to withstand the vibration of launch and hazards of low earth orbit. The projected SSF upgrade capability of the 486 with 20MIPS at 20MB is considerable. Typically the tools deployed are fat, therefore placing the support software on a diet will yield much more. If allowed, the use of virtual memory would enhance the performance capability. Also, incorporation of a multiprocessor architecture would add MIPS, but more importantly provide for function isolation, mainly for traditional computing tasks such as controls. For robotics, video and speech, another architecture may be necessary. It must be noted, however, for most efficient use and optimal system design, the multiprocessor system must be an original design rather than an add-on capability.

Low Earth Orbit: Equatorial (SSF)

Since the Space Station Freedom will be a permanent facility, upgrades and configuration changes will take place on-orbit. In its lifetime, national priorities will change, user needs and mission requirements will change, technology will evolve and components will become obsolete. Utilizing mature technology tools and applications is among the requirements for achieving high-level engineering fidelity. While striving for successful design, deployment and operation, important concerns continue to be operations costs and reliability. With both concerns in mind, it is essential to incorporate new technology into the program. Realizing that crew is the most scarce resource, productivity is crucial in meeting assembly, user and

servicing requirements. The evolution mission scenarios are crew intensive while at the same time science missions will grow and demand for crew time will increase. Also in on-orbit assembly, checkout and launch of Lunar/Mars vehicles the crew time will be essential. Advanced automation programs are designed to enhance the baseline SSF capabilities to improve productivity and reliability, reduce operations costs and prevent technological obsolescence. Applications are under development for thermal control, power, life support, data management and mission control. All of this will enable SSF evolution. To realize these applications, however, the capabilities on which to deploy them must also be available. As it currently stands, however, the SSF design is not sufficient for the advanced automation activities.

SSF Requirements and Platform Overview

The SSF is designed to sustain a 30 year mission with first element launch scheduled for March 9, 1995 and assembly complete in 1999. It is managed by four NASA work packages and three international partners. The station is designed to house an 8-person crew. Its mission is to support international scientific research labs investigating physics, material and life sciences and performing astronomical and earth observation. The SSF is also intended to support the Lunar/Mars missions. The entire station design will weigh 320 tons at 250 miles and be 445 feet long with 100 foot solar arrays.

SSF Functionality The computational processing capabilities onboard are designed in a distributed architecture with a hierarchical functional order. The overall station will be maintained at the topmost level by the operations management system OMS, supporting both the operations management applications and the operations management ground applications OMGA. The station's basic operations supported by this are CHeCS, C&T, GN&C, TCS and DMS. Communications onboard the station are through the DMS and to the ground is through TDRSS, at a rate of 300 MB/s. Advanced Automation support onboard is to enable intelligent data gathering, and it must be fast and compact. That onground must entail robust FDIR yet is *not bounded by resources.*

SSF System Design The basic system proposed is as follows: the hardware for the standard data processor SDP is to be a rad-hard VHSIC 80386 at 20MHz with 4MB ram and operating at a 4 MIPS performance level. It is to be built with commercial-of-the-shelf products, have a 100 MB/s core network using FDDI in a dual counter rotating ring architecture. The local network(s) will be enabled by a 1553 bus with performance of 1 MB/s. The mass storage unit is 250MB with an average access time of 250 msec and a data transfer rate at 1MB/sec. The radiation total dose for the platform is 10Krads.

The software is based on the Lynx operating system with an Ada run time environment and event management and application scheduling. The network operating system provides data/application access to remote SDPs. The standard services provides runtime object data base and sensor and effector access. The user support environment provides human computer interface routines and the data storage and retrieval system is for file handling.

Rescoping the System The SSF has evolved through several scrub activities. Cost and size are the motivating factors for the scrubs. In 1989 the DMS scrub goal was 66% power reduction. To achieve this, virtual memory was removed, DBMS was put off (in hibernation), several components/aspects like 802.4, P-MPAC, Bridge, BNIU, low power EDP, gateway and GPS were "deferred", and the TGU was assigned to

be updated from ground. In the 1990 scrub, some of the 25,000 sensors may be eliminated, replacing onboard software to the ground. Weight, power and EVA time (WPE team) will also be focused upon.

Requirements for Growth and Evolution

Using the previously described baseline design as a foundation, evolutionary systems must be added on to provide advanced capabilities. Those of primary concern were indicated in a study by Dave Weeks [Space Station Freedom Automation and Robotics: An Assessment of the Potential For Increased Productivity, Dec. 1989]. Many areas were identified to benefit from use of automation. Those stressed in Mr. Weeks' report were:

Integration of Subsystem Elements and Operations
Instrumentation and Sensor Reconfiguration
Subsystem Status Monitoring and Fault Detection
Onboard Automated Element Test and Verification
Inventory Management Requirements
Onboard Training Needs
Payloads and Payload Operations
EVA/IVA Requirements for Maintenance
Increased Mission EVA/IVA Requirements for SEI Accommodation
Need for Fault Management and Fault Tolerance

Each of these are discussed in a report analyzing the Data Management System (DMS), detailing the currently projected capabilities [see bibliography]. The following section presents some of the evolutionary requirements identified in systems currently being developed to fill these areas.

Case Study: The ECLSS An effect of continuing the DMS scrub activities with the focus of minimizing power, weight and cost, has been to defer much computational processing to the ground. In fact, only those systems and activities that are hard real-time critical have been kept onboard. Recognizing that the station would eventually have automation capabilities onboard, those operating on ground are assured of 'future' placement onboard and that the hooks and scars are in place. One such system that is affected by this is the Environmental Control Life Support System (ECLSS). With the time critical systems onboard, much of the analysis and decisions come from telemetered data and commands via the over-burdened TDRSS to and from Space Station Control Center.

Baseline ECLSS The goal of the ECLS system for the SSF is to utilize previously unused closed loop life support technology to minimize resupply and return logistic penalties. Maintaining respirable atmosphere, potable and hygienic water, and emergency operation are its primary duties. System weight, power and volume constraints have definitely influenced the ECLSS design. The availability of on-orbit computational resources will impact the overall level of automation, including the number of sensors and effectors. Complete closure of the ECLSS loop is optimal, but due to limitations in onboard processing capability has been deferred until later in the SSFP lifetime.

Functionally, the ECLSS is standalone, although it requires services from supporting systems, such as electrical power, DMS, thermal control and man systems. All supporting systems have finite resources available, due to power, processor performance, etc. and the ECLSS must coexist with other Space Station

systems. Man systems is different in that it serves as the interface between ECLS subsystems and the crew.

The ECLSS is designed to operate in a range of automation capability, from a limited autonomous capability providing corrective and preventative maintenance with crew established manual overrides and non-critical fault isolation and redundancy management to full automation. It is designed so that onboard computers perform most of the process control functions, interactions with other systems, fault detection functions, criticality 1 ORU fault isolation and, if the proper manual overrides are inhibited by the crew, the ECLSS can perform emergency reconfiguration or take appropriate response to an emergency, such as suppress a fire. Considering the life- critical nature of the ECLSS, nearly all automated functions that support it are critical.

Designed to be primarily automated on-orbit, the breakdown between ground and space processing has not been completely defined. However the effects of the power, weight and cost enforcements are leaving little choice to the system designers. The currently allocated resources for each subsystem have been reduced so much that only real-time critical functions are to be housed onboard. Basically anything that can be achieved on ground must be. Also this will make it easier for expected upgrades to be made.

The current onboard ECLSS supervisory system design, using soft real-time , utilizes a total of 1 SDP worth of processing power and memory size (4 MIPs/4 meg). That is, although several SDPs are shared among subsystems, the total power and memory capacity total 1 SDP. The hard real-time system is distributed among approximately 20 MDMs and the composite size is 10 to 12 MDMs. The fault sensors here are directly connected to the MDMs. It is only the supervisor section that interfaces with the DMS. This is for information storage, crew updates and further data analysis.

Automated ECLSS The reason for automation is that more can be achieved with less. Although the current design of the ECLSS ensures enough sensors to be in place for critical tasks, there are no leak detection sensors. To compensate for this fault detection capability, the advanced automation program will analyze the characteristics of failures downstream in the model base and then detect the leak origin or near vicinity.

The automated ECLSS consists of rule-based fault-detection algorithms for baseline on-orbit ECLSS regenerative systems, with initial operating configuration (IOC) model-based diagnosing on the ground. The model based diagnosis systems will move to flight software when computer resources on board permit. The design of the ECLSS automation system specifies that the rule-based fault detection algorithms be on-board and complemented with model based diagnosis autonomy in the ECLSS ground support center. The primary obstacle to achieving this is a lack of computational resources on orbit and on the ground. The computational resources of the SSF must be upgraded to allow migration of high fidelity automatic fault diagnosis software on-board. The on-board fault detection algorithm will meet the Ada task interface.

Consistent water drinkability would be assured through automation of the water quality monitor output. This requires real-time processing of chemical and/or microbial analysis in the life support control system. Onboard processing would require fast symbolic and/or neural net processing.

The current lab system for the advanced automation ECLSS is using Sun RISC, porting to Sun 386i and SunOS using KATE, ART/Ada, Lisp and TAE+. The software is a mix of 80% symbolic and 20% numeric code for which the detection is 500KB and the diagnosis is 4MB. On-orbit applications have been limited to

fault detection because the model-based fault diagnosis requires more processing capabilities than are available. The primary effect of this is that reasoning about the faults is deferred to the ground. A large sacrifice in fault tolerance of the system, the turnaround time from detecting a fault to its isolation and recovery, would be delayed by twice the amount of time it takes to transmit over the TDRSS link. In a critical situation such as a gas leak in the Hab or Lab on the respirable air, this extra time could be intolerable.

The major software hooks and hardware scars necessary for evolution to a more autonomous ECLSS have been identified. The advanced subsystem FDIR requires component sensors to be available from the RODB within 1 second, allowing subsystem control loop latency of 5 to 10 seconds. This leaves real-time fault detection and fault preventive reconfiguration to use 3 to 8 seconds, with communication to the subassembly monitoring process taking 2 seconds. Also called for is software process location transparency (dynamic memory allocation). This was explicitly removed from baseline. The automation efficiency would be increased by the use of model-based reasoning tools, like KATE and ART/Ada for early design knowledge capture. It is recommended that these tools be added to the SSF SSE. This model-based reasoning approach to subsystem FDIR would allow minimal use of explicit leak detection sensors by inferring leaks using the baseline process control sensors. The result would be the same yet require less onboard resources.

For the current laboratory system, which is a scaled down version of the initial design, 4 MIPs with 4 MB will be acceptable. By the nature of the task, it is a slow system and it works. The turnaround time on sensor sample analysis is at worst case minutes, so there is enough time to wait for commands from the ground based control center. With increased use of the station, the demands on the system will grow. Also this system does not account for onboard fault handling beyond detection. There is not enough capability available to meet this growth. To support timely inclusion of this automation technology, symbolic processing in a real-time environment, dynamic memory allocation and much more memory are necessary.

Case Study: Power Management and Distribution Another system targeted for evolutionary upgrades with increased automation is the Power Management And Distribution (PMAD) system. The PMAD Automation Evolution project is promoting the operation of a highly autonomous, user-supportive PMAD system for the Space Station Freedom HAB/ LAB modules with a fully integrated user override capability. Automated aspects of the system include immediate power system safing, short term load shedding, limit checking and reporting, redundant load switching, schedule implementation, fault detection, fault isolation, fault diagnosis, scheduling load prioritization, and dynamic scheduling. The critical functions of the PMAD system are implementation of normal operation, FDIR, scheduling and load prioritization. Though initially targeted to support the Station from the ground, it is to be eventually installed onboard.

The current laboratory system uses Common Lisp and Common Lisp Object System (CLOS) on the Symbolics and Motorola 68010 computers. To better reflect the end environment of the SSF, it is being ported to Solbourne 5/501 and Q-max 80386's. The initial objectives of the project can be met with hardware currently used, but as the hardware system is redefined and trimmed to more closely resemble resources available for use with SSF, sacrifices are imminent. More computing power at the Symbolics level is needed. With more than 50K lines of code, of which 45% is symbolic, dynamic memory allocation must be allowed. To meet the current processing requirements, a total of 20 Vax MIPs is required for the Solbourne, 2 MIPs for the 80386's and about 5 MIPs for the Symbolics. Also necessary is 0.5 MB memory at the controllers and 2 to 3 MB at higher levels. If a Symbolics-oriented machine is not available, the capabilities it would give must be provided otherwise a new system design is necessary. It is obvious, during the port of

the system, that the 80386 is not fast enough and the memory allocated is not sufficient. Without the automation onboard, the ramifications here, as in the ECLSS, could be costly.

Advanced Automation Methodology Project The objective of the Advanced Automation Methodology Project (AAMP) is the development of an engineering methodology for advanced automation systems for use in the SSF program guidelines. The two independent systems in this project are the Advanced Automation Network Monitoring System (AANMS) and the Recovery Procedure Selection Application (RPSA). Upon completion, each being integrated into existing SSFP testbeds.

Advanced Automation Network Monitoring System The AANMS involves the development of a network monitoring system capable of intelligent fault monitoring of FDDI-based networks. Designed to provide predictable, hard-deadline scheduling in real-time within the SSF constraints, the total time to respond to any fault is limited to within 1.0 seconds. This system will automatically detect, isolate and diagnose network faults based on data passing on the network. The network monitor is important for telerobotics and docking-critical systems. Therefore it is necessary to identify bottlenecks quickly so that the system can be reconfigured with minimal loss. The AANMS is to also provide a flexible user interface for semi-automated systems. The primary focus here is to prove functionality, so hardware is not currently a prime concern. When the system concept is proven it will be ported to the state-of-the-art system technology.

The driving requirement for porting systems to the SSF is to use COTS. However the technology required to successfully accomplish this job is not available. The current laboratory system is based on a Silicon Graphics experiment with NetVisualizer: conventional network monitoring software, running on their 16 MIPs personal IRIS 4D/25 workstation. This software was able to process only 4% of the tested FDDI traffic. This 4% is 25 times less than would be the required computing power for all of FDDI traffic. Hence, for processing all of the network traffic, a system throughput of 16*25=400MIPs is necessary to ensure no loss of data. Yet this is for data capture and extraction functions only. Processing of this data would require additional capability. This performance is not in the baseline design, nor is it currently identified in the upgrades. The memory requirements for storing and analyzing the data are prohibitive. The FDDI will generate (worst case) 12.5 MB/sec. of data. For data capture this implies 40 MB of RAM required for each 3 seconds. Other functions are projected to require around 24 MB. This implies a minimum of 40+24=64 MB. In order to reach the evolutionary objectives, memory size can grow to require from 128 to 256 MB and up. Each hour of raw data will require 12.5 MB/sec times 3600 sec/hour = 45 GB/hour. FDDI is already too fast for the 386 platform.

The Flight Telerobotic Servicer One presentation at the Workshop was given by Martin Marietta on the Flight Telerobotic Servicer Data Management Processing System (FTS DMPS). Here it was indicated that the prime computational driver is the critical path "around the loop" flow. This system, in simplified terms, consists of the workstation at which the telerobot is controlled and the telerobot itself. The workstation has a hand controller feeding commands to the 80386/387@16MHz system with 256KB RAM. This is linked via a 1553 bus to the telerobot control computer. The originally specified requirements for the FTS was that the throughput be 2.25 MIPS, 2MB memory, 32 Bits floating point, running Ada on a 1553 Bus. The baseline selected 80386 fits these requirements and the mission has not realized sacrifices due to this. However, the FTS program is undergoing rescoping and changes for which the selected SDP cannot fulfill the requirements. These are due to the growth in the around the loop timing, due in part to the increased algorithm complexity, better software implementation definition and reduced SDP performance (3.59 MIPs).

The current plan for evolution is to provide a margin of 50% processor usage and 35% RAM; however this does not address around-the-loop timing. Spare bus nodes are available for additional processors, but parallel processing does not improve around-the-loop timing - the algorithms used here are serial. To keep the compatibility with SSF, the 80486 will be the evolutionary processor, with an expected increase in performance of 2 to 3 times that of the 386. Because of the critical timing requirements, FTS departed from the SSF selection of backplane for the SDP and selected the 1553 Bus. This allows up to 5 cpus attached, whereas the SSF design allowed for 2 maximum.

The DMPS Architecture Study, initiated by GSFC, indicated future requirements of 200Hz around the loop (vs 50Hz) with increased vision processing in line with increased autonomy and graphics simulation for path planning preview and training. These capabilities can not be provided by the 80386. Without a design from the beginning with a faster system, these are not evolutions that can be realized quickly, without sufficient redesign in space. The FTS would probably remain as it is deployed, with little or no autonomy, initially or evolutionary.

It is recommended that a faster computational platform, with increased performance by a factor of 6, and multiple cpu's (at least 4) plus spare slots for growth be used in the baseline design. As previously stated, no sacrifices have yet been realized to fit to the chosen platform but control algorithms, software and hardware reviews are conducted to squeeze out performance. In this project so far, the primary lesson learned is that the performance margin, initially set at 25% for around-the-loop control algorithm growth was too small. This is being compensated for by targeting Ada RTE to bare machine, thus eliminating the operating system overhead. Greater emphasis must be placed on performance margins.

EVA Retriever The EVA Retriever system project currently under development at Johnson Space Center shows promising results. The objective of the EVAR is to demonstrate and develop selected technologies necessary for a free flying robot to autonomously retrieve an object which has accidently separated from the SSF. The computational requirements for this project were as follows. Although a need for high throughput was recognized, the exact requirement was unknown. The system must have expandability to handle presently unknown requirements. It must also allow for high level language programmability, have an available high level development system that is easily adapted to a multi-programmer environment, use low power and allow fault tolerance to be easily incorporated. Based on these requirements, the working system design is a distributed, transputer based central computer with processors in existing hardware unchanged. The subsystems are linked together by the Intel Bitbus serial data system. Some portions are located onboard the robot, others are remotely located. Although a working system has been demonstrated and continues to be built upon, it uses the transputer extensively. This system, yielding 10-plus MIPs performance, affords much flexibility and performance. However, it is not space or flight qualified, and no indications were given that this will be provided for. A common problem shared by many demonstration projects is that their successful demonstration of functionality is based on state-of-the-art ground based hardware that has no comparable spaceborne capabilities.

SSF Evolution Capabilities

The evolution plan for advanced automation is in place and showing progress, with many of the the systems designed to provide some type of FDIR. Three advanced automation projects targeted for implementation on the SSF that contributed to the Computational Requirements Assessment are ECLSS,

PMAD and AANMS. The FTS program also provided computational processing requirements of the baseline system and their plans for including automation. The general response to the assessment from NASA advanced automation programs was that the computational capabilities required to achieve success, as initially defined in their own program, far exceeded the capabilities allocated by the DMS. That is, a majority of the programs either scaled back their end functionality in order to achieve some form of success on the selected computational architecture, or they indicated that they require additional capability - up to 50 MIPS per processor and beyond.

The need for automation capabilities in the SSF is clear. The limitations in the DMS baseline capabilities are cited as the reason for postponing automation to evolutionary upgrades. To enable growth and evolution, the SSF Program must focus on providing a readily extensible hardware and software architecture, beginning with baseline. The continued development and use of flight system automation and ground operations applications focusing on automated status monitoring, fault detection, isolation and recovery using knowledge-based system techniques is necessary. This will be accomplished by providing advanced processors, network architectures and software development tools, which will prevent obsolescence and reduce power consumption and cost. Embedded fault tolerance and system security, improved processing, memory, data storage, and communication network performance would ensure reliability and credibility of the end system. Together, all of this would *reduce training and sustained operations costs*. The evolution of the SSF would be through First Element Launch to Assembly complete to Lunar Vehicle operations to Lunar and Mars operations, and beyond.

LEO Polar (EOS, CSTI)

The differentiating factor between LEO Equatorial and LEO Polar is the mission type. Whereas the SSF is designed for human life, these systems are for science experiments and have no crew. They tend to be the most benign of the off-ground systems to design because the environmental factors are comparable to those of the SSF and their mission functions are communications and scientific return.

The Earth Observing System (EOS)

The EOS is a major component of the NASA Mission to Planet Earth Program, a part of the Global Change research program. By leveraging international participation, the goal is to develop a comprehensive understanding of the Earth as a system. This will be accomplished by deploying two series of three observatories. The A-series is for imaging and sounding while the B-series is for altimetry and atmospheric chemistry. Each of the six systems is designed for a 5 year life-span. The first will be launched in FY 98 with the others following every 2.5 years thereafter. In the A-series, there will be 13 instruments onboard, with an average data rate of 30Mbps and peak data rate of 219 Mbps. The average power will be 2.6kW and peak power is 3.6 KW. The mass is 2845Kg. Series-B also has 13 science instruments with 14Mbps average and peak data rate. The power will also vary with average at 3.2 KW and peak of 4.2 KW. The A-series has 6 instruments with peak rates less then 200Kbps, 5 with between 200Kbps and 20Mbps and only 2 at greater than 20Mbps. The B-series has 11 instruments with less than 200Kbps and 2 up to 20 Mbps.

The science user requirements are simply stated, " We want all of the data - in the raw". This is to ensure that the correct data are archived and the scientists can process to higher level products on the ground. Since they can save the data indefinitely, reprocessing can be performed when improved algorithms are available. They can also generate new products when new algorithms are available. With direct broadcast of the data and direct downlink, they are provided a limited set of data for field use. Due to this contention for the downlink, not all instruments can be used all of the time. This automatically presents a sacrifice in mission return.

The requirements for the flight data system of each satellite are that it at least be tolerant of single hard failure. The primary function is to accept and forward streams of instrument and platform data to the ground, providing 0.5 terabits of storage of housekeeping, engineering and instrument data. The computational platform shall have 0.8 MIPs available for application software and 1 MB memory available. This system is not to exceed 1127 pounds, not exceed 641W, and have a design life equal to or greater than 5 years. The platform must have dual redundant computers and provide the majority of platform processing resources. The selected computer is the 1750A. This meets the computational power, weight, power and radiation requirements. It is space qualified, tolerant to single event upset and has multiple vendors in the marketplace. The 1750A computer is compatible with the 1553B for communication with platform subsystems/instruments. The 1750A is a mature architecture, and no rescoping changes are anticipated. There have been no sacrifices needed due to this selection and no known bottlenecks.

The future application considered for this program is to maintain a direct downlink for the data. This would provide more data to remote users. Although there are no limitations realized in the design, the program managers have indicated that the current instrument data rates exceed portable ground station capabilities to collect it. This has caused the program to reduce the data rate. To compensate for this, plans are being considered for future expansion, possibly using CHRPS.

The objective of CHRPS (Configurable High Rate Processor System) is to provide the architecture, system control and high rate data interfaces to support onboard data processing for space systems. Applications for this are data compression, information extraction and higher level product generation. The benefits foreseen are that downlink data bandwidth would be reduced, the data would be provided to ground receiving stations with modest computing power so that the data could be used readily, and further enable the use of telescience.

Although it was indicated several times that this system of satellites has realized no sacrifice in mission, their design was made to the available hardware following the requirement of no onboard processing. This overall requirement preempts the use of all of the instruments to their full potential, thus sacrificing design to the mission.

Unmanned SEI - Mars Lander, Rover, and Telerobotics

Most of the robotics requirements in this study were provided by engineers working on the JPL telerobotic testbed. Members of this laboratory were responsible for the FTS study on providing advanced computing requirements as a function of capabilities. An additional focus is Mars rover technologies.

In this lab, recognizing that the biggest limiting factor to robotics is processor speed and memory size and acknowledging the absence of good space qualified components, the testbed does not use space hardware. Focusing on the heuristics, the software is designed for space. When completed, the software is put on "fastest possible" available space-qualified hardware.

The focus at JPL is on fully autonomous, on line task planning. The task planning code is currently being moved from a Microvax and Symbolics to Sun4's. This port alone will greatly increase the performance. However, the evolution to the fastest workstations available increases excessive predictions of what can be realistically achieved on the fastest space qualified processors available.

A recent study for the FTS program [see bibliography], reported that the 386/486 processors would provide sufficient processing capability as long as the right software environment is used. A 10-mil cycle time requirement was determined, in excess of this time the operation would not have a true telepresence feel. The requirements presented are for teleoperated systems only. For fully autonomous systems the requirements are much greater and cannot be fulfilled by this selected set of processors. Also, because the FTS is a teleoperated system, there is not a high bandwidth problem. In this mode, the laboratory has determined that the FTS uses 50% capacity of the 386; but as autonomy grows, the remainder will be exhausted quickly. The move toward autonomy implies an exponential growth in the amount of data to interpret, which requires a corresponding upgrading of interpretation mechanism of the data.

Telerobotics

In an interview with Jake Matejevic, the lead systems engineer for telerobotics at JPL, it was indicated that the absence of qualified components is a major difficulty in providing telerobotic capability in space. This lab. has accomplished "reasonable" things with commercial hardware. There is not enough business available for industry to spend so much money on qualifying processors for us, therefore processors are not available. It must also be remembered that after qualification, it is not the same processor. When you design control systems, you have to know your end environment and processor.

In this laboratory, the 68020, VME and "C" all seem to be recurring components that are used in architecture design. This was a transition from initially whole systems, coming from a need for increased flexibility and being able to close the control loop.

Current work focuses on implementing a model matching data set version of worksite for vision in collision avoidance. Planning sites have used Symbolics a ruggedized symbolic processor is not in the foreseeable future. Therefore they used C based in 68020. They would like a single board symbolic processor that would put data acquisition and data processing in same box and therefore have near real-time. Currently, at least two hours of computing are needed to build map and create a plan on the map. This is too slow.

After initially recognizing limitations and what we can expect from current hardware, we should decide what should be shown to the operators on the ground, and then prioritize. All this leads to the question: is what we want to do feasible to accomplish and fund? If so, then put the algorithm, hardware and software ideas together into a complete implementation. Using small teams of engineers focusing on whole systems, the hardware and software are designed concurrently.

It is increasingly obvious to this community that further developments of flight qualified end systems are necessary. The question is, should the development environment necessarily be equivalent to the end product? It is very tempting to develop on advanced workstations using RISC processors, but the results are not always deployable.

The laboratory is based on the 68000 microprocessor family. The successful demonstrations to date have led to a search for more capable processors for Sun4 class and RISC-68040 class. However qualifiability and software environment availability are prime concerns in the selection of computational processors. Another concern in system selection is providing for the experience base in-house. Processor and system selection is often not based on requirements but rather a matter of heritage. Since the 68000 family has historically been used, the new processor(s) selection also must consider the current sizable investment in 68020 family. Because of the portability of current boards and system development, this cannot be ignored.

A Space Control System Designer's Viewpoint In one interview, a standard conflict between engineers and scientists was indicated. As stated, the scientists think that system developers overestimate the computational requirements of robotics and the developers think that the scientists underestimate them. Some scientists contend that if the specified requirements are too high, their project may get cancelled. The processing requirements on each side differ by two orders of magnitude. In order to support the processing needs of the algorithms, especially for real-time, large computational capabilities are required. Yet, the scientists believe that conventional processors are sufficient. They have a difficult time divorcing these computational capabilities from the labs that they are used to working in. Most scientists are used to ground-based processing and know very little about space processors. Even now, the 68020 is not providing enough capability. The 32016 is the best processor available now for space qualified systems development, and it is slotted for use in CRAF/Cassini. Based on the capabilities needed now for what the scientists want to achieve, the RH-32 should have been available two years ago.

Requirements cannot be satisfied as requested, with time frames being the strongest limitations. Given the most aggressive time schedule and funds, it would still be impossible to achieve requested functionality because the computational performance is not available. The incorporation of multiprocessors would increase the performance throughput but be exchanged with other unresolved problems such as the need for synchronization, function anonymity and inherent fault tolerance.

Interconnect technology is the primary limiting factor. A good system is as good as its weakest link, and the interconnect technology is usually the weakest link, particularly in real-time control applications like robotics.

RobotLab2 Another lab was set up for 2D object tracking experiments and large robot manipulation tests. As the lab is expressly set up for scenario definition and testing, no flight equipment is used or considered. The development system used is based on an 8800 cluster host w/ 80-020 processor for controllers. With the focus on teleoperations, the current computational processing capabilities are ok. However, transitioning to autonomous systems is different story. The current set up would not capably provide for the processing requirements.

The hardware entailed in the lab is: data cube: VME card cage, 8 vision processing boards with 68030 host. 1 board is digitizer, 1 threshold image, 1 low pass filter, etc. all run at a frame rate. Bottleneck is the host whenever you have an application without a special board.

This lab has found that the 68030 is good for most processing functions, however a multiple or parallel processor is needed for efficient image processing.

For robot vision (image processing), a special purpose system would be necessary. Frames are processed in real-time as they come in every 1/30 second, and each frame requires .25MB. However, some applications here are so slow that no one wants to code and debug them. Therefore if it runs an application adequately, it can be sped up later.

The biggest processing requirements identified in a recent in-house study of PI's were image processing and it's integration into control systems.

Rover Technologies

Development of rover technology for Lunar and Mars exploration is a difficult task because requirements for this unique scenario do not exist. The overriding functional determining design to is that the rover is to perceive its environment and plan it's path. Every meter travelled requires x amount of processing. For this task, it is estimated that 100 GFlops capability for 60 meters/day would be successful.

A demonstration was made at JPL in May of 1990 in which a laboratory rover ran from dawn till dusk, accomplishing 2 cycles for 10 meters. Reports of what can be achieved by the current testbed vary by a factor of 10, ranging from 5 to 50 meters. This system used 25000 lines of real-time code in C, T (a Lisp derivative), and VX-works.

The computational and data storage requirements for the planetary rover are presented in Volume 1. The planetary rover must sufficiently support computational requirements of onboard navigation activities, which involves large databases, stereo correlation, terrain matching and path planning. Robotic processing includes the real-time command, control and data management of science and engineering subsystems. The summarized requirements presented are detailed by K. Lambert of JPL.[see bibliography] The simplest of scenarios indicated, requiring 0.5 to 2 MIPs capability, is currently pushing the performance limits of available processors. The construction vehicles requirements of 500 MIPs to 500K MIPs are well beyond most processing capabilities of ground-based technology. The goal of relating this information to the available space-qualified processors, will probably never be met, at least not in a timely manner.

Manned SEI

Planet Surface Systems

A Lockheed ESC presentation at the workshop entitled the *Space Exploration Initiative Planet Surface Systems Computation Needs* presented general requirements. From case studies performed for the Office of

Exploration, this was to point out designs for potential Lunar and Mars exploration programs, including expeditions, observatories and outposts. The study produced the recommendation to design for the moon then for the Mars program which would establish a Lunar outpost, followed by a Mars expedition, and then settling a Mars outpost.

The issue of technology needs was identified as an essential part of the feasibility study. Objectives of the outpost will be met by five different areas. Planet Science will perform sample collection, terrain traverses, maintain geophysics station and perform site surveys. Platform Science will focus on astronomy, physics, biomedicine and materials. The Infrastructure consists of habitation, command and control, enabling equipment, transportation node, supply depot and energy. The Learning Center and Test Bed is responsible for the equipment, operations human factors and logistics of the outpost. The Resource Development portion is responsible for propellants, volatiles, metals, ceramics and energy.

The timeframe for establishing the outposts indicates a first piloted flight to the Lunar emplacement (to enable key capabilities and establish initial facilities) to be in September 2000 and the first piloted expedition to Mars in October of 2014 with the first piloted evolution to Mars in 2023. While admitting that now is too early to have a processor selected or an idea of the size of software required, through the definition of the top level requirements, some of these architectural implications are undeniable. Throughout the design process it must be realized that this is to provide a safe haven for crews of 4 to 8 for 6-to-12 month tours. The requirements for the construction vehicles in the previous section can only be the starting point here. With the continued sustenance of human life for long periods of time, coupled with the environmental factors of radiation and remoteness being more stringent than the Space Station, these computational requirements must be accurately identified early in the design phases - to ensure fully functional, capable and reliable systems.

Deep Space (comet, planetary)

One of the primary reasons for NASA's established reputation is the continued success of the planetary spacecraft. A speaker at the workshop, Bob Bunker of JPL, provided the computational evolution of these systems.

Voyager

The first comprehensive survey of the Outer Planets was conducted by Voyager. The Voyager mission was to visit the outer planets: Jupiter and Saturn, and later extended to Uranus and Neptune or Pluto, using gravity assist. This mission took 3.5 years to reach Saturn and 12 years to reach Neptune. This mission involved 11 science instruments and 13 experiments.

Two primary challenges for the mission were indicated in this presentation. Automation involved automated fault tolerance and accurate antenna pointing and spacecraft control. The other challenge was the environment- handling the external radiation of high E electrons and protons, internal radiation of neutrons, micrometeroids, thermal control and externally induced ESD.

The onboard computation for the Voyager was maintained by three computers: the FDS, CCS and AACS, with performance at about 100 Kips (add, sub, shift, etc.), 4K x 18 bit memory, plated wire, discrete SSI and few MSI chips, software all assembly coded. Time and memory margins at launch were 0.

Galileo

The Galileo mission was to probe Jupiter's atmosphere, orbit Jupiter for a thorough survey of the Jovian System, carrying 15 science instruments and 17 experiments. The challenges presented by automation in this project were much more extensive. The one-way light time has an encounter time of 37m. The extended lifetime of the prime mission is about 10 years. Galileo utilizes a dual redundant system involving finer granularity and more options than that of Voyager. Automated fault tolerance was extended and doomsday attitude recovery utilized. Accurate antenna pointing and spacecraft control also had safing of instruments during thruster firings, trim of spacecraft inertial properties during and after burns - wobble, nutation, spin axis alignment. Star referenced inertial position updates, even during turns with no accurate sun knowledge used and compensation of spacecraft disturbances, such as tape recorder, fuel slosh, etc. For accurate platform knowledge and control, the spacecraft had onboard gyro drift compensation, complex gymbal motion displace axes, automatic compensation for non-linearities; continuous spectrum of slew rates, pointing control maintained during slews - active damping at start and end through profile control. Control of dual spin spacecraft was maintained. Onboard knowledge of inertial references was used: target or inertially referenced pointing - EME 50 coordinates. Problems included higher rate slews, faster settling times, more flexible structure, dynamically interacting spacecraft elements-fuel slosh, boom flexures,and rapidly changing mass properties. Problems for the Galileo due to the environment were much the same as Voyager, also cited were launch vibration, shuttle safety and bay environment, contamination control from biprop thrusters.

The onboard computation for Galileo was based on two computers, CDS (merged FDS, CCS) and the AACS. CDS performed at 100 KIPS with a simple ISA. It used the RCA 1802 in a distributed architecture of 5 cpus, with 128 KBytes, and was block redundant, assembly coded in HAL/S. Leadless Ceramic Chips on ceramic two-sided substrates were used which caused problems. The AACS computer at launch had cost for hardware alone >$50M after the program was initially set for $20M. A 1750A class machine, run at 250 KIPS, had a total dose hardness to 50K rads(Si) but all other subsystems have a 150K Rad(Si) requirement, and SEU hardness to >37 Mev/mg/cm^2. The flight code software at launch (cost $25M), used a JPL written operating system (GRACOS) and 10K SLOC in HAL/S and assembly code. Time and memory margins have been negative since 1983.

The major challenges cited in this talk are that **technology is driven by limitations in onboard power**. Because of this, the use of CMOS technology became a requirement where possible. Programs always underestimate requirements of speed, memory, etc. Power requirements and heat dissipation are big factors as are microcode errors, compiler bugs and long term software maintenance. Fault protection strategies were organized in levels: WDT, keep alive, hot spare memory and heartbeat. SRAM failures were given due consideration. Single Event Upsets in 1981 were great major cost drivers: making a $25M computer system end cost more than $50M in 3 to 4 years. Other problems were not identified because there was no worst case analysis performed, it was qualified by margin testing only.

The emphasis from this arena is that it is the control requirements, not the data requirements, that drive computer needs. This is not typically recognized until too late! Speed margins are at least as important as

memory margins but less visible to management: hence management is less willing to expend resources to extend speed margins which ends up being quite costly! Here, the SEU problem surfaces unpredictably because technology has tended to change during the design phase. For reliable systems, projects must stay away from two-sided assemblies. When a system is built to meet in the middle, the middle is typically in a different place and the fit isn't there. Also, flight software is very expensive at $2200/SLOC. NASA requirements for space missions are unique, and NASA should not assume that DoD projects will produce products to meet our needs. When testing, a system cannot be over-tested - keep testing till it breaks and then test some more.

CRAF/Cassini

The near future of on-board computing for planetary spacecraft is in CRAF/Cassini. It is a 3 axis stabilized spacecraft with a 12 year mission lifetime, tolerating 75K Rads(Si) and SEU Hardness to >80 Mev/mg/cm^2. The pointing knowledge and control is similar to Galileo, but enhanced. The key new technology used in this project is KA band transmitter, Fiber Optic Rotation Sensor (FORS), momentum wheels and Common Flight Computer.

The historical perspective of the satellites is that the computers used every 10 years have significantly increased performance from 5 KIPS to 5 MIPs while at the same time decreased the power required, increased density while decreasing real-estate size and decreasing cost.

The trends seen for space systems is that they will have greater autonomy. This is because they will be in areas that are less predictable, they will be afforded more performance, and have onboard more elaborate sensing and control. They will be involved in more hazardous missions and hence require greater fault and damage tolerance. This will also require improved maintainability, with more reliable code updates and graceful accommodation of changes. With the enhanced autonomy and newer areas of exploration, more data will be available. This will require imaging and broad spectrum instruments which require extensive onboard data preprocessing for analysis, selection of specific data and compression before transmission.

The data system of this spacecraft is driven by the science requirements of current missions. Data rates for instruments are the drivers. Scientists want their data -- whole and raw. Data systems need to evolve to perform more space-based signal processing that is easier to process on ground rather than scale down the computer to process in space.

A primary difference cited between the spacecraft technologies and SSF is that here, once deployed, the system is unserviceable, therefore it needs to be self assured of reliability.

Autonomy in this arena carries its own definition. "Autonomy" in the context of this community requires that the system can survive 24 hours with no commands, therefore being in "safe mode" of self preservation.

Scientists set the requirements and wish list - no matter what is offered in capability, they want more: faster, etc. Whatever you come up with, they always want more.

General Issues from the Interviews

The following presents specific issues and concerns identified throughout the survey:

Provisions for a space qualified Lisp processor system drew both support and criticism. Those who support the system banked on the provision of a space qualified Lisp system for their Lisp-based automation applications. However the AI section at JSC stressed that symbolic architecture is neither warranted nor needed and that there would be no success fielding applications on symbolic hardware. This is based on experience in applications written and executed in Lisp, Art and KEE. In this effort, they ran into one dead end after another and hence achieved no fielding. While working closely coupled with the operations group, they disliked the Lisp systems. So they transfered to a conventional language and hardware => CLIPS.

More extensive capabilities are needed for space applications. The projected performance of the space qualified 386 is not enough to satisfy the requirements for reliable automation. The 486 and 68040 performances are getting closer but still not enough. Lisp is esoteric in engineering world. There are inherent problems of using Lisp on conventional architectures.
'AI programmers are undisciplined". Software engineering principles are not (generally) followed.

At JSC, we are not building applications but tools. Core Clips inference engine takes 180K memory which is still questionable for use on the SSF.

In the SSF program, there is almost no automation onboard and a typical application is allocated 512K memory. The current 386 speed is a serious issue. However the biggest issue is the size of RAM at 1 to 4 MB of memory -- we really need 30 to 60 MB RAM (particularly given that no virtual memory is allowed).

Among other concerns, heat and power use were cited, but the biggest is software size down to keep cost down and keep costs associated with the ground.

AI technology in conventional architecture still needs capability of 486 technology (at least) ~20MIPs spaceborne. This would reduce maintenance costs. A well designed AI application may be easier to maintain than conventional languages because the further you go up the spectrum of high order languages, the easier they are to read, maintain, and re-use. Hence a potential is there for reducing operational costs which would be significant. Even if maintenance costs increase, reduce the number of operations support. It is easier to catch mistakes of maintenance rather than operations people. Configuration control is easier to have in maintenance rather than operations.

Plain Jane automation applications FDIR multiple faults, model based reasoning, requires 1 to 1.5 MB memory (AI approach is not deterministic ... like KATE).

ART to Ada - maintain ART version. Ada version faster than Lisp on Vax (big deal) but C version is even faster! Overall throughput is the primary selling point of cpu's and architecture- it is not just languages, etc.

There is no question that we need advanced/more powerful data compression, analysis, control, housekeeping. Our problem is the crews' acceptance of automation i.e.AUTOLAN of Shuttle for fully

autonomous landing, astronauts will not use it. A multi-billion dollar vehicle with 7 astronauts presents too much risk. But Dave Weeks' study showed areas of strength. The astronauts do have too much to handle.

Our on-board processor for SSF is not sufficient.

Find much support for new architecture. SVMS lacked adequate demonstration of efficient execution of Ada, C, FORTRAN. This may be overcome by representative benchmarks in each language.

It must be recognized that the Spaceborne life-cycle is longer than ground based. The motivation for providers will enhance. If the 80486 is available in 5 years and then viable for 5 to 10 years, given its acceptance. The demand for CPU power is not going down.

Advanced processors: common sense to those of us in CS world - we want and need further processing capability.

Shuttle: <1 MIPS, 108K memory to work with. We also run into the old generation ideas of "software... aaaaagh!" To implement parallel processing - consider reliability and maintenance - its a tradeoff.

The majority of maintenance failures are mechanical.

Software cost will increase if capabilities are not scaled up!

NASA finds itself now constantly trying "to put 10 pounds of doodoo in a 5 pound bag"; this is very costly. But if working within a percent range of capability is costly, then increase RAM then need to increase MIPs. Therefore need to increase throughput and memory simultaneously.

General Issues Raised in the Workshop

Too often a program specifies a design instead of functionality. NASA needs to do a better job of specifying parameters and requirements, and let the engineers do the design. This would give us better ability to track how the design should change as specifications change.

How can we prevent the cost of software from being the limiting factor for processor upgrade for manned spacecraft. *Suggestions*: Develop an intermediate code such as p-code.

Access to real data over a long period of time rather than "simulated data" or a sample of real data for a short burst like "ten minutes on a tape" is critical to gaining confidence in a new software by an operations staff. Controllers fail to be impressed by a system that works beautifully on the problem they encountered last month, and rightfully so. *Suggestions*: Run old data and system side-by-side with old data and new system to verify performance and establish feasibility on standard benchmarks or scenarios. Run old and new system side-by-side with capability to toggle "new" back to "old", if something looks weird to controllers or pilots observing, for at least several mission critical events. This may take days to weeks, depending on the context. Retire old system only after share out has proven that new system is robust on an interesting class of mission critical events. Increased cost for new-technology-injection SW testing and delivery. Overlaps of "old" and "new" systems will find bugs before they are fatal.

Average time from freeze technology to launch is 7 yrs. Therefore have "concept" for any long lifetime system. Autonomy is a key!

We need a common indication(s) of performance for the space computing community. It must also include memory access as well as processor speed.

Microscopic benchmarks are acknowledged as inadequate. System/ subsystem-level benchmarks are required. What would be the attributes of a good system-level benchmark? What good candidates exist?

Software costs are dominating total system costs. Are software costs inversely proportional to the speed/size of the computer? I.e., is it less work to write code for better computers or those in which you are not computer-bound? Can this be quantified?

Recommended Follow-on Activities

1) refining the user requirements in a more uniform specification. This assessment identified many concerns that are shared among various programs. In the refining process, these common concerns are expressly addressed and further quantified.

2) The issue of non-existent or non-responsive benchmarks relevant to NASA applications has been raised. Benchmarks are key to assessing the extent requirements are met. This issue is being addressed.

3) communication of common computational processing requirements: among NASA projects as well as to Industry is not solidly established. Methods of enabling broader, more efficient information sharing are being addressed.

REFERENCE LIST AND SUGGESTED READINGS

Advanced Automation Network Monitoring System Concept Document Revision 5.0
--Lockheed Engineering and Sciences Company; LESC-27811, April 1990.

Advanced Automation Testbeds Functional Requirements Definition
--MITRE Corporation; Steven E. Bayer, et al., Dec. 1988.

A Review of Space Station Freedom Program Capabilities for the Development and Application of Advanced Automation
--MITRE Corporation; MTR-88 D00059, Dec. 1989.

Assessment Of Technology Alternative for Telecommunications, Navigation, and Information Management for Lunar and Planetary Exploration Based on 1989 Case Studies
--Denise S. Ponchak, John E. Zuzek, Kenwyn J. Long;
Lewis Research Center; Oct. 1989.
Final Report
--Dr. Edward F. Miller, Denise S. Ponchak, John E. Zuzek, &
Kenwyn J. Long; Lewis Research Center

Automated Electric Power Management and Control for Space Station Freedom
--Pamela A. Mellor, James L. Dolce; Lewis Research Center
Joseph C. Krupp; Decision Science Applications, Inc.; NASA TM 103151, Aug. 1990.

Automating Security Monitoring and Analysis for Space Station Freedom's Electric Power System
--James L. Dolce; Lewis Research Center
Dejan J. Sobajic & Yoh-Han Pao; Case Western Reserve University;
NASA TM 103148, Aug. 1990.

Automation & Robotics Human Performance
--Robert W. Mah, Ph.D.; Special Assessment Agent, Advanced Missions
Technology Branch, Information Sciences Division, NASA-Ames
Research Center; Aug. 1990.

Automation and Robotics Report- January 1990
--McDonnell Douglas Space Systems Company
Space Station Division; MDC H4659.

Automation and Robotics Report- May 1990
--McDonnell Douglas Space Systems Company
Space Station Division; MDC H6409.

Automation of the Environmental Control and Life Support System
--Brandon S. Dewberry; Space Station Evolution Symposium, Feb. 1990.

Autonomous Power Expert System
--Jerry L. Walters, Edward J. Petrick, Mary Ellen Roth, and
Long Van Truong; Lewis Research Center
Todd Quinn and Walter M. Krawczonek; Sverdrup Technology, Inc.;
Goddard Converence on Space Applications of Artificial Intelligence, May 1990.

Design, Development, Integration: Space Shuttle Primary Flight Software System
--William A. Madden and Kyle Y. Rone

Dynamic Function Process ..Simulation.. Space Station Freedom
--Engineer: J. Dolce; Designer: R. Overy

Earth Sciences Requirements for the Information Sciences Experiment System
--Edited by: D.E. Bowker, S. J. Katzberg, and R.G. Wilson
Langley Research Center

Electric Power Scheduling: A Distributed Problem- Solving Approach
--Pamela A. Mellor and James L. Dolce; Lewis Research Center
Joseph C. Krupp; Decision Science Applications, Inc.; NASA TM 103149, Aug. 1990.

Engineering Analysis for Assembly and Checkout of Space Transportation Vehicles in Orbit
--Boeing Aerospace Company; D615-11900, Feb. 1989.

Event- Logic Network Power System Automation Advanced Development Block Diagram Space Station Freedom
--Engineer: J. Dolce; Designer: R. Overy

FTS Requirements for Data Management and Processing
--Dr. Michael Ring; Advanced Technology and Research Corporation; May 1990.

Information Sciences Experiment System (ISES)
--Langley Research Center
August 10, 1988 and July 21, 1989

Manned Mars Mission On-Orbit Operations FTS Capabilities Assessment: Final Report
--Frank G. Gallo and Stewart W. Jackson: Robotics and Servicing Systems Group- Fairchild Space Company; June 1989.

Multi-path Redundant Avionics Suite (MPRAS) System/Subsystem Requirements Document
--The Boeing Company; D180-30579-2, May 1989.

Planetary Rover Computational & Data Storage Requirements Version 1.3
--Ken Lambert, Romney Katti, and Robert M. Manning
Jet Propulsion Laboratory; JPL Doc D-6948, Dec. 1989.

Research On Advanced Engineering Software For In-Space Assembly and the Manned Mars Spacecraft: Final Report to the Office of Exploration
--Rockwell International Corporation; Space Transportation Systems Division; Expert Systems Application Group; June 1989.

Robotic Lunar Surface Operations: Engineering Analysis for the Design, Emplacement, Checkout and Performance of Robotic Lunar Surface Systems
--Boeing Aerospace and Electronics D 615-11901, Jan. 1990.

Space-Borne Computing for the Year 2000 and Beyond
--Ravi K. Iyer and Prith Banerjee; UILU-ENG-88-2265, Dec. 1988.

Space Station Automation of Common Module Power Management Distribution
--Martin Marietta Aerospace Denver Astronautics Group; MCR-89-516, Feb. 1989.

Space Station Freedom Automation and Robotics: An Assessment of the Potential for Increased Productivity
-- David J. Weeks, Sponsoring Organization: Advanced Development Program Strategic Plans and Programs Division; Office of Space Station- NASA Headquarters; Dec. 1989.

Space Station Freedom Program Advanced Automation; Volumes I, II, & III
--MITRE Corporation; MTR 89-W00271-01, Dec. 1989.

Space Station Freedom Program Capabilities for the Development and Application of Advanced Automation
--Steven E. Bayer; MITRE Corporation; Dec. 1989.

Study Reports for Space Station Technology Development Mission Requirement Definition for Advanced Automation and Robotics-- A Reproduced Copy of N88-70310
--Boeing Aerospace Company
reproduced by: NASA Scientific and Technical Information Facility; D180-30636-1, Nov. 1987.

Telerobot Response Requirements (for the FTS telerobot)
--Thurston L. Brooks; STX Corporation; STX/ROB/90-03, March 1990.

The Space Station Assembly Phase: Flight Telerobotic Service Feasibility ; Volume I & II
--Jeffrey H. Smith, Max A. Gyamfi, Kent Volkmer, and Wayne F. Zimmerman; Jet Propulsion Laboratory; Sept. 1987.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-103860		

4. Title and Subtitle	5. Report Date
Computational Needs Survey of NASA Automation and Robotics Missions Volume 2: Appendixes	May 1991
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Gloria J. Davis	A-91093
	10. Work Unit No. 549-03-61

9. Performing Organization Name and Address	11. Contract or Grant No.
Ames Research Center Moffett Field, CA 94035-1000	
	13. Type of Report and Period Covered Technical Memorandum

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
National Aeronautics and Space Administration Washington, DC 20546-0001	

15. Supplementary Notes

Point of Contact: Gloria J. Davis, Ames Research Center, MS 244-4, Moffett Field, CA 94035-1000
(415) 604-4858 or FTS 464-4858

16. Abstract NASA's operational use of advanced processor technology in space systems lags behind its commercial development by more than eight years. One of the factors contributing to this is the fact that mission computing requirements are frequently unknown, unstated, misrepresented, or simply not available in a timely manner. NASA must provide clear common requirements to make better use of available technology, to cut development lead time on deployable architectures, and to increase the utilization of new technology.

This paper provides NASA, industry and academic communities with a preliminary set of advanced mission computational processing requirements of automation and robotics (A&R) systems. The results were obtained in an assessment of the computational needs of current projects throughout NASA. The high percent of responses indicated a general need for enhanced computational capabilities beyond the currently available 80386 and 68020 processor technology. Because of the need for faster processors and more memory, 90% of the polled automation projects have reduced or will reduce the scope of their implemented capabilities. The requirements are presented with respect to their targeted environment, identifying the applications required, system performance levels necessary to support them, and the degree to which they are met with typical programmatic constraints. Volume 1 includes the survey and results. Volume 2 contains the Appendixes.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Automation, Computational requirements, Automation and robotics, Robotics, Aeronautics, Space science	Unclassified-Unlimited Subject Category - 31

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	79	A05

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