

**MINUTES OF THE
39TH COMPUTER RESOURCES INTEGRATION
MANAGEMENT
MEETING**

12 March 1991

19980302 079

BMDOTIC

DTIC QUALITY INSPECTED

DISTRIBUTION STATEMENT A

**Approved for public release
Distribution Unlimited**



TELEDYNE

BROWN ENGINEERING

Cummings Research Park • Huntsville, Alabama 35807

U2909

Accession Number: 2909
Title: Minutes of the 39th Computer
Resources Integration Management
Meeting, 12 March 1991
Contract Number: DASG60-87-C-0042
Corporate Author or Publisher: Teledyne Brown Engineering,
Huntsville, AL 35807
Publication Date: Mar 14, 1991
Pages: 00150
Comments on Document: Contains copies of briefing
presentations.
Descriptors, Keywords: SDS Computer Resource Management
Productivity Ada Software SEIC Reuse
NTF NTB L2SS PROTO+ MICOM ISSI SDDS
RATIONAL Model Simulation

**MINUTES OF THE
39TH COMPUTER RESOURCES INTEGRATION
MANAGEMENT MEETING
12 March 1991**

PREPARED FOR:

**UNITED STATES ARMY
STRATEGIC DEFENSE COMMAND**

CONTRACT: DASG60-87C-0042

PREPARED BY:

**TELEDYNE BROWN ENGINEERING
CUMMINGS RESEARCH PARK
HUNTSVILLE, ALABAMA 35807**

PREPARED ON: 14 March 1991

Minutes of the 39th CRIM

1. The 39th CRIM was held 12 March 1991 at 1300 hours in Room 2D1100. The theme for this CRIM was Software Productivity, with presenters from both the government and the private sector about lessons learned and Ada development tools to enhance productivity. The meeting began with Mr. Frank Poslajko presenting the agenda. Next, Mr. Poslajko presented a slide depicting various areas of software productivity. In addition, a fact slide was presented on the Ada Fundamentals Workshop scheduled to begin March 25.

2. Mr. Michael Walker presented the SEIC Software Activity Status. Mr. Walker discussed the ongoing activities related to the documents SEIC is responsible for developing. A discussion occurred about distribution of SEIC documents with a general consensus of the need for broader distribution in the USASDC community. Mr. Walker presented an update of the Trusted Software Development Methodology with a recommendation for its' incorporation into the Software Engineering Support Environment (SESE) System Specification. Next, he provided insight into projects under development in the field of Trusted Software.

3. Mr. Terry Gill of Carnegie Mellon University presented highlights of the Software Reuse Program for the SDIO Software Engineering Integration Center (SSEIC) at the National Test Bed. Mr. Gill touched upon the major topics (i.e., determining SDS software reuse requirements, establishing a concept of operation) for the Software Reuse Program. Mr. Gill presented an organizational chart of how SSEIC views the Level 2 System Simulation (L2SS) Software Reuse Organization. Mr. Gill tentatively identified the top level players, but he requested USASDC provide inputs for the element level reuse experts. In addition, Mr. Gill presented the basic goals of the reuse program.

4. Mr. Bruce Lewis of the Software Engineering Directorate, MICOM, discussed the Ada Policy as it is applied at MICOM. Mr. Lewis presented the mission of the MICOM Life Cycle Software Engineering Center (LCSE) and a list of the projects which are under development using Ada almost exclusively. Mr. Lewis displayed some of the "Key" Ada policy issues, and discussed certain ambiguities which need to be addressed with the contractors. Mr. Lewis presented lessons learned in getting the contractor to try Ada for the first time. Finally, he presented several slides displaying the productivity of the Ada language in association with good software engineering practices.

5. Ms. Elizabeth S. Kean of Rome Laboratory presented the Aspects of Reusability in PROTO+. This tool is being developed by International Software Systems, Inc (ISSI). Ms. Kean discussed the system technology and the two levels of reuse foreseen in the system (e.g., low level components, major functional component with subcomponents). In addition, Ms. Kean presented the major characteristics of the PROTO+ tool. She discussed how the PROTO+ tool supports reuse, and what additions would be added in the future. At present, the PROTO+ tool is virtually identical to SDDS (refer to para. 6), but the PROTO+ tool will deviate in the future. Finally, Ms. Kean walked through a demonstration of the tool.

6. Ms. Kathryn H. Hiles of Teledyne Brown Engineering, representing Jackie Cristina from BM/C³ Technology Branch, presented the SDS Common Framework. Ms. Hiles discussed the major differences in the SDS common framework and the Strategic Defense Development System (SDDS) tool now under development by ISSI. Ms. Hiles presented a slide with the eight basic requirements of the common framework, and then discussed the full life support cycle of the framework. In addition, Ms. Hiles presented the current status of the SDDS tool and its current

ability to only support four of the eight requirements. Ms. Hiles presented several slide showing the areas of SDDS which would have to be emphasized to allow conformance to the SDS common framework and explained ISSI's contract was being redirected to expand these areas.

7. Mr. Tom Matson of Rational presented an overview of the Rational system and some of the projects on which Rational has proven its capabilities. Mr. Matson discussed the challenges facing the developers of extremely large Ada systems and the goals Rational had for assisting in the development environment. Mr. Matson presented the main goals as automating life cycle activities, improving quality and reliability, shortening schedules, and helping the transition to Ada. In addition, Mr. Matson discussed the design facilities within the Rational environment. Mr. Glenn Hughes II concluded the presentation with a look at two examples of the benefits of the Rational design environment.

8. Mr. Paul Larson of Integrated Systems Inc. presented the ISI tools for modeling, simulation, code generation, and real-time testing. Mr. Larson discussed the conventional real-time software/control system development loop, and the basic problem of the inability to detect errors until the integration and testing of the software. Mr. Larson presented a slide on the ability of prototyping to detect errors earlier in the design, but this approach still involved looping back to make changes to the design. Finally, Mr. Larson presented ISI's software and hardware tools and explained how they could be used to avoid retracing steps in the design of a system.

9. The meeting was adjourned by 1505. The 40th CRIM is scheduled for 9 April 1991.

39TH COMPUTER RESOURCES INTEGRATION MEETING
12 March 1991
LIST OF ATTENDEES

<u>NAME</u>	<u>ORGANIZATION</u>	<u>TELEPHONE</u>
Dr. Davies	CSSD-TD	895-3520
Frank Poslajko	CSSD-SP	955-1995
Pete Cerny	CSSD-SP	955-3069
Ted Allen	TBE	726-1285
Ben Herrin	DRC	(703)521-3812 x6034
Terry Gill	CMU/NTB	(719)380-2465
Robert Ellis	TBE	726-2748
Liz Kean	Rome Laboratory	(315)330-2762
Rachel Ramey	CSSD-CA-S	955-3124
Tom Matson	Rational	(813)885-8955
Kevin Haar	Rational	(314)428-8640
Glenn Hughes II	Rational	(301)897-4024
Bill Burrows	SFAE-SD-GBR-E	955-5877
Bill Shelton	CSSD-CA	955-3612
Lance Smind	Rational	351-2715
Henry Kunkel	Boeing	464-4437
Michael Walker	SEIC/GE-HSV	355-8086
Paul Larson	Integrated Systems	(408)980-1500
LTC Steve Rice	SLKT	955-3643
Sandra Brazelton	GBR-I	955-5798
Reggie Anderson	GBR-E	955-5981
Tom Nuttall	CSSD-TE-P	955-3909
Dr. Ron Green	SFAE-SD-GST-D	722-1844
Dr. Virginia Kobler	CSSD-SA-BT	955-3857
Kathy Hiles	TBE	726-2350

39th Computer Resource Integration Management (CRIM) Meeting
Action Items

- | | |
|---|----------------------------|
| 1. Provide a status update on the Software Organization and Development at NTF. | John Hawk - ext 3920 |
| 2. Schedule status briefing on SDS committees to include purpose, accomplishments, plans, and schedules. | Frank Poslajko - ext 1995 |
| 3. Establish a Data Reduction Planning Committee. | Barbara Rogers - 722-1518 |
| 4. Disc 4 to coordinate with Ada 9X project office on Ada language deficiencies. | Bob Johnson - AV 227-0259 |
| 5. ADCCS project office to report on the number of Ada waiver requests submitted to DISC 4. | Denise Jones - 895-3397 |
| 6. Discuss broadening the distribution of SEIC documents to the USASDC community. | Frank Poslajko - ext 1995 |
| 7. Provide location of controlled experiment for Trusted Software Case Study. | Terry Gill - (719)380-2465 |
| 8. Arrange demonstration of ISI tools. | Frank Poslajko - ext 1995 |
| 9. Addition of Trusted Software Requirements to Software Engineering Support Environment (SESE) System Specification. | Frank Poslajko - ext 1995 |

PRESENTER: FRANK POSLAJKO

- 1) AGENDA**
- 2) ACTION ITEMS**



39th CRIM



12 Mar 91

COMPUTER

RESOURCE

INTEGRATION

MEETING

THEME: SOFTWARE PRODUCTIVITY

Agenda **39th Computer Resources Integration Management Meeting** **12 March 1991** **Conference Room 2D700, 0810-1500**

0810-0820	Introduction	Frank Poslajko - ext. 1995
0820-0830	SEIC Software Activity Status	Mike Walker/Terry Starr 883-1170
0830-0855	NTB Software Development Highlights	CPT Andrews/T. Gill 719-380-2465
0855-0920	MICOM Software Center Support	D. Copeland/B. Lewis 876-3931
0920-0955	RADC System Definition, Requirements, And Specification Reuse Tool	Liz Kean AUTOVON 587-2762
0955-1010	SDS Software Engineering Support Environment (SESE) System Specification.	Jackie Cristina - ext. 1337
1010-1020	Break	
1020-1100	Rational Software Support Environment	Tom Matson - 813-885-8955
1100-1145	ISI Ada Development Environment for Boeing on Freedom Space Station	Paul Larson 408-980-1500 x239
1145-1200	Ada Implementation Plan	Frank Poslajko - ext. 1995

Agenda

39th Computer Resources Integration Management Meeting

12 March 1991

Conference Room 2D1100, 0815-1500

1300-1315	Action Items	Frank Poslajko - ext. 1995
1315-1325	SEIC Software Activity Status	Mike Walker - 883-1170
1325-1340	NTB Software Development Highlights	CPT Andrews/T. Gill 719-380-2465
1340-1355	MICOM Software Center Support	D. Copeland/B. Lewis 876-3931
1355-1405	SDS Software Engineering Support Environment (SESE) Development Status	Jackie Cristina - ext. 1337
1405-1420	RADC System Definition, Requirements, and Specification Reuse Tool	Liz Kean AUTOVON 587-2762
1420-1440	Rational Software Support Environment	Tom Matson - 813-885-8955
1440-1500	ISI Ada Development Environment for Boeing on Freedom Space Station	Paul Larson 408-980-1500 x239

Software Productivity

Repositories
RAPID
IBM/SAIC
KDEC
ELI/ARC Reuse
CECOM/SPS 1st Release Sept 91
COSMIC
SDITIC
NESC
SW Components
EVB-GRACE Comp
CAMP
BOOCH
MATH MODELS

Training

- DCDS
- Ada Workshops
- Simulations
- BRAT
- UAH
- SEI
- AIDT

Software Development Environments

- DCDS
- SDDS (Interface Framework)
- RATIONAL - Freedom Space Station
- ISI Ada Development
- Ada A_Z - TASC

SW Tools

- CDRL Tracker
- C2 DATE
- LOGICAN-2167A
Tailoring Tool
- COCOMO

SW Engr Practice

- Ada Language
- Portability
- REUSE
- Tailoring (2167A)
- Prototyping
- Open Architectures

Quality

- Testing
(Quality Factor)
- Trusted SW
- REUSE

39TH Computer Resource Integration Meeting (CRIM)

Action Items

- 1. Provide a status update on the Software Organization and Development at NTF.** **John Hawk - ext 3920**
- 2. Schedule status briefings on SDS committees to include purpose, accomplishments, plans, and schedules.** **Frank Poslajko ext 1995**
- 3. Establish a Data Reduction Planning Committee.** **Barbara Rogers 722-1518**
- 4. Require higher management signature on notification of software policy compliance for solicitations and contracts.** **Frank Poslajko ext 1995**
- 5. DISC 4 to coordinate with Ada 9X project office on Ada language deficiencies.** **Bob Johnson AV227-0259**
- 6. ADCCS project office to report on the number of Ada waiver requests submitted to DISC 4.** **Denise Jones 895-3397**



Review of USASDC SW Policy / Notification of Compliance

CSSD- (715e)

SUBJECT: Review of the U. S. Army Strategic Defense Command (USASDC) Software Policy;
Notification of Compliance for Solicitations and Contracts

FOR CSSD-CM

TITLE OF CONTRACT REQUIREMENT PACKAGES:

I have evaluated the above Contract Requirements Package (CRP) and it is my judgment that:

A. _____ Requirements are adequate and in compliance with U.S. Army Strategic Defense Command (USASDC) Software Policy; Therefore, a waiver will not be provided with Contract Requirements Package (CRP).

B. _____ Requirements are not in compliance with USASDC Software Policy and a waiver is being provided. (Submit a copy of the waiver to Software Engineering Division.)

PE Manager / Project Manager / Office Chief

Date

2nd Ada Fundamentals Workshop

- **Starts 25 Mar 91**
- **8 hours/week for ten weeks**
 - **Mondays 4 hours classroom**
 - **Session makeup through video program**
 - **Fridays 4 hours on VAX terminals solving problems**
 - **Provide flexibility to access terminal other times**
 - **PC modem access**
- **16 new participants selected**
- **Previous workshop participants can makeup missed sessions to obtain certificate (12 People)**
- **Each classroom session to be video taped by IMO services**
- **Certificates to be awarded based on performance**
- **Goal is to have majority of attendees awarded certificates**

PRESENTER: MR R. M. WALKER

SEIC SOFTWARE ACTIVITY STATUS



**SEIC SOFTWARE ACTIVITY STATUS
FOR COMPUTER RESOURCE
INTEGRATION MEETING (CRIM)
MARCH 12, 1991**

DR. R.M. WALKER



SOFTWARE ENGINEERING

- SOFTWARE ENGINEERING SUPPORT ENVIRONMENT (SESE) SYSTEM SPECIFICATION (CDRL A096)
 - COORDINATION DRAFT ISSUED JANUARY 31, USASDC COMMENTS RECEIVED
 - REWRITTEN AND RESTRUCTURED
 - INCORPORATES TECHNICAL APPROACH DEVELOPED BY SDS CRWG SOFTWARE ENGINEERING ENVIRONMENT (SEE) COMMITTEE
 - COMMENTS INTEGRATION/ISSUE RESOLUTION AT SDIO SEE COMMITTEE MEETING MARCH 11 & 12
 - CONTROLLED VERSION PLANNED RELEASE MAY 17
- SDS SOFTWARE STANDARDS, PRACTICES AND CONVENTIONS (CDRL A095)
 - PREVIOUS COMMENTS INCORPORATED
 - REISSUED JANUARY 17
 - REDLINED DOCUMENT TO BE REVIEWED AT SDS CRWG STANDARDS COMMITTEE MEETING APRIL 5
 - CONTROLLED VERSION PLANNED RELEASE MAY 24



SOFTWARE ENGINEERING

- SDS SOFTWARE DEVELOPMENT ROLES AND RESPONSIBILITIES (CDRL A139)
 - POC DRAFT PLANNED RELEASE JUNE 14
 - CONTROLLED VERSION SEPTEMBER 30
 - CONTENTS TO BE INCORPORATED INTO SDS CRLCMP
- COMPUTER RESOURCE LIFE CYCLE MANAGEMENT PLANS (SDS, ARMY & AIR FORCE)
 - SDS CRLCMP (CDRL A057) WILL BE UPDATED AND ISSUED FOR CRWG REVIEW JULY 15
 - ARMY AND AIR FORCE CRLCMPs (CDRLs A058, A149) WILL BE UPDATED AND ISSUED SEPTEMBER 30



TRUSTED SOFTWARE DEVELOPMENT METHODOLOGY

- **TRUSTED SOFTWARE DEVELOPMENT METHODOLOGY (CDRL A075)-
PRINCIPLES, COMPLIANCE CRITERIA, AND CRITERIA CLASSES BEING
REFINED**
 - **INCLUDES A MODEL SOFTWARE DEVELOPMENT PLAN AND TRUST METHODOLOGY
TEMPLATES TO IMPLEMENT PROCEDURES**
 - **EXTENDED TO ALL 2167A SOFTWARE DEVELOPMENT AND ACQUISITION LIFE CYCLE
PHASES**
 - **INCLUDES A MODIFIED SOFTWARE LIFE CYCLE (PROTOTYPING)**
- **RECOMMENDATIONS FOR TRUSTED SOFTWARE DEVELOPMENT
METHODOLOGY INCORPORATION INTO THE SESE SPECIFICATION UNDER
DEVELOPMENT**
- **SECURITY TRAINING COURSE FOR SOFTWARE SECURITY UNDER
DEVELOPMENT**
- **EVALUATING SOFTWARE ENGINEERING INSTITUTES (SEI) METHODOLOGY TO
ASSESS SOFTWARE DEVELOPERS MATURITY, FOR APPLICATION TO TRUST
DEVELOPMENT**



TRUSTED SOFTWARE DEVELOPMENT METHODOLOGY

- **TRUSTED SOFTWARE CASE STUDY UNDERWAY**
 - **ANALYTICAL IMPACT ASSESSMENT AND TECHNICAL APPROACH DEFINED TO QUANTIFY EFFECTS AND RESULTS OF USING TRUST METHODOLOGIES**
 - **EXPERIMENT AND DEVELOPMENT PLANS FOR CONTROLLED EXPERIMENT BEING PREPARED FOR REVIEW MARCH 21**

PRESENTER: TERRY GILL

**NATIONAL TEST BED
SDIO SOFTWARE ENGINEERING INTEGRATION CENTER
(SSEIC)**

**National Test Bed
(NTB)
Strategic Defense Initiative Organization
(SDIO)**

**Software Engineering Integration Center
(SSEIC)**



A division of Carnegie Mellon University

Unclassified



Software Engineering Integration Center(SSEIC)

- Purpose
- Status
- Programs
 - Software Integration and Test Facility(SITF)
 - Software Productivity Tools Program
 - Software Reuse Program
 - Software Measuring/Monitoring/Costing Program



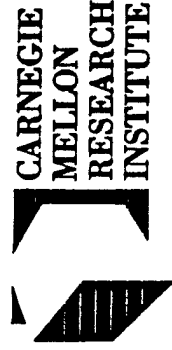
A division of Carnegie Mellon University

Unclassified



SSEIC PURPOSE

1. Provide an SDI integration and test laboratory which uses a framework that can integrate simulation software, operational software, or actual hardware for SDS integration testing and/or software/hardware evaluation.
2. Evaluate software development productivity tools for SDI-wide use.
3. Implement a distributed software reuse program for SDI-wide use.
4. Assist in developing and implementing a SDI-wide software quality evaluation and metrics program.
5. Evaluate programs during L2SS where applicable(i.e. Reuse and Metrics).



A division of Carnegie Mellon University

Unclassified



SSEIC STATUS

- NTB provided SDIO program direction through DEM/VAL topics(formally Key Goal #15).
- NTB system specification and statement of work(SOW) rewritten to accommodate SDIO direction.
- NTBIC response to new NTB system spec and SOW being evaluated by NTBJPO.
- New NTB system spec and SOW scheduled for finalization/implementation in April 91.

NTBJPO currently seeking additional government slots to fulfil forecasted manning requirements.



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

SOFTWARE INTEGRATION AND TEST FACILITY(SITF)

- Define SDS integration and test requirements.
- Develop SDS software integration and test standards based upon requirements.
- Develop a SITF concept of operations.
- Initiate buildout of new NTB building(circa 4th quarter FY91).



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

SOFTWARE PRODUCTIVITY TOOLS PROGRAM

- Previously implemented as the NTB technical insertion laboratory(TIL) but unfunded currently for this fiscal year.
- New SSEIC program will focus on productivity tools for SDI-wide application and provide a laboratory for investigating productivity tools.
- Program goals include determining SDI development tool requirements, evaluating the tools, and recommending tool implementation for SDI-wide use, if applicable.



A division of Carnegie Mellon University

Unclassified

NTB
NATIONAL TEST BED

SOFTWARE MEASURING, MONITORING, & COSTING

- Continue work begun under former NTB software center to validate a set of metrics for SDI-wide application.
- SDS software quality evaluation program being established through the SDS CRWG Software Quality Evaluation(SQE) Committee.
- SQE committee goals include developing a SDS QEP Plan(SQEP) and validating a set of metrics and metric tools using the SDS Software Metrics Evaluation Plan(SMEP).
- SDS SQE Committee consists of key SSD(Lt Dale Brown), SDC(Mr Tom Nuttall), and NTB(Capt Emily Andrew) personnel.



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

SOFTWARE REUSE PROGRAM

- Determine SDS software reuse requirements
- Establish a concept of operations.
- Start building a trusted repository of small parts initially and more generalized(abstract) parts later.
- Provide connectivity through RAS sites.
- Implement a prototype program on L2SS.

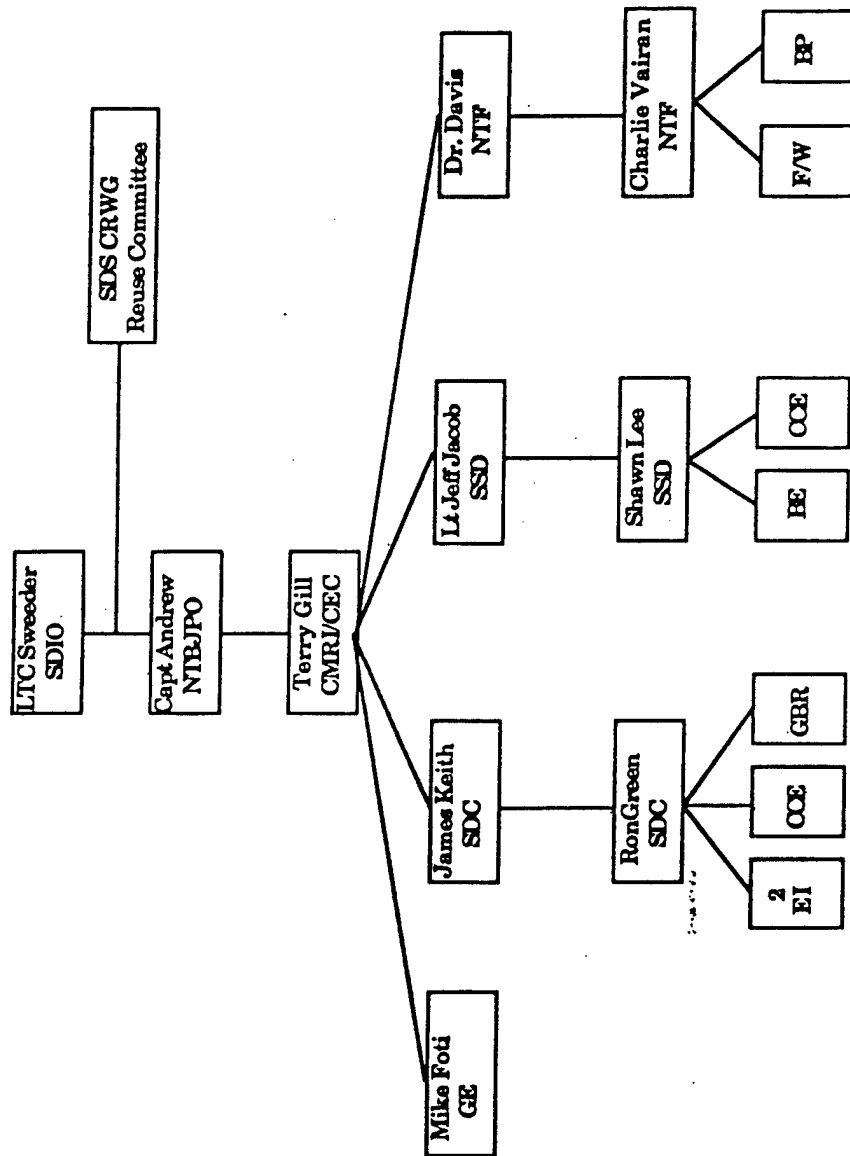


A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

L2SS Software Reuse Organization



L2SS Software Reuse Goals

- Open Communication Channels to Primes & Elements
- Minimally Impact L2SS Build 1 Schedule
- Identify/Secure Common L2SS 'Small' Parts
- Identify/Construct Common L2SS Generalizations
- Provide Initial Reuse Training/Education
- Implement L2SS Reuse Concept of Operations
- Install/Populate SAIC/STARS Repository at NTB
- Survey/Evaluation Software Reuse Repository Architectures
- Establish RAS Connectivity & Distributed Libraries
- Demonstrate the Viability of Reuse
- Report Build 1 Lessons-Learned



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

Open Communication Channels to Prime & Element Contractors

- Interact during Requirements Analysis Phase
- Interact during Design Phases
- Interact during Coding and Testing Phases



A division of Carnegie Mellon University

Unclassified



Identify/Secure Common L2SS 'Small' Parts

- Standard Constants(e.g. gravity, earth,..)
- Standard Math Library and Utilities(e.g. NAG, KESS)
- Standard ADT's(e.g. GRACE, Booch)



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

Domain Analysis for L2SS Build 1

- Identify Existing SDS Domain Studies
 - System Simulator
 - Command Center
 - Communications
- Consolidate Existing Studies
- Develop Initial Cataloging System for SDS Reuse Repository
- Continue Domain Analysis for L2SS Full Life Cycle



A division of Carnegie Mellon University

Unclassified

N²B
NATIONAL TEST BED

Identify/Construct Common L2SS Generalizations

- Identify L2SS Reusable Components from Domain Analysis
- Construct New L2SS Reusable Components from Domain Analysis
- Re-engineer Reusable Components into Common (Abstract) Models



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

Define L2SS Reuse Concept of Operations

- Legal and Contractual Issues
- New Procedures and Responsibilities
- Reuse Library Processes and Transactions
- Reuse Incentive Program Ideas



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

Install/Populate SAIC/STARS Repository at NTB

- Ada Constants, Algorithms, KESS Elements
- C & FORTRAN Routines from System Sim. Framework
- Ada Components Identified from Domain Analysis
 - "Small" Parts
 - "General" Parts



A Division of Carnegie Mellon University

Unclassified

NTB
NATIONAL TEST BED

Survey/Evaluation Software Reuse Repository Architectures

- SAIC/IBM STARS Repository
- USASDC-W/Softech RAPID Repository
- USAF/Westinghouse RAASP Repository
- McDonnell Douglas CAMP Program
- NASA/Barrios AutoLib Repository



A division of Carnegie Mellon University

Unclassified

NB
NATIONAL TEST BED

Establish RAS Connectivity & Distributed Libraries

- Central Repository for Configuration Management
- Electronic Bulletin Board for Reuse Information Network
- Electronic Mail System for Reuse Communication and Component Updates
- Coordinate Development of "Local" Reuse Library



A division of Carnegie Mellon University

Unclassified

N B
NATIONAL TEST BED

TRW's Ada Success	Ada Pros and Cons	
-------------------------	-------------------	---

Advantages:

Control	Single Representation Format for Design and Code Consistent Tools, Training and Metrics Demonstrable Insight Into Intermediate Products
Productivity	Ada Supports a High Quality NAS Architectural Approach Uniform Structure and Generic Building Blocks Ada Prohibits a Large Class of Programming Errors
Quality	Expands The Number of Potential Software Solutions Self Documenting Ada Prohibits a Large Class of Programming Errors

Disadvantages:

Increased Resources	Ada Compilers Do Substantial Work Requires More CPU and Disk
Performance Sensitivity	Broader Potential Performance Impacts
Environment Maturity	From Excellent to Very Poor; Dependent on Host/Target Choice

PRESENTER: BRUCE LEWIS

Ada POLICY APPLIED AT MICOM

Ada Policy Applied at MICOM

**Bruce Lewis
Software Engineering Directorate
Feb. 12, 1991**

**Mission: MICOM Life Cycle Software
Engineering Center (LCSE)**

**PROVIDE MCCR EXPERTISE TO SUPPORT COMMAND'S
WEAPON SYSTEMS OVER THEIR LIFE CYCLE. COMMAND
FOCAL POINT FOR MCCR.**

- **ACQUISITION AND DEVELOPMENT (AD)**
 - **MCCR TECHNOLOGY ASSESSMENT**
 - **SOFTWARE ENGINEERING SUPPORT TO PO**
 - **MCCR INPUTS TO RFP, CRMP**
 - **SOURCE SELECTION EVALUATION**
- **SOFTWARE VERIFICATION AND VALIDATION (V&V)**
 - **REQUIREMENTS, DESIGN, CODE
AND TEST VERIFICATION**
 - **SOFTWARE/SYSTEM VALIDATION**
- **POST DEPLOYMENT SOFTWARE SUPPORT (PDSS)**
 - **MAINTAIN/ENHANCE SOFTWARE OF FIELDED
SYSTEMS**
 - **FIELD PROBLEM ANALYSIS/SUPPORT**
- **INTEROPERABILITY TESTING**

Ada Systems at SED

<u>MISSILES/WEAPONS</u>	<u>COMMAND AND CONTROL</u>
RADIO FREQUENCY INTERFEROMETER	FDDM CHS
HELLFIRE DIGITAL AUTO PILOT ADVANCED MISSILE SYSTEM - HEAVY	FAAD C2I
LINE OF SIGHT ANTI-TANK HELLFIRE OMS HELLFIRE LONGBOW	
AAWS-M MLRS TGW	<u>WOULD HAVE BEEN Ada</u>
TOW SIGHT IMPROVEMENT PROGRAM KE ASAT	TACIT RAINBOW FOTL NLOS

KEY Ada POLICY ISSUES

- **AMOUNT OF ASSEMBLER ALLOWED AND WHY**
Demonstrated need - Do it in Ada first.
Assembler source / Ada source
15 % of what ? System? Tactical? CSCI? Processor?
- **WHEN WAIVERS ARE REQUIRED**
Prior to Milestone I approval - before contract on POP
If 15% proves to be inadequate
- **DEFINITION OF NDI**
Contractor's - Anything already existing
Ours - Commercial or fielded based on AR 70-1
- **SUPPORTABILITY STATEMENT REQUIREMENT**
Keeps those with a life-cycle perspective involved
We also validate technical, cost, schedule claims

AR70-1 para 4-9 "NDI includes commercial "off the shelf " items, material developed and in use by other U. S. military services or Government agencies, and material developed and in use by other Allies.with adequate supportability."

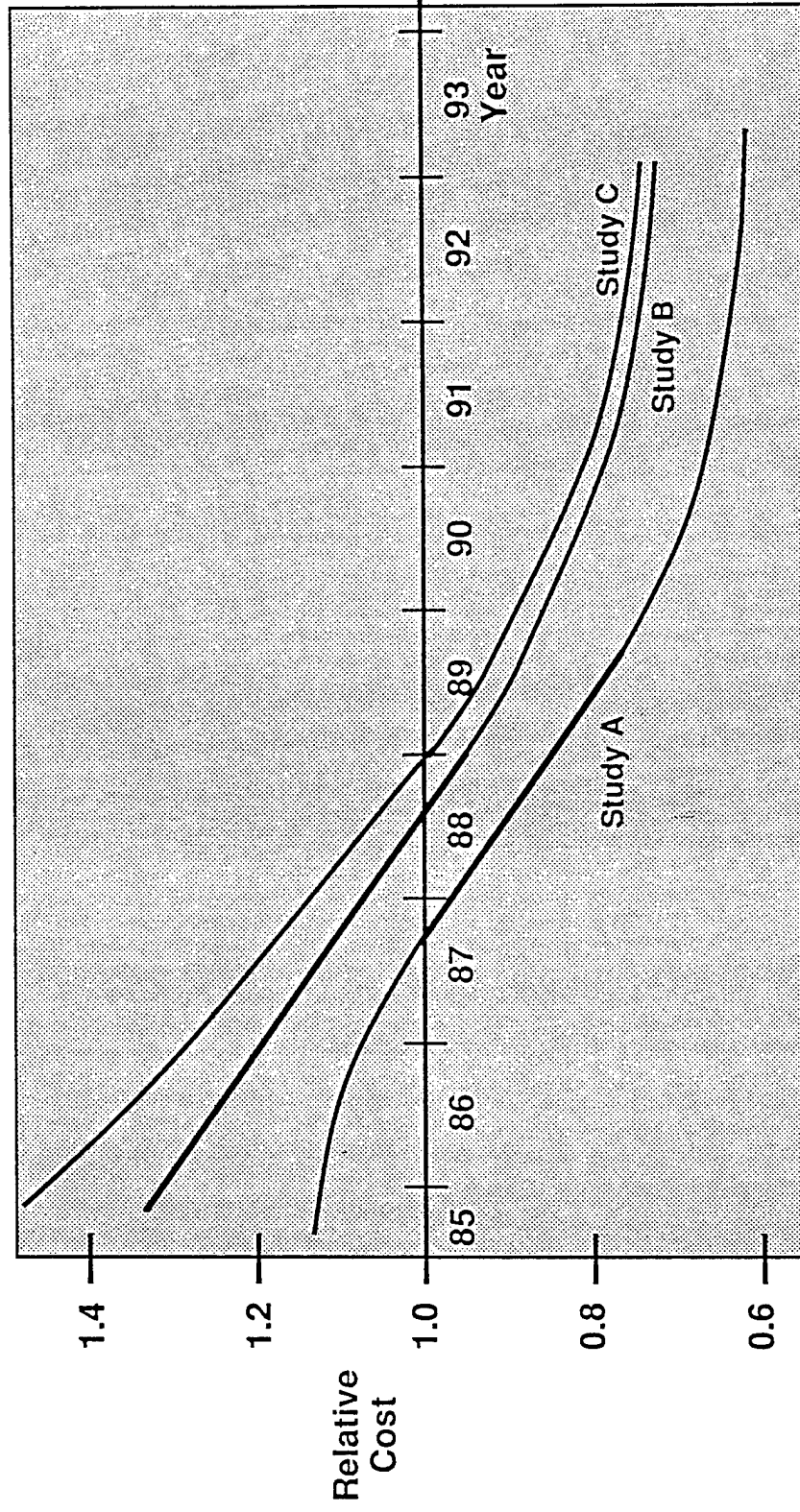
Lessons Learned on Getting First Use

- Typically difficult to get to use Ada the first time, thereafter no problem
- Requires government software engineers who understand the benefits and issues and are willing to stand up for the use of Ada.
- Requires support with policy and at each level of government management. Loop holes will be used or invented.
- Typical Areas of Resistance:
 - Management more often than technical, management risk, and investment adverse (software technical people want it)
 - Perception that language or compilers are not efficient (changes after benchmarking current compilers)
 - Wants to use last phase hardware/software for a bid advantage. (But often inadequate causing significant problems)

Lessons Learned for Program Success

- Contractor more open to try new technology in Tech Demo or Proof of Principle.
- Training important part of technology transition, especially to obtain software engineering benefits as well as Ada.
- CASE tools must support Ada features/methods to be effective. Otherwise may hinder more than help.
- Adequate processing power key to getting well engineered software in Ada or any language.
- Reserve and growth requirements for throughput and memory critical.
- Adequate schedule to allow software engineered, reliable, maintainable, enhanceable, reusable design rather than rework of POP design.
- Get a few (or more) Ada experienced personnel as leads on the development team.
- Software management personnel need Ada program management and Ada software engineering training.

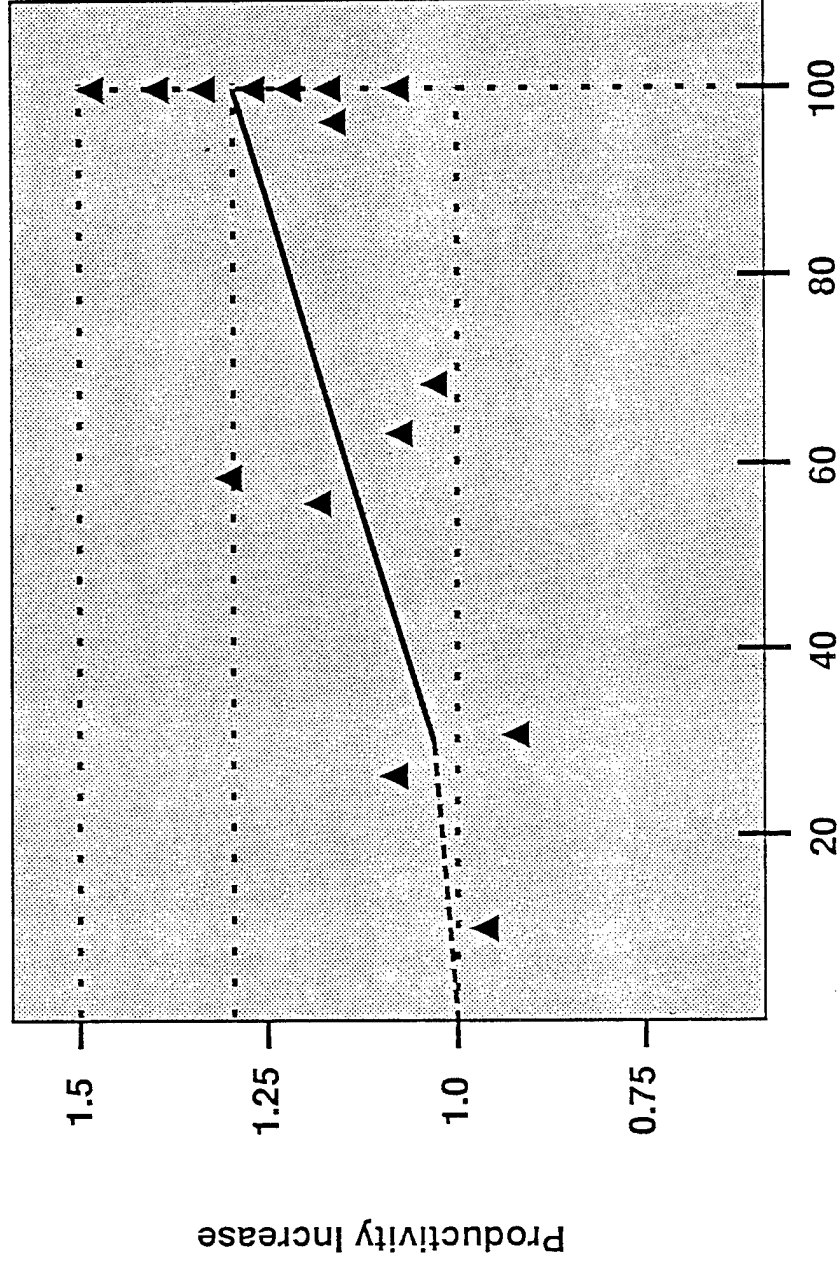
Early Predictions: Effect of Ada on Software Development Costs



Source: B. Boehm, "Software Cost Estimation Using COCOMO", 1987

Productivity of Ada

Ada at NTT



SYSL portion = 100%

Ada portion = 100%

Source: K. Tamaya, "Using Ada at NTT",
WAda S'90

Ada at NTT

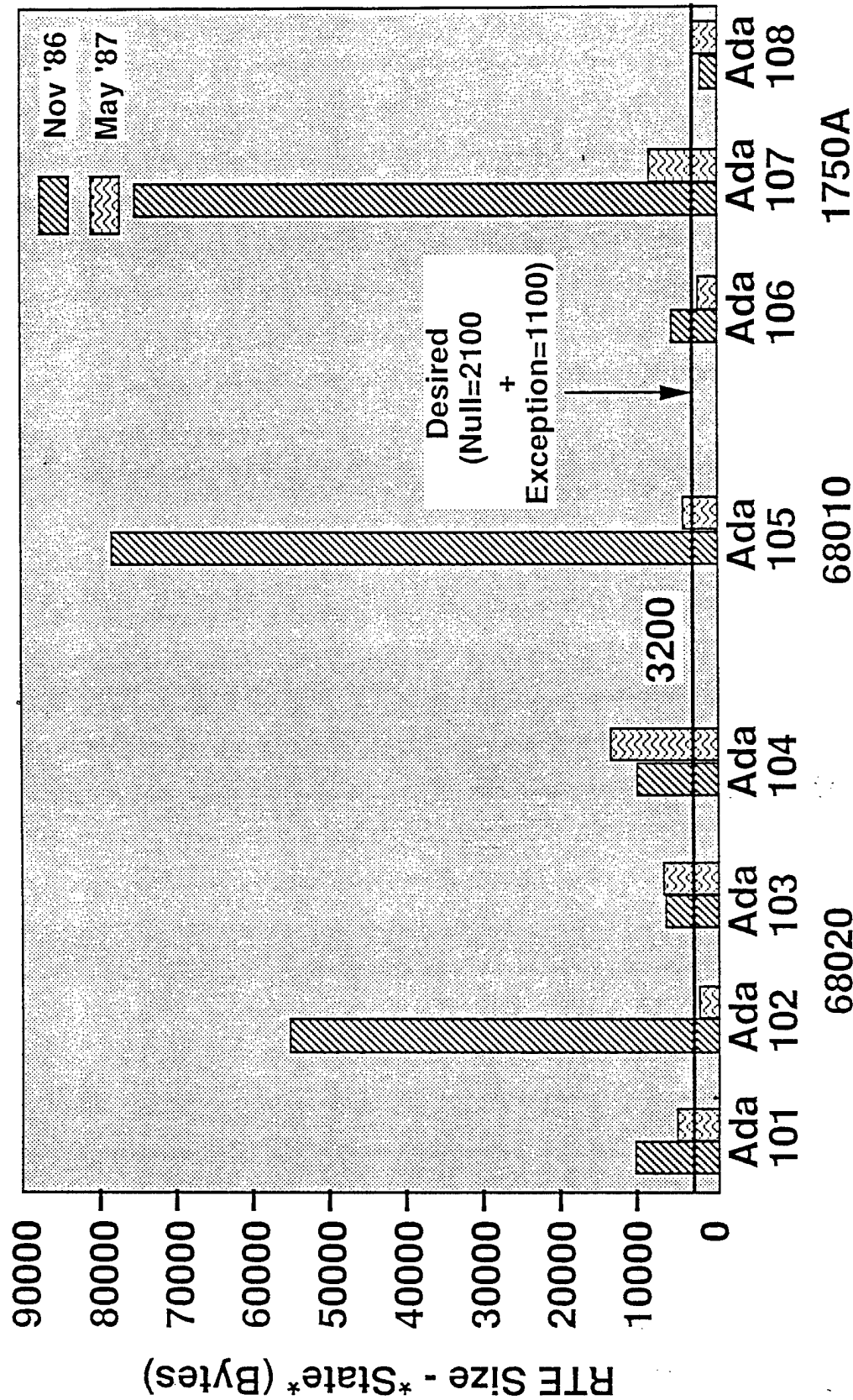
- When Ada is used 100% productivity improved 30%
- Productivity expected to increase more when using software engineering methodology like OOD
- NTT has written over 2 million lines of Ada
- SYSL is a PL/1 like systems programming language used on mainframes and for real time software in telecom microprocessors
- NTT has adopted Ada instead of SYSL because it is more reliable and easier to maintain

Development Cost Data

Ada Projects which have made their productivity numbers public are showing about a 20% decrease in costs (over time) after they have made the transition to what is called "the Ada mindset." This mindset involves learning and applying new software engineering principles, modern methods like OOD (Object-Oriented Design), and advanced packaging concepts and tools, as well as the Ada programming language itself.

These experiences are based upon composite facts gleaned from conducting postmortems on over 140 completed Ada projects worldwide. These projects have delivered in excess of 50 million lines of Ada code over the past five years.

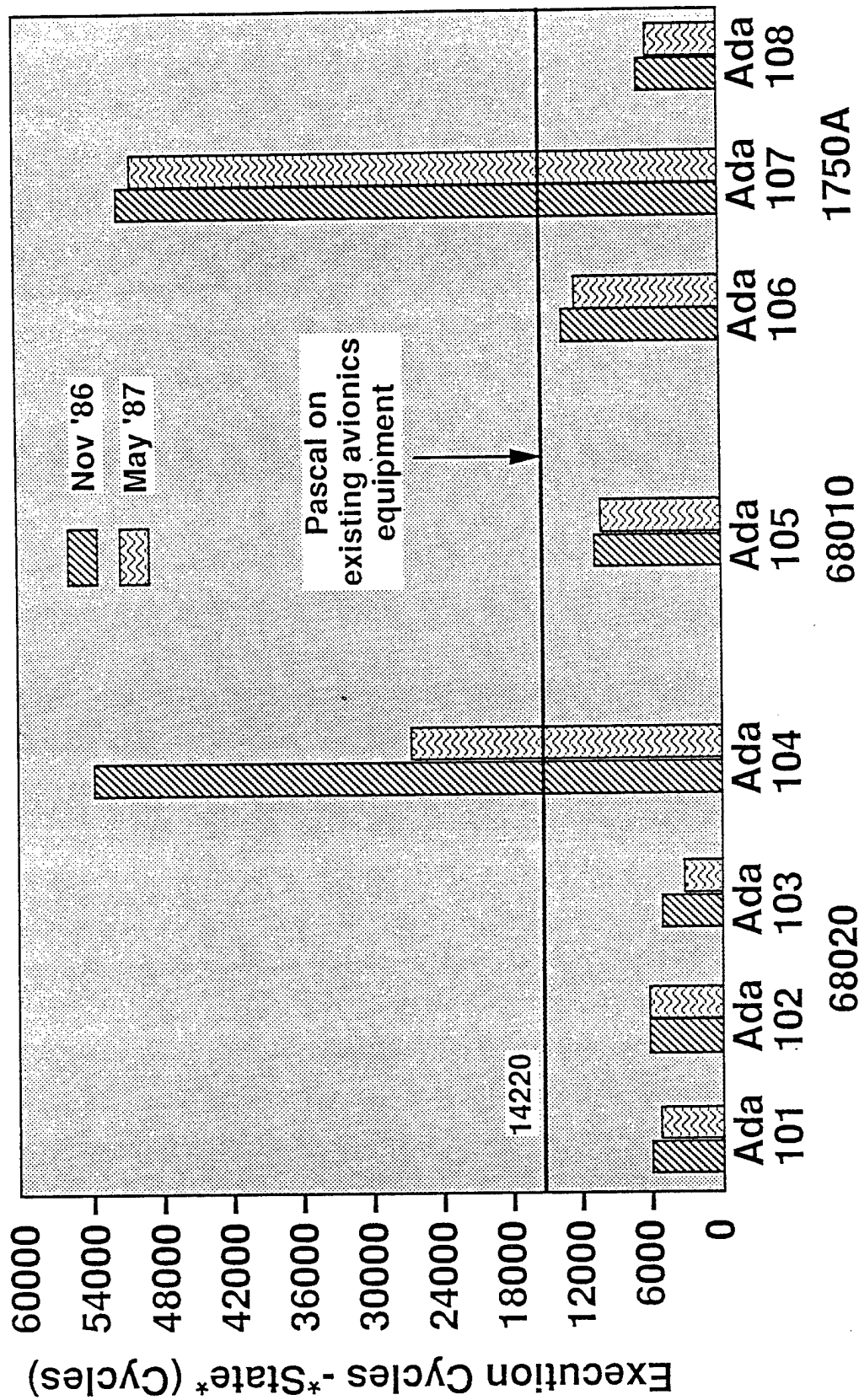
Boeing Commercial Airplanes Compiler Benchmarking



Note: The desired value was the average of the survey of avionics suppliers for this type of program.

Source: BCA, SIGAda Seattle, Aug., 1987.

Boeing Commercial Airplanes Compiler Benchmarking



Note: The Ada version includes exception handling which is unavailable in Pascal. Suite consists of application specific benchmarks from 3 avionics suppliers.

Source: BCA, SIGAda Seattle, Aug., 1987.

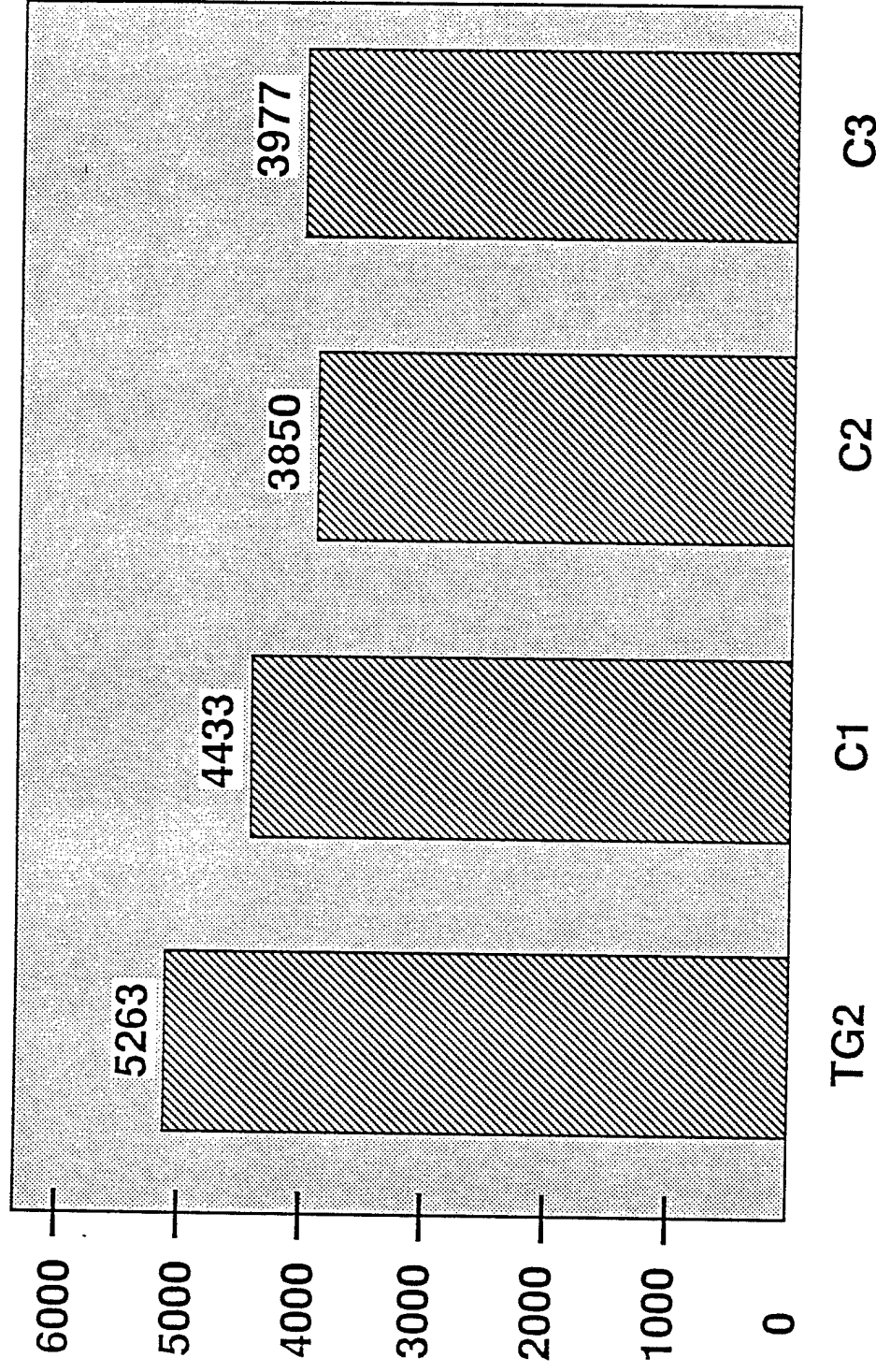
7J7 Avionics/Flight Systems

TECHNOLOGY ITEM:	DECISION GATE FOR PROGRAM COMMITMENT:	ASSESSMENT OF RISK FOR INCORPORATION	
		DECISION GATE	6 MONTHS LATER:
LVPS	05/16/88	NONE	NONE
FLY-BY-WIRE	05/31/88	LOW	NONE
VLSI COUPLER	07/01/86	NONE	NONE
	04/01/88	NONE	NONE
RACU	09/01/87	MED*	LOW*
IACS	09/01/87	MED*	LOW*
VERY HIGH RELIABILITY CHIP SET	11/25/87	LOW	NONE
	04/15/88	MED	LOW
FIBER OPTIC SENSORS	01/21/88	TLA:MED	TLA: LOW
		REST:HIGH	REST:MED
FIBER OPTIC LINEAR			
DATA BUS	01/21/88	MED	LOW
PRIMARY FLT CONTROL			
FBW ACTUATION	05/31/88	LOW	NONE
FLAT PANEL DISPLAYS	06/27/88	MED	LOW
FAULT TOLERANT ADIRS	02/02/89	MED	LOW
ADA	11/13/87	LOW	NONE

*BETTER ASSESSMENT 09/01/87

Source: B. Pflug, BCA, Aug 26 1987,
SIGAda Seattle

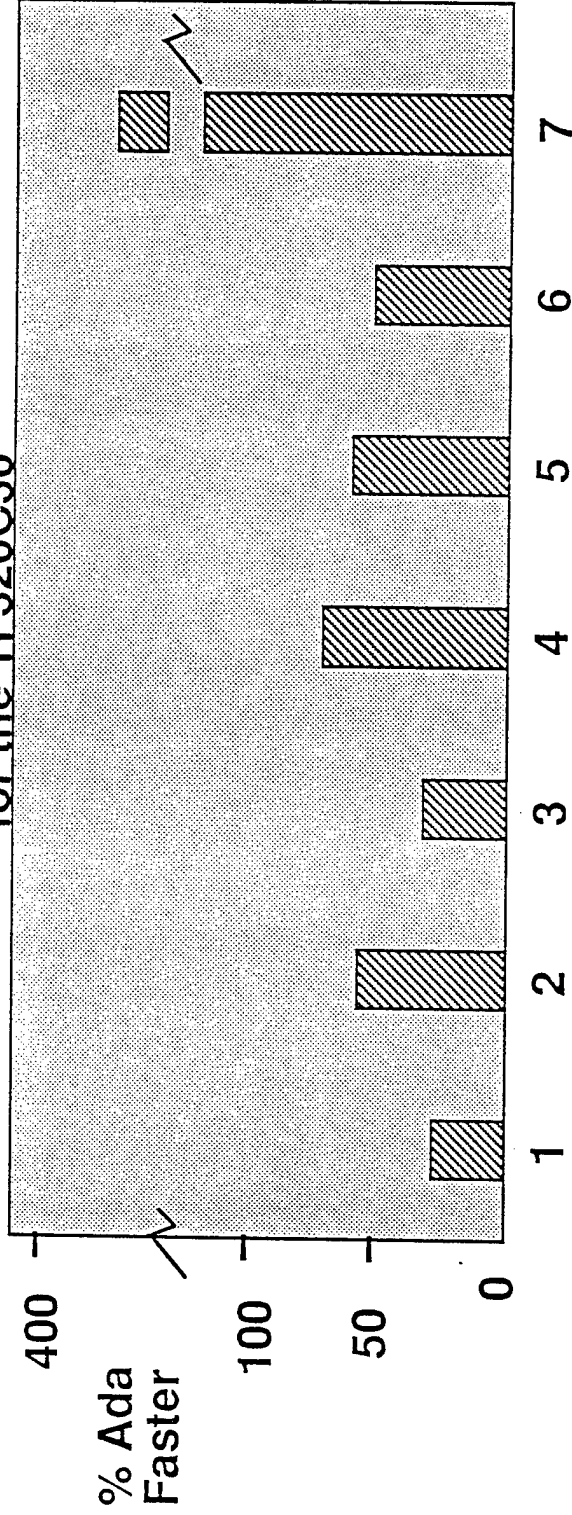
PIWG Dhrystone



The TeleGen2 Optimizing Ada Compiler surpasses the top C compilers on the Sun-3 system.

Tartan Ada outperforms latest "C" compiler

Tartan Ada Version 3.0 vs. Optimized C Version 4.0
for the TI 320C30

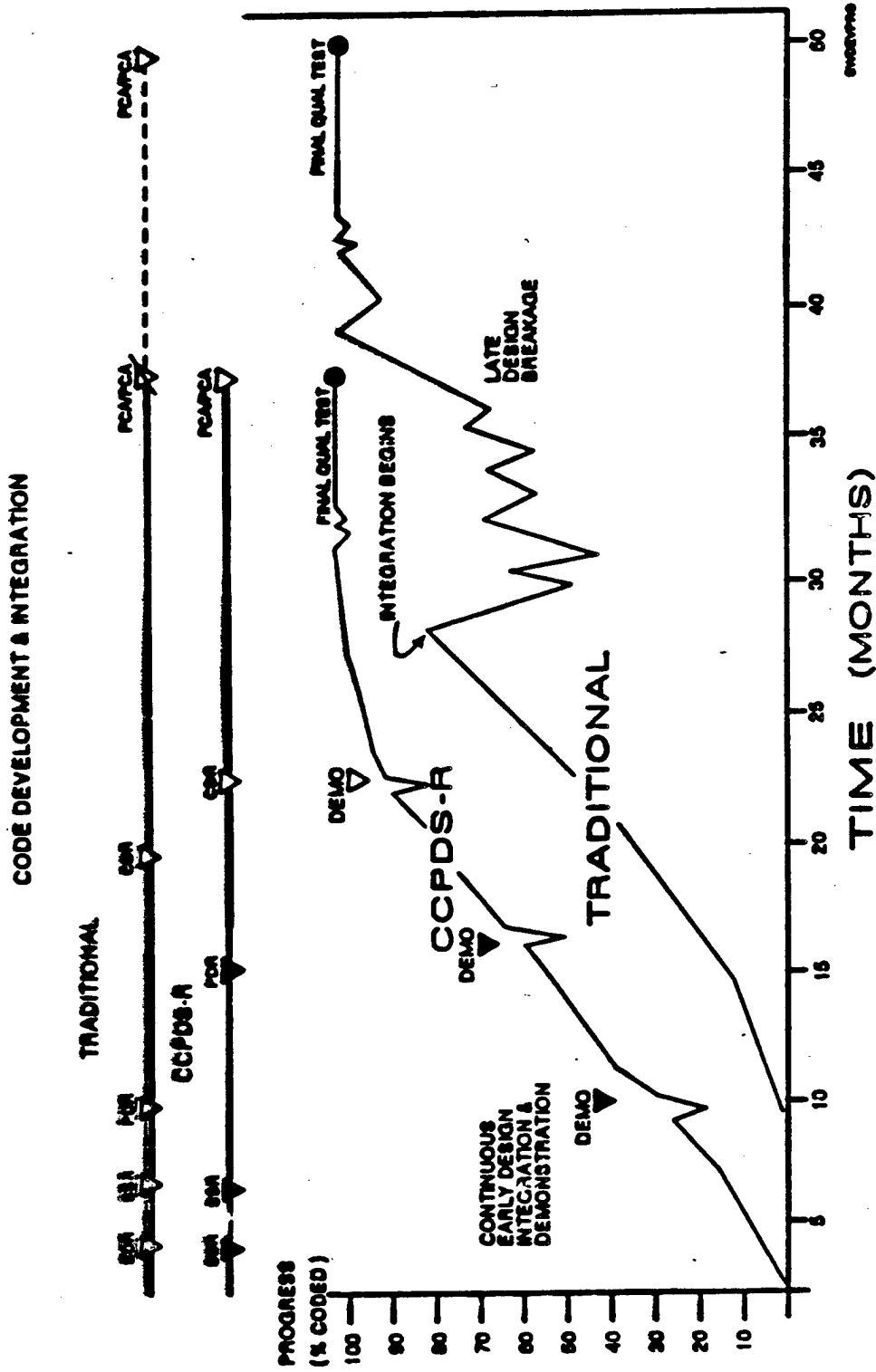


Figures for the Speed Comparison

Benchmark	Tartan Ada	C Compiler	Ratio	% Faster for Ada
1	386	472	1.22	22
2	422	648	1.53	53
3	40	51	1.27	27
4	118	201	1.70	70
5	888	1429	1.60	60
6	958	1429	1.49	49
7	696	3040	4.36	336

Application Specific Benchmarks Supplied by
MLRS-TGW Contractor

Source: Tartan Real Times
Dec. 1990



TRW's Ada Success	CCPDS-R Quality Metrics Actuals	TRW
-------------------------	---------------------------------	-----

Metric	Definition	CCPDS-R Value
Rework Proportions	$R_E = \frac{\text{Effort}_{\text{Rework}}}{\text{Effort}_{\text{Total}}}$ $R_S = \frac{\text{SLOC}_{\text{Reworked}}}{\text{SLOC}_{\text{Configured}}}$	6.7% 13.5%
Modularity	$Q_{\text{mod}} = \frac{\text{Total Breakage}}{\text{No. of SCOs}}$	$53 \frac{\text{SLOC}}{\text{SCO}}$
Changeability	$Q_C = \frac{\text{Total Effort}}{\text{No. of SCOs}}$	$15.7 \frac{\text{Hrs}}{\text{SCO}}$
Maintainability	$Q_M = \frac{R_E}{R_S} = \frac{\text{Productivity}_{\text{Development}}}{\text{Productivity}_{\text{Change}}}$	.49

SCO: Software Change Order - Discrete Configuration Baseline Change

TRW's Ada Success	CCPDS-R Changeability Evolution	TRW
-------------------------	---------------------------------	-----

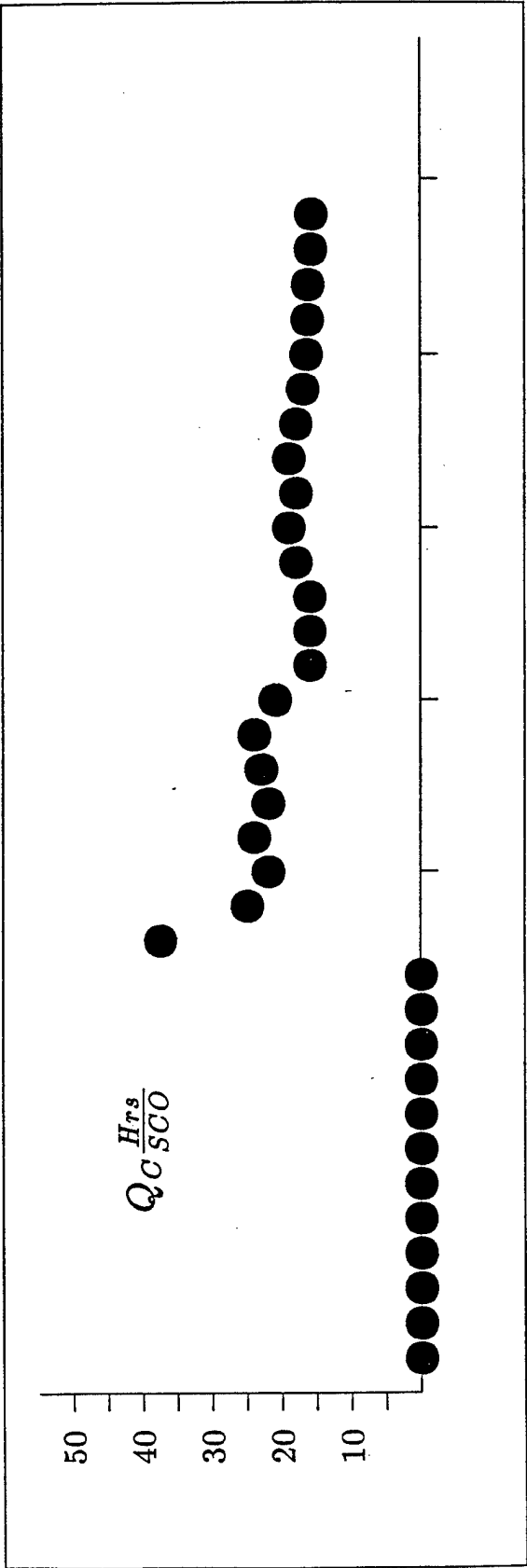
Conventional:

- Cost of Change Increases With Time

CCPDS-R:

- Cost of Change Stabilizes, (Even Decreases) With Time

CCPDS-R Changeability Evolution



PRESENTER: ELIZABETH S. KEAN

ASPECTS OF REUSABILITY IN PROTO+



ASPECTS OF REUSABILITY IN
PROTO+

ELIZABETH S. KEAN

ROME LABORATORY/COEE
GRIFFISS AFB NY 13441-5700
(315) 330-2762/AV 587-2762



OUTLINE

- SYSTEM DEFINITION TECHNOLOGY
- "REUSABLE" REQUIREMENTS SPECIFICATIONS
- WHAT IS PROTO+?
- HOW WILL PROTO+ SUPPORT REUSE?
- PROTO+ WALKTHROUGH
- PROTO+ HARDWARE/SOFTWARE CONFIGURATION



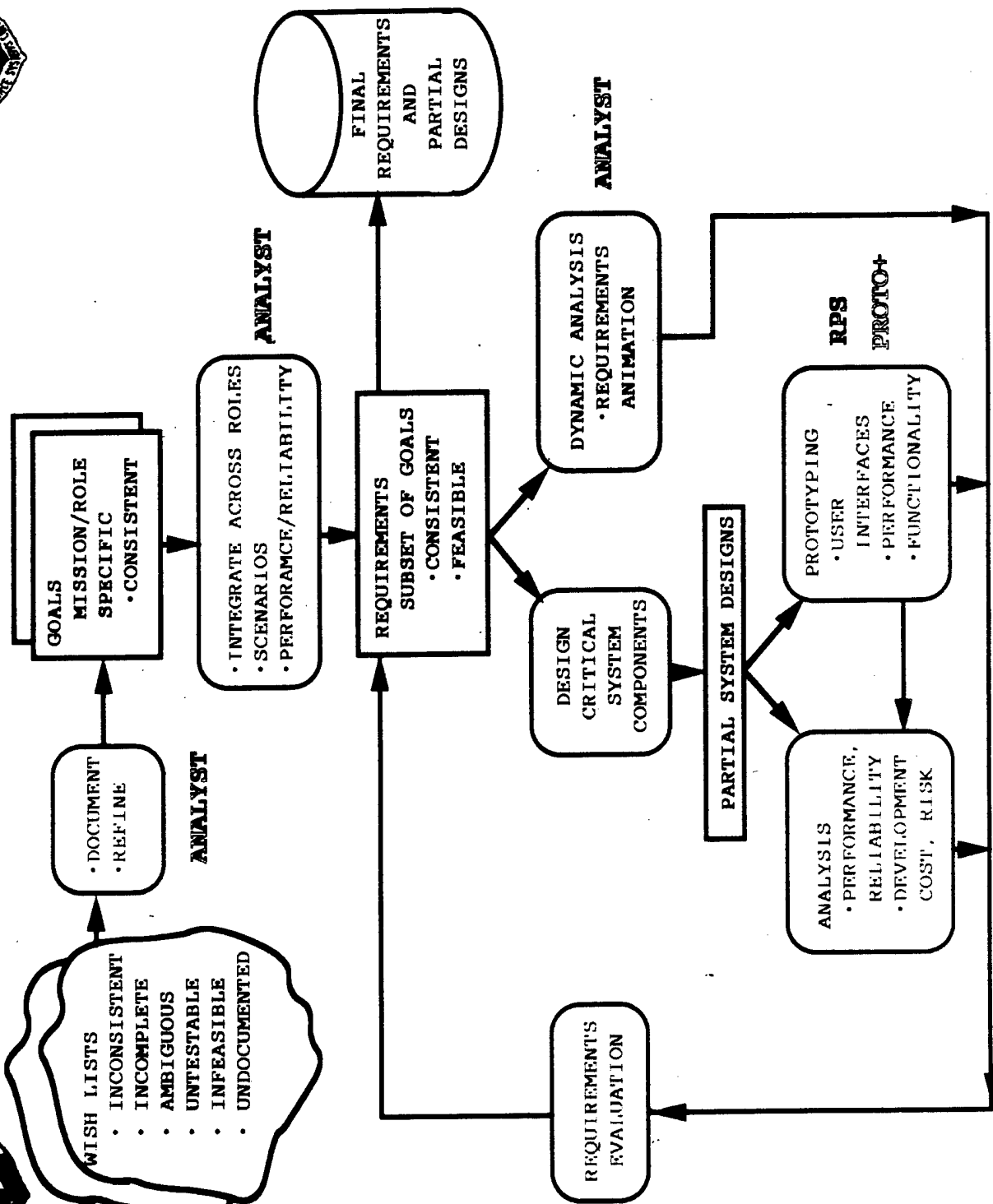
SYSTEM DEFINITION TECHNOLOGY



- TASKED WITH DEVELOPING METHODOLOGY AND TOOLS TO SUPPORT THE DEVELOPMENT OF REQUIREMENTS SPECIFICATIONS FOR LARGE C3I SYSTEMS
- DEFINED REQUIREMENTS ENGINEERING PROCESS MODEL
- DEVELOPED ENVIRONMENT OF TOOLS TO SUPPORT THE PROCESS MODEL



REQUIREMENTS ENGINEERING PROCESS MODEL





"REUSABLE" REQUIREMENTS SPECIFICATIONS

TWO LEVELS OF REUSE:

- LOW LEVEL COMPONENT - e.g., LIBRARY ROUTINE
- COMPONENT CONSISTING OF HIERARCHY OF SUB COMPONENTS - e.g., AN ENTIRE OR MAJOR PIECE OF FUNCTIONALITY

SPECIFY WITH REUSE IN MIND



WHAT IS PROTO+?

- VERY HIGH LEVEL SYSTEM PROTOTYPING TOOL
- EXECUTABLE SPECIFICATION LANGUAGE FOR
FOR DESCRIBING SYSTEM FUNCTIONALITY
- OBJECT ORIENTED DATA MANAGEMENT SYSTEM
FOR STORING SYSTEM GRAPHS AND DATA



PROTO+ CHARACTERISTICS:

- REPRESENT/ANALYZE/MEASURE PROTOTYPE TIMING CHARACTERISTICS
- CONCURRENT PROCESSES AND COMMUNICATION CHANNELS
- REUSABLE COMPONENTS FOR A C3I DOMAIN
- TOOLS FOR BUILDING/MANAGING REUSE LIBRARY
- TOOLS FOR BUILDING AND TAILORING REUSABLE COMPONENTS



PROTO+ USERS

- SYSTEMS ANALYST - LITTLE OR NO PROGRAMMING ABILITIES, HOWEVER KNOWLEDGEABLE IN C3I DOMAIN
- PROGRAMMER - ADDS SPECIAL PURPOSE CAPABILITIES/FUNCTIONAL ALGORITHMS



HOW WILL PROTO+ SUPPORT REUSE?



- PROTO+ AS IT EXISTS TODAY:

REUSABLE COMPONENT LIBRARY AND
SUPPORTING TOOLS

ABILITY TO "CUT AND PASTE" BETWEEN
COMPONENTS

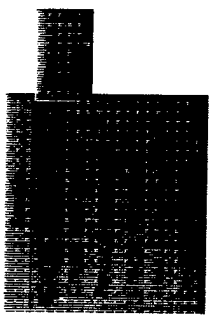
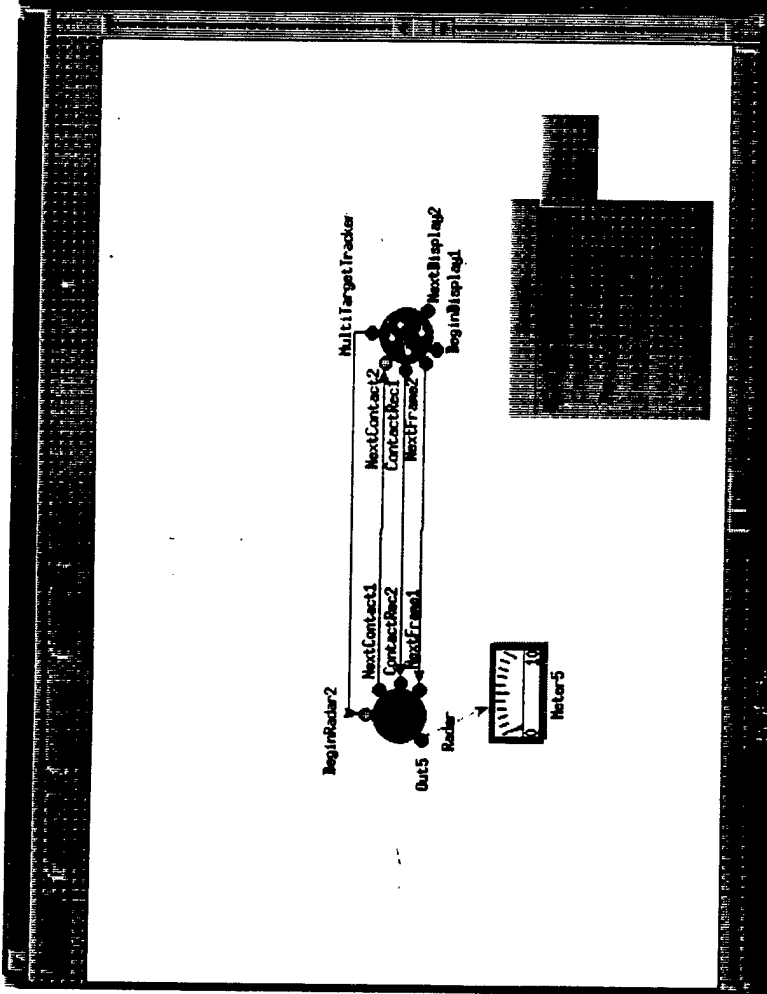
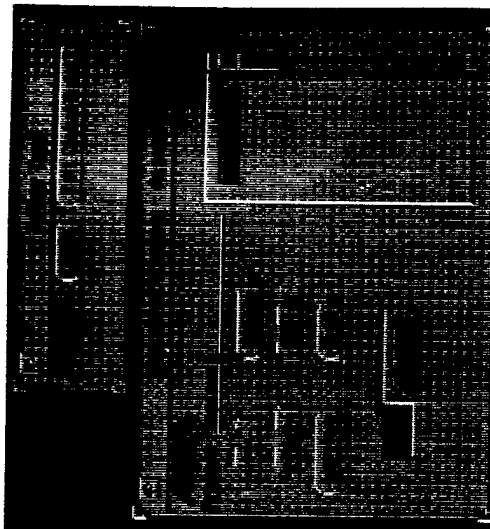
METHODOLOGY FOR DEVELOPING REUSABLE
COMPONENTS

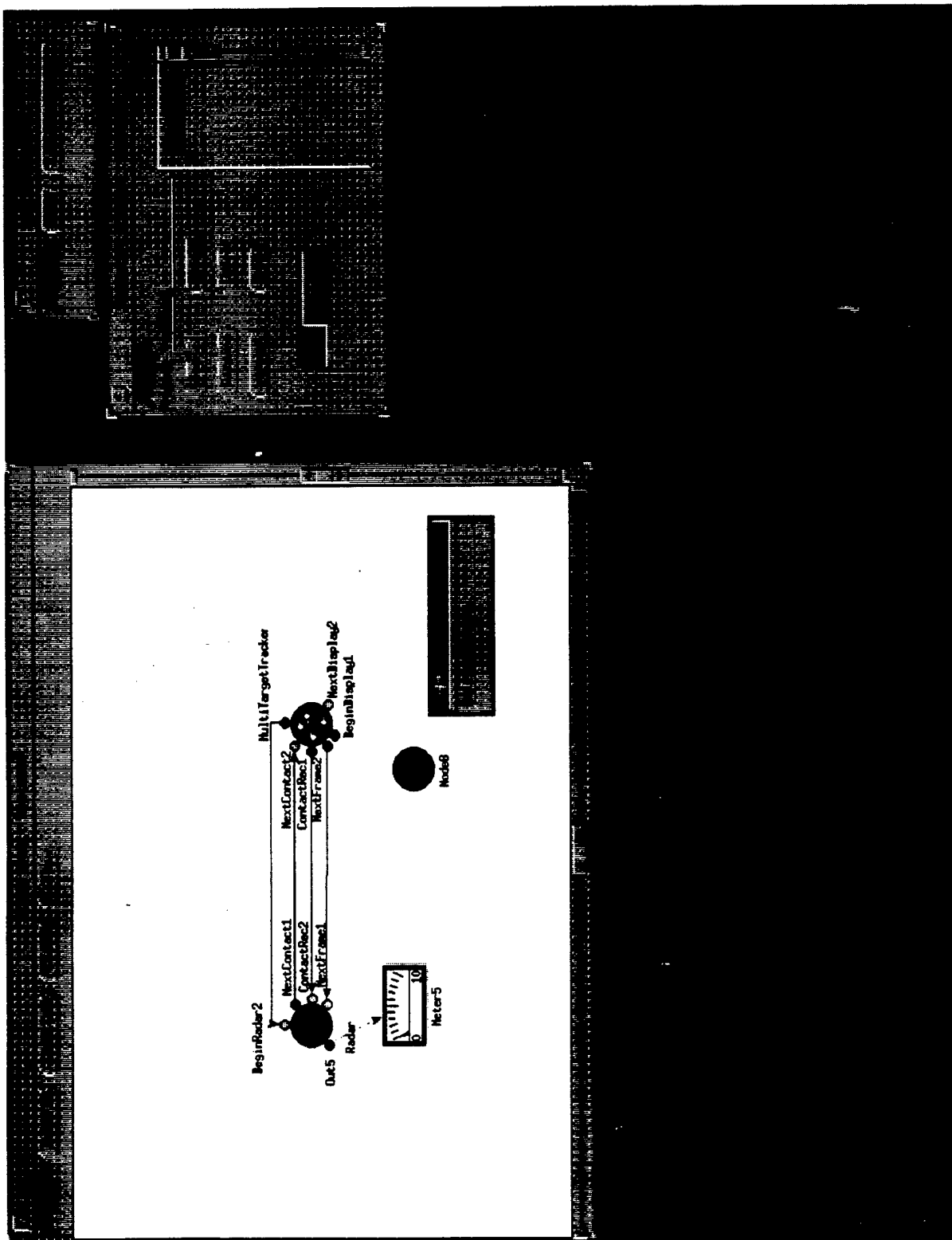
- PROTO+ OF THE FUTURE:

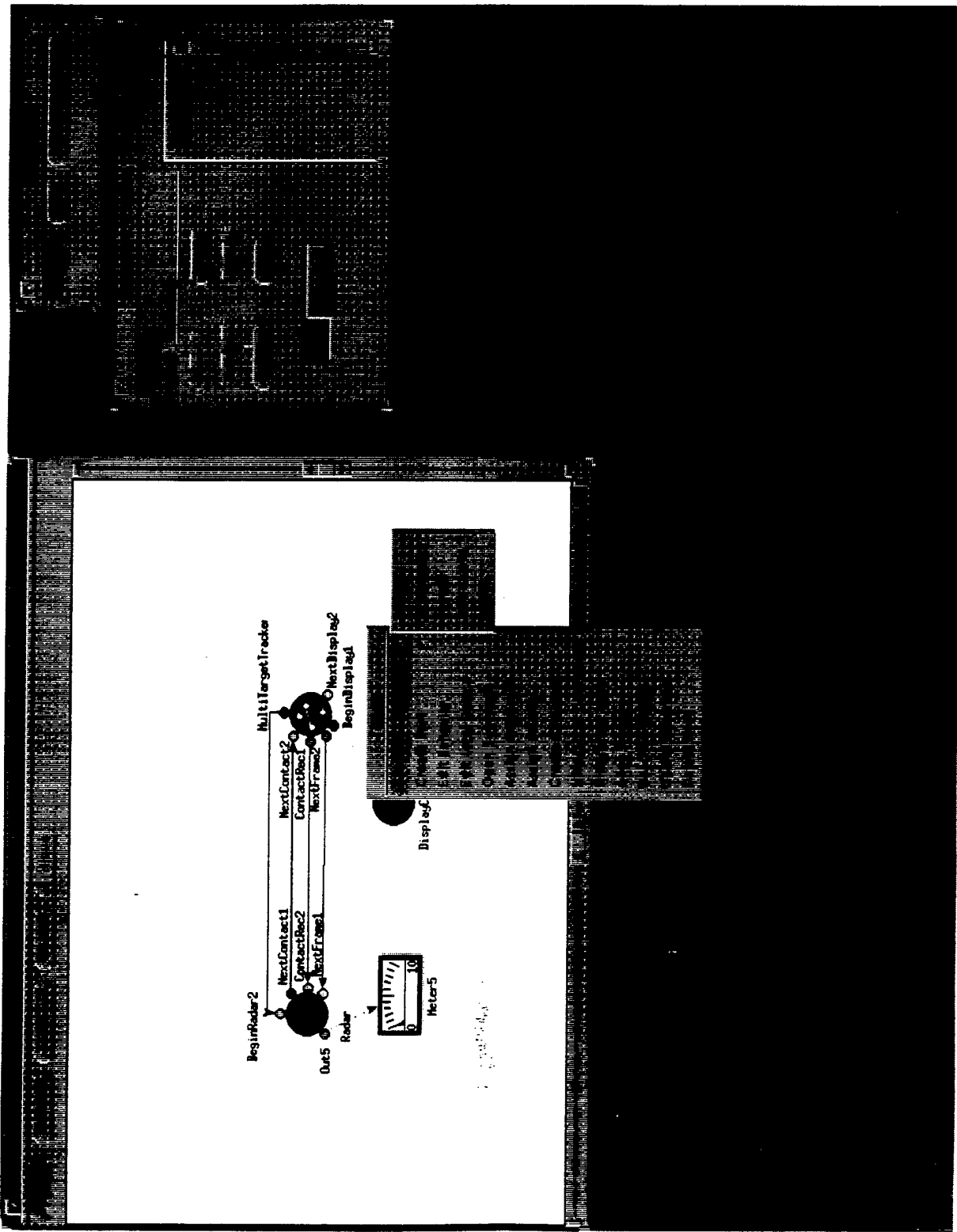
ADD "GLOBAL" LIBRARY FUNCTIONALITY

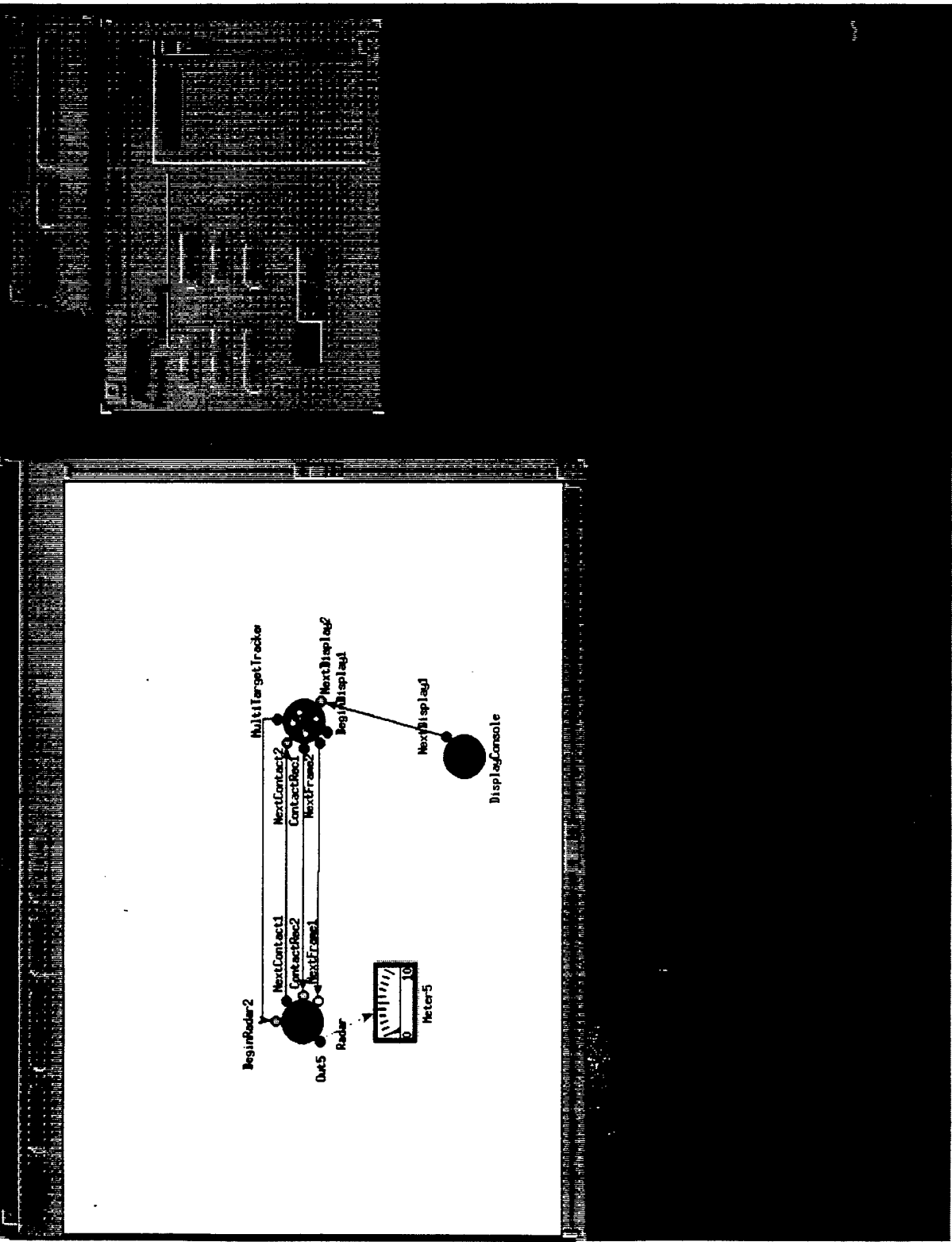
IMPROVE REUSE METHODOLOGY

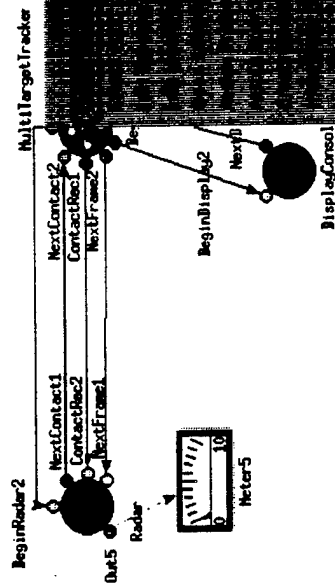
DEVELOP C3I TEMPLATES OF REUSABLE SPECS

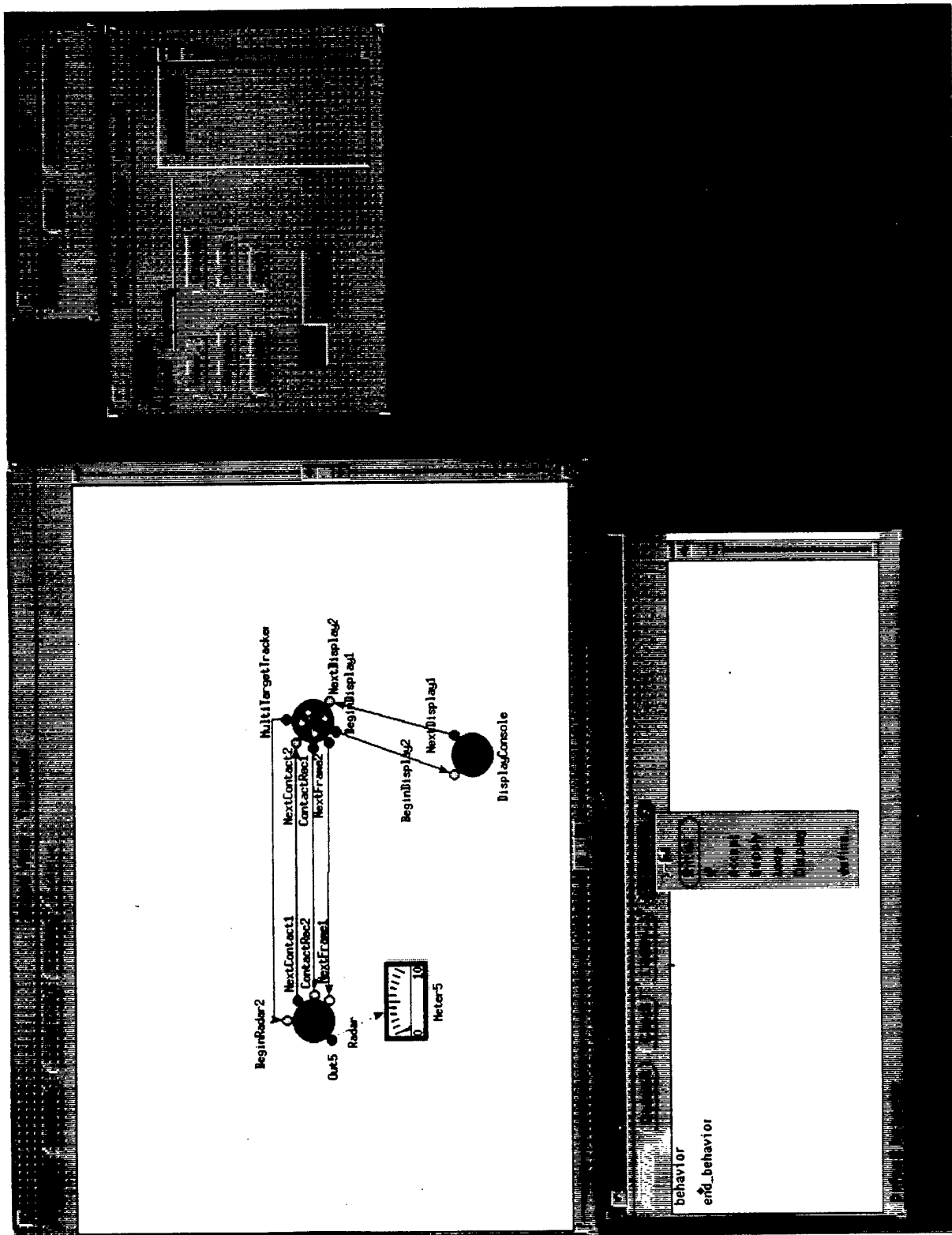


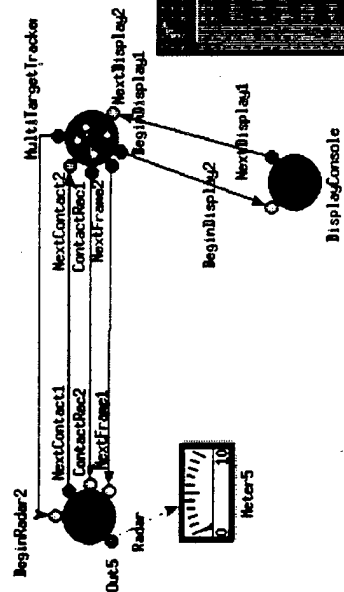








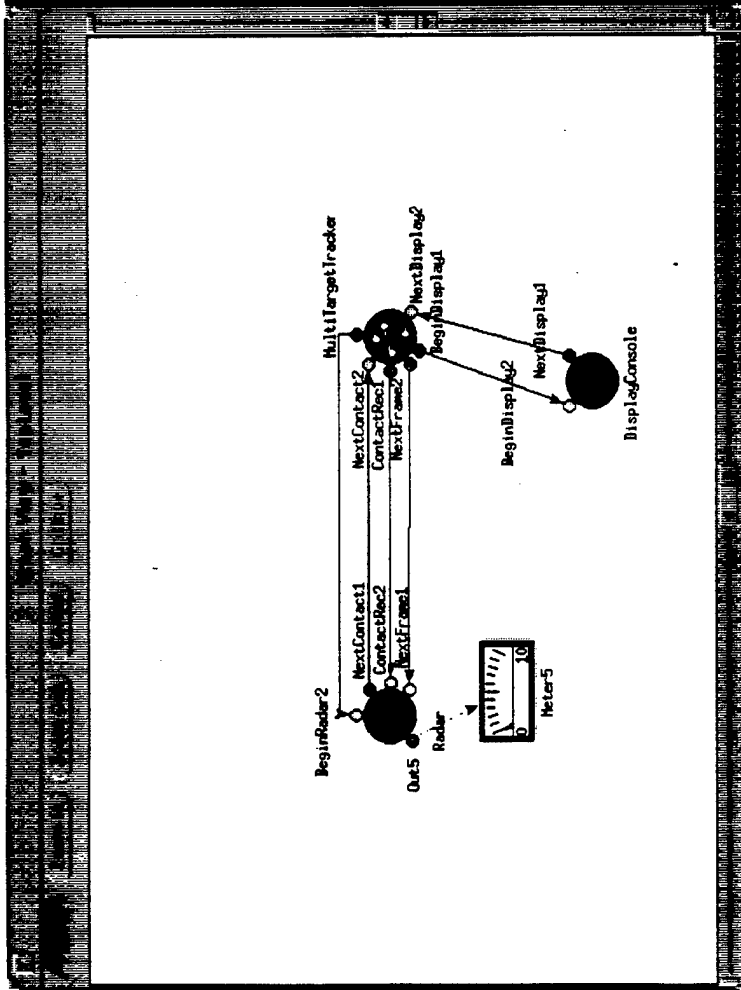
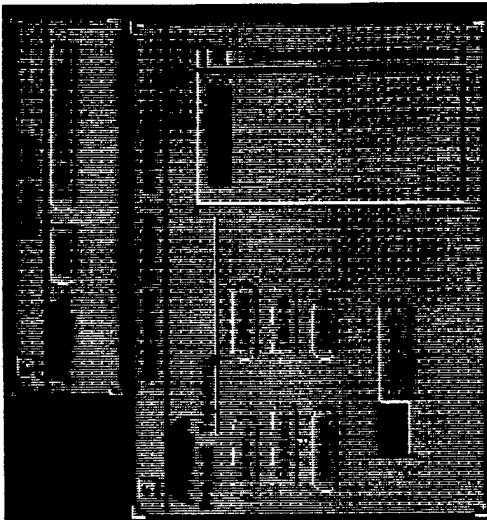




```

behavior
  accept (BeginDisplay2) :-
    NextDisplay1 <<
  end_behavior

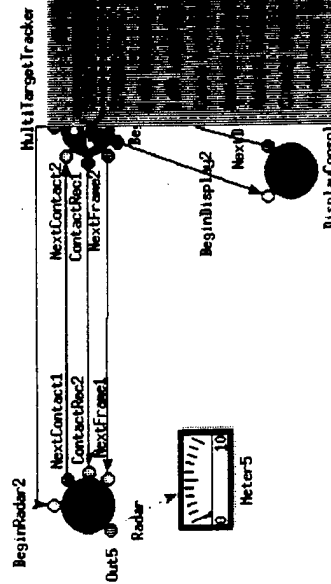
```

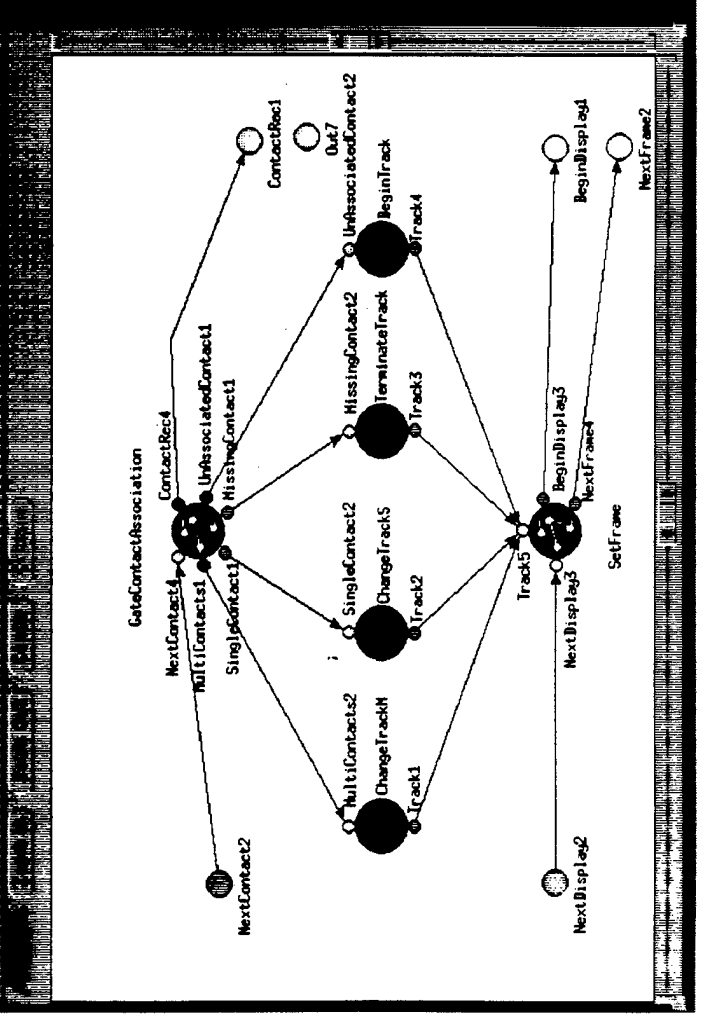
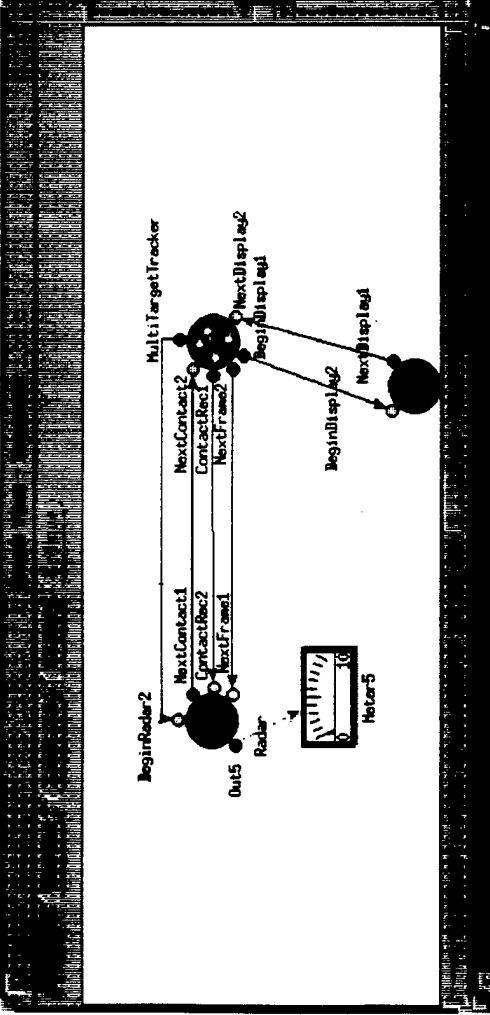
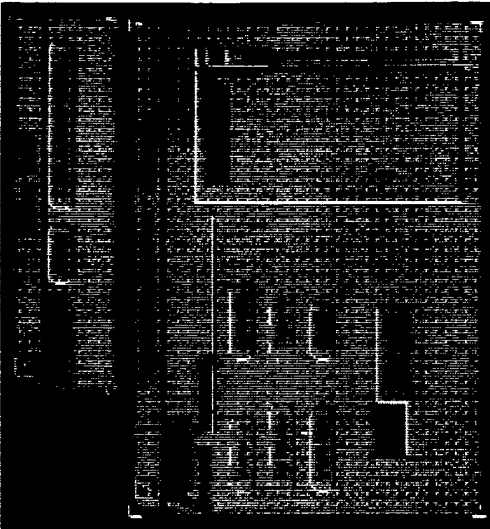


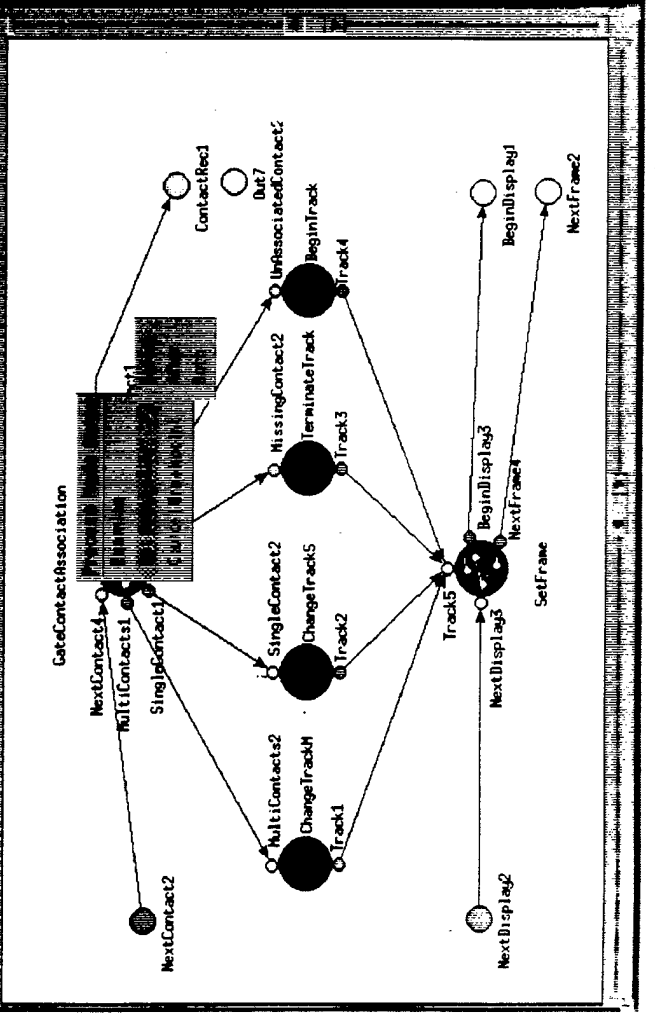
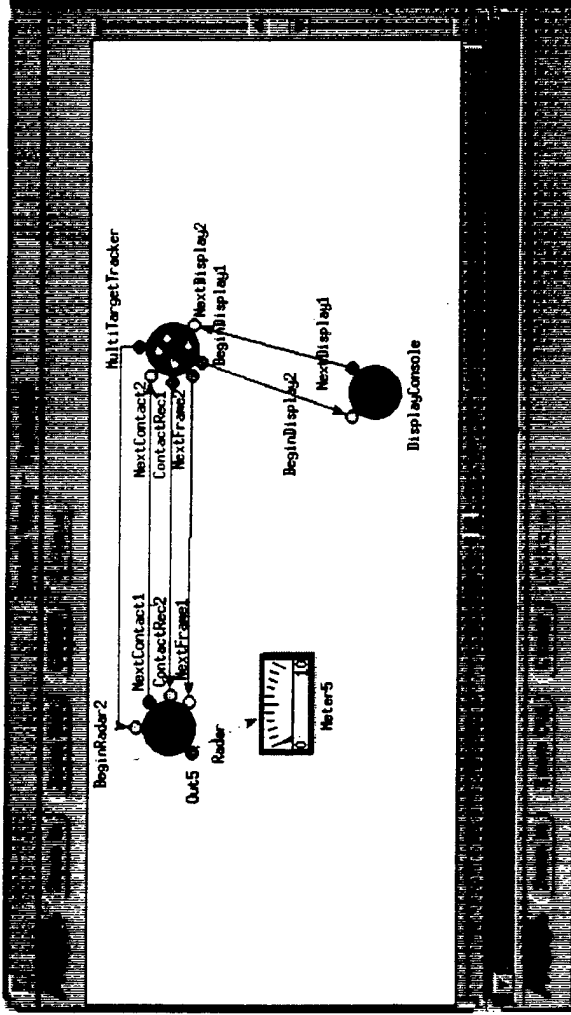
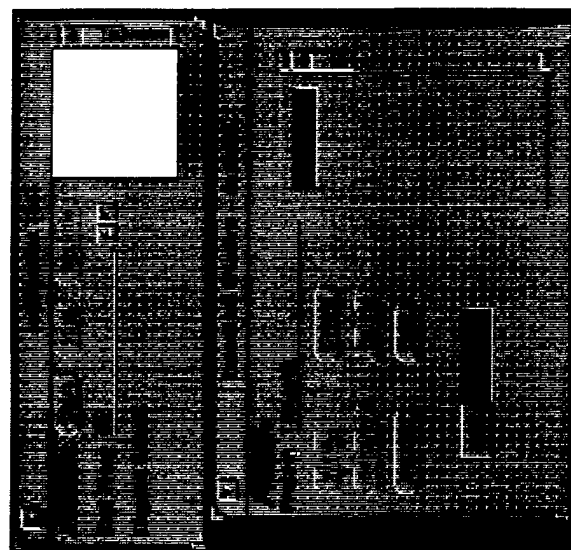
```

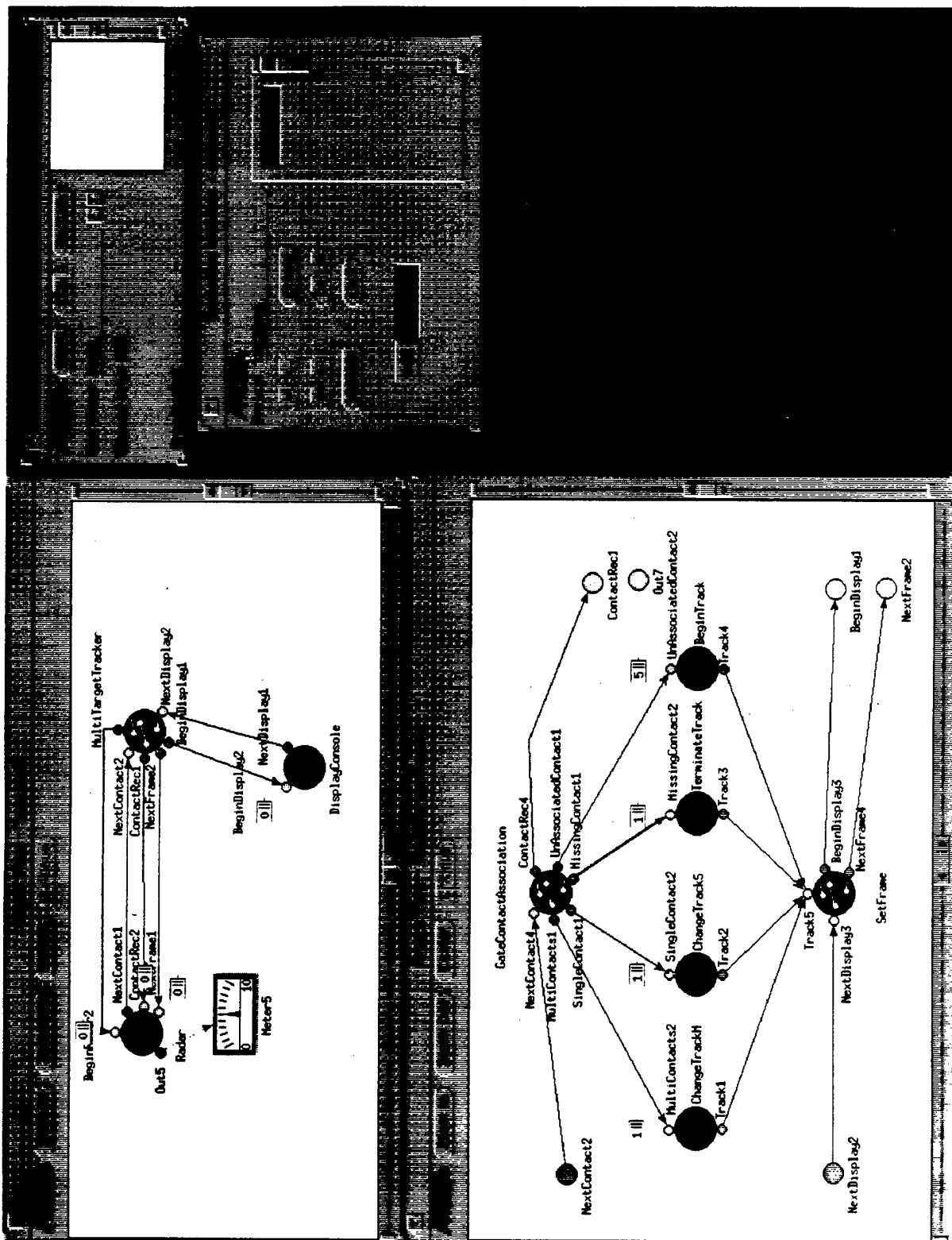
behavior
  accept (BeginDisplay2) :-
    NextDisplay1 << newframe (1)
  end_behavior

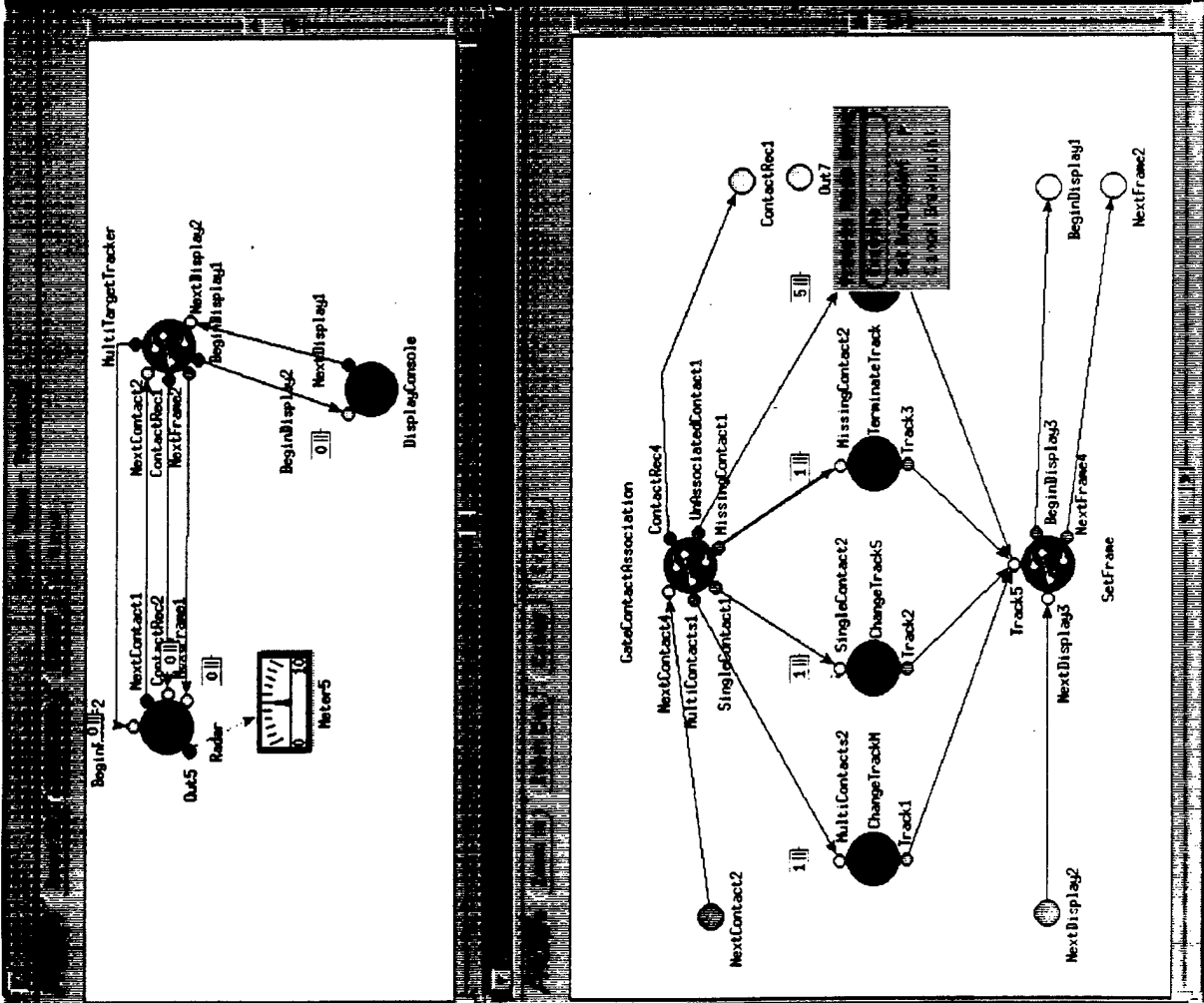
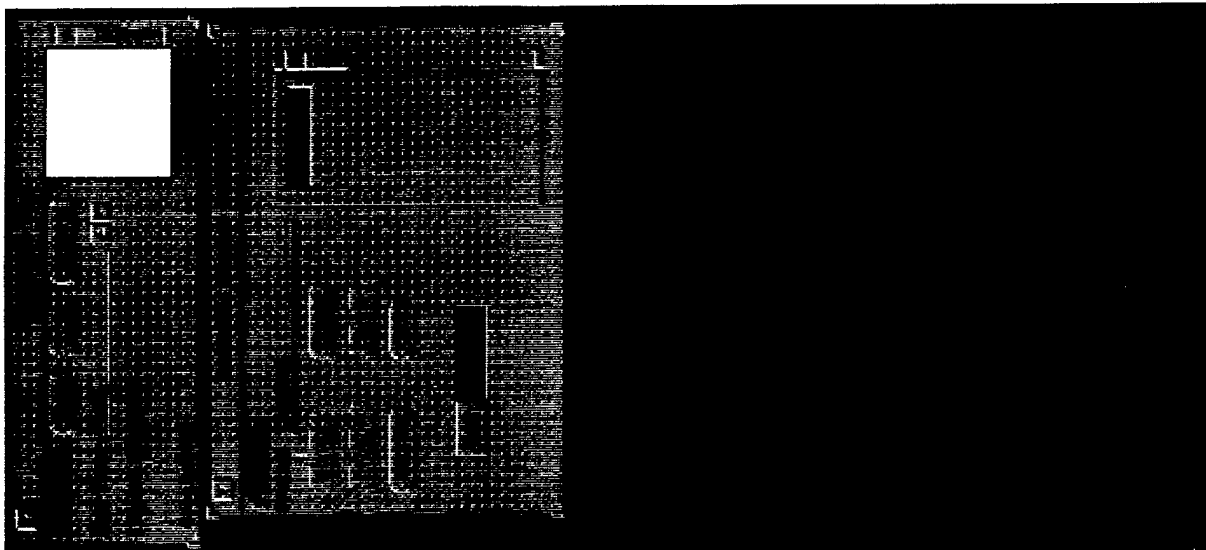
```

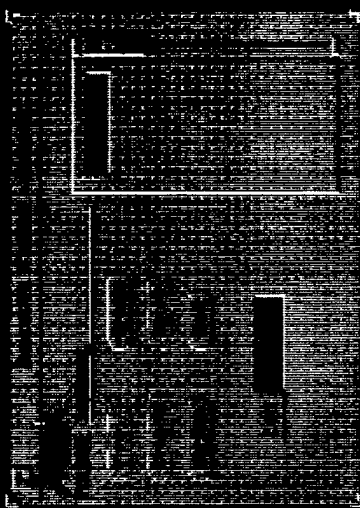


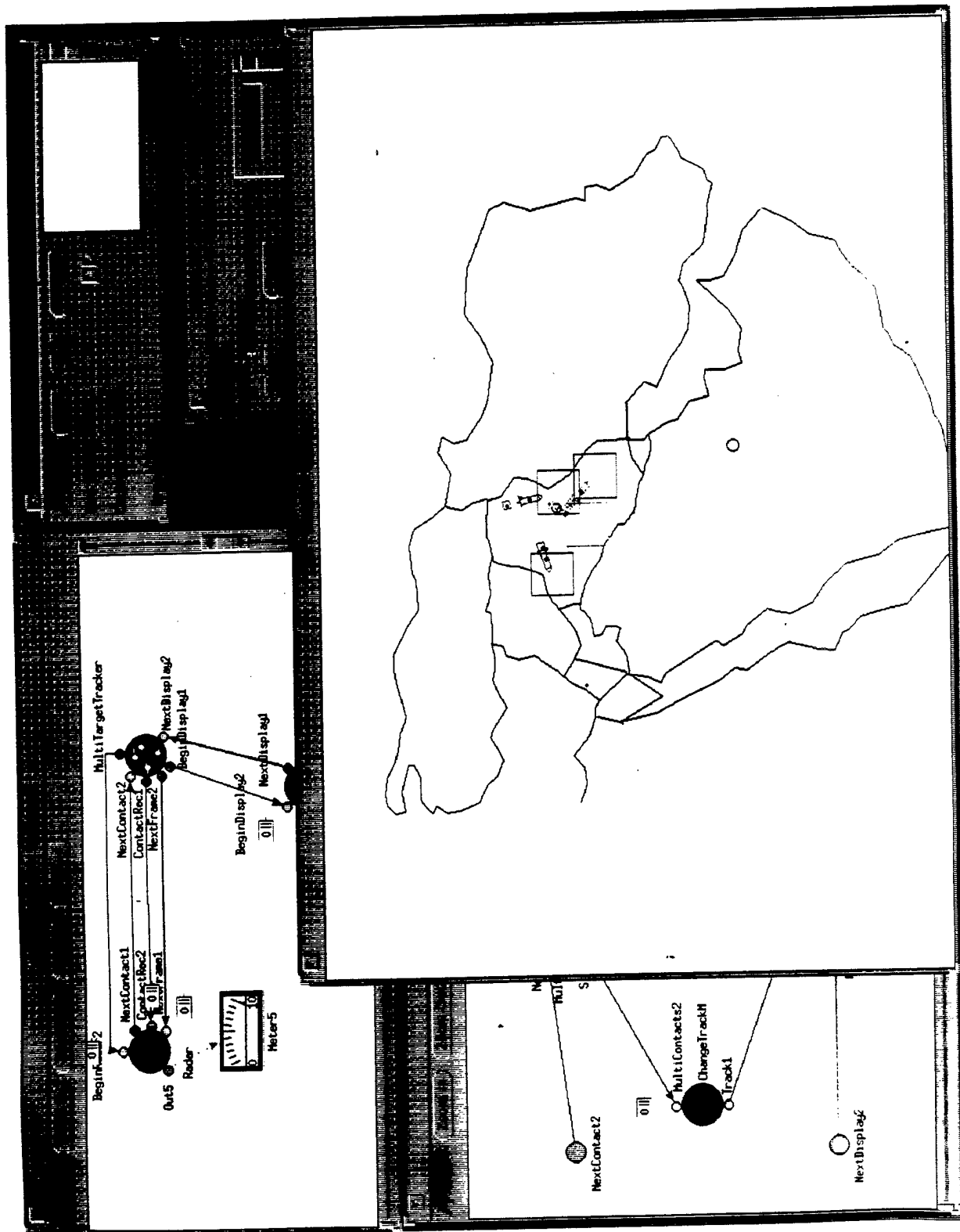


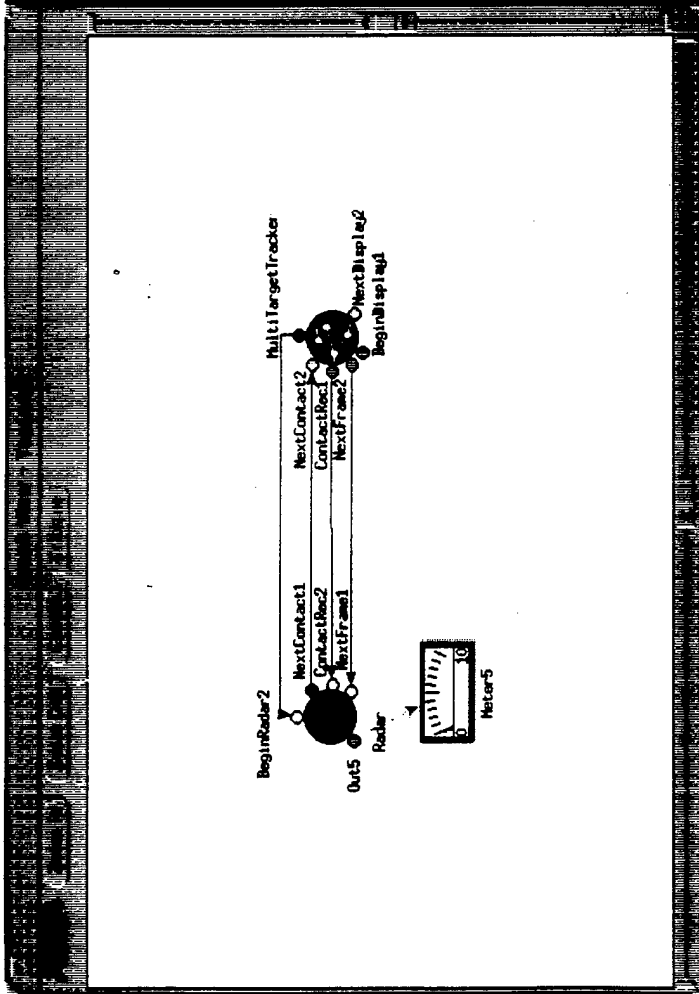
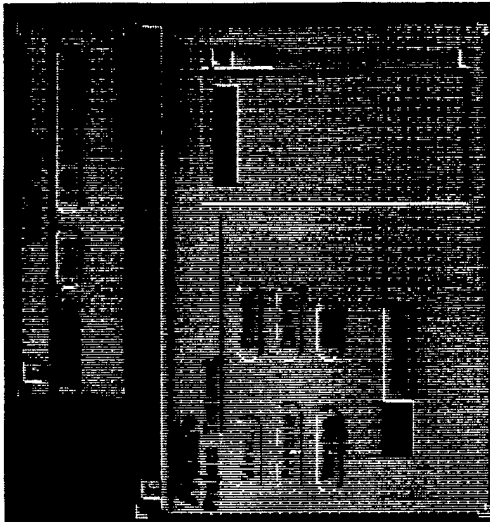


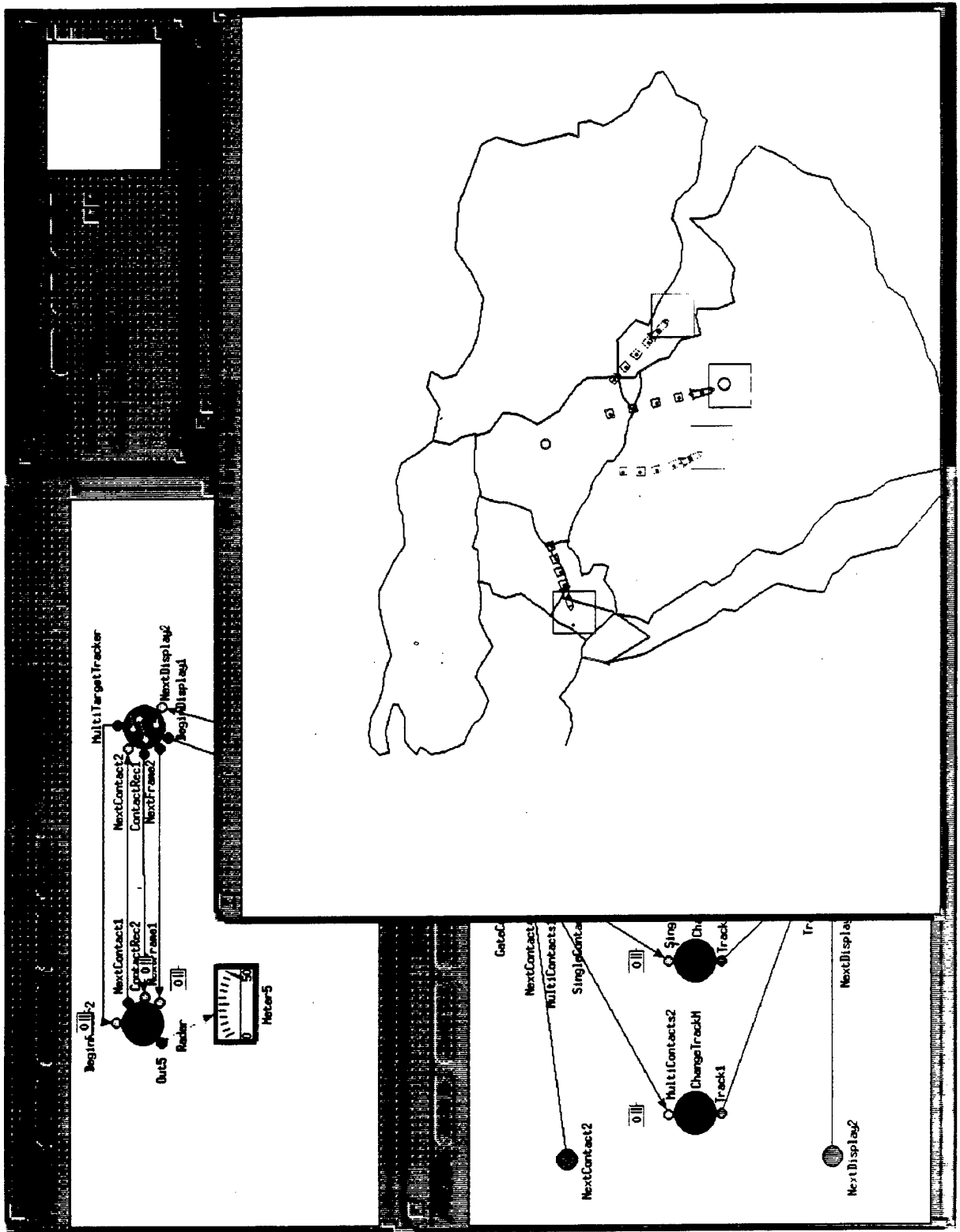


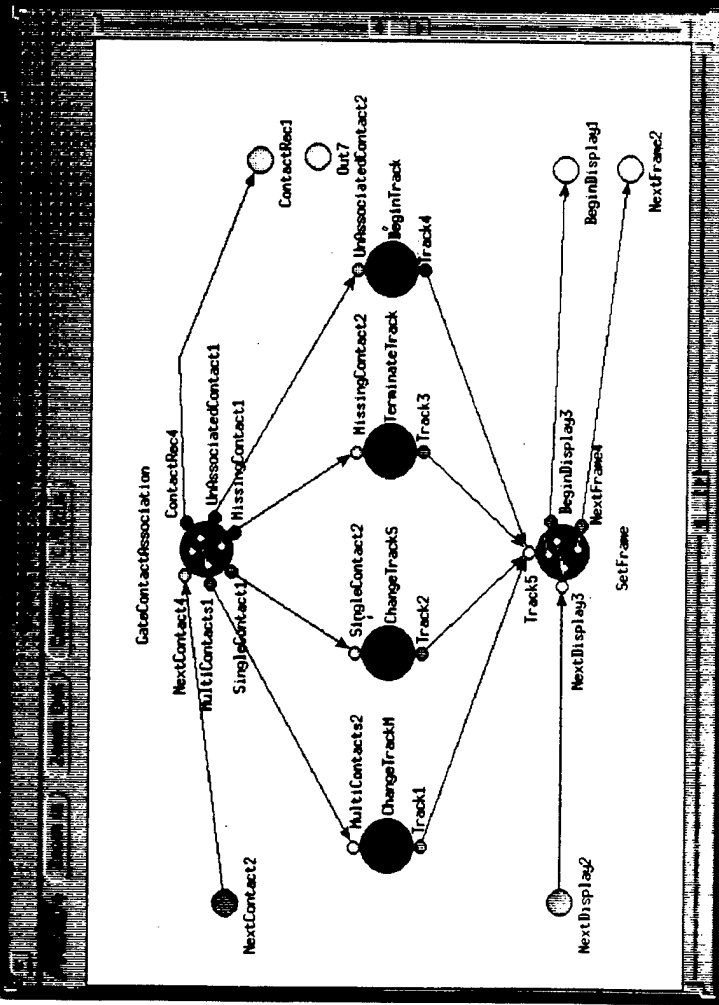
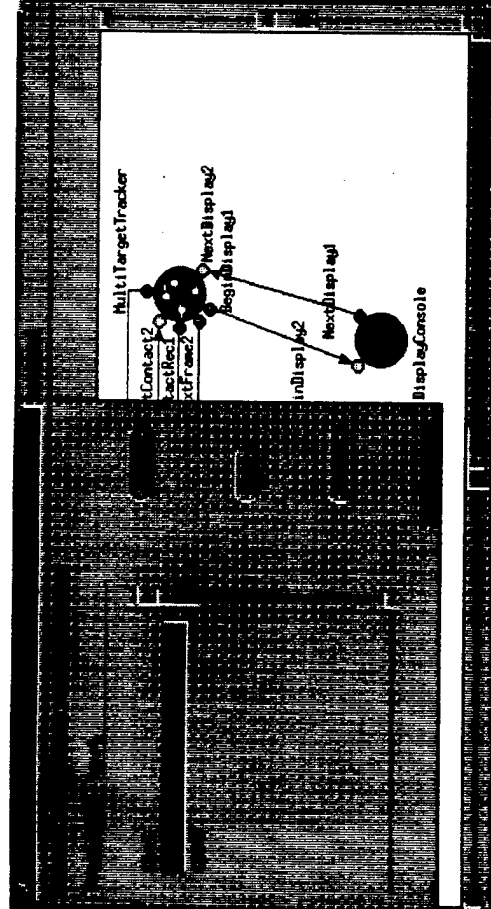
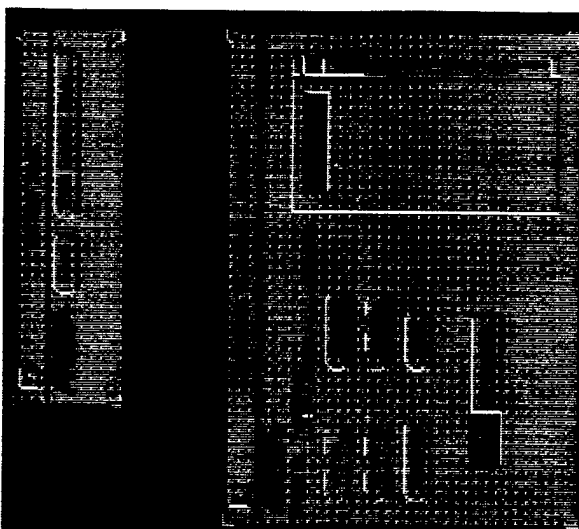


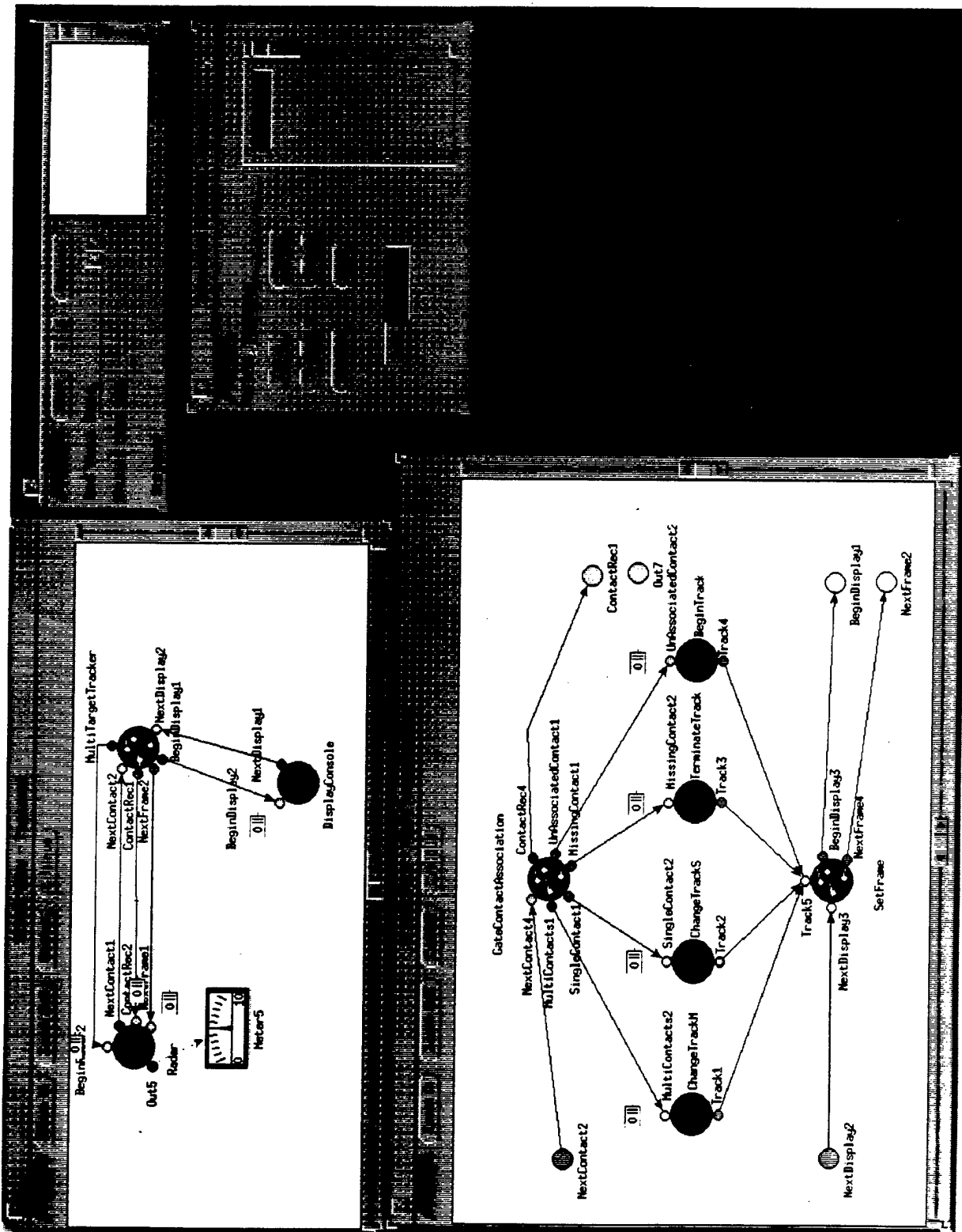














HARDWARE SOFTWARE CONFIGURATION

- SUN4 - SPARC STATION WITH SUN OS 4.1
- ONTOS OBJECT ORIENTED DBMS
- XVIEW (CURRENTLY DELIVERED WITH 4.1)
- PROTO+ SOFTWARE

***NOTE:

PROTO+ WILL BE AVAILABLE 30 APRIL 1991 WITH
DRAFT DOCUMENTATION

PROTO+ WILL BE AVAILABLE 30 JULY 1991 WITH
"APPROVED" DOCUMENTATION

PRESENTER: KATHRYN H. HILES

SDS COMMON FRAMEWORK



UNCLASSIFIED



SDS COMMON FRAMEWORK

KATHRYN H. HILES
TELEDYNE BROWN ENGINEERING
12 MARCH 1991

UNCLASSIFIED

UNCLASSIFIED



UNCLASSIFIED

SDS SOFTWARE ENGINEERING COMMON FRAMEWORK



- Provides the Software Engineering Environment (SEE)
- Portable to UNIX Based Hardware
- Defines interfaces where external tools can exchange and share data
- Defines the Standard Information Model
- Supports and Accommodates the Full 2167A Life Cycle Development
- Provides the Basis for Software Reuse
- Facilitates the Collection of Quality and Management Metrics
- Provides the Software Process Management

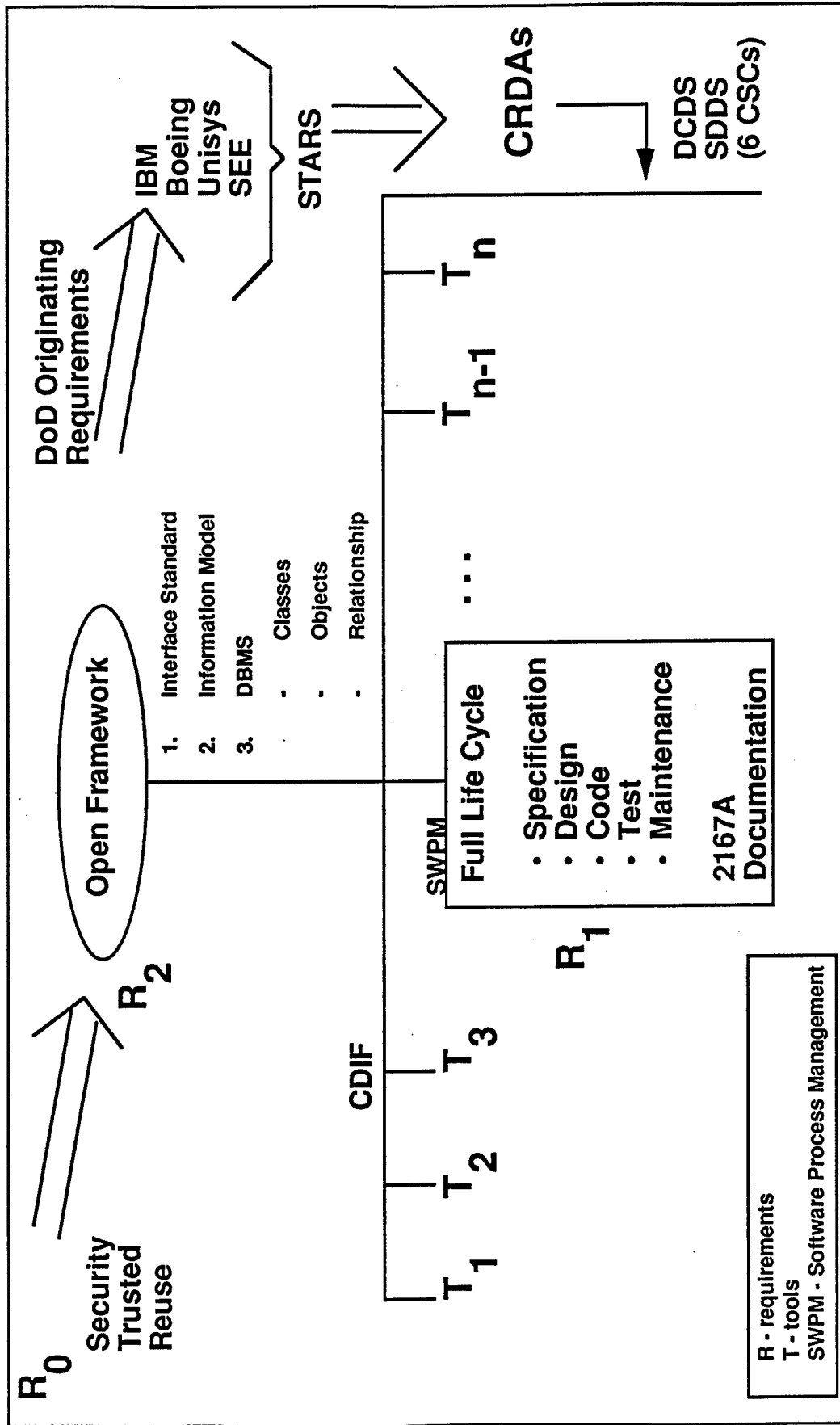
UNCLASSIFIED

UNCLASSIFIED



UNCLASSIFIED

FULL LIFE SUPPORT CYCLE



UNCLASSIFIED

UNCLASSIFIED



UNCLASSIFIED

SDS SOFTWARE ENGINEERING COMMON FRAMEWORK



- Provides the Software Engineering Environment (SEE)
- Portable to UNIX Based Hardware
- Defines interfaces where external tools can exchange and share data
- Defines the Standard Information Model
- Supports and Accommodates the Full 2167A Life Cycle Development
- Provides the Basis for Software Reuse
- Facilitates the Collection of Quality and Management Metrics
- Provides the Software Process Management

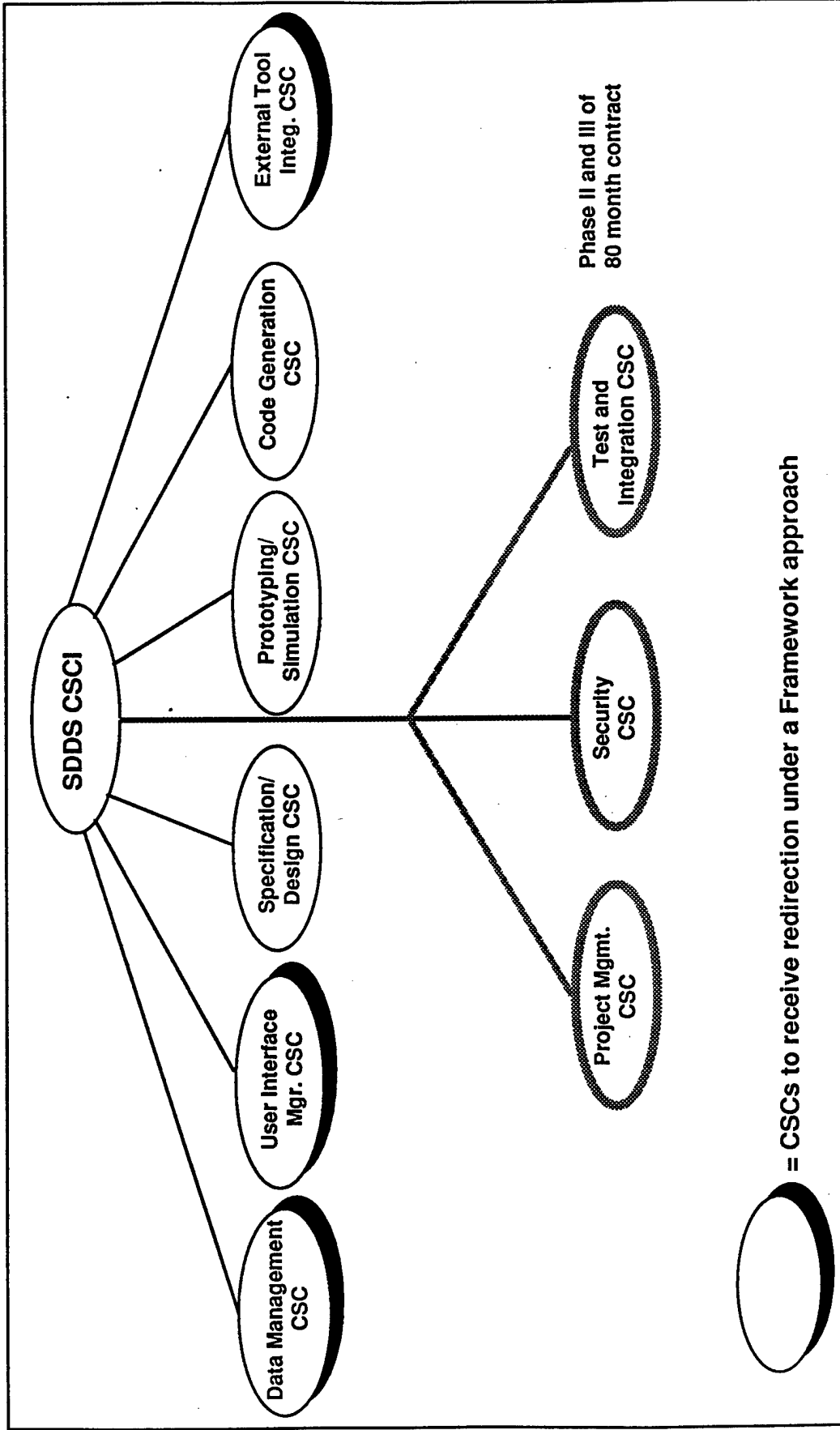
UNCLASSIFIED

UNCLASSIFIED



UNCLASSIFIED

SDDS FRAMEWORK CAPABILITIES



= CSCs to receive redirection under a Framework approach

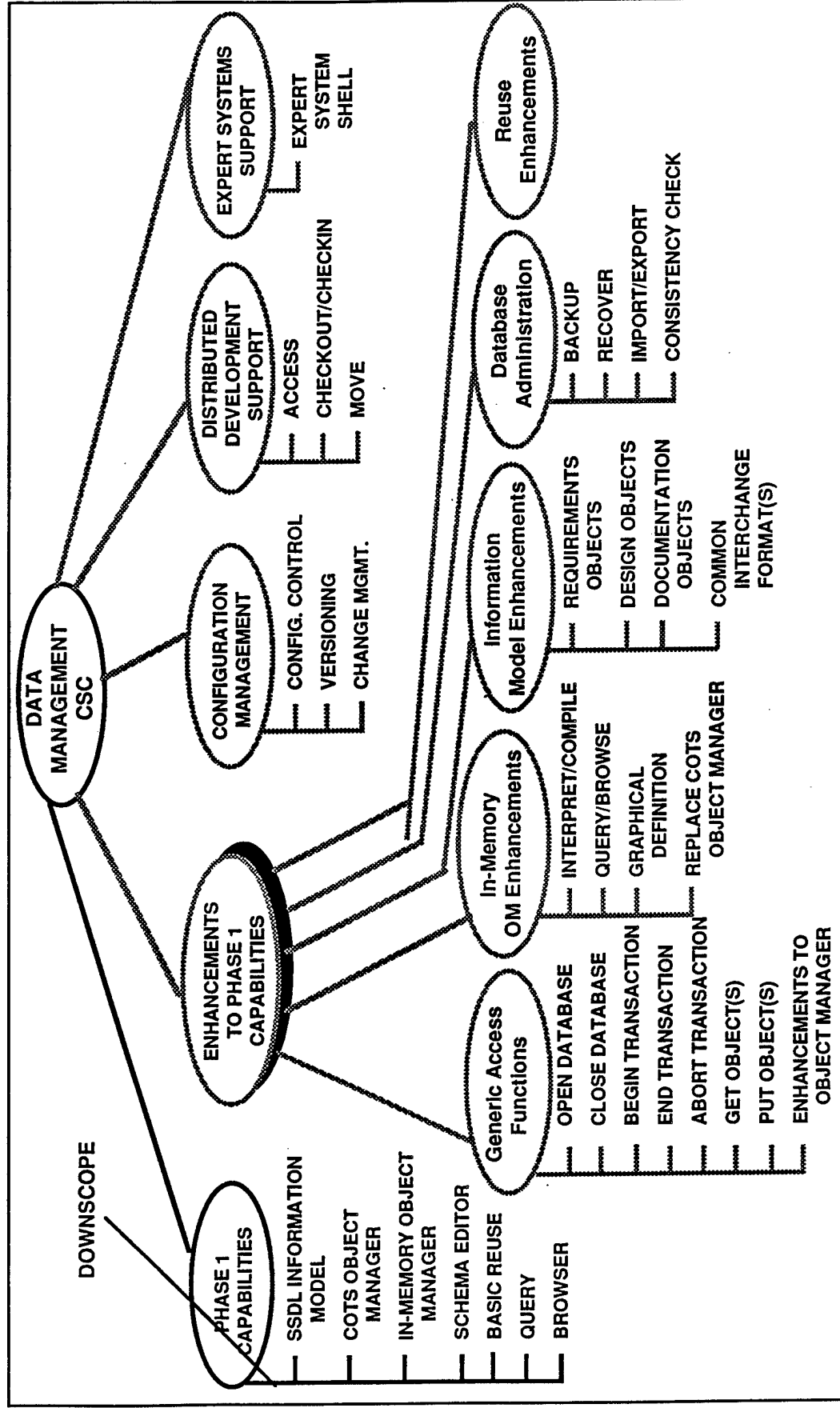
UNCLASSIFIED

UNCLASSIFIED



UNCLASSIFIED

SDDS's DATA MANAGEMENT CSC



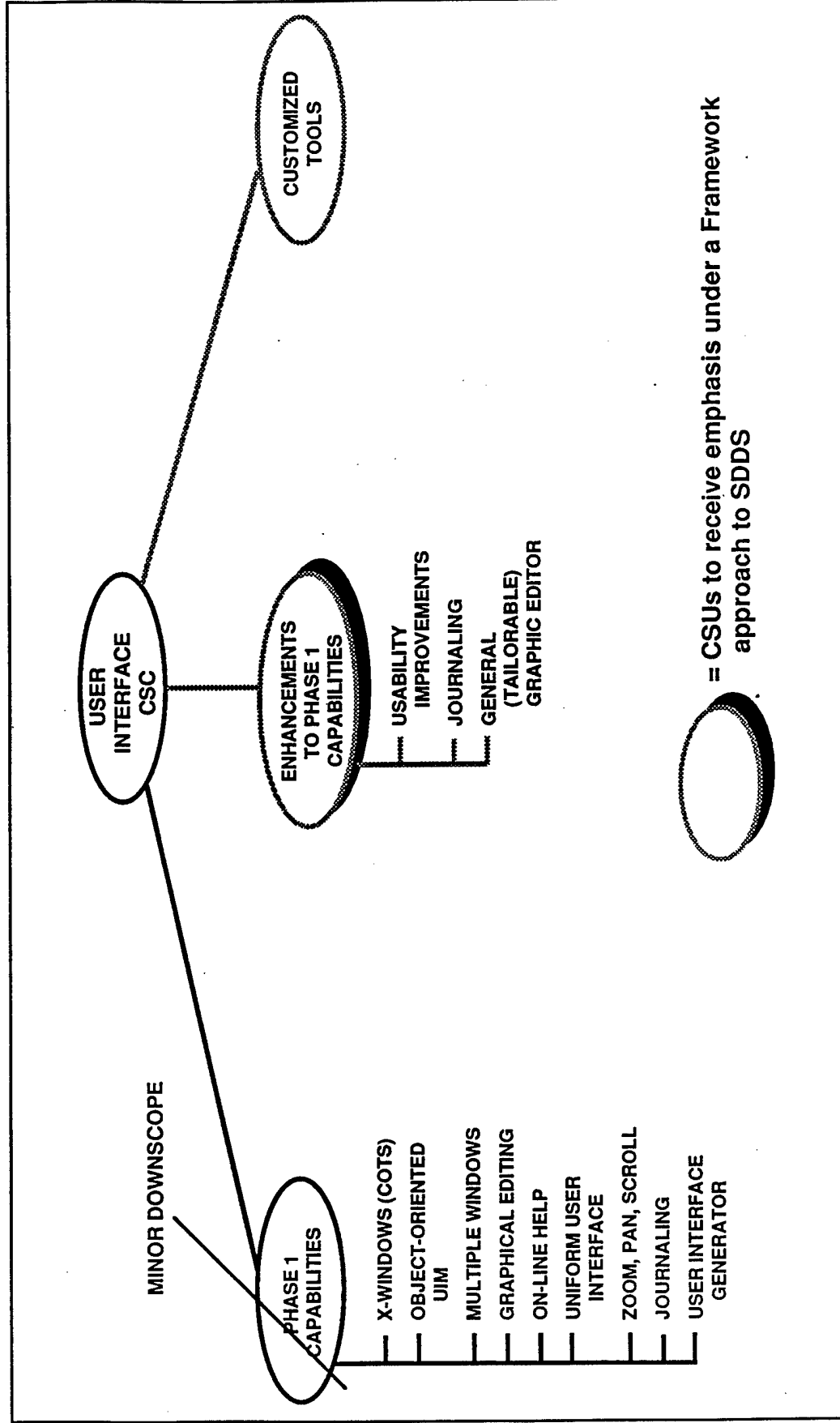
UNCLASSIFIED

UNCLASSIFIED



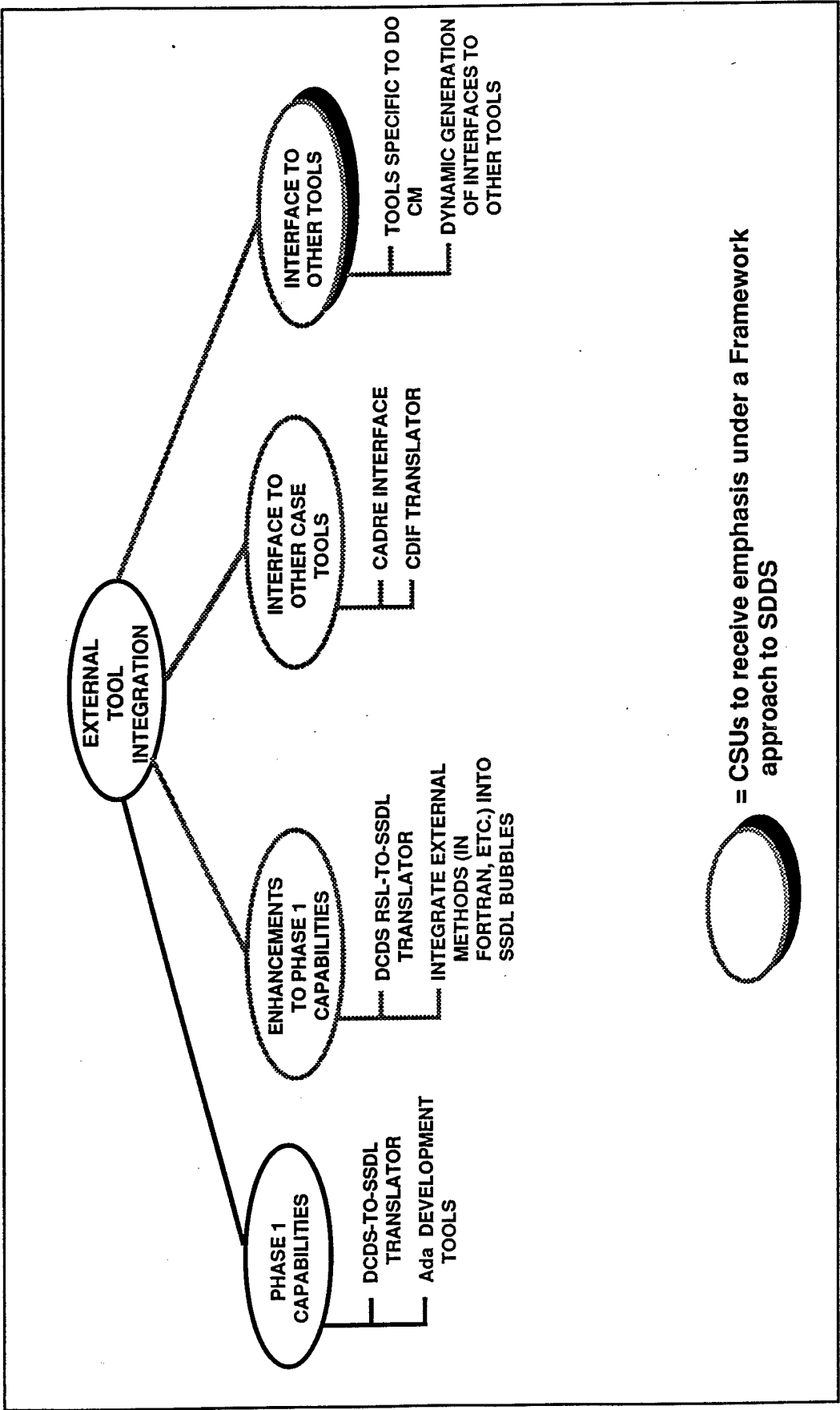
UNCLASSIFIED

SDDS's USER INTERFACE CSC



UNCLASSIFIED

UNCLASSIFIED





UNCLASSIFIED

SDDS COMMON FRAMEWORK DEVELOPMENT PLANS



- Common Framework Requirements from System Engineer Incomplete
- SDIO/SDC Task Force to Lead Rewrite SEE Document
- Prioritize Requirements in SDDS Scope
- FY92 Targeted for Redirection for Framework Development

UNCLASSIFIED

UNCLASSIFIED

PRESENTER: TOM MATSON/GLENN HUGHES II

THE RATIONAL ENVIRONMENT

THE RATIONAL ENVIRONMENT

Advanced Technology for Developing Ada Software Systems

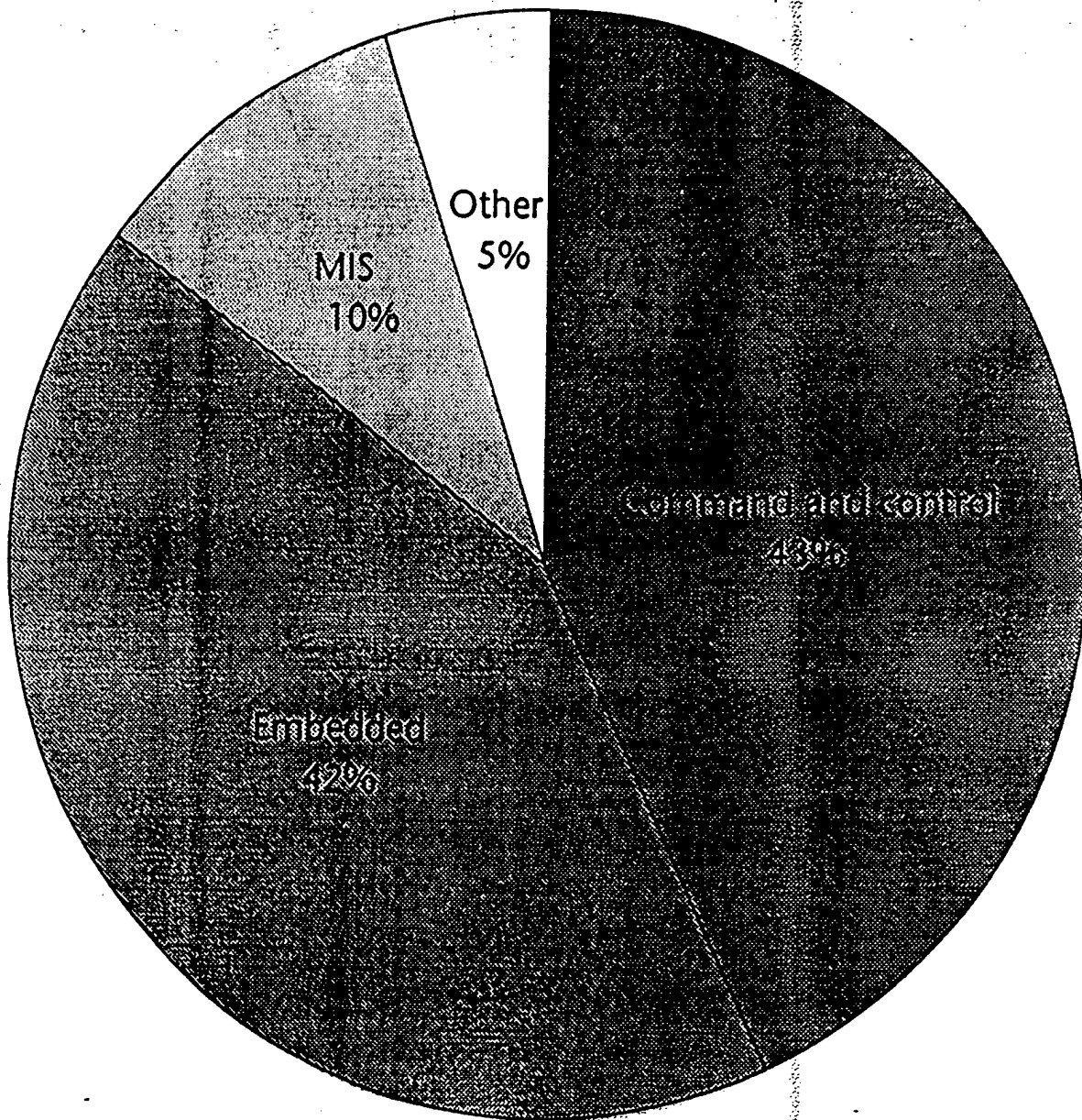
RATIONAL'S LARGEST CUSTOMERS

IBM	All major Ada programs including FAA's Advanced Automation System (AAS) for air traffic control
Bofors Electronics AB	All Ada projects including FS 2000 C³ and weapons control system
U.S. Army	ATCCS/CHS program, major C³ and MIS systems including AFATDS, SIDPERS-3, and STANFINS-R
Rockwell	Commercial avionics and space craft electrical power systems
Lockheed	NASA/SSE for Space Station Freedom, major C³ systems, and classified programs
McDonnell Douglas	Space Station Freedom, C³, and extra vehicular activity (EVA) systems
Ferranti Computer Systems	Naval C³/weapons control, trainers, and power plant automation systems
TRW	Major C³ systems including CCPDS-R, AWIS, FAAD C²I, and classified programs
Martin Marietta	Air traffic control, SDI/NTB, C³I, and robotics systems
U.S. Air Force	Major C³, MIS, and avionics systems

RATIONAL

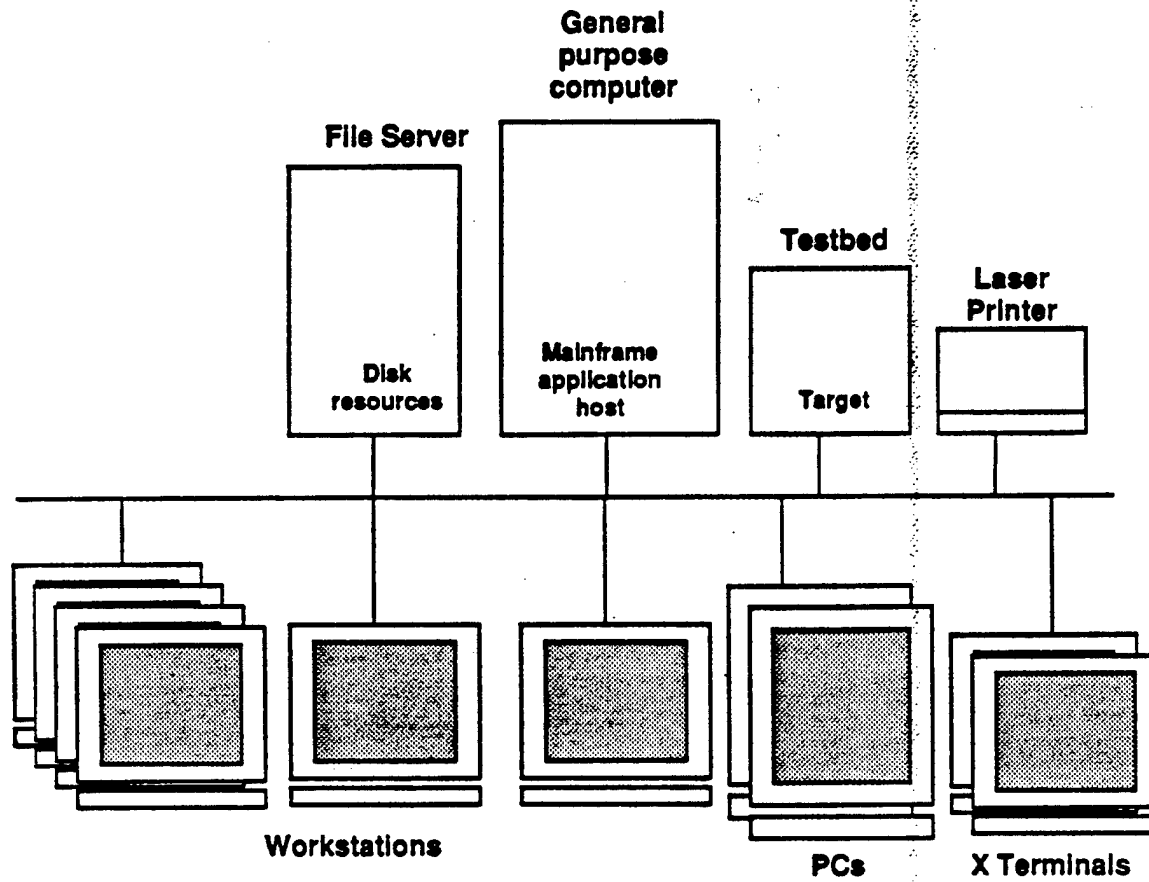
Customer Applications

Percentage of processors



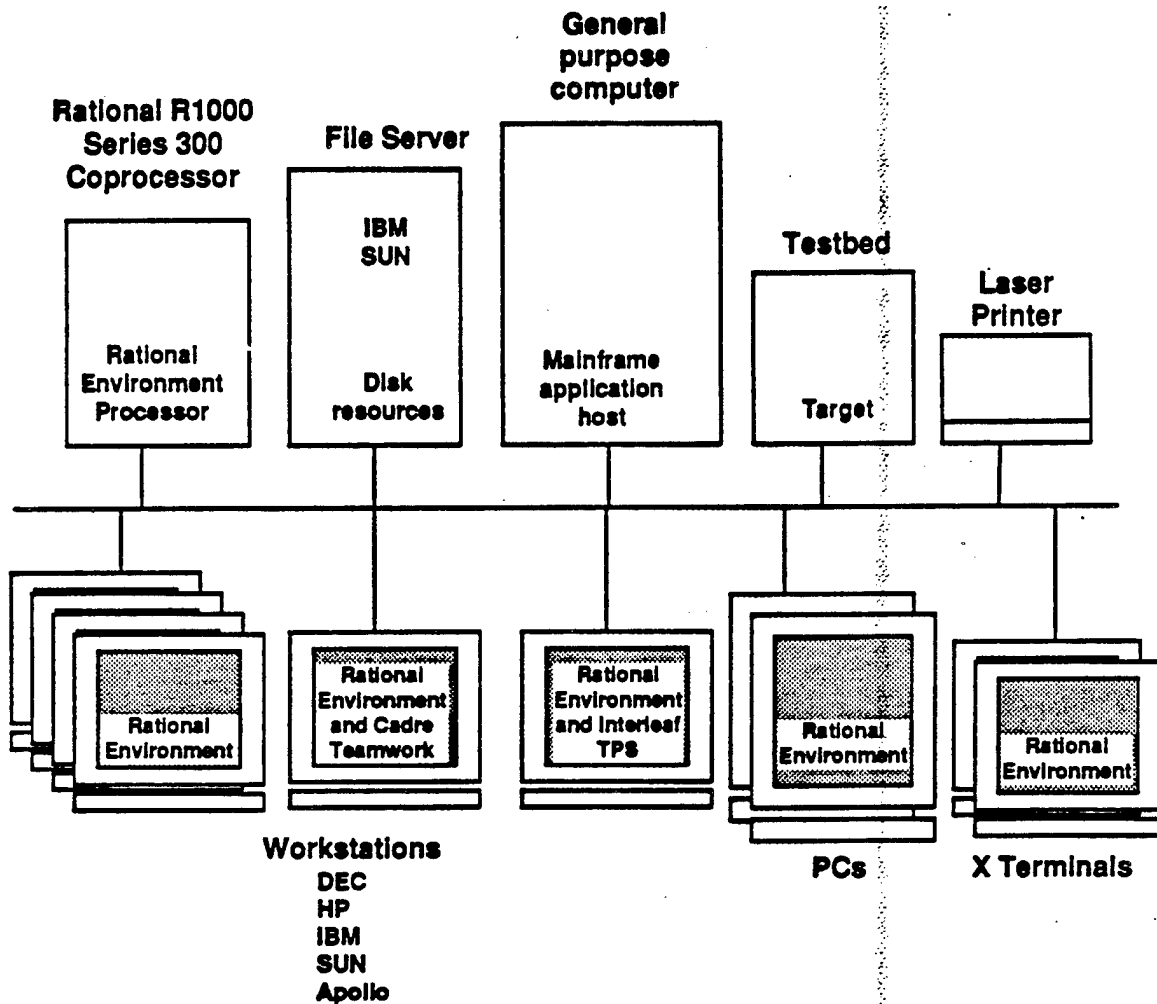
TRENDS IN SOFTWARE ENGINEERING

- **Iterative process model based on object-oriented methodology**
 - **Emphasis on graphical representations**
 - **Extensive use of prototyping**
 - **Emphasis on reusability**
- **Heterogeneous target environment**
 - **UNIX (including real-time applications)**
 - **Workstations will dominate**
- **Merger of systems engineering and software engineering**
- **Networked, workstation-based development environments providing increased automation**



RATIONAL

RATIONAL ENVIRONMENT ADDS SOFTWARE ENGINEERING SUPPORT TO NETWORKS



RATIONAL

Federal Aviation Administration Advanced Automation System (AAS)

Prime Contractor: IBM Federal Sector Division

Purpose: Replace United States Air Traffic Control System

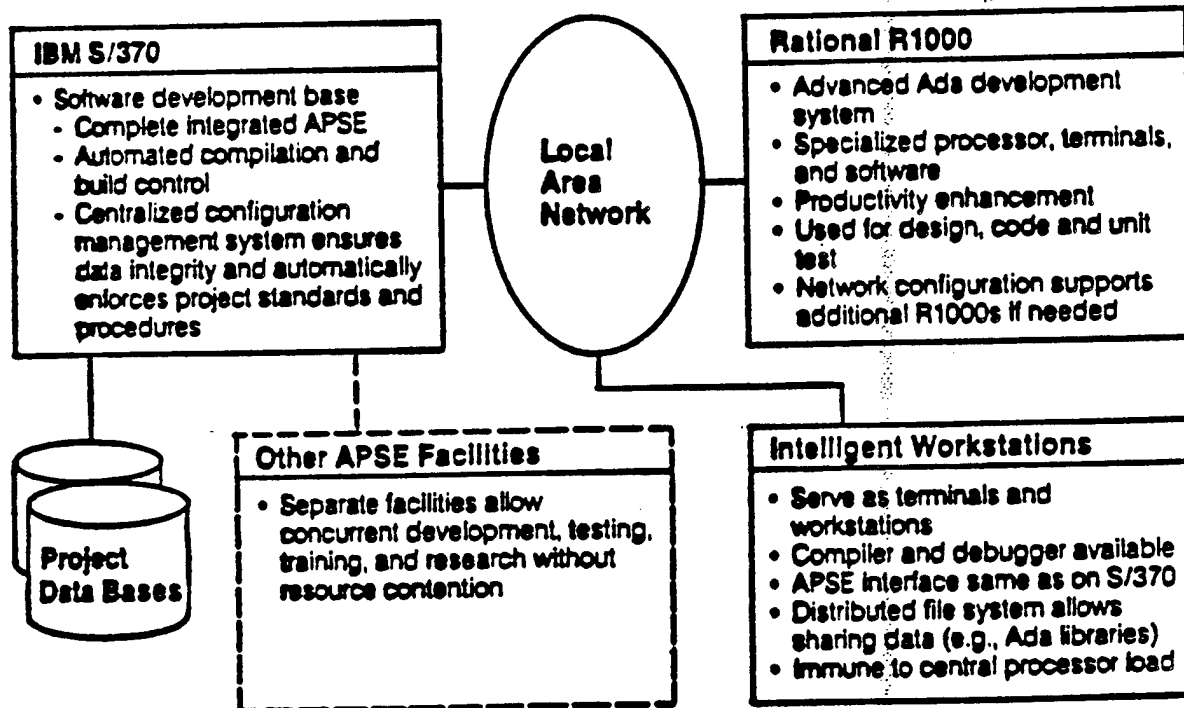
Size: 2.5M lines of code (Ada)
Maximum software engineering staff - 500
Development schedule - 10 years

Status: Systems architecture defined
Prototyping in progress
250 engineers trained

RATIONAL

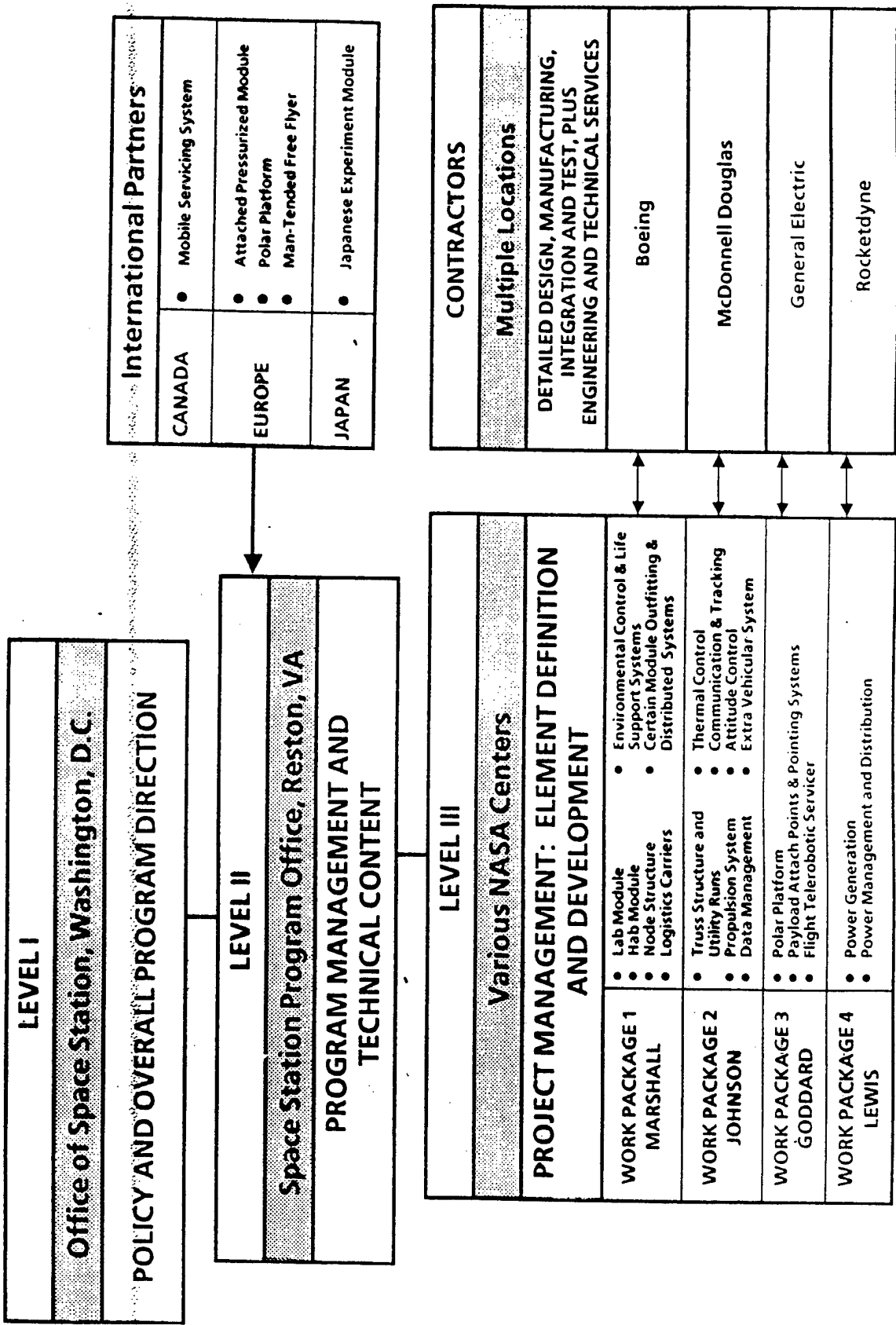
AAS SOFTWARE DEVELOPMENT ENVIRONMENT

FSD's Ada Programming Support Environment (APSE) configuration features a powerful central processor, the Rational R1000 system, and network intelligent workstations that distribute the high-resource demands associated with Ada development.



RATIONAL

NASA SPACE STATION FREEDOM PROGRAM ORGANIZATION





SOFTWARE PRODUCTION INITIATIVE

SSE OBJECTIVE/GOALS

o OBJECTIVE:

- ESTABLISHMENT OF A COMMON SOFTWARE PRODUCTION AND MAINTENANCE ENVIRONMENT FOR THE NASA SPACE STATION FREEDOM PROGRAM (SSFP)

o GOALS:

- SUPPORT THE LIFE CYCLE MANAGEMENT OF SSFP OPERATIONAL SOFTWARE
- MINIMIZE THE OVERALL LIFE CYCLE COST OF THAT SOFTWARE
- FACILITATE THE MIGRATION OF CAPABILITIES FROM GROUND TO SPACE
- FACILITATE THE INTEGRATION OF SSFP OPERATIONAL SOFTWARE



WHAT IS A NASA SSE?



- * COMMON SOFTWARE DEVELOPMENT METHODOLOGY

- * COMMON STANDARDS AND RULES

- * COMMON INTEGRATED TOOLS

- DOCUMENTATION

- DESIGN

- CODE

- SOFTWARE TESTING

- MANAGEMENT

- CONFIGURATION MANAGEMENT



WHY A NASA SSE?

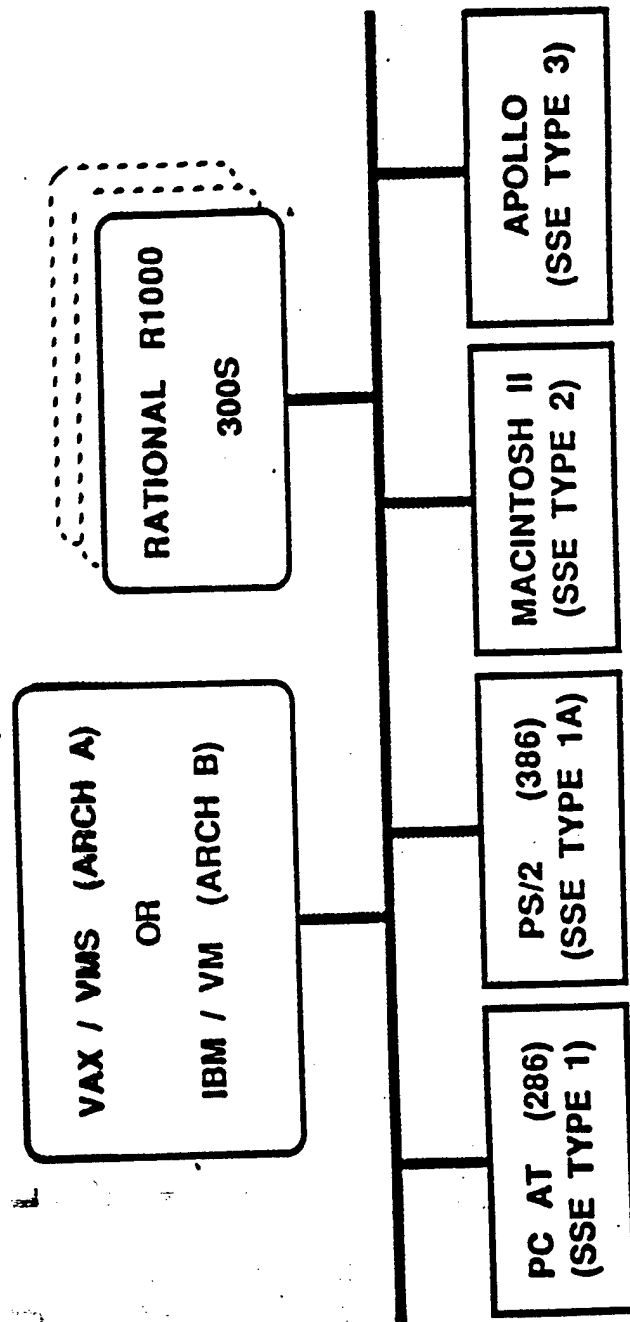


- * 30+ CONTRACTORS DEVELOPING SSFP SOFTWARE
- * 2+ MILLION LINES OF Ada FLIGHT SOFTWARE
(initial load)
 - * 30 YEAR PROGRAM
 - * REDUCE RISK AT INTEGRATION
 - * REDUCE LIFECYCLE COST



SOFTWARE PRODUCTION INITIATIVE

THE SSE HARDWARE ARCHITECTURE





WHY WE SELECTED RATIONAL



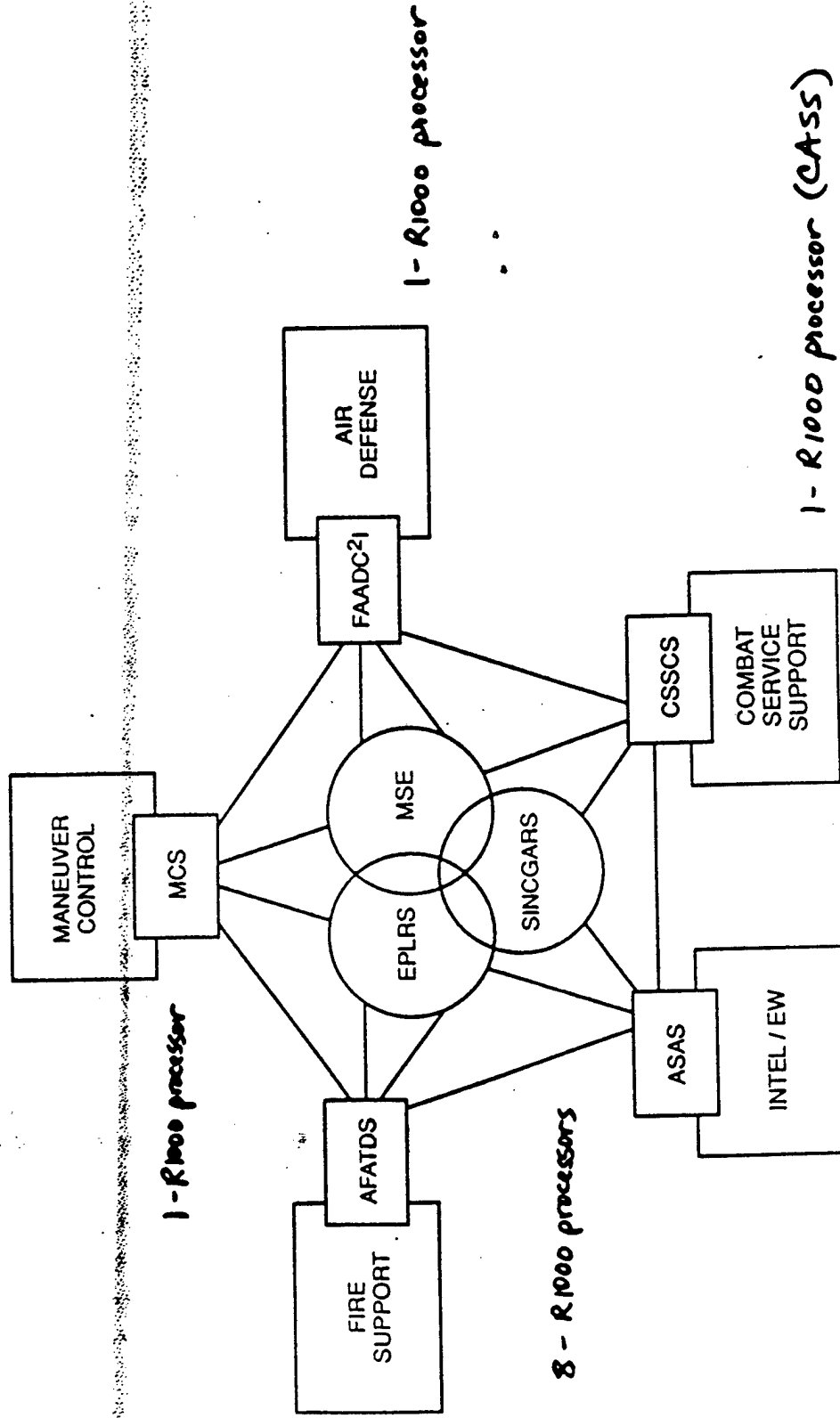
-
- * Ada IS ONLY APPROVED LANGUAGE
 - * REALIZATION THAT COTS IS THE ONLY PATH
 - * SOFTWARE DEVELOPMENT TOOLS NEEDED IN 1990
 - * RATIONAL OFFERED THE MOST -- NOW
 - RDF
 - TBU
 - RPI
 - RTI
 - CMVC
 - CDF
 - * COMMITMENT & SUPPORT FROM RATIONAL



ATCCS ENVIRONMENT

w/ Rational Development Systems

ARMY PROGRAMS

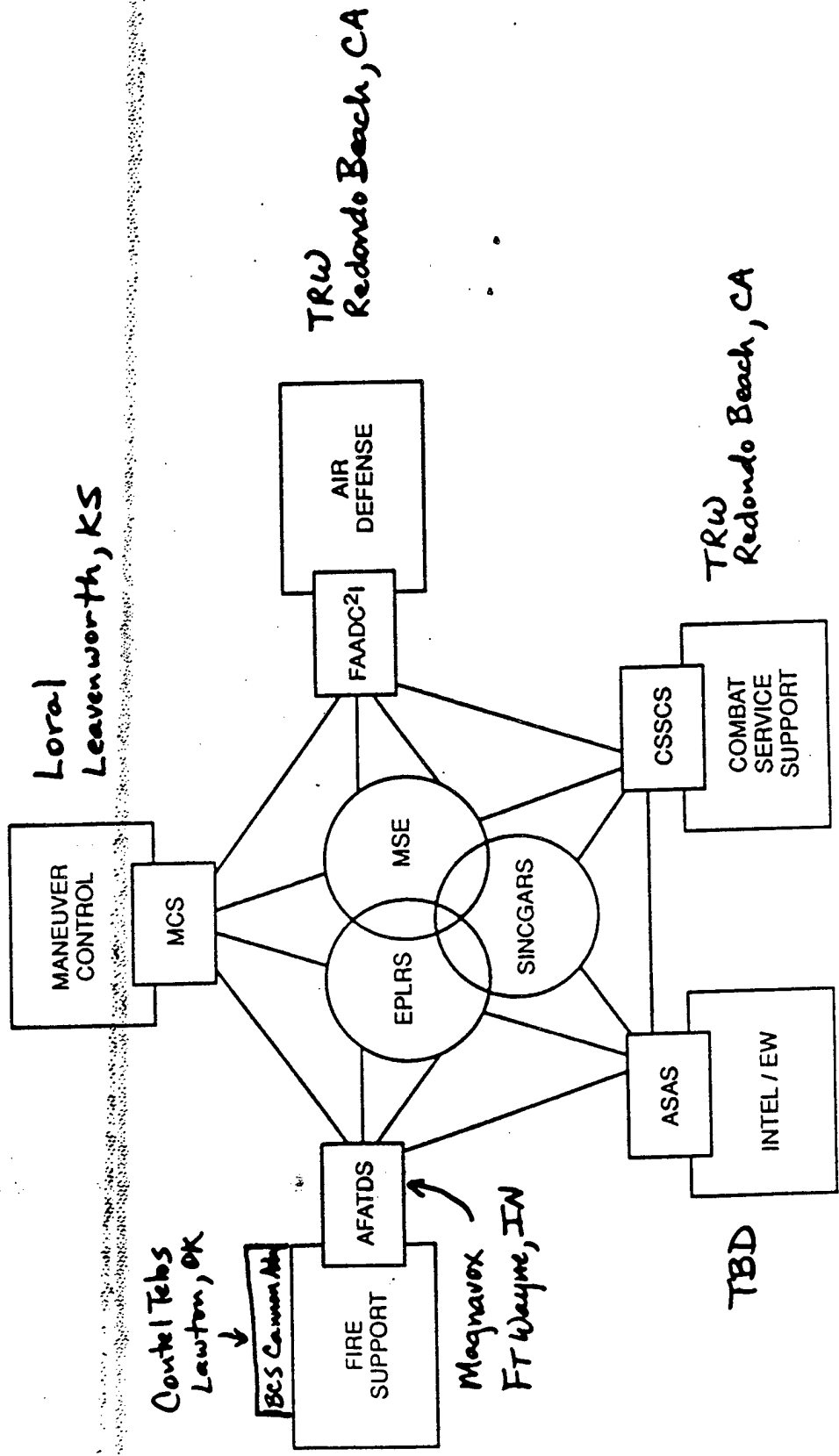


- C2 NODES PRIORITY TO VERTICAL INTEGRATION
- C2 INTEROPERABILITY REQUIREMENTS POORLY DEFINED
- REQUIREMENTS DEVELOPMENT PROCESS EMPHASIS ON NODAL ELEMENTS
- NODAL ELEMENTS IN VARIOUS PHASES OF ACQUISITION PROCESS



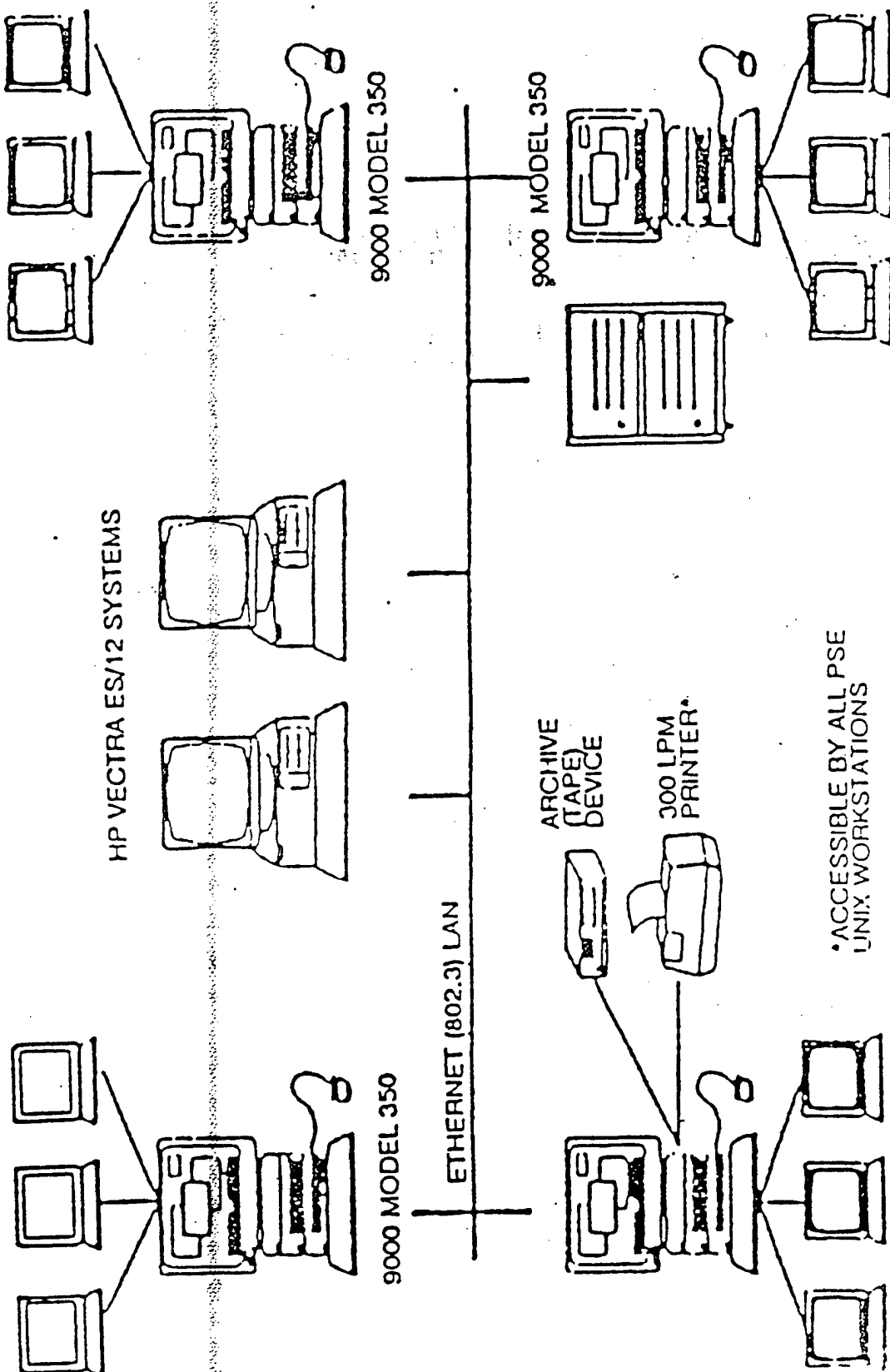
ATCCS ENVIRONMENT Contractors

ARMY PROGRAMS



- C2 NODES PRIORITY TO VERTICAL INTEGRATION
- C2 INTEROPERABILITY REQUIREMENTS POORLY DEFINED
- REQUIREMENTS DEVELOPMENT PROCESS EMPHASIS ON NODAL ELEMENTS
- NODAL ELEMENTS IN VARIOUS PHASES OF ACQUISITION PROCESS

CHARACTER TERMINALS (12 TOTAL)



ATCCS BASIC PSE SYSTEM

Rational Status Update

- ECP 077 is approved
- ATCCS Rational Users
 - TRW/FAADC2I
 - Magnavox/AFATDS
 - Contel Telos/BCS Cannon Ada
 - TRW/CASS
- Other Army Rational Users
 - TRW/AWIS
 - CSC/STANFINS-R
 - Statistica/SIDPERS-3
 - CSC/STARFLARS-M
 - SofTech/RAPID
- Rational-HP integration products are on schedule

IMPACT OF PRODUCTIVITY INCREASES ON PROJECT COST

Productivity Increase (%)	Aggregate Code Production Rate	Eng-Months Saved	\$ Saved
25	250	800	7,667,000
50	300	1,333	12,778,000
100	400	2,000	19,167,000
150	500	2,400	23,000,000
200	600	2,667	25,566,000
250	700	2,857	27,381,000
300	800	3,000	28,156,000

Assumptions:

1. Project size equals 800,000 source lines of code.
2. Baseline code production rate equals 200 source lines of code per engineering-month.
3. Cost per engineering-month is the loaded cost including wages, benefits, facility allocation, management burden, and profit burden. For this case, the cost per engineering-month is assumed to be \$9,583, or \$115,000 per engineering-year.
4. Aggregate code production rate is the number of source lines of delivered code divided by the aggregate number of engineering-months required from project start through first delivery.
5. The number of engineering-months saved is equal to the number of engineering-months required at the increased productivity less the number required at the base rate.
6. \$ Saved equals the number of engineering-months saved multiplied by the cost per engineering-month.

RATIONAL

BUSINESS CASE SUMMARY

	Bofors	CSC	IBM Systems Integration Division
Application	Integrated command and weapon-control system for ships	MIS/Financial accounting system	Command and control-satellite ground system
Size (source lines of code)	1,500,000 x 5 (5 ship designs)	2,000,000	650,000
Status	First ship delivered	Acceptance testing	Top-level design (PDR) completed
Productivity Improvement	118%	106%	93%
Cost savings	\$31.9M	\$24M	\$1.0M
Payback (savings/ investment)	6.5x	9.7x	1.4x*

* Payback is computed through end of top-level design phase only.

Productivity = $\frac{\text{source lines of code delivered}}{\text{total engineering hours}}$

RATIONAL

NASA Space Station Leads Way In Software Development

By Bill Suydam

The software technology for adaptable reliable systems (STARS) program could learn a few lessons from NASA. Although the DOD is still hell-bent on developing its own software-development environments, NASA recently decided to purchase commercial, off-the-shelf solutions for its Space Station. NASA's decision to use Rational Inc.'s Rational Environment rather than build its software support environment (SSE) from scratch, as is being done in the STARS program, blazes a new path in software development the DOD would be wise to follow.

The original NASA software development plan for the Space Station was a disaster. It seems fairly clear that the problems were inherent in the way the original Space Station software development program was designed.

That original plan called for handling the overall project's design and configuration management through a program called automated process control environment (APCE) to be put in place by Lockheed through a subcontractor. APCE turned out to be, in the words of current NASA SSE Project Manager Mike Gremillion, "user-vicious."

But project management in programs the size of the Space Station change frequently. Deputy Director Robert Moorehead joined the program in 1989 and rightly concluded that we could probably build a useable sidewalk to the Space Station before the Space Station software itself would ever be developed. Frank Barnes, Lockheed's program manager for the SSE surveyed the available options and proposed dumping the APCE in favor of the Rational Environment. Barnes redirected the SSE project, and in March of this year, Gremillion was brought on board. He instituted a formal configuration-control program for the project.

Gremillion currently estimates that the project has saved from a year to a year and a half in development time



The proposed NASA Space Station has benefited from the use of off-the-shelf software. (Photo courtesy of Intelligent Light.)

by substituting the Rational Environment for the APC. Unfortunately, as he freely admits, that puts the Space Station program just "not quite as far behind." Meanwhile, selection of the target compiler for the code being developed is still underway, with Alsys and DDC-I proving the finalists in the contest. That compiler will run on the VAX/VMS or IBM platforms and produce code for a bare 80386 platform as well as the multiplexors/demultiplexors (MDMs) of the Space Station platform itself. The Rational incremental compiler is serving as the development compiler for the project.

The SSE itself comprises more than just a combination of software tools for configuration management. It includes several components — software-engineering tools, hardware tools,

operating system interfaces, software-development rules and procedures, and software standards. The SSE, in turn, will be used to provide a development environment for the Space Station's data management system (DMS). The DMS is really more than its name suggests, since it comprises a set of application-level services that rest on the underlying Lynx real-time Unix operating system.

The choice of an existing development system for the Space Station, especially compared to the custom-coded alternative originally proposed, is clearly justified. But we think the larger lesson here is one that should have been learned many times over not only by NASA, but by STARS, the DOD and other agencies responsible for major software-development projects.

An undertaking such as the Space Station program is so vast that no single vendor's solutions really encompass all that is required. But going with off-the-shelf solutions whenever possible — especially in software development, where most of the time and money get spent in a major program — is a step in the right direction. It should be a matter of policy in programs such as the Space Station to rely on commercially available development environments, leaving the design of such environments to the commercial vendors whose products must compete in the marketplace. It is inappropriate for the design of those environments to be turned over, in effect, to government committees.

Nor does this one change in practice and philosophy redeem the entire NASA Space Station Freedom program. Current funding and staffing for the program are totally inadequate to the goal of launching the Space Station by the turn-of-the-century. And, as this goes to press, NASA still has not announced the vendor of the target compiler, although this decision was reportedly made some months ago. □

PRESENTER: PAUL LARSON

Ada CODE GENERATION FOR REAL-TIME SOFTWARE

Ada Code Generation for Real-Time Software:

Space Station Freedom
Satellite Flight Software Testing

March 12, 1991
39th CRIM
Huntsville, AL

Topics to be Discussed

- ◆ An Introduction to Integrated Systems, Inc.
- ◆ Modifying the Conventional Software Development Process
- ◆ ISI Tools for Modeling, Simulation, Code Generation and Real-Time Testing
- ◆ Application Examples
 - Space Station Freedom, Boeing Aerospace
 - Satellite Flight Software Testing
- ◆ Summary



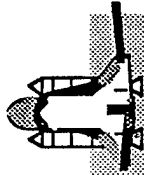
An Introduction to Integrated Systems, Inc.

- ◆ Principle Facilities in Santa Clara, California
- ◆ Incorporated in 1981
- ◆ Approximately \$20M in Sales,
125 Employees, Publicly Held
- ◆ Provider of Tools and Consulting Services
for Real-Time Systems Development



Examples of Real-Time System Design Applications

Aerospace



Aircraft flight path control
Satellite pointing and tracking
Jet engine control
Hydraulic system management
Space station environmental control

Automotive



Engine fuel flow management
Power-train performance optimization
Emission reduction system
Electronically controlled transmission systems
Four wheel steering system
Active suspension system
Anti-lock braking system
Cruise control system

Industrial



Thickness control in aluminum rolling mills
Quality optimization in cement kilns
Optimization of yield in distillation columns
Material handling
Accuracy and speed of wafer steppers

Computer Peripherals



Disk drive servos
Servos for optical disk
Printer controllers

Others

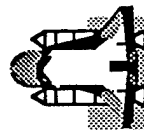
Robot task planning and path optimization
Power electronics design
Power plant control
Emergency/safety management systems



ISI Provides Competitive Edge:

- Higher Reliability
- Higher Performance
- Lower CGS

Examples of ISI Customers



Aerospace

Boeing
Kawasaki
Lockheed
MartinMarietta
Matra
NASA
Northrop
Rockwell

Automotive



BMW
Bosch
Ford
General Motors
Honda
Motorola
Nissan
TRW

Manufacturing



ALCAN
Amoco
Fluor
Fuji Film
General Electric
Kaiser Aluminum
Kao
Nikon

Computer Peripherals



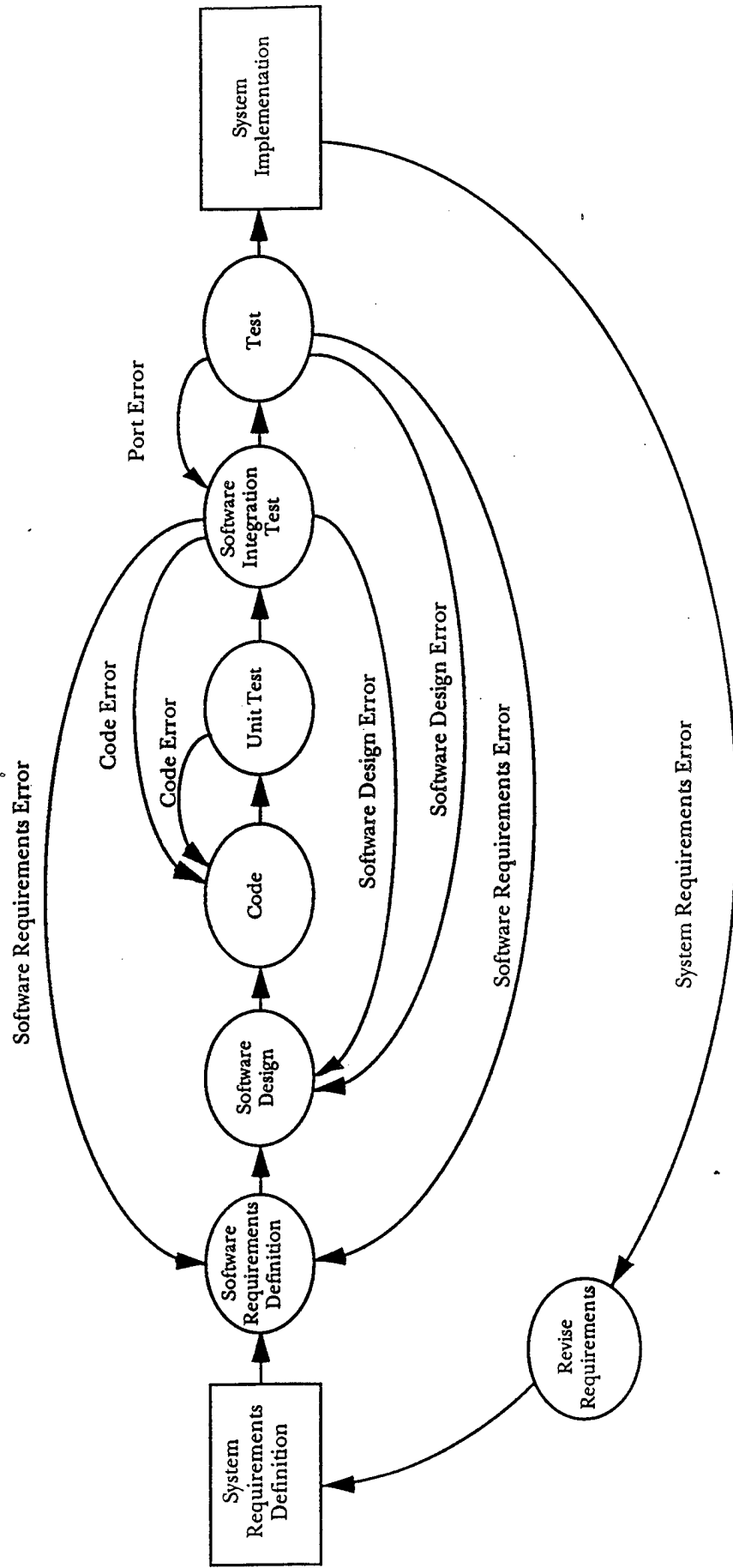
Control Data
Data General
Digital Equipment
Fujitsu
General Electric
Honeywell
Matsushita Electric
Olympus

Universities

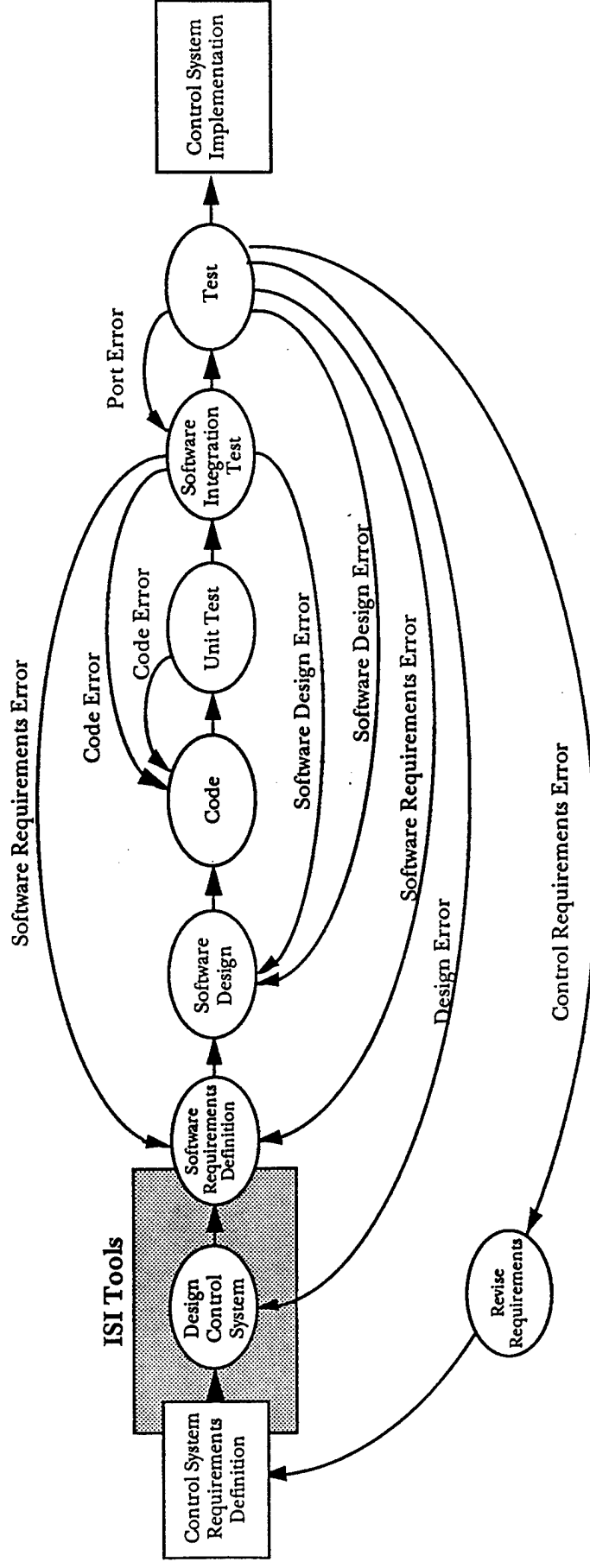
Carnegie Mellon
Iowa State University
MIT
Ohio State
Stanford
Univ. of California
Univ. of Colorado
Univ. of Michigan



Conventional Real-Time Software Development



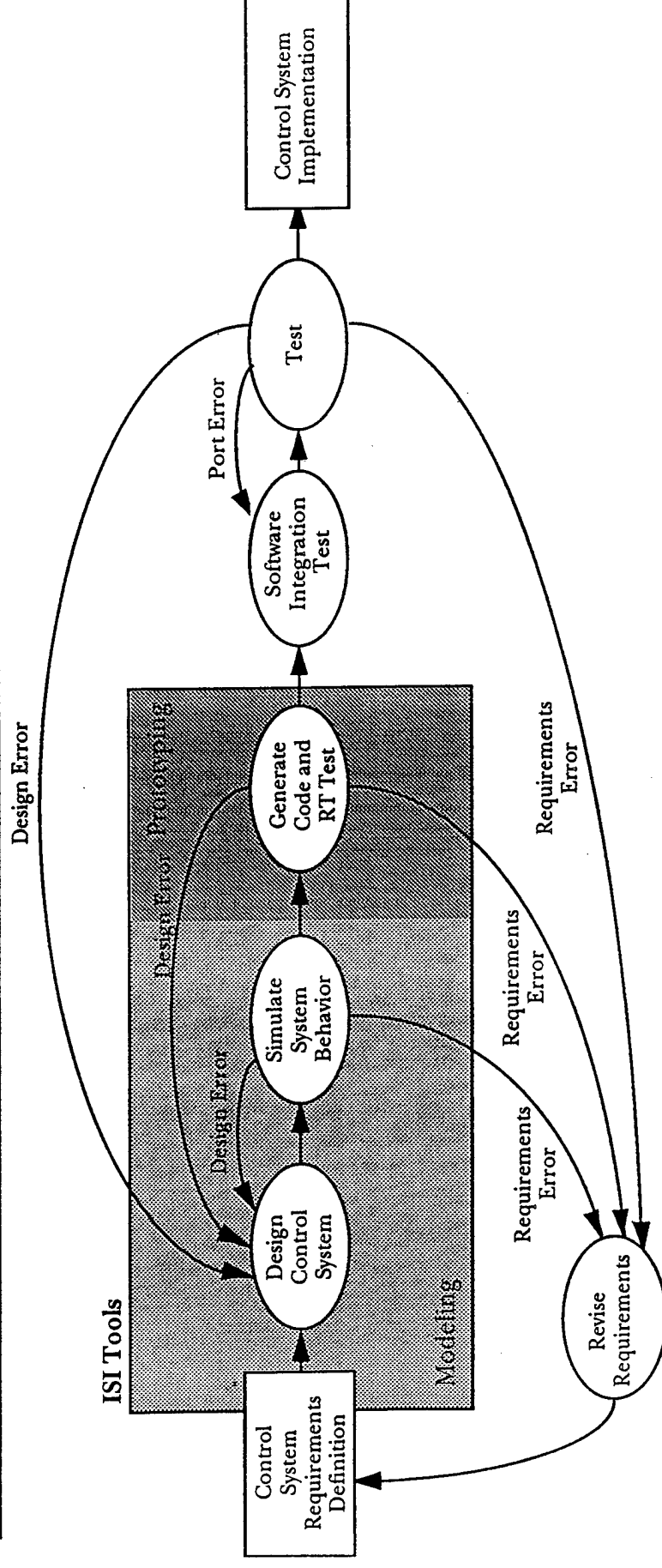
Conventional Real-Time Control System Development



- ◆ For complex controllers, an additional "Design Control System" step is required
- ◆ ISI's Tools have traditionally performed control analysis, design and simulation



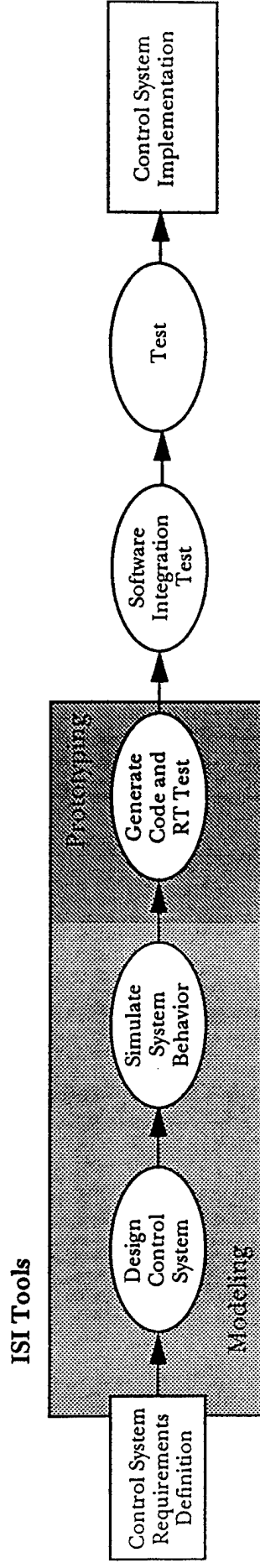
Prototyped Real-Time Control System Development



- ◆ ISI Tool enhancements support modification of the control design process
 - Requirements errors are identified earlier
 - Software integration and test reliability is enhanced
 - Risk, time and cost to develop are reduced



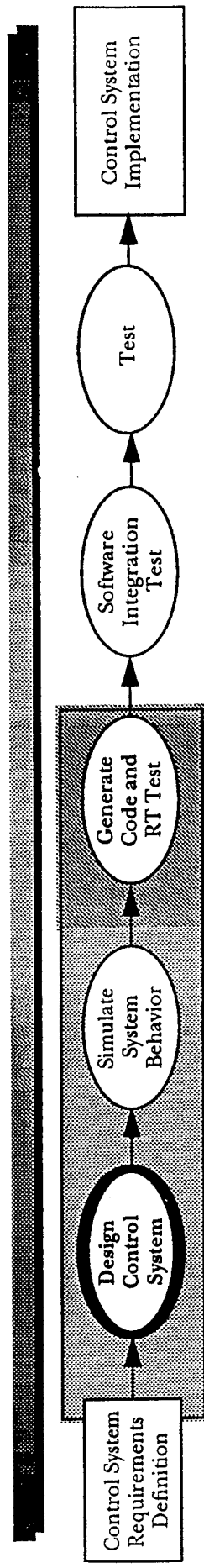
ISI's Software and Hardware Tools



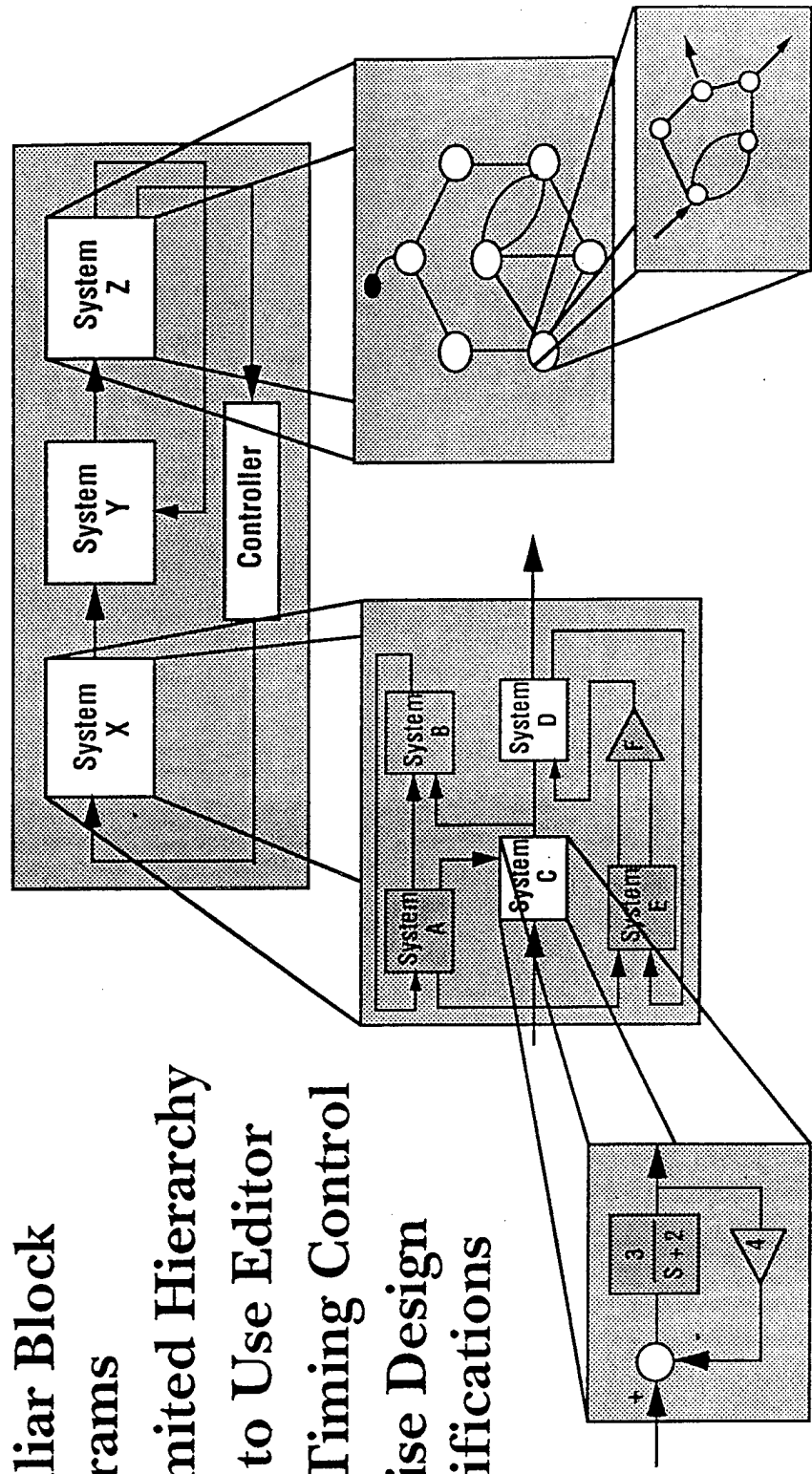
- ◆ Graphical Modeling
- ◆ Simulation
- ◆ Code Generation in Ada, C, or FORTRAN
- ◆ Real-Time Simulation, Test, Control and Data Acquisition



Modeling with the SystemBuild Block Editor



- ◆ **Familiar Block Diagrams**
- ◆ **Unlimited Hierarchy**
- ◆ **Easy to Use Editor**
- ◆ **Full Timing Control**
- ◆ **Precise Design Specifications**





Algebraic

Gen Algebraic Expression

12

Y= U1/U2 - 3*U1*U3 + U2**2

Nonlinear

Saturation

Dynamic Continuous

State Space

SS: 4	99
0	1
0	0
0	1
0	0
0	0
0	1
0	0
0	0
1	2
2	3
3	4
4	0

Integrator

1

S + 2

Trigonometric

ASIN (u)

Exp & Log

LOG (u)

e^u

$u^{3.14}$

SQRT

Dynamic Discrete

State Space

SS: 4	99
0	2
1	1
1	0
0	1
0	0
0	0
1	1
1	0
2	2
2	3
3	4
4	0

Transfer Function

Z

Z - 1

Waveform Generators

General

Boolean

Less Than

Coordinate Translation

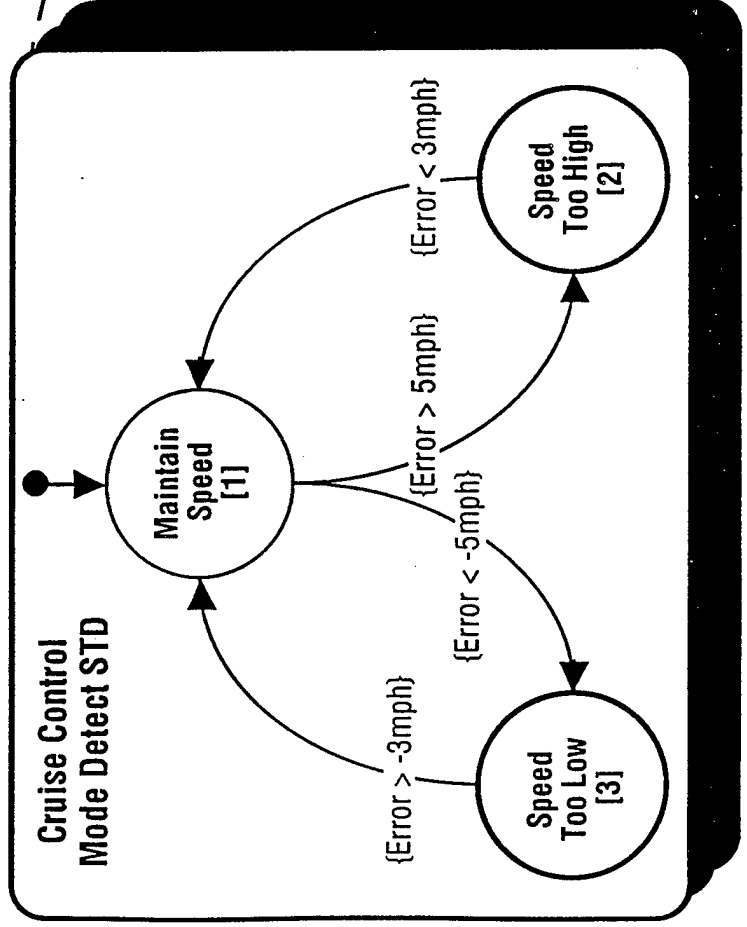
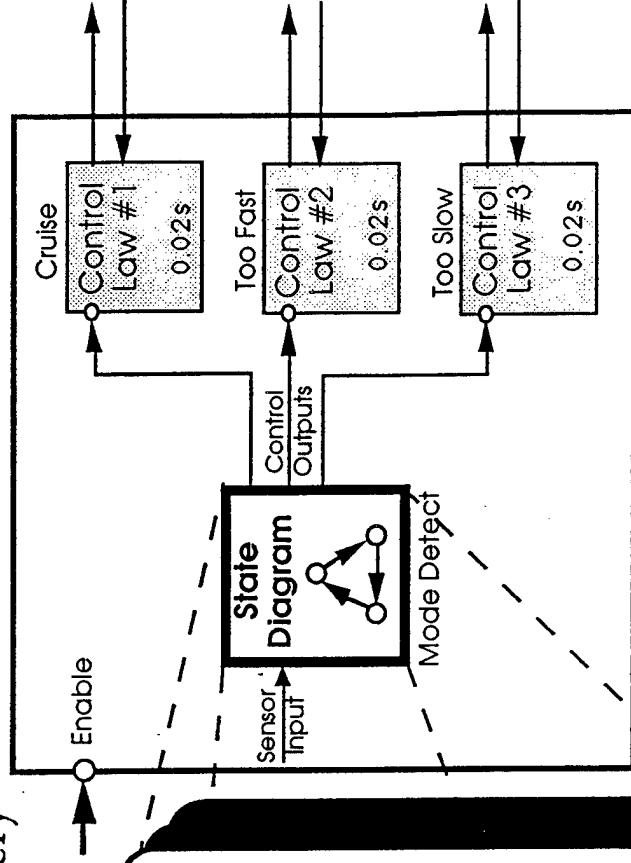
3 Axis Rotate

y = T123 + u

State Transition Diagrams

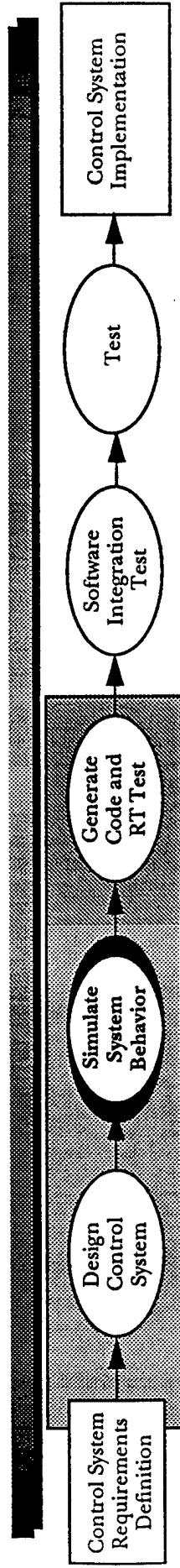
◆ Finite state machine implementation for:

- Adaptive suspension control
- Switchable engine controllers for power or economy
- ABS fault diagnostics and recovery
- Cruise control strategies

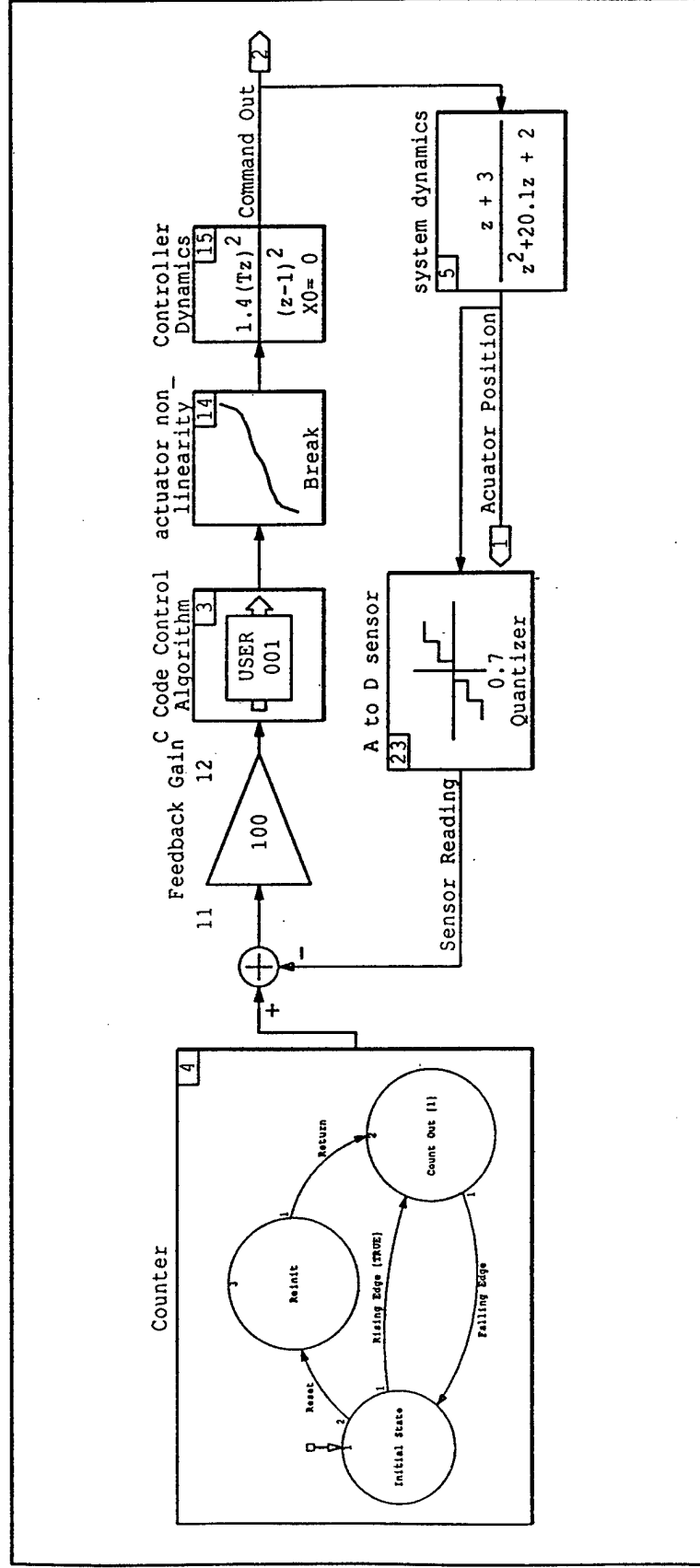




SystemBuild Simulations



What You See is What You Simulate





Simulation Control Panel Editor

Same Editing Paradigms Used in SystemBuild

Interactive Animation Icons

35 Block Strip Chart

20 Block Strip Chart

10 Block Strip Chart

Bar

Multiple Bar

Objects

Circular

Altimeter

Slide

Value

Value

Value

Time

Time

Time

0.00E+00 0.10E+03

0.00E+00 0.10E+03

0.00E+00 0.10E+03

1.0

4.0

5.0

9.0

11.

12.

6.0

3.0

7.0

8.0

13.

10.

14.

- More

1

2

3

4

5

6

7

8

9

+ More

KB

Connection Editor

1

2

3

4

<SET>

<ON>

<SPEED>

<BRAKE>

CANCEL

ADD

DELETE

DONE

Interactive Animation

Circular

Slide

0.00E+00 0.10E+03

0.00E+00

LED

LED

LED

SWITCH 1

SWITCH 2

SWITCH 3

On

On

On

Off

Off

Off

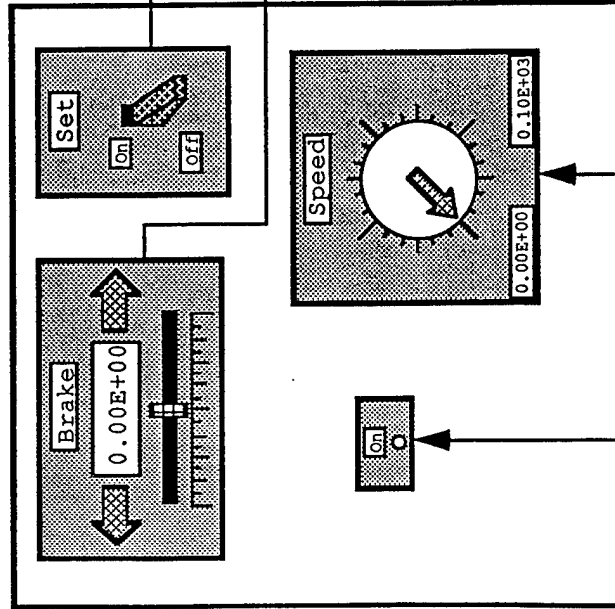
LOAD PICT	DEFINE	DRAG	NORMAL	ANIMATE
SAVE PICT	MODIFY	REDUCE	FIT WINDOW	STOP
DETAIL	DELETE	ENLARGE	HARDCOPY	HOLD
RTF NAMES	COPY	LABELS ON	HELP	EXIT



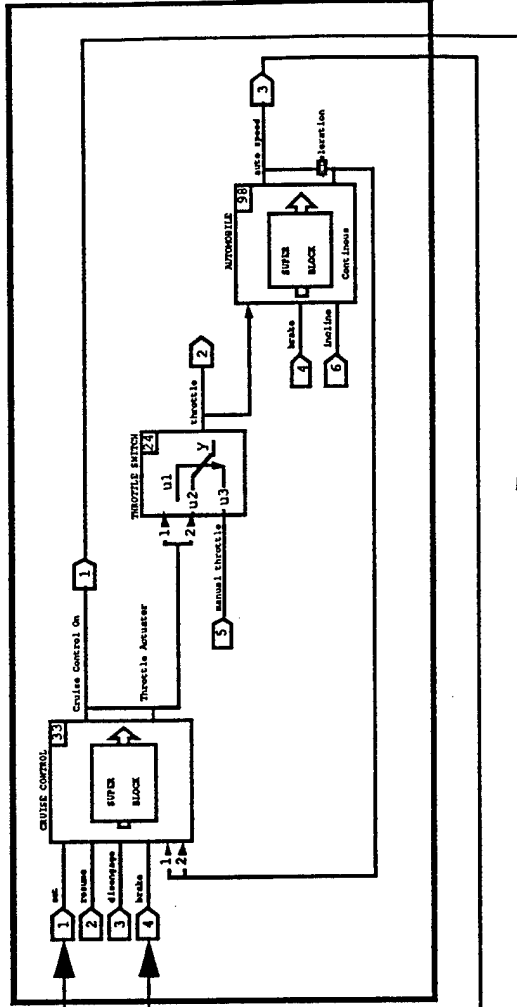
Interactive Simulation

The ability to monitor and control the dynamic system while it is simulating

Interactive Display

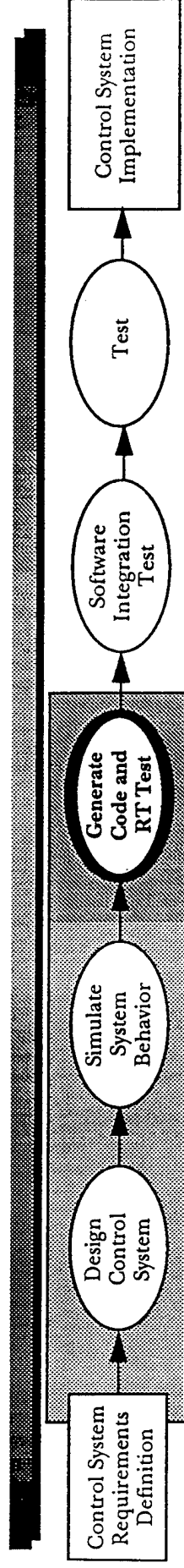


SystemBuild Simulation



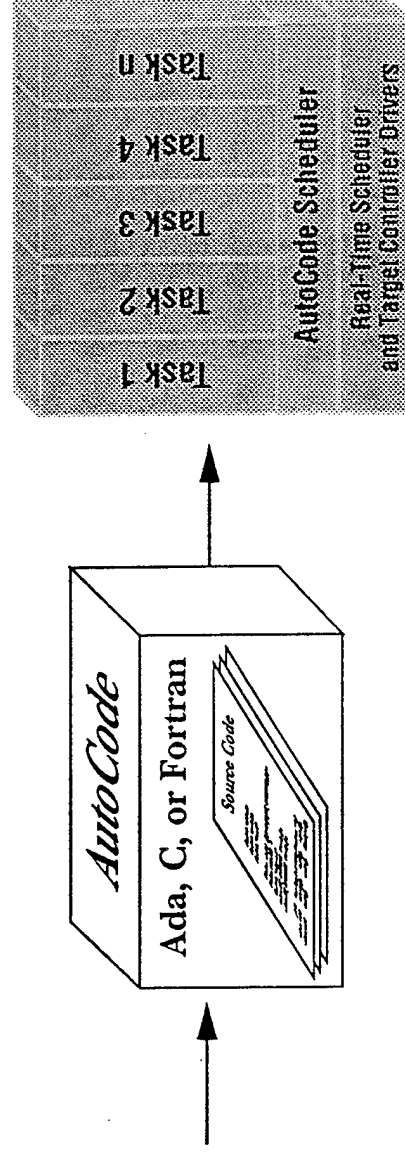
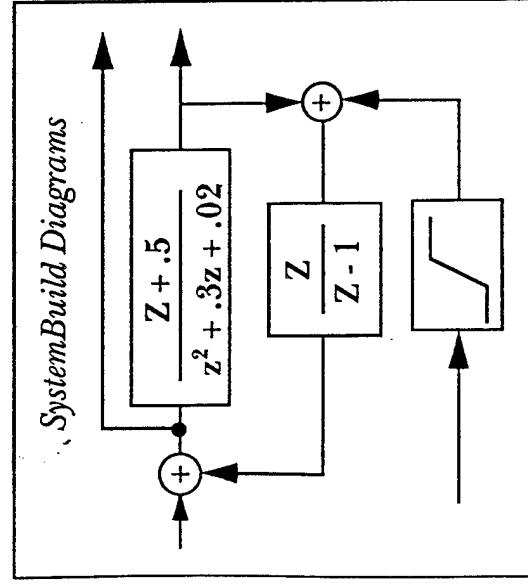


Automatic Code Generation in Ada, C, or Fortran



- ◆ Automatically Resolves Schedules Issues
- ◆ Traceable Back to Diagram
- ◆ No Detailed Code Debugging
- ◆ Minimal Real-Time Coding
- ◆ Works with Other Real-Time Operating Systems

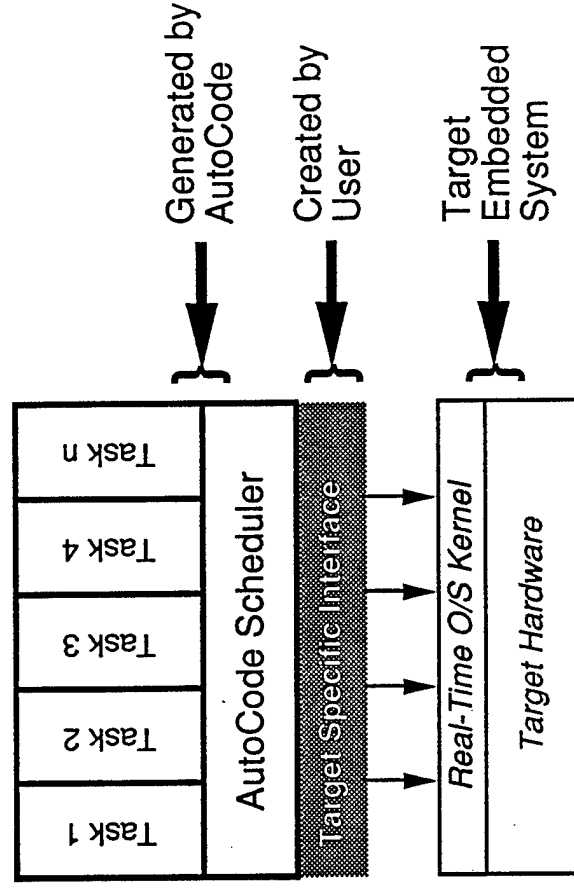
Design Database





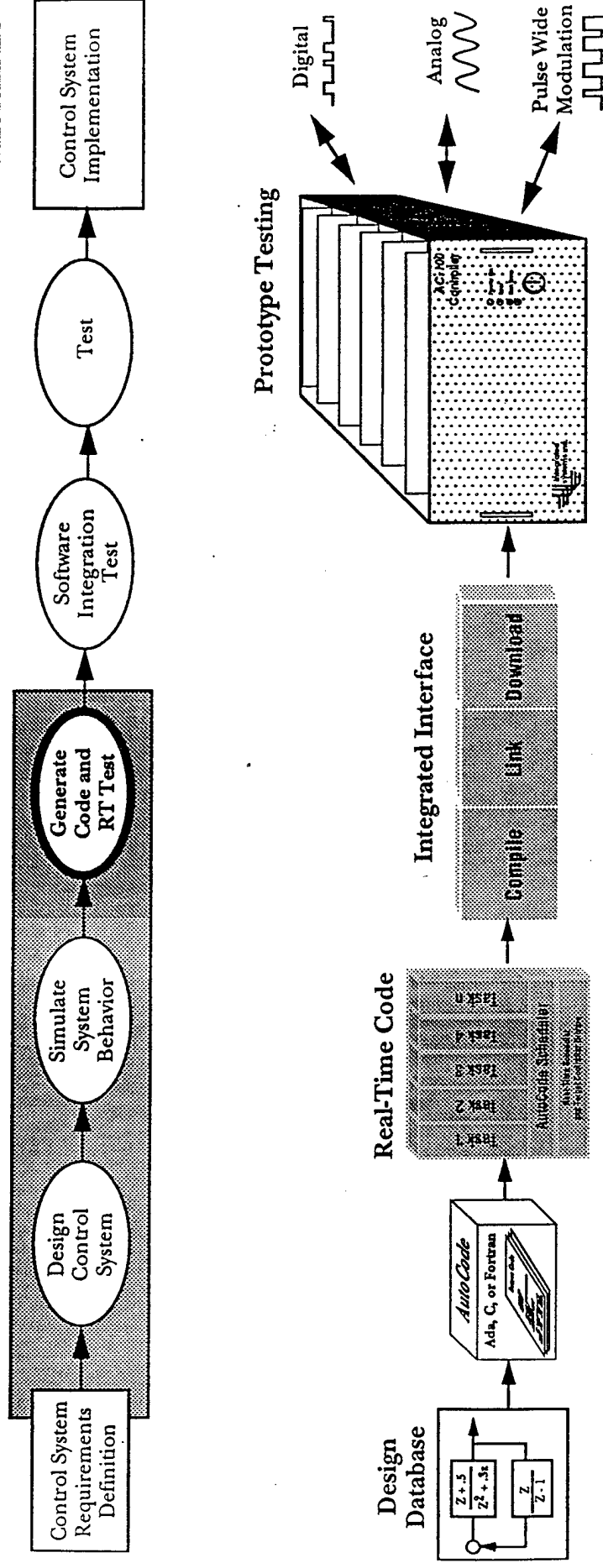
Complete Generation of Application Layer

- ◆ Generates Complete Multi-Tasking Application
- ◆ Generates Scheduler
 - Preemptive
 - Priorities Determined by Rates
- ◆ User Provides Target Specific Layer
 - Input/Output Drivers
 - Enable/Disable Interrupts
 - Fatal Error Recovery
 - Optional Background Task





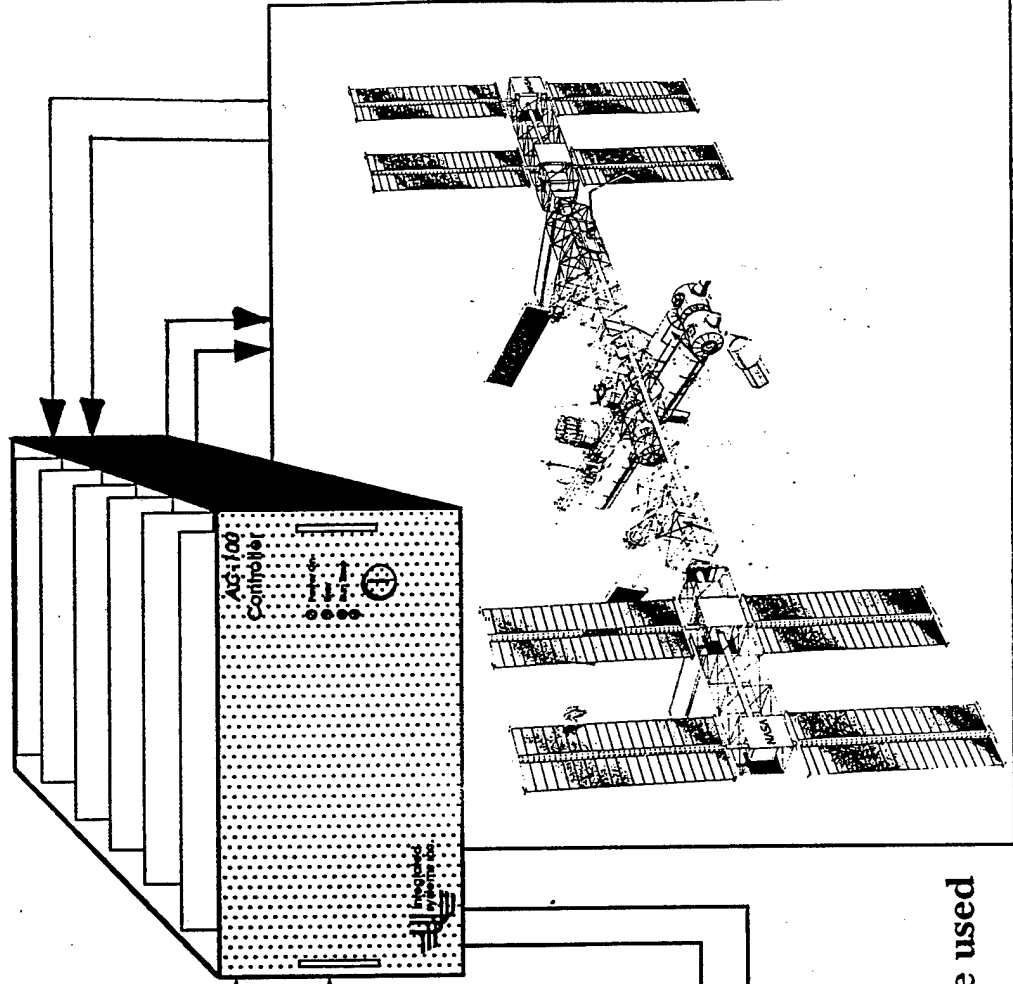
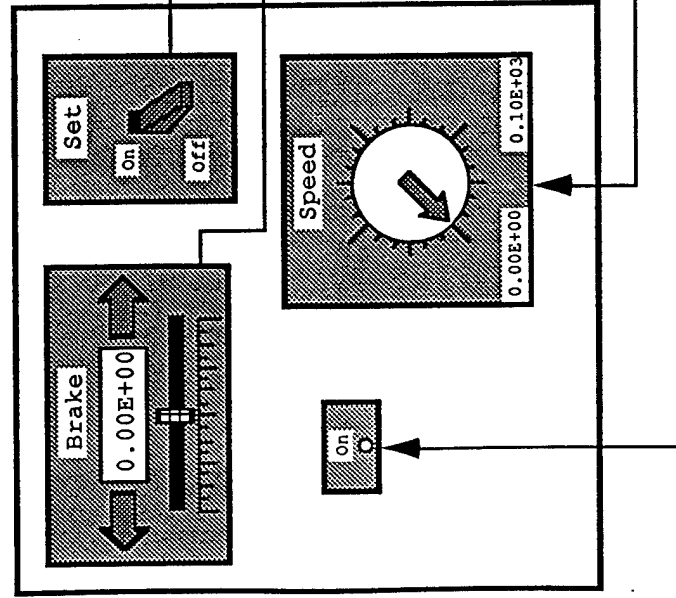
Real-Time Simulation, Test, Control and Data Acquisition



- ◆ Integrated Design to Implementation Solution
- ◆ Fast Design Iterations
- ◆ Performance Tuning
- ◆ Systems Engineers can Test the Solution without Programming

Interact with the Prototype

Interactive Display



Actual Plant

- ◆ View Controller and System Behavior
- ◆ Low Cost, Flexible Instrumentation for a Real-Time Controller
- ◆ Same Panels Used with SystemBuild can be used with a Controller Prototype

Application Example

◆ Space Station Freedom, Boeing Aerospace

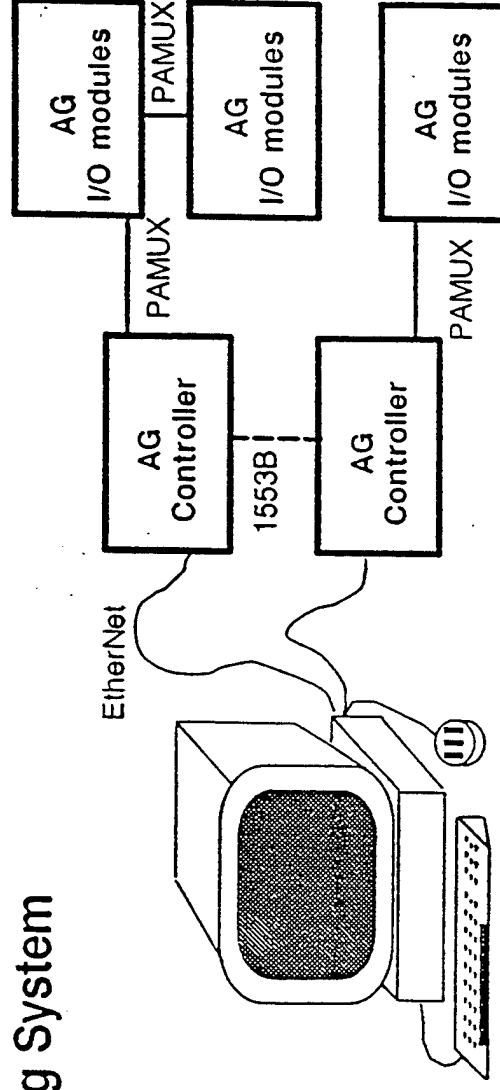
- Boeing's responsibilities include environmental control and life support systems, thermal control, fluid management
- Approximately 15 sub-contractors are participating in the development of software which Boeing must manage and integrate
- An Iterative Rapid Prototyping Design Methodology was adopted to modify the front end of the conventional software development cycle
- ISI's Tools support this modification and enforce consistency through a toolset referred to as the Application Generator (AG)
- System design changes are graphically implemented on a workstation, Ada source code is generated, compiled, downloaded and executed on the AG controller
- Prototype software is generated rapidly and prototype hardware can be exercised to support iterative testing and design



Boeing Application Generator Hardware

AG I/O Modules
OPTO-22 Optically Isolated
16 Analog Channels
32 Digital Channels

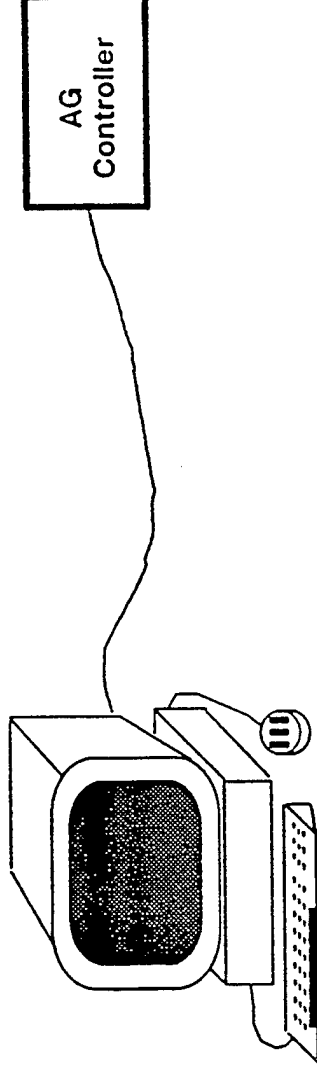
AG Workstation
VAXstation 3100 Model 30
19" Color Monitor
VMS Operating System



AG Controller
80386 microprocessor
Multibus II Backplane
80186 Communications Processor



Boeing Application Generator Software



AG Workstation

MATRIXx

Linear Analysis

Graphics

AutoCode

Model Development

Simulation

Interactive Animation

Operator-in-the-Loop Sim

Hardware Connection Editor

Ada Code Generator

DDC-I Ada Compiler System

Document Generator

DDC-I Kernel

I/O Drivers

- 1553B Bus
- PAMUX Bus
- Ethernet

Application Software

- Generated Code
- Scheduler



Advantages of AG Based Development

- ◆ Control System Engineers are able to implement algorithms in a control domain context
- ◆ Design drawings are maintained consistent with current design
- ◆ Reduces number of steps required for error identification, promoting early detection of design and requirements errors
- ◆ Reduces the number of steps required for error correction
- ◆ Eliminates the need for labor intensive and error prone hand coding of algorithms
- ◆ Permits early testing through simulation and hardware emulation prior to availability of actual hardware
- ◆ Provides real-time control and display of processes including hardware-in-the-loop simulations



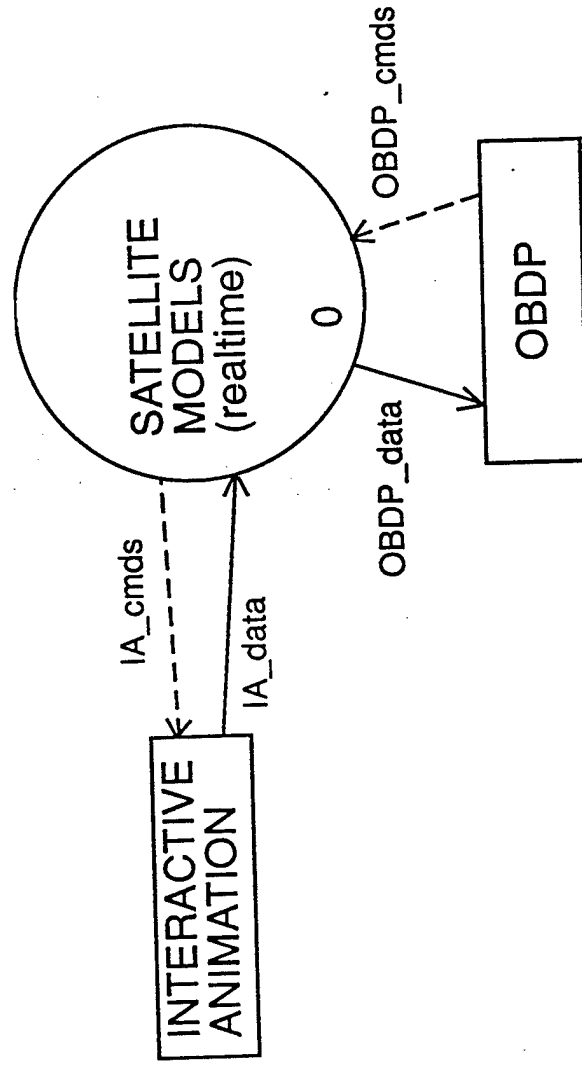
Application Example

◆ Satellite Flight Software Testing

- Major aerospace company interested in rapidly developing a real-time simulation/emulation which could interact with and verify independently developed flight software
- Cadre's Teamwork used for System Requirements Definition and documentation
- ISI's Tools used for detailed satellite model development, simulation, Ada code generation, real-time simulation, and interface to satellite flight computer (OBDP)

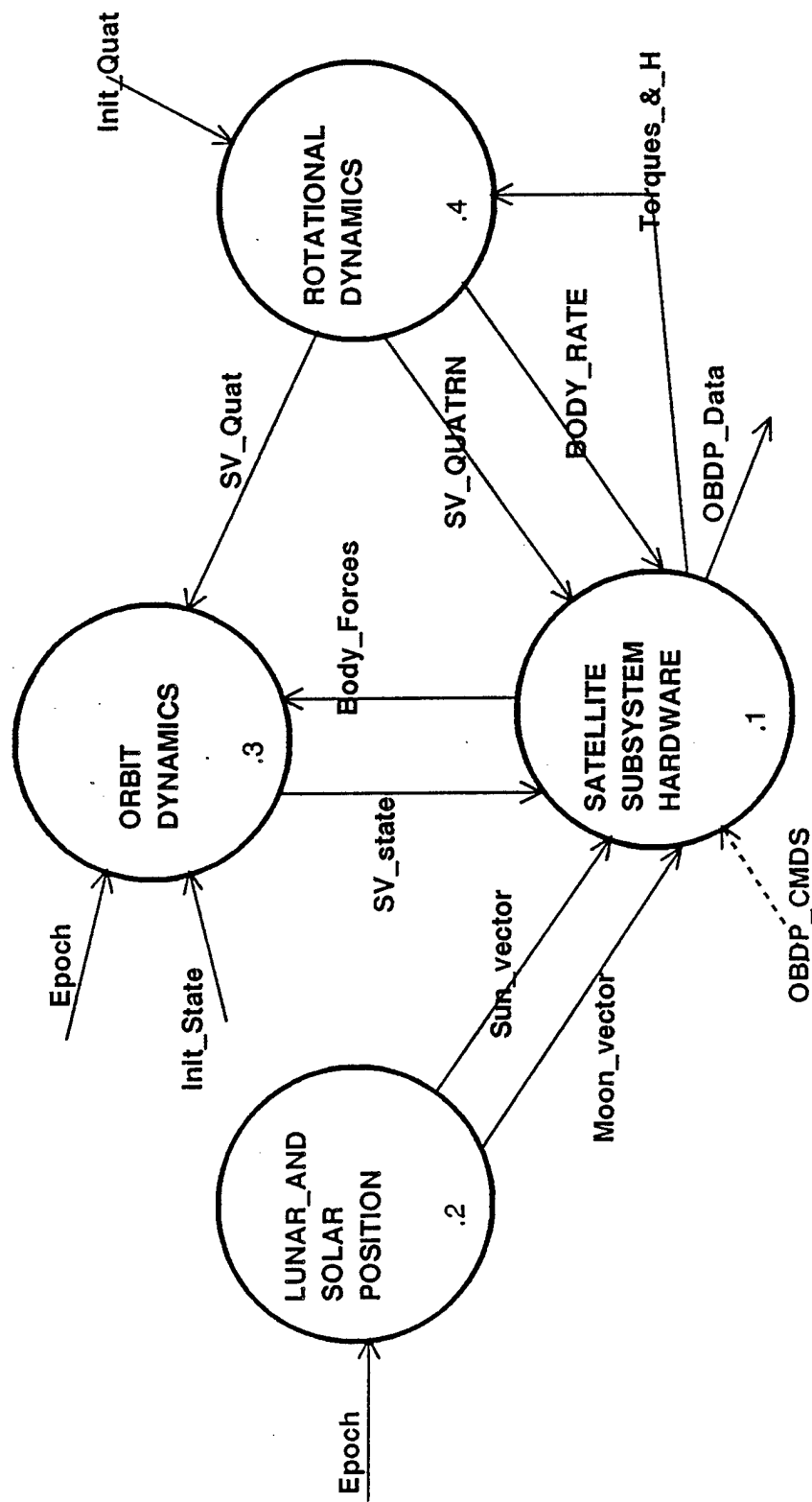
System Requirements Definition

- ◆ Cadre's Teamwork:
 - CASE Tool for performing systems analysis and design
 - Implements the Hatley/Perbhai structured analysis approach to systems and software design
 - Below is the highest level systems model developed in Teamwork (DFD 0):



First Level Decomposition of DFD 0

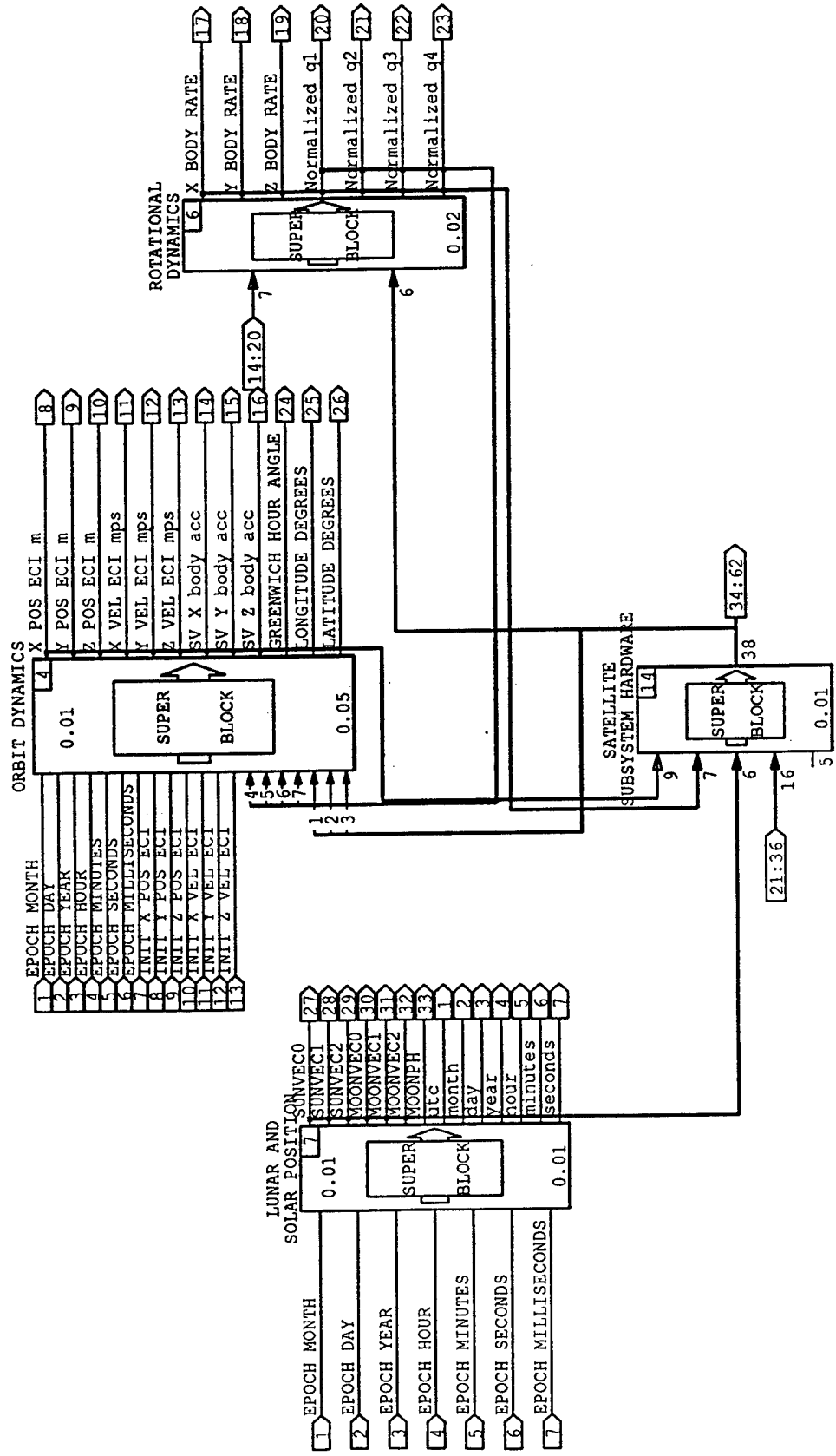
- ◆ DFD 1 modeled in Teamwork:





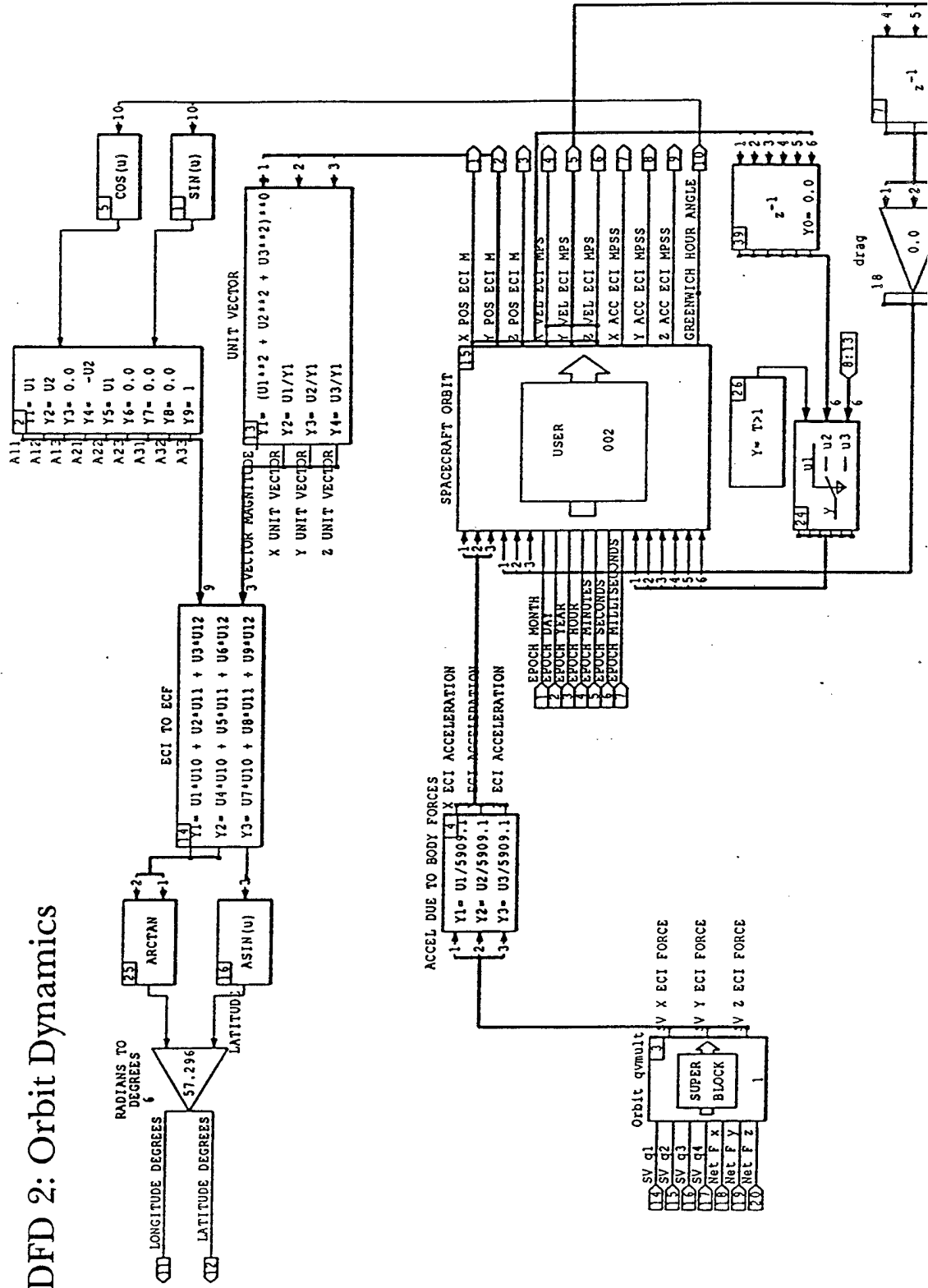
Translation from Teamwork to ISI's Graphical Environment

- ◆ DFD 1 modeled in SystemBuild



Further Decomposition of Orbit Dynamics

◆ DFD 2: Orbit Dynamics

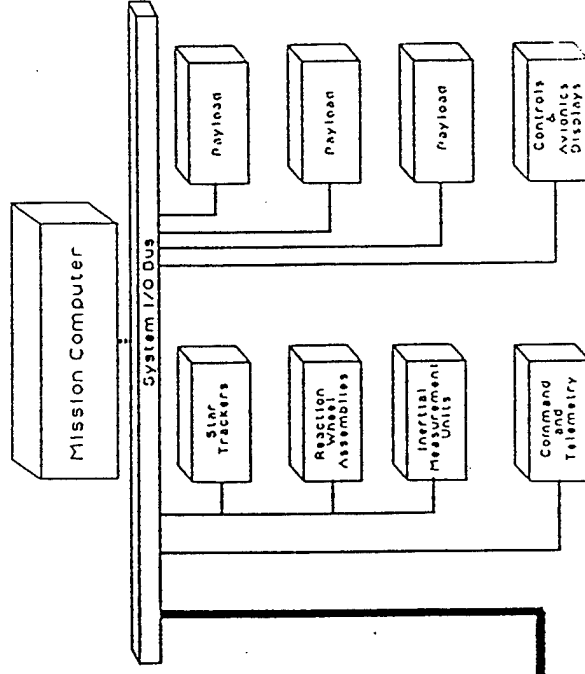
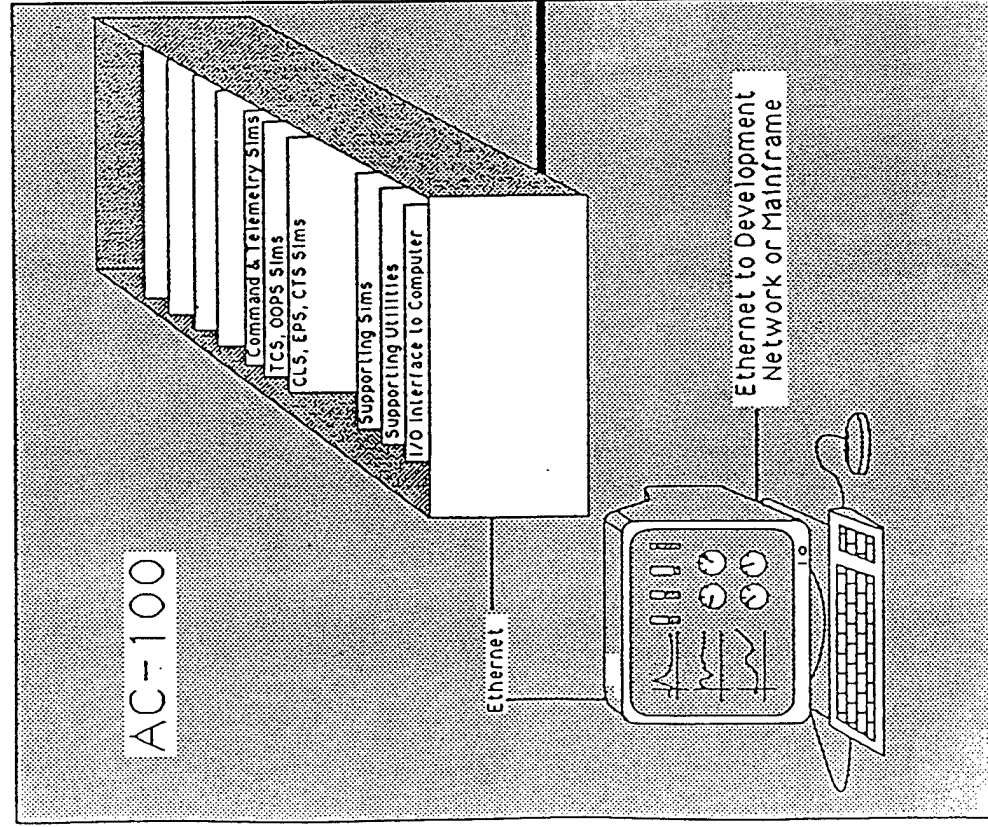




Simulation/Emulation Hardware Configuration

SUBSYSTEM/PAYLOADSIMS

HARDWARE - IN - LOOP



- ◆ VAX3100 workstation
- ◆ Concurrent processor with 11 80386 cards (22 MFLOP)
- ◆ Extensive hardware interface capability
- ◆ Concurrent multi-rate simulation and control

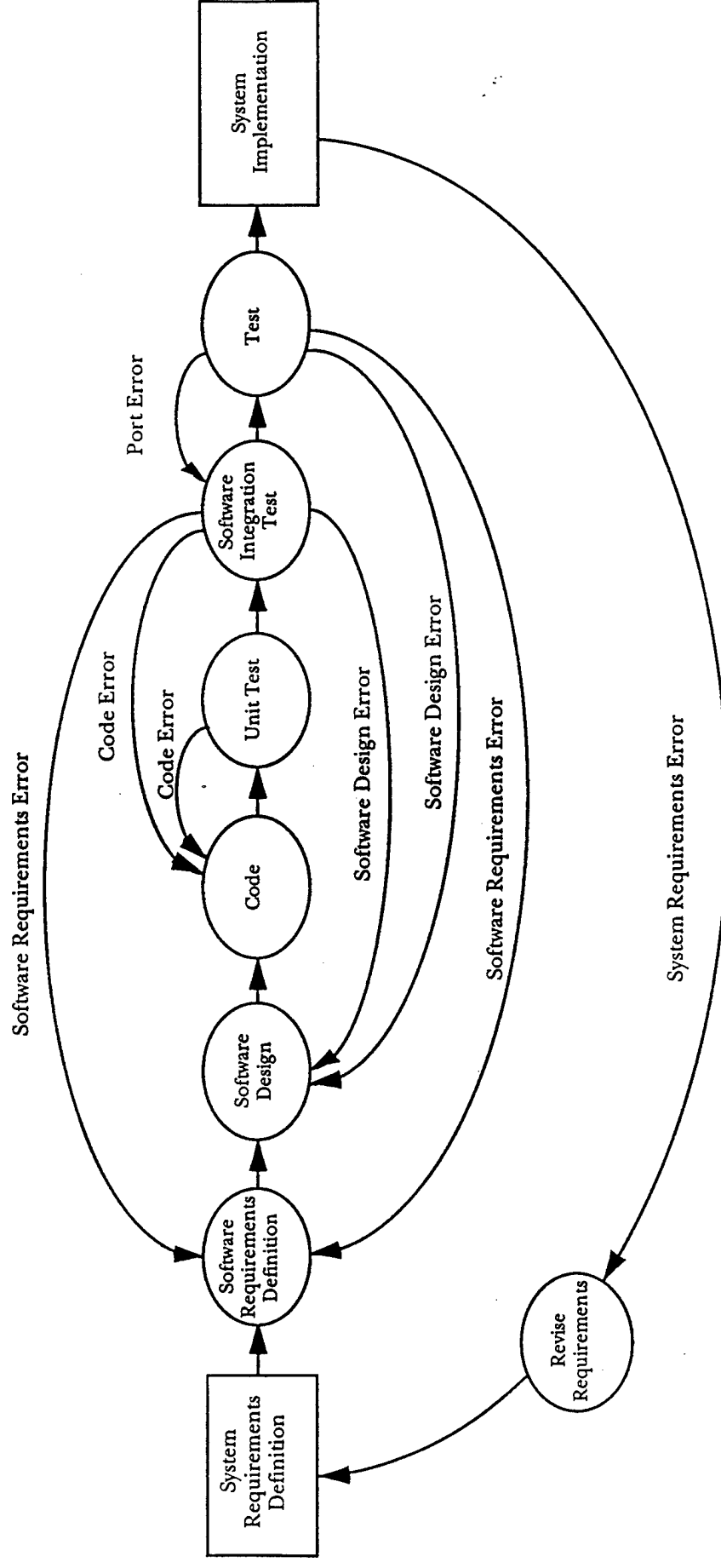


Satellite Simulation/Emulation

Advantages

- ◆ Graphical based programming environment is easy to use
- ◆ Automatic Ada Code Generation eliminates coding of algorithms
- ◆ Ability to interface to flight computer verifies flight software and/or hardware subsystems
- ◆ Reuseable library of standardized spacecraft and space environment models are developed:
 - Standard User interface to simulation facility
 - Standard for definition of simulation models (purpose, function, assumptions, data, design approach, . . .)
 - Standard simulation "structure" (definition of order, rates, dependencies of execution, . . .)
 - Standard graphical programming simulation "language"
 - Standard simulation services (control, data logging, data display, I/O communication with host operating system and external devices . . .)
 - Standard for introduction of fault conditions into simulations to simulate subsystem/payload failures (i.e., fault tolerance)

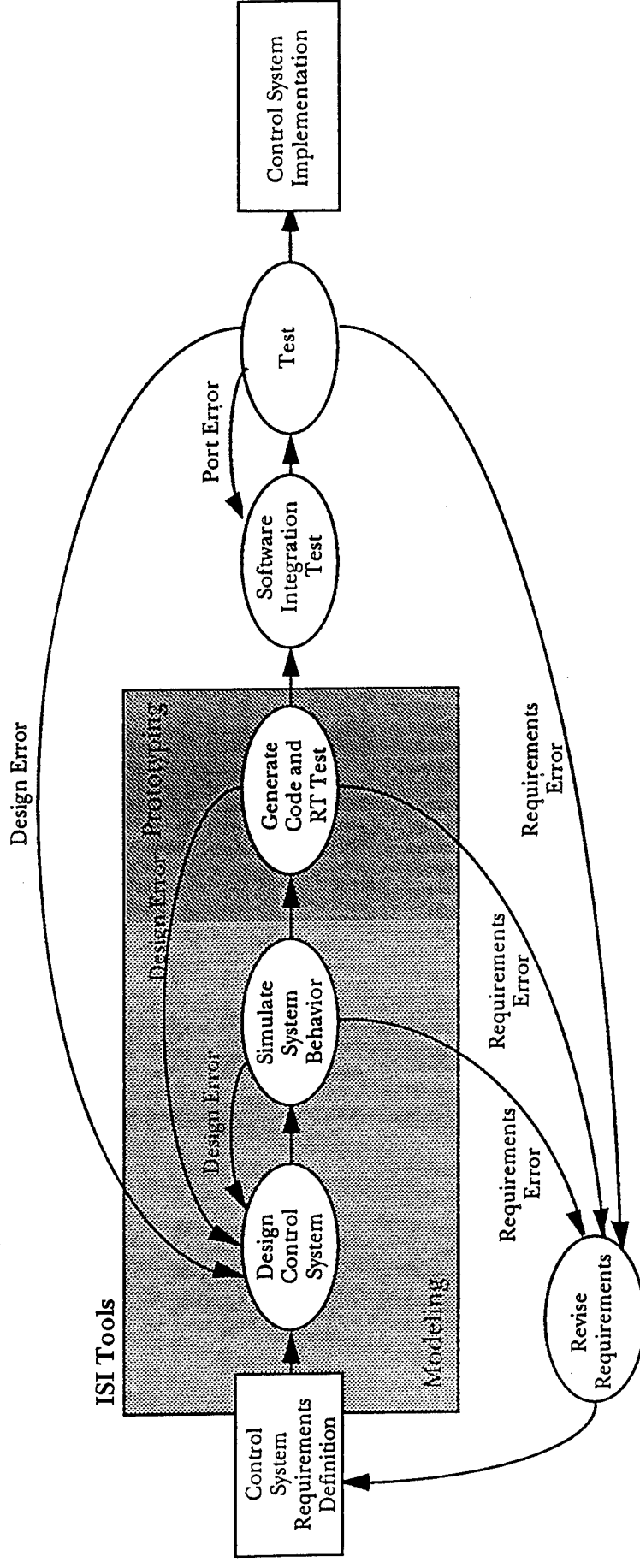
Summary



- ◆ The conventional software development process is nearly concluded before it is possible to identify errors introduced at the Requirements Definition stage
- ◆ These errors are costly to identify and correct



Prototyped Software Development

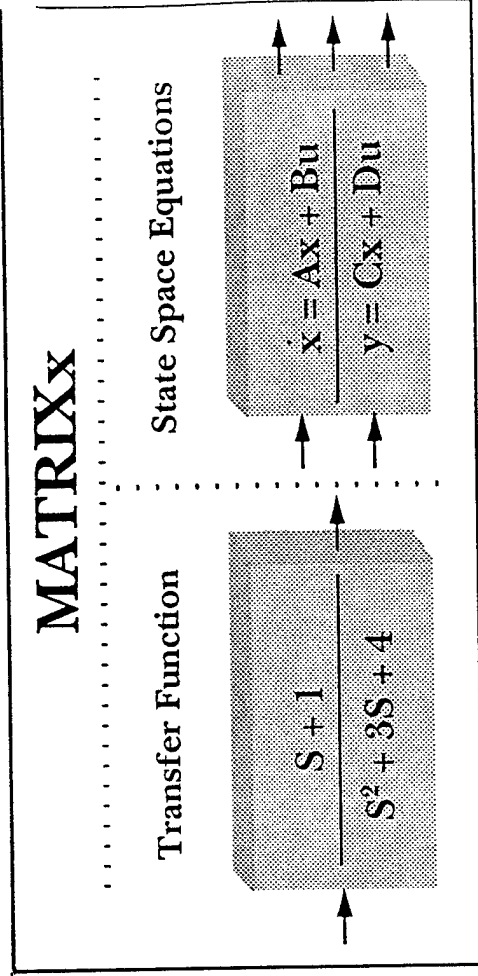
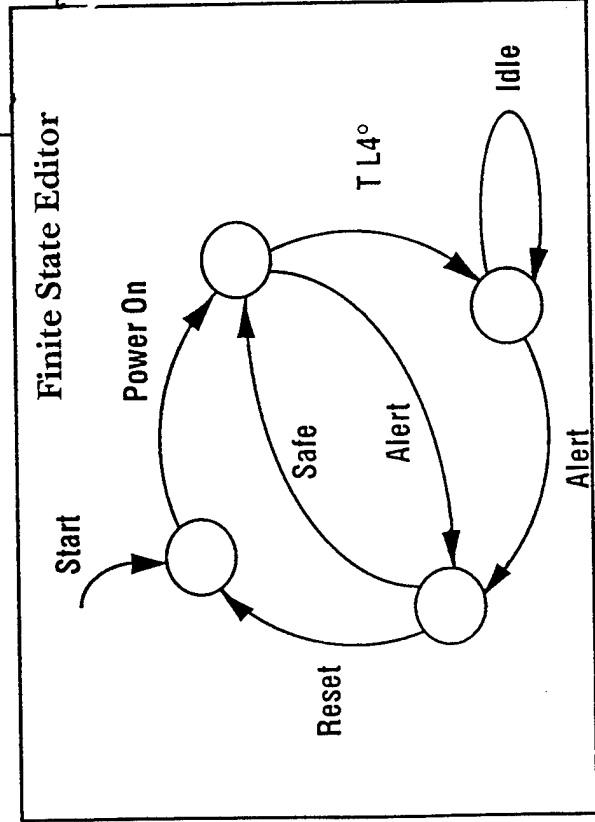
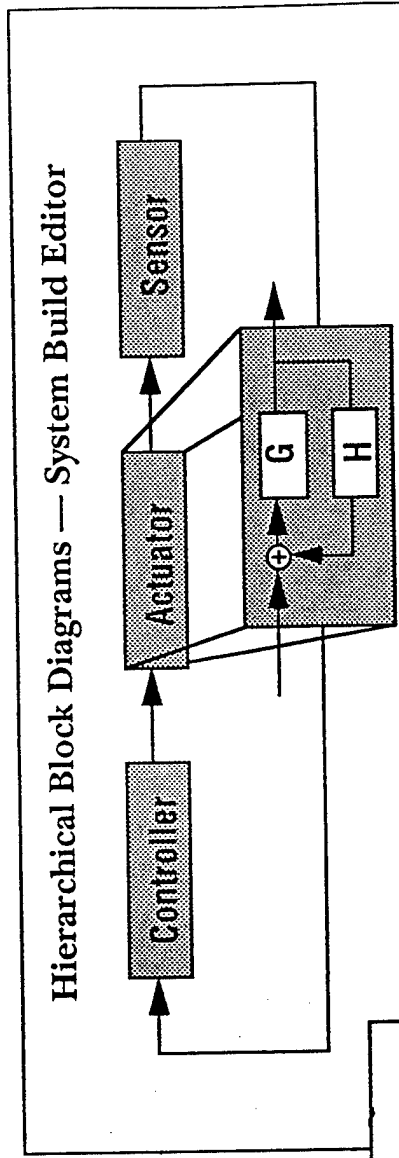


- ◆ ISI offers tools that allow many real-time software development efforts to be compressed
- ◆ The result is an efficient path to simulation and real time testing
- ◆ Rapid error identification, economical correction, and improved maintenance follow



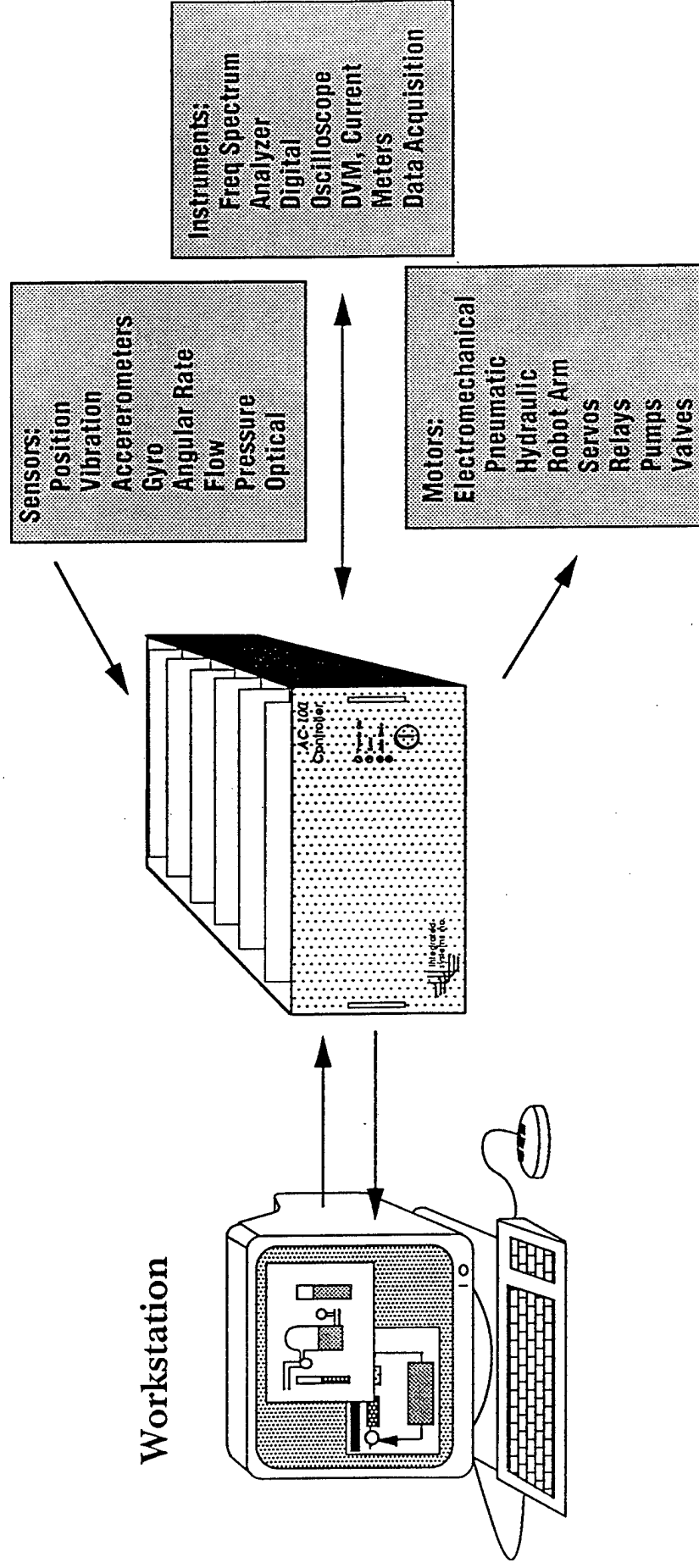
A Graphical Development Environment

- ◆ Easy to use
- ◆ Facilitates communication
- ◆ Enforcement of standards
- ◆ Support of re-useable modules





Simplified Coding and Real-Time Testing



- ◆ Ada, C and FORTRAN code generators
- ◆ Configurable, turn-key, real-time execution



Software Development Suitable for ISI's Tools

- ◆ Real-time software for embedded control as well as simulation of the process and its environment
- ◆ Anything which can be characterized in a control-flow/data-flow model:
 - Pointing and Tracking Systems
 - Engine Control Simulation
 - Flight Control Simulation
 - Advanced Robotics
 - Missile Guidance and Control
 - Real-Time Hardware-in-the-Loop Simulation
 - Man-in-the-Loop Simulation
 - Interceptor Tracking/Trajectory
 - Status Determination for sensors, interceptors, etc.
 - Or . . . ?



ISI's Product and Service Focus

System Design and Simulation

Analysis and Design	Modeling and Simulation
MATRIXx	SystemBuild
DSP Module	Interactive Animation Module
Control Design Module	HyperBuild Module
Robust Control Module	RemoteSim Module
System ID Module	
Optimization Module	

Real-Time Software Development

Real-Time Code Generation
AutoCode/Ada Module
AutoCode/C Module
AutoCode/Fortran Module

Rapid Prototyping

Implementation and Testing
AC-100/C System
Additional Processor Boards
AC-100/Ada System

Engineering Services

Design Experts
Control Systems
Robotics
Real-Time Software
Expert Systems
Signal Processing
Structural Analysis
Fluid Mechanics

22 MAR 1991

MEMORANDUM FOR SEE DISTRIBUTION

SUBJECT: 40th Computer Resource Integration Meeting (CRIM)

1. Subject meeting is scheduled for 9 Apr 91, 0800-1500, in room 1C1600. The main focus of this meeting is parallel processing capabilities and technology within the U.S. Army Strategic Defense Command (USASDC). These monthly CRIMs are held to provide the Technical Director an integrated status of significant computer resource activities, developments, and issues in the development of projects, programs, and systems. Organizations involved in related parallel processing activities not being briefed at this meeting should contact this office for presentation at the next CRIM. An open meeting will be held in the morning to provide all USASDC personnel an opportunity to interact with the briefers and ask questions on the products and services presented. An executive overview will be presented to Dr. Davies on the same day from 1300-1500.

2. An agenda for this meeting (enclosure 1), action items (enclosure 2), and the minutes of the 39th CRIM (enclosure 3) are provided. A status review of each action item by the action officer is requested at the meeting. Please notify this office of any corrections or additions to the minutes.

3. Future meetings will include briefings from the various USASDC organizations on their software development, quality evaluation, parallel processing, and other related software programs. It is therefore requested that all organizations provide representatives to this meeting in preparation for these briefings. The point of contact is Frank Poslajko, 955-1995.

4. Security clearances for participants external to this command are to be sent to:

Frank Poslajko
U.S. Army Strategic Defense Command
ATTN: CSSD-SP
P.O. Box 1500
Huntsville, Alabama 35807-3801

FAX No. (205) 955-3958

3 Encls


O. PETER CERNY
Chief, Systems and Programs
Integration Office

CSSD-SP

SUBJECT: 40th Computer Resource Integration Meeting (CRIM)

DISTRIBUTION:

SFAE-SD-AST/COL Stewart
SFAE-SD-GST/Jerry Cavender
SFAE-SD-GST-D/Dr. Ron Green (w/o encl 3)
SFAE-SD-GBR/COL Meier
SFAE-SD-GBR-E/W. Burrows, Reggie Anderson (w/o encl 3)
SFAE-SD-GBI/James Katechis
SFAE-SD-GBI-I/Bill Roberson, Bill Mobley (w/o encl 3)
SFAE-SD-HED/Alan Sherer
SFAE-SD-HED-E/Terri Russell, Gisele Wilson (w/o encl 3)
SFAE-SD-ASP-E/Donna Brock
CSSD-CM/Billy Perkins
CSSD-CO
CSSD-NT-LO/John Hawk (w/o encl 3)
CSSD-DE/Dr. Lavan
CSSD-DE-C/Bill Hughes, Greg Heinen (w/o encl 3)
CSSD-KA/COL MacNeill
CSSD-KA-R/LTC Harrison (w/o encl 3)
CSSD-LC/R. Hamilton
CSSD-SL/LTC Steve Rice
CSSD-IM/Bob Ogle, Sue Martin (w/o encl 3)
CSSD-IN-T (w/o encl 3)
CSSD-TM/COL T. Kunhart
CSSD-TM-I/David Hayes (w/o encl 3)
CSSD-KE/Dr. Wilkinson
CSSD-KE-T/R. Curtis, Eugene Sanders (w/o encl 3)
CSSD-PA/Ed Vaughn (w/o encl 3)
CSSD-SA/LTC Thomas Shpakowsky
CSSD-SA-E/COL Paul Mullek (w/o encl 3)
CSSD-SA-EE/Ron Liedel (w/o encl 3)
CSSD-SA-EE/Mark McClellan (w/o encl 3)
CSSD-SA-ET/Pam Caruso, David Parsons, Bill McCormick (w/o encl 3)
CSSD-SA-B/LTC John Wilson (w/o encl 3)
CSSD-SA-BA/MAJ James DeLary, Katy Attilio, James Caudle
(w/o encl 3)
CSSD-SA-BB/Paro Perrett (w/o encl 3)
CSSD-SA-BT/Dr. Virginia Kobler, Jackie Cristina (w/o encl 3)
CSSD-SA-BE/Dr. Doyle Thomas (w/o encl 3)
CSSD-SA-SA/Jim Burch, Mark Walczyk (w/o encl 3)
CSSD-SA-SS/James Keith (w/o encl 3)
CSSD-SO/Don Marsh
CSSD-TE/COL James Green
CSSD-TE-P/John Hartman, Tom Nuttall (w/o encl 3)
CSSD-TE-O/Mickey Jones (w/o encl 3)
CSSD-AT/Dr. Shelba Proffitt, Dick Lenning
CSSD-AT-P/D. Satterfield
CSSD-TD/Dr. William Davies
CSSD-SD/Dr. James Fisher
CSSD-SD-S/Tony Muzzi (w/o encl 3)
CSSD-CA-S/W. Shelton

PRELIMINARY AGENDA
40TH COMPUTER RESOURCE INTEGRATION MEETING
9 APRIL 1991
CONFERENCE ROOM 1C1600, 0800-1500

0800-0810	Introduction	Frank Poslajko 955-1995
0810-0825	SDIO Parallel Processing Activities, Organization, etc.	Dr. Leslie Pierre 702-693-1826
0825-0850	NTB Software Development Highlights	Cpt Andrews/T. Gill 719-380-2465
0850-0910	ARC Architecture Overview Environment System Spec.	Bob Cooley 955-1354
0910-0940	A2P Program	Steve Risner 955-3848
0940-1000	Touchstone Hypercube (ips 860)	Dr. Ron Green 722-1844
1000-1010	Break	
1010-1030	Fault Tolerant Processor (FTP)	Bettie Upshaw 955-3704
1030-1050	Georgia Tech Parallel Function Processor Status	Gene Sanders 955-5813
1050-1110	Acousto-Opic Processing	James Fears/ Dr. Dorsett 955-3798
1110-1130	Residue Number System/ Real Time Waveform Processor	Charles Kiss 955-3798
1130-1150	Solid State Optics Processor	Luis Lopez 955-4817
1150-1210	High Performance ARC Network (HPAN)	Luke Huffman (Colsa) 922-1512
1210-1300	Lunch	

Enclosure 1

PRELIMINARY AGENDA
40TH COMPUTER RESOURCE INTEGRATION MEETING (Cont)

1300-1305	Action Items	Frank Poslajko 955-1995
1305-1310	SDIO Parallel Processing Activities; Organization, etc.	Dr. Leslie Pierre 702-693-1826
1310-1315	SDI Technical Information Center	Mike Metrione 703-521-3812 x6029
1315-1330	NTB Software Development Highlights	Cpt Andrews/T. Gill 719-380-2465
1330-1340	ARC Architecture Overview Environment System Spec.	Bob Cooley 955-1354
1340-1350	A2P Program	Steve Risner 955-3848
1350-1400	Touchstone Hypercube (ips 860)	Dr. Ron Green 722-1844
1400-1410	Fault Tolerant Processor (FTP)	Bettie Upshaw 955-3704
1410-1420	Georgia Tech Parallel Function Processor Status	Gene Sanders 955-5813
1420-1430	Acousto-Opic Processing	James Fears/ Dr. Dorsett 955-3798
1430-1440	Residue Number System/ Real Time Waveform Processor	Charles Kiss 955-3798
1440-1450	Solid State Optics Processor	Luis Lopez 955-4817
1450-1500	High Performance ARC Network (HPAN)	Luke Huffman (Colsa) 922-1512

40TH COMPUTER RESOURCE INTEGRATION MEETING (CRIM)
ACTION ITEMS

- | | |
|---|---------------------------------|
| 1. Provide a status update on the software organization and development at the National Test Facility. | John Hawk
5-3920 |
| 2. Schedule status briefings on SDS committees to include purpose, accomplishments, plans, and schedules. | Frank Poslajko
5-1995 |
| 3. Establish a Data Reduction Planning Committee. | Barbara Rogers
722-1518 |
| 4. DISC4 to coordinate with Ada 9X project office on Ada language deficiencies. | Bob Johnson
AUTOVON 227-0259 |
| 5. ADCCS project office to report on the number of Ada waiver requests submitted to DISC4. | Denise Jones
895-3397 |
| 6. Broaden distribution of SESE specification document to CASE developers. | Frank Poslajko
5-1995 |
| 7. Provide location of controlled experiment on trusted software development costs. | Dr. Mike Walker
883-1170 |
| 8. Incorporating reuse in the trusted software document and the SEI industry self assessment. | Dr. Mike Walker
883-1170 |
| 9. Schedule a Rationale and ISI tools demonstration | Frank Poslajko
5-1995 |

Enclosure 2