08:01:35 OCA PAD AMENDMENT - PROJECT HEADER INFORMATION 02/12/91

Active

Project #: C-43-611 Cost share #: Rev #: 4
Center # : 10/24-6-R6863-0A1 Center shr #: OCA file #: 99
Work type : RES
Contract#: DAKF11-86-D-0015-0027 Mod #: ADMIN. Document : DO
Prime #: Contract entity: GTRC

Subprojects ? : N
Main project #:

Project unit: SRC Unit code: 02.010.307
Project director(s):
PUTNAM W O III COMPUTING (404)894-5551

Sponsor/division names: ARMY / AIRMICS
Sponsor/division codes: 102 / 017

Award period: 891116 to 910516 (performance) 910516 (reports)

Sponsor amount	New this change	Total to date
Contract value	0.00	173,879.00
Funded	0.00	173,879.00
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: INTEGRATION OF SEVERAL COMPUTER SYSTEMS WITHIN A HETEROGENEOUS ENVIRONMENT

PROJECT ADMINISTRATION DATA

OCA contact: William F. Brown 894-4820

Sponsor technical contact

DR. GERARD C. MCCOYD, COR/AIRMICS
(404)894-3107

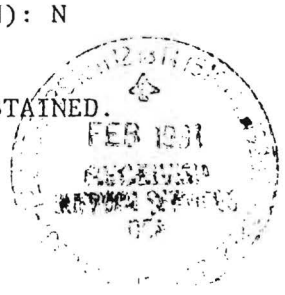
115 O'KEEFE BLDG.
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GA 30332-0800

Sponsor issuing office

MS. JANIE WRIGHT/AFZK-DOC-AS
(404)752-2469

CONTRACT ADMINISTRATION DIVISION
BLDG. 184
FT. MCPHERSON, GA 30330-5000

Security class (U,C,S,TS) : U ONR resident rep. is ACO (Y/N): N
Defense priority rating : DO-A7 GOVT supplemental sheet
Equipment title vests with: Sponsor X GIT
EXCEPT ITEMS LESS THAN $5K/EA VEST W/GIT IF PRIOR PURCH. APPROVAL OBTAINED.
Administrative comments -
ADMIN. REVISION TO CORRECT UNIT CODE FROM COMPUTING TO SRC.
PREVIOUS REVISION CHANGED PROJECT NUMBER FROM A-70-611 TO C-43-611.



GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 07/24/92

Project No. C-43-611_____ Center No. 10/24-6-R6863-0A1_

Project Director PUTNAM W O III_____ School/Lab SRC_____

Sponsor ARMY/AIRMICS_____

Contract/Grant No. DAKF11-86-D-0015-0027_____ Contract Entity GTRC

Prime Contract No. _____

Title INTEGRATION OF SEVERAL COMPUTER SYSTEMS WITHIN A HETEROGENEOUS ENVIRONMEN

Effective Completion Date 910516 (Performance) 910516 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	911219
Final Report of Inventions and/or Subcontracts	Y	920131
Government Property Inventory & Related Certificate	Y	_____
Classified Material Certificate	N	_____
Release and Assignment	Y	920109
Other _____	N	_____

Comments_____

_____

Subproject Under Main Project No. _____

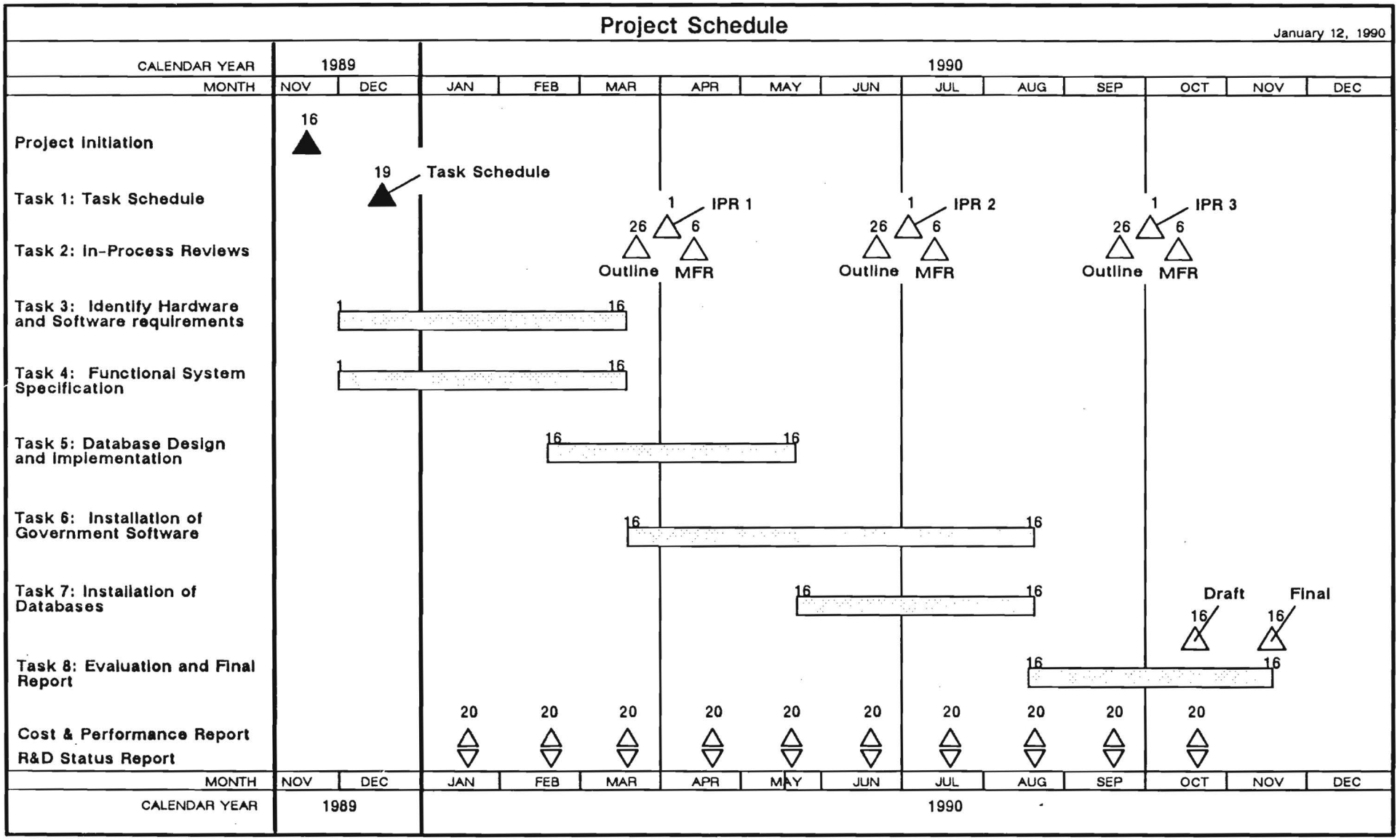
Continues Project No. _____

Distribution Required:

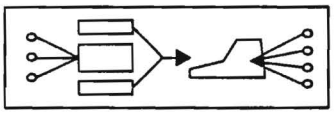
Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other _____	N
_____	N

A-70-611

Integration of Several Computer Systems Within a Heterogeneous Environment



The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for November 16 to December 31, 1989
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: January 12, 1990

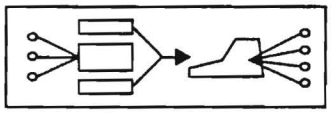
Expenditures from 11/16/89 through 12/31/89

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	7,934.39	7,934.39	63,995.00	56,060.41	12
Materials and Supplies	0.00	0.00	6700.00	6,700.00	0
Overhead	4,959.00	4,959.00	44,184.00	39,225.00	11
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**12,893.39**	**12,893.39**	**149,879.00**	**136,985.61**	**9**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for January, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: February 1, 1990

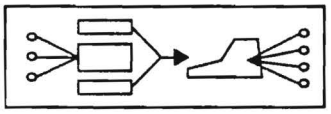
Expenditures from 1/1/90 through 1/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,736.25	12,670.64	63,995.00	51,324.36	20
Materials and Supplies	0.00	0.00	6,700.00	6,700.00	0
Overhead	2,960.16	7,919.16	44,184.00	36,264.84	18
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**7,696.41**	**20,589.80**	**149,879.00**	**129,289.20**	**14**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for February, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

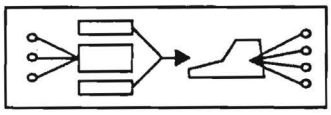
Date: March 1, 1990

Expenditures from 2/1/90 through 2/28/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,736.25	17,406.89	63,995.00	46,588.11	27
Materials and Supplies	35.00	35.00	6,700.00	6,665.00	0.5
Overhead	2,982.03	10,901.19	44,184.00	33,282.81	25
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**7,753.28**	**28,343.08**	**149,879.00**	**121,535.92**	**19**

C-43-611

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for March, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: April 1, 1990

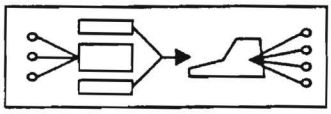
Expenditures from 3/1/90 through 3/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,736.25	22,143.14	63,995.00	41,851.86	35
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	2,960.16	13,861.35	44,184.00	30,322.65	31
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**7,696.41**	**36,039.49**	**149,879.00**	**113,839.51**	**24**

A-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for April, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: May 1, 1990

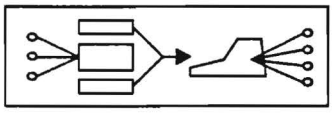
Expenditures from 4/1/90 through 4/30/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,736.25	26,879.39	63,995.00	37,115.61	42
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	2,960.16	16,821.51	44,184.00	27,362.49	36
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**7,696.41**	**43,735.90**	**149,879.00**	**106,143.10**	**29**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for May, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: June 1, 1990

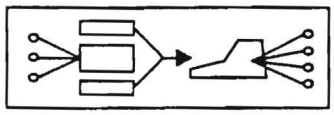
Expenditures from 5/1/90 through 5/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,736.25	31,615.64	63,995.00	32,379.36	49
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	2,960.16	19,781.67	44,184.00	24,402.33	45
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**7,696.41**	**51,432.31**	**149,879.00**	**98,446.69**	**34**

A-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for June, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: July 1, 1990

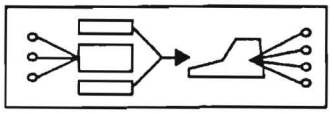
Expenditures from 6/1/90 through 6/30/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	5,092.82	36,708.46	63,995.00	27,286.54	57
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	3,183.01	22,964.68	44,184.00	21,219.32	52
Equipment	0.00	0.00	35,000.00	35,000.00	0
Total	**8,275.83**	**59,708.14**	**149,879.00**	**90,170.86**	**40**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for July, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

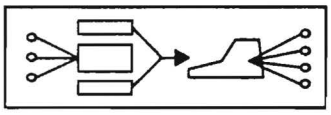
Date: August 1, 1990

Expenditures from 7/1/90 through 7/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	5,514.36	42,222.82	63,995.00	21,772.18	66
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	3,446.48	26,411.16	44,184.00	17,772.84	60
Equipment	43,998.68	43,998.68	59,000.00	15,001.32	75
Total	**52,959.52**	**112,667.66**	**173,879.00**	**61,211.34**	**65**

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332

A-73-611



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for August, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: September 1, 1990

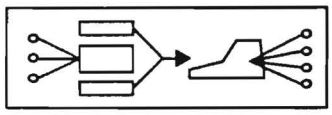
Expenditures from 8/1/90 through 8/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	6,648.50	48,871.32	63,995.00	15,123.68	76
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	4,155.31	30,566.47	44,184.00	13,617.53	69
Equipment	0.00	43,998.68	59,000.00	15,001.32	75
Total	**10,803.81**	**123,471.47**	**173,879.00**	**50,407.53**	**71**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for September, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: October 1, 1990

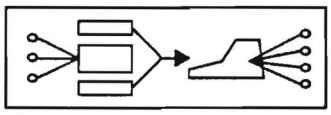
Expenditures from 9/1/90 through 9/30/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	5,686.20	54,557.52	63,995.00	9,437.48	85
Materials and Supplies	0.00	35.00	6,700.00	6,665.00	0.5
Overhead	3,553.88	34,120.35	44,184.00	10,063.65	77
Equipment	1,308.26	45,306.94	59,000.00	13,693.06	77
Total	**10,548.34**	**134,019.81**	**173,879.00**	**39,859.19**	**77**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for October, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: November 1, 1990

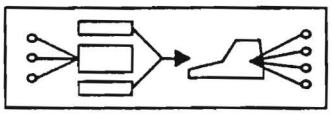
Expenditures from 10/1/90 through 10/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	5,125.06	59,682.58	63,995.00	4,312.42	93
Materials and Supplies	114.39	149.39	6,700.00	6,550.61	2
Overhead	3,274.66	37,395.01	44,184.00	6,338.99	86
Equipment	0.00	45,306.94	59,000.00	13,693.06	77
Total	**8,514.11**	**142,533.92**	**173,879.00**	**31,444.08**	**82**

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for November, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: December 1, 1990

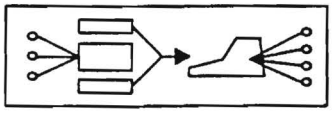
Expenditures from 11/1/90 through 11/30/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,973.06	64,655.64	63,995.00	-660.64	101
Materials and Supplies	0.00	149.39	6,700.00	6,550.61	2
Overhead	3,108.16	40,503.17	44,184.00	3,680.83	92
Equipment	0.00	45,306.94	59,000.00	13,693.06	77
Total	**8,081.22**	**150,615.14**	**173,879.00**	**23,263.86**	**87**

C-70-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for December, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: January 1, 1991

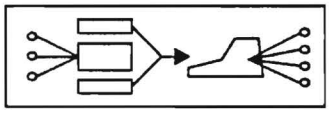
Expenditures from 12/1/90 through 12/31/90

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	4,973.06	69,628.70	63,995.00	-5,633.70	109
Materials and Supplies	3,587.65	3,737.04	6,700.00	2,962.96	56
Overhead	5,350.44	45,853.61	44,184.00	-1,669.61	104
Equipment	858.67	46,165.61	59,000.00	12,834.39	78
Total	**14,769.82**	**165,384.96**	**173,879.00**	**8,494.04**	**95**

0-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for January 1991
Project C-43-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: February 1, 1991

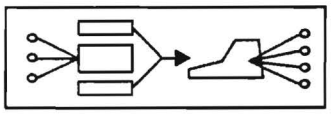
Expenditures for January, 1991

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	0.00	69,628.70	63,995.00	-5,633.70	109
Materials and Supplies	0.00	3,737.04	6,700.00	2,962.96	56
Overhead	0.00	45,853.61	44,184.00	-1,669.61	104
Equipment	0.00	46,165.61	59,000.00	12,834.39	78
Total	**0.00**	**165,384.96**	**173,879.00**	**8,494.04**	**95**

C-43611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Cost and Performance report for February 1991
Project C-43-611
Integration of Several Computer Systems Within a Heterogeneous Environment

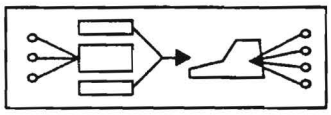
Date: March 1, 1991

Expenditures for February, 1991

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	0.00	69,628.70	63,995.00	-5,633.70	109
Materials and Supplies	0.00	3,737.04	6,700.00	2,962.96	56
Overhead	0.00	45,853.61	44,184.00	-1,669.61	104
Equipment	0.00	46,165.61	59,000.00	12,834.39	78
Total	**0.00**	**165,384.96**	**173,879.00**	**8,494.04**	**95**

C-43-611

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

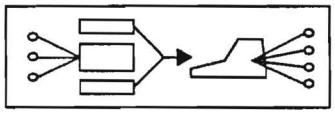
Subject: Cost and Performance report for March 1991
Project C-43-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: April 1, 1991

Expenditures for March, 1991

	This Period	CTD	Budget	Balance	%Exp.
Personal Services	0.00	69,628.70	63,995.00	-5,633.70	109
Materials and Supplies	300.00	4,037.04	6,700.00	2,662.96	60
Overhead	187.50	46,041.11	44184.00	-1,857.11	103
Equipment	2,200.00	48,365.61	59,000.00	10,634.39	82
Total	**2,687.50**	**168,072.46**	**173,879.00**	**5,806.54**	**97**

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for November 16 to December 31, 1989
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: January 12, 1990

During the period from November 16 through December 31, 1989, Background information was collected for the project. Documentation on the ANSWER, RAID, and IOIS projects was obtained and reviewed. Further documentation will be required before these systems can be completely understood. An IPR for the ANSWER project was held at AIRMICS which provided an introduction to that system and its capabilities. A search was begun for commercial products, both software and hardware, which will help meet the requirements of the project. Sun versions of popular PC products such as Lotus 123 and Dbase have been located. A product information database was set up.

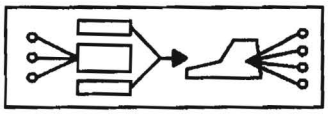
Current goals are:

Move Sun 3/50 workstation from AECAL to O'KEEFE and connect it to the AIRMICS network.

Study documentation on ANSWER, RAID, and IOIS projects - obtain copies the software for evaluation and experimentation.

Begin design of systems to support AIRMICS in the areas of Project Management, Planning and Budgeting, and Presentation & Publishing.

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for January, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: February 1, 1990

During the month of January, the documentation for ANSWER, RAID, and IOIS was examined. We are studying these systems to determine whether they can be installed and integrated into the AIRMICS testbed network.

The Sun 3/50 workstation was moved to the O'Keefe building and installed on the AIRMICS local area network. Bill Putnam will be using it for the duration of the project and will work primarily at AIRMICS.

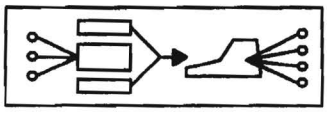
We are examining the AIRMICS Information Architecture Reference Model (IARM) for use as a basis for the development of a testbed network to explore issues and demonstrate applications of open systems technology and interoperability. Some refinement of the IARM will be required.

Current goals are:

Study documentation on ANSWER, RAID, and IOIS projects - obtain copies of software for evaluation and experimentation.

Examine IARM and determine whether it can be used as an implementation guide for open systems development

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for February, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: March 1, 1990

During the month of February the examination of ANSWER, RAID, and IOIS documentation continued. It appears that the systems are not yet far enough along in development to be installed and integrated in the AIRMICS testbed. To ease the integration AIRMICS should install the MIT X Window System in the testbed environment and should include Sun SPARC workstations in the testbed. The task of obtaining and installing the X Window software will be performed under this project. Plans for enhancement of the testbed network will be developed.

Study of the IARM continued. The model is being changed to accommodate the Army's Information Systems Architecture Circa 1997. We need to get more information on open systems standards and technologies.

Current goals are:

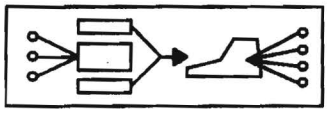
Obtain and install X Window System software in AIRMICS network

Develop plan for testbed network enhancement

Continue examination of IARM and integrate with ISA97

Gather information on open systems standards and technologies

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for March, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: April 1, 1990

During the month of March the X Window System software was obtained and loaded onto a Sun 3 workstation in the AIRMICS network. The system is quite large (over 150MB of source code) and will require some study to configure and build.

Work was also done on the development of the IARM and its integration with the ISA97 plan. The two will be merged into a single model of information system architecture. Information on open systems standards and technologies is being gathered.

AIRMICS PC systems have been integrated into the network using PC-NFS software for TCP/IP communications. Supported applications include remote printing, file transfer, electronic mail, and remote file systems mounted from Sun workstations and the AIRMICS file server. We are looking at interface compatibility products to provide a consistent user interface between DOS and UNIX systems.

Current goals are:

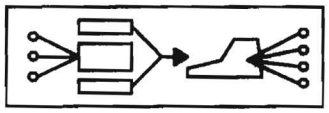
Develop plans for AIRMICS open systems testbed network enhancement

Gather information on open systems standards and technologies

Configure and build X Window System

Continue work with IARM and ISA97 integration

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for April, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: May 1, 1990

During the month of April the primary focus was on the installation of the X Window System on the Sun 3 workstations. The software was loaded, configured, and compiled and is being tested. Once Sun 3 installation is complete the software will have to be re-compiled for the 386i workstations. We are also investigating X products for the DOS systems in the network.

The focus of the project has shifted from integration of ANSWER, RAID, and IOIS to refinement of the IARM into the ISA 97 Compliant Architecture Model (ICAM). The ICAM will describe the components of an information system and relate them to open systems standards and technologies. Bill Putnam will work on the definition of the Entry and Operating System layers of the model.

Current goals are:

Complete installation and testing of X on Sun equipment

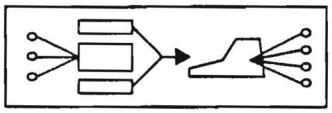
Gather information on X terminals and workstations for consideration in testbed network planning

Develop ICAM Entry and Operating System definitions

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for May 1 to May 31, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: June 12, 1990

During the period from May 1 to May 31, 1990 several steps were taken to continue the upgrading of the AIRMICS research network. Information was collected on the costs and capabilities of several commercial hardware and software products of interest to AIRMICS including X servers for PC systems, X terminals, and diskless/dataless SPARC workstations. A report describing the information collected and considering the pros and cons of the technologies is being prepared. Also, software to control the uninterruptible power supply on the AIRMICS file server was installed. A special cable will be required to connect the UPS control port to the server due to the incomplete implementation of the RS232 standard on the server's ALM serial card unit. Installation of the X window system on the Sun 3 systems is complete and is beginning on the 386i systems. SunOS 4.1 for the Sun 3 systems arrived but cannot be installed until the memory upgrades for the 3/50 workstations are in place.

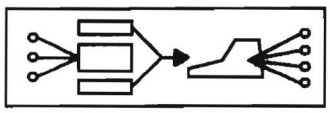
Current goals are:

Install X11R4 on the Sun 386i systems

Complete and distribute report on workstation technologies.

Select a technology to replace VT100 terminals with some type of bit-mapped workstations (X terminal, PC with X, or SPARC SLC)

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for June 1 through June 30, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: July 1, 1990

During the period from June 1 through June 30, 1990, efforts were focused on the development of an ISA 97 architecture model and proof of concept testbed. The objectives are:

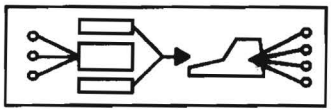
- to demonstrate the feasibility of a distributed computing system based on Open Systems standards
- to examine the current state of interoperability of Open Systems standards
- to explore the transition to Open Systems by migrating selected STAMMIS systems into the open systems environment

To do this we must first gain a better understanding of the open systems standards and environment. We will therefore focus on the identification of applicable standards and try to obtain documentation describing them. We will also be gathering information on commercial implementations of the standards which could be obtained for use in the AIRMICS testbed network.

Current goals are:

- identify open systems standards and reference documents
- identify commercial products which implement open standards
- develop a plan for an open systems testbed network at AIRMICS
- refine the architecture model for ISA 97

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for July, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: August 1, 1990

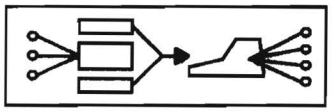
During the month of July work continued on the refinement of the ISA 97 architecture model, which is now known as the ISA 97 Compliant Architecture Model (ICAM). Attention was focused on the description of the Entry, or User Interface Module and the Operating System Module. The objective is to write a high level description of these modules of the model and illustrate them with slides. The descriptions will show the top-level functions of the modules and identify applicable technologies and standards.

Time was also spent planning the upgrade of the AIRMICS testbed network. It was decided that the network should be upgraded with the addition of Sun SPARC workstations, with two systems configured as servers and five systems as dataless client workstations. One of the servers will also be equipped with a second ethernet card to allow the partitioning of the network into subnets in the future. An equipment list was prepared and the order was placed to Sun Microsystems.

Current goals are:

- develop descriptions of ICAM Entry and Operating System modules
- illustrate the ICAM modules with slides
- identify technologies and standards applicable to ICAM modules
- obtain and install upgraded testbed equipment
- obtain and study open systems reference documents

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for August, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: September 1, 1990

During the month of August the descriptions of the ICAM Entry and Operating systems modules were completed and slides illustrating the modules were developed. Suggested revisions to the material, including the separation of the narrative description from the graphics and the extension of the narrative were made.

The Sun workstations equipment for the testbed is on order and is expected to arrive in September. Planning for the installation was begun. Some conversion of data will be required, primarily for Interleaf documents. An automated script for document conversion should be possible.

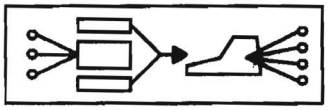
Orders were placed to the National Technical Information Service (NTIS) for the Federal Information Processing Standards (FIPS) publications relating to the POSIX operating systems standard and the GOSIP network standard.

We are considering installing X Window System software on the PC and Mac systems so that they can be integrated into the ICAM testbed with a consistent user interface. Information on X server products for those machines is being gathered.

Current goals are:

- refine ICAM Entry and Operating System module descriptions
- plan installation and integration of new testbed equipment
- obtain and review POSIX and GOSIP documentation
- identify and obtain additional open systems standards information
- obtain and review information on X Window products for PCs and Macs

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for September, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: October 1, 1990

During the month of September the material describing the ICAM Entry and Operating Systems modules were revised and expanded. The two modules were decomposed into component functions and these functions described. The expanded material was formatted with Interleaf and placed on the AIRMICS file server for inclusion with other ICAM briefing material.

Two reports on the portability of GUI-based applications from proprietary environments to the X Window System were prepared. The reports are based on the experience gained from developing the WYWO application first in the SunView environment and then porting it to X using both the XView portability package and toolkit and the MIT X toolkit.

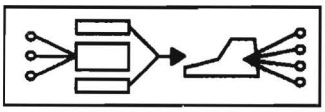
The FIPS publications for POSIX (151) and GOSIP (146) arrived and are being studied. We hope that these publications will give us a basic understanding of the standards and will lead us to other documentation. Our goal is to develop a transition strategy or strategies for the migration to open systems.

Information on X Window system products for PC and Mac systems was obtained and is being evaluated. PC-Xview appears to be a good choice for the DOS systems.

Current goals are:

- refine ICAM Entry and Operating System module descriptions
- study POSIX and GOSIP documentation
- identify and obtain additional open systems standards information
- obtain and review information on X Window products for PCs and Macs

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for October, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: November 1, 1990

During the month of October discussion of the ICAM model continued at weekly meetings held at AIRMICS. The Entry and Operating Systems modules were given further attention, with the intent of defining services provided by those modules to the Application module.

The Sun workstation equipment arrived and was installed in the AIRMICS network. We are in the process of moving user accounts to the new machines, installing Interleaf and other applications software, and converting data files to the SPARC format. An automated script has been developed for the Interleaf data conversion.

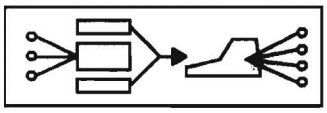
Backup of the AIRMICS file server has become a problem due to the number of 1/2 inch tapes required. The process now takes 10 tapes and most of a day to perform, making the system unavailable for substantial periods. We are looking into the possibility of getting an 8 millimeter Exabyte tape drive which would back up the entire system on a single 2.5GB tape unattended.

We are gathering information on COBOL compilers for the Sun SPARC systems to use in porting STAMMIS applications into the testbed environment. We are also looking at DOS emulator software for SPARC systems and UNIX work-alike software for the DOS systems. This software would give us a common user environment at the command interpreter level on both types of systems.

Current goals are:

- continue refinement of ICAM modules and services
- study POSIX and GOSIP documentation
- complete integration of new Sun equipment
- get Exabyte tape system for AIRMICS server backup
- identify COBOL compiler and other software for testbed

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for November, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: December 1, 1990

During the month of November the integration of the new Sun SPARCstations into the AIRMICS network was completed. User accounts are now accessible on all Sun systems and files are mounted from the AIRMICS server using NFS. The conversion of Interleaf documents is still ongoing, and is being done with the automated conversion script, which must be run at night, one user at a time.

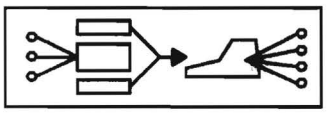
Refinement of the ICAM Entry module services continued and created interest in GUI standards and development environments. The OpenLook GUI standard is implemented on the Sun SPARCstations and is operational in the AIRMICS testbed. The MIT X Window System which is installed on the Sun 3 equipment can interoperate with OpenLook, but is not integrated - it has a different "look and feel". We wish to standardize the look and feel of the GUI across all systems. This will be addressed in three ways: 1) install the MIT X release on the SPARCs as well as the Sun 3 systems; 2) obtain OpenLook for the Sun 3 systems; 3) obtain the Motif GUI standard software (the competitor to OpenLook) and install it on all UNIX systems. This will allow us to explore the different GUI environments and compare them.

We are still gathering information on X products for the PC and Mac. We have placed an order for PC-Xview for DOS systems. The product will be installed on a PC in the AIRMICS lab and will work with PC-NFS to give X Window System compatibility for the PC.

Current goals are:

- continue refinement of ICAM services
- install PC-Xview on lab PC and evaluate
- obtain OpenLook and Motif for all Sun equipment
- obtain Exabyte tape backup system for AIRMICS server

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for December, 1990
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: January 2, 1991

During the month of December the exploration of GUI standards continued. The Motif software distribution was ordered through the Georgia Tech Office of Information Technology. The OpenLook software for Sun 3 workstations was obtained, but will require that those systems be upgraded to SunOS 4.1 before installation. Plans for the upgrade were discussed, and it was decided to wait until the 8mm tape backup system for the server was operational before doing the upgrade.

The MIT X11R4 release software was copied onto the SPARCstation in preparation for compilation and installation.

Ethernet cables and transceivers were ordered for the ICAT testbed network. This equipment will be used to install the testbed systems on a subnet separate from the AIRMICS administrative network so that testing and development activities do not affect other AIRMICS computing activities.

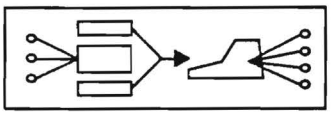
The 8mm Exabyte tape backup system for the AIRMICS file server was ordered.

The PolyShell UNIX work-alike for DOS systems was installed on the AIRMICS file server and mounted on the PC systems using PC-NFS. The software provides a UNIX command interpreter (the C shell) and many UNIX utilities in the MS-DOS environment. The product is being evaluated to determine whether it can provide a consistent user environment between PC systems and UNIX workstations.

Current goals are:

- install X11R4 release on SPARC workstations
- obtain and install Motif on Sun equipment
- continue refinement of ICAM services
- configure and install ICAT research subnet

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for January 1991
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

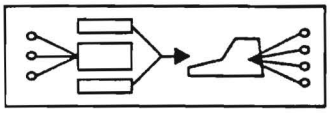
Date: February 1, 1991

During the month of January, 1991 no work was performed on the project. The Personal Services budget for the project was fully expended at the end of December, 1990. The termination date of the project was extended to 16 May 1991 by the sponsor to provide continuity with a follow-on project which has not yet been funded. It is expected that during the remainder of the project term we will complete the acquisition of equipment identified in the earlier phases of the project, but that no further research can be conducted until additional funds are available.

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

Subject: Status report for February 1991
Project C-43-611
Integration of Several Computer Systems Within a Heterogeneous Environment

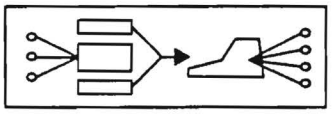
Date: March 1, 1991

During the month of February, 1991 no work was performed on the project. The Personal Services budget for the project was fully expended at the end of December, 1990. The termination date of the project was extended to 16 May 1991 by the sponsor to provide continuity with a follow-on project which has not yet been funded. It is expected that during the remainder of the project term we will complete the acquisition of equipment identified in the earlier phases of the project, but that no further research can be conducted until additional funds are available.

C-43-611

The Software Engineering Research Center

Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

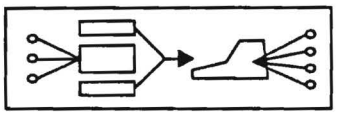
Subject: Status report for March 1991
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: April 1, 1991

During the month of March, 1991 no work was performed on the project. The Personal Services budget for the project was fully expended at the end of December, 1990. The termination date of the project was extended to 16 May 1991 by the sponsor to provide continuity with a follow-on project which has not yet been funded. It is expected that during the remainder of the project term we will complete the acquisition of equipment identified in the earlier phases of the project, but that no further research can be conducted until additional funds are available.

C-43-611

The Software Engineering Research Center
Georgia Institute of Technology / Atlanta Georgia 30332



To: G. C. McCoyd, AIRMICS
W. F. Brown, OCA

From: Bill Putnam
Software Engineering Research Center

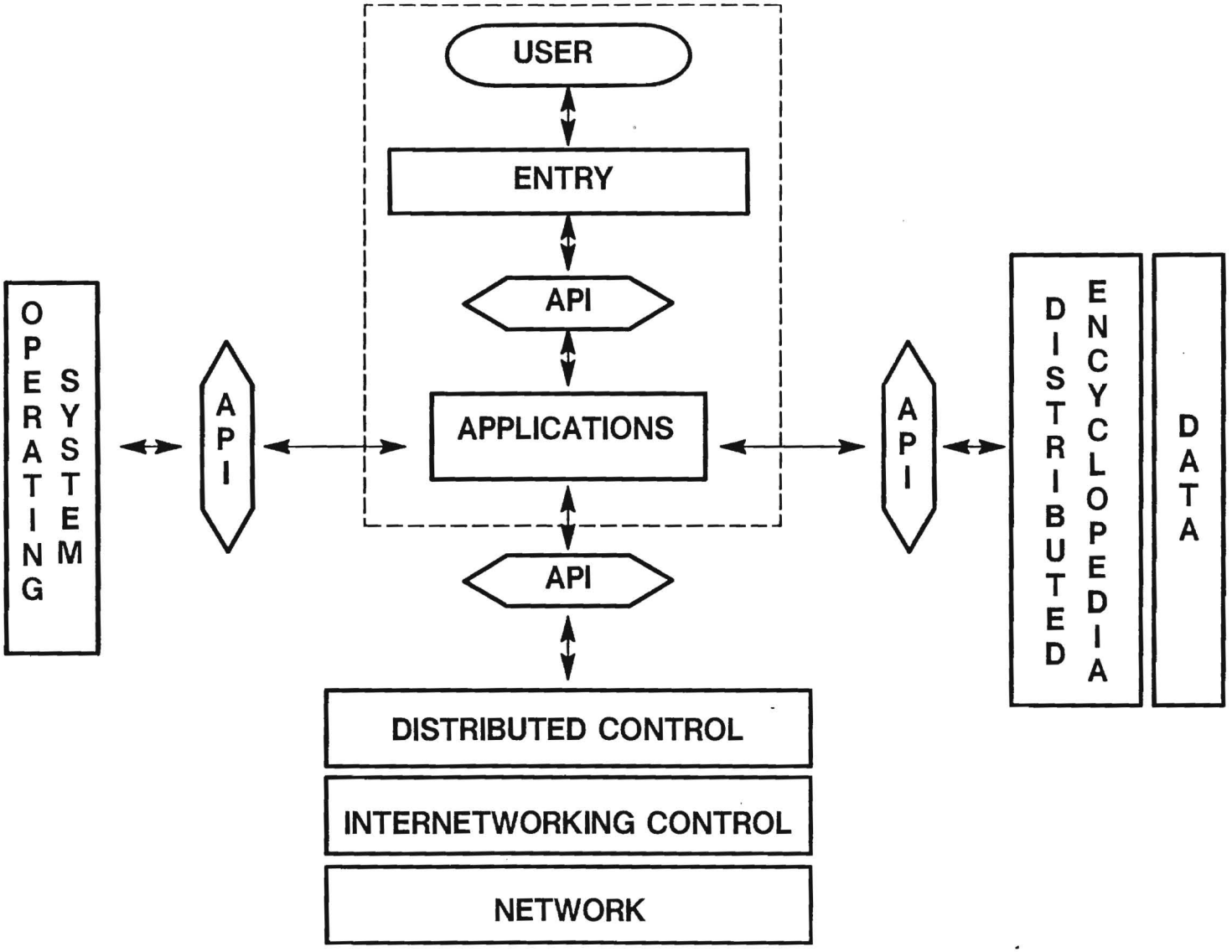
Subject: IPR Memorandum for Record
Project A-70-611
Integration of Several Computer Systems Within a Heterogeneous Environment

Date: February 8, 1991

On February 7, 1991, a meeting was held at AIRMICS to review the status of project A-70-611, Integration of Several Computer Systems Within a Heterogeneous Environment. In attendance were Bill Putnam (GIT), John Mitchell, Jay Gowens, Binh Nguyen, and Reginald Hobbs (all of AIRMICS). Bill Putnam described the status of the work completed and in progress, and delivered copies of slides and materials developed for the ICAM presentation. Copies of the slides are attached. After the project status was reviewed, plans were made for the remaining term of the project. These plans focused on the development of the ICAT research network and the definition of the next phase of work to be done on the ICAM and the Open Systems Environment transition study.

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards



USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

The user's view of the application is shaped by the entry module. At this level the nature of the interaction between user and application is defined. Input and output devices, such as displays, keyboards, and pointing devices, are provided and standards for their operation are defined. The user sees the devices and the rules for their operation.

The application developer sees the details of the interaction between the devices and the application software. The developer uses toolkits to build support for the devices into the application. These toolkits contain the implementation details of standard components of the user interface, such as menus, scrollbars, buttons, and so on.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

Client–to–Client and Client–to–Server Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications. The Inter–Client Communications Conventions Manual defines the standards. All X applications should comply. Compliance is voluntary: non–compliant applications may still work to some extent.

Programming Standards – X11 toolkits (Xt, Athena, HP, OpenLook, Motif) provide basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

This slide present definitions of the primary concepts of the User Interface and Application Standards.

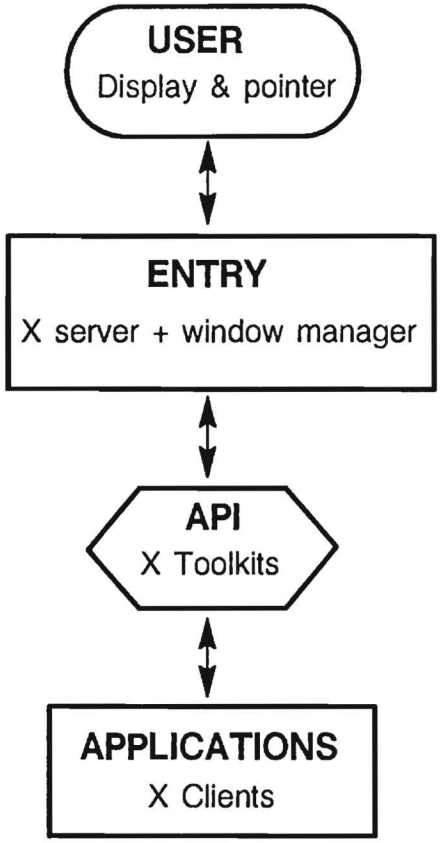
AIRMICS

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager



Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

X Toolkits

Xt HP OpenLook
Athena Motif

X Client Applications

xterm – terminal emulation
xcalendar – scheduling
xmail – electronic mail

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

In this view of the User-to-Application modules we show some of the existing standards and implementations of the interfaces.

The user has available a selection of input and output devices including monochrome and color displays, keyboards, mice, touchpads, trackballs, and so on.The operation of these devices is under the control of the Entry module, which implements some user interface model. The most common model today is the Graphical User Interface (GUI), which usually provides a workspace ("the desktop") containing logically distinct regions ("windows") for applications to display information and accept input. The workspace and its components are under the control of a process which is separate from the applications. Applications must cooperate with this process via standard interfaces to use system display and input resources.

The X Window system is a widely accepted GUI for many hardware platforms. It uses the desktop and window metaphor and is based on the client-server model. The "X server" manages the physical display resources. Applications are "clients" of the server and make requests to use resources.

The Application Program Interface (API) between the application and the X server consists of programming libraries called "toolkits". These toolkits supply the basic building blocks of the user interface: the windows, menus, scrollbars, buttons, etc.

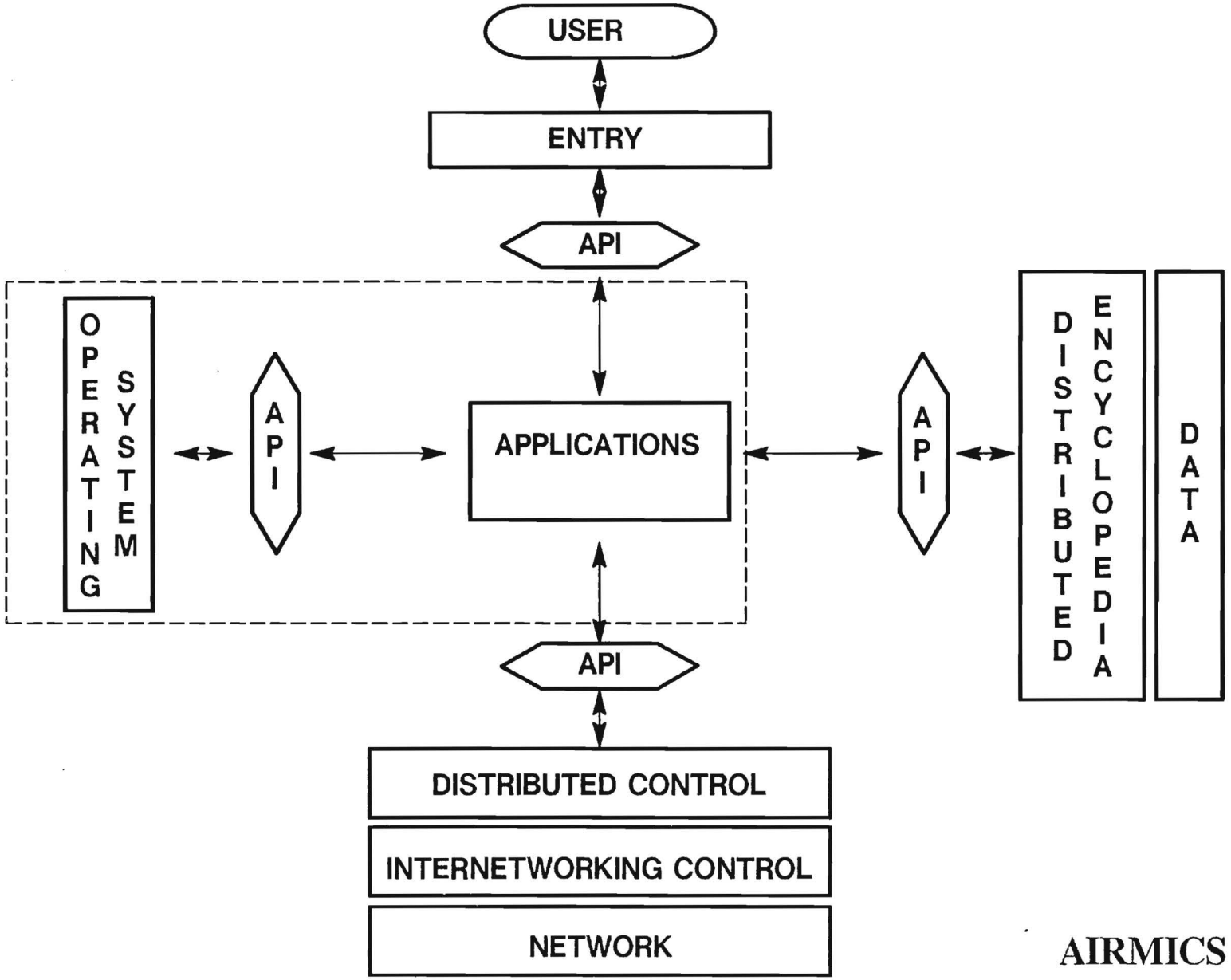
The other component of the Entry module is the "look and feel" of the interface. This is defined by the arrangement of the windows, the assignment of functions to buttons, the style and operation of menus, and so on. These elements are specified in documents called style guides or interface specifications. Some of the rules and constraints in the style guide are built into the toolkits used by the application developer. Others are left to the display server or some independent process for implementation and enforcement.

In the X Window system most of these details are implemented by the "window manager", a process which coordinates the fine detail of the GUI operation, leaving the X server free to handle the raw resources. This separation of function allows users great freedom to customize their computing environments to suit their individual tastes and needs. The X system is independent of any look and feel. The applications can be made independent also. The look and feel component of the GUI is then easily changed or replaced without affecting the majority of the system.

Some examples of style guides, window managers, X toolkits, and typical application programs are shown above.

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

In this section we discuss the interface between the operating system and the application.

The operating system controls the allocation and usage of the computer system resources such as storage, processor time, and input/output devices. Applications interact with the operating system (API) by making requests through library functions and system calls. These are provided to the application developer in the programming libraries and toolkits in the program development environment.

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide application developers with access to system functions and resources through standardized components at various levels.

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users. Applications must conform to the operating system standards for resource allocation, execution, and communications.

Utilities – programs used to maintain system resources

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

This slide present definitions of the primary concepts of the Operating System and Application Standards.

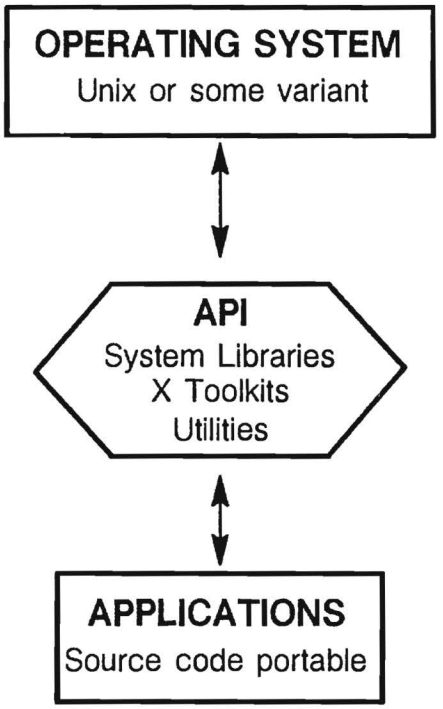
AIRMICS

8/90

BP 11/2/90

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



Command Shells
ksh - korn shell
csh - C shell
sh - Bourne shell

CPU Architectures
MC 680x0
Intel 386/486
SPARC
MIPS

POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Libraries

math	network access & management
windowing	file & data management

Utilities
account management
user registration
backup & recovery

From the Operating System perspective, *everything* looks like an application.

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

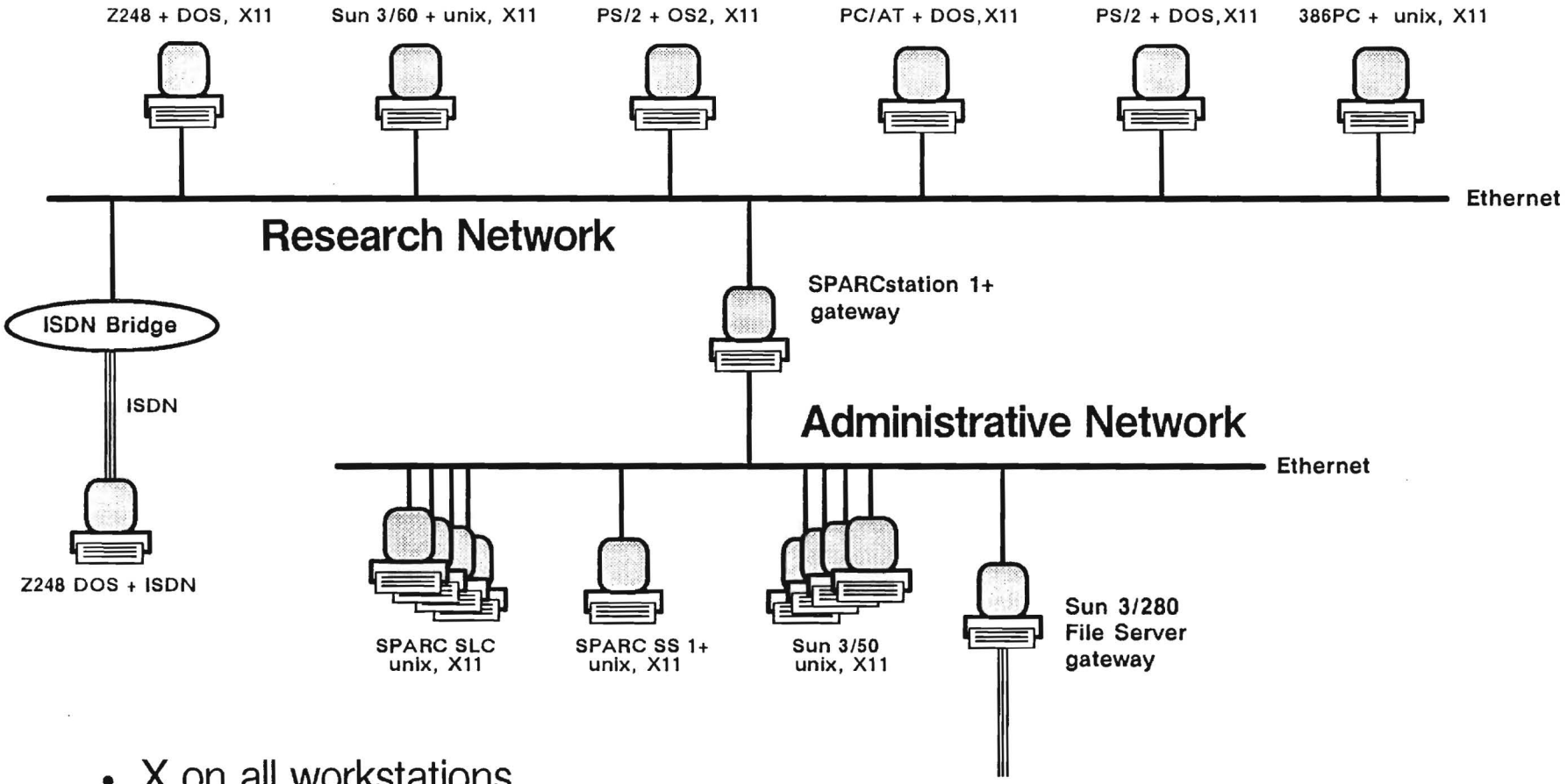
The desire for common operating system across all hardware platforms lead to the development of the POSIX operating system standard. POSIX specifies the methods of resource allocation and management, interprocess communication, and other details of system operation. It specifies the basic utilities to be included in the system for maintenance and operation. These are implemented by standard system function libraries and utility programs.

Many of the concepts of the unix operating system are present in the POSIX standard, and many of the available unix implementations are or soon will be POSIX compliant. Some of the POSIX compliant systems and their components are shown above. Command shells provide a direct user interface to the operating system. Libraries are used by application developers. Utility programs are used to maintain system resources.

POSIX is designed to run on many different hardware architectures, including those shown above.

ISA 97 Conceptual Architecture

Development and Evaluation Testbed



- X on all workstations
- POSIX Operating System Interface
- GOSIP communications on networks
- SQL + RDBMS

USAISEC

ISA 97 Conceptual Architecture

Development and Evaluation Testbed

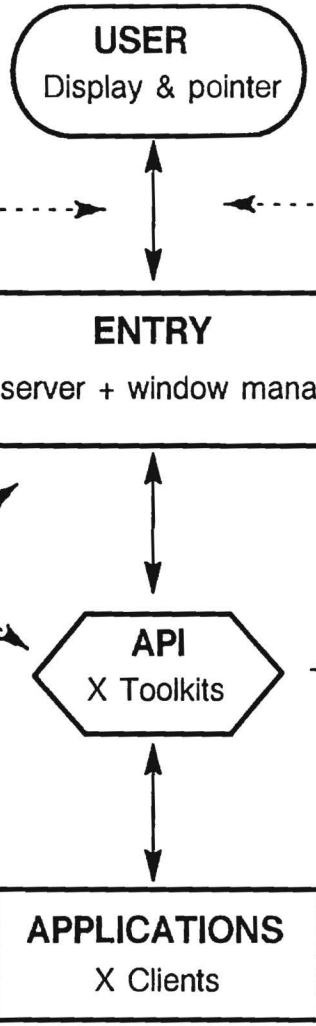
This slide shows the ISA 97 Architecture Development and Evaluation Testbed being developed at AIRMICS.

The testbed demonstrates the concepts described above in a heterogeneous, distributed, multiply connected network environment. Its features include a portable distributed GUI based on the X Window system, three different hardware architectures (Intel 80x86, Motorola 680x0, RISC), GOSIP-compliant local and wide area networking with ISDN and Ethernet, and POSIX-compliant operating systems.

AIRMICS

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards



Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.

Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager

Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

X11R4 toolkits (Xt, Athena, HP, OpenLook, Motif) – a toolkit provides basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

Client – Client Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications.

ICCCM – Inter Client Communications Conventions Manual: new applications must comply

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

ISA 97 Conceptual Architecture

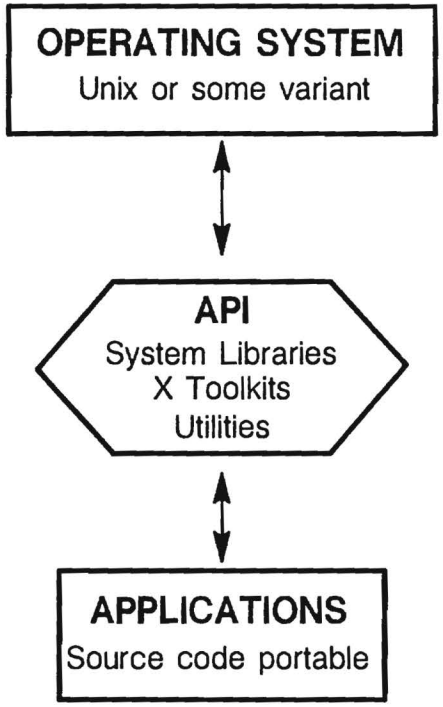
Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide access to system functions and resources for applications

Utilities – programs used to maintain system resources

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users.



POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Command Shells

ksh – korn shell
csh – C shell
sh – Bourne shell

Libraries

math
windowing
network access & management
file & data management

Utilities

account management
user registration
backup & recovery

CPU Architectures

MC 680x0
Intel 386/486
SPARC
MIPS

From the Operating System perspective, *everything* looks like an application.

AIRMICS

DRAFT

Integration of Several Computer Systems Within a Heterogeneous Environment

Final Report

by

William O. Putnam
Ian E. Smith

Software Research Center
College of Computing
Georgia Institute of Technology

prepared for
The Army Institute for Research In Management Information,
Communications, and Computer Science
(AIRMICS)

US Army Contract DAKF11-86-D-0015-0027
GIT Project Number C-43-611

June, 1992

Contents

1.0 Introduction

The original objective of this project was to expand and evaluate the concept of the Information Architecture Reference Model (IARM) by integrating a set of independently developed applications into a heterogeneous environment consisting of several hardware systems, several operating systems, and several database management systems. Upon examination of the systems to be integrated, we found that they were to early in their development to be ported into the testbed environment and integrated at this time. We also identified elements of the AIRMICS IARM testbed environment which required modification before the integration could take place.

Once these issues had been identified, the focus of the project shifted to two areas: preparing the AIRMICS testbed for the integration, and the modification and enhancement the IARM to incorporate it into the Army's Information Systems Architecture Circa 1997 plan (ISA97). This report describes the evolution and enhancement of the testbed environment and the IARM/ISA97 model.

2.0 Summary of Activities

During the period from November 16 through December 31, 1989, background information was collected for the project. Documentation on the ANSWER, RAID, and IOIS projects was obtained and reviewed. An IPR for the ANSWER project was held at AIRMICS which provided an introduction to that system and its capabilities. A search was begun for commercial products, both software and hardware, which will help meet the requirements of the project. Sun versions of popular PC products such as Lotus 123 and Dbase have been located. A product information database was set up.

During the month of January, the documentation for ANSWER, RAID, and IOIS was examined. These systems were studied to determine whether they could be installed and integrated into the AIRMICS testbed network.

The Sun 3/50 workstation was moved to the O'Keefe building and installed on the AIRMICS local area network. Bill Putnam will be using it for the duration of the project and will work primarily at AIRMICS.

We also began examining the AIRMICS Information Architecture Reference Model (IARM) for use as a basis for the development of a testbed network to explore issues and demonstrate applications of open systems technology and interoperability. Some refinement of the IARM will be required.

During the month of February the examination of ANSWER, RAID, and IOIS documentation continued. It appears that the systems are not yet far enough along in development to be installed and integrated in the AIRMICS testbed. To ease the integration AIRMICS should install the MIT X Window System in the testbed environment and should include Sun

SPARC workstations in the testbed. The task of obtaining and installing the X Window software will be performed under this project. Plans for enhancement of the testbed network will be developed.

Study of the IARM continued. The model required changes to accommodate the Army's Information Systems Architecture Circa 1997. A need for more information on open systems standards and technologies was identified.

During the month of March the X Window System software was obtained and loaded onto a Sun 3 workstation in the AIRMICS network. The system is quite large (over 150MB of source code) and required extensive study to configure and build.

Work was also done on the development of the IARM and its integration with the ISA97 plan. We decided to merge the two into a single model of information system architecture. Information on open systems standards and technologies was gathered.

AIRMICS PC systems were integrated into the network using PC-NFS software for TCP/IP communications. Supported applications include remote printing, file transfer, electronic mail, and remote file systems mounted from Sun workstations and the AIRMICS file server. We began examining interface compatibility products to provide a consistent user interface between DOS and UNIX systems.

During the month of April the primary focus was on the installation of the X Window System on the Sun 3 workstations. The software was loaded, configured, compiled, and tested. Once Sun 3 installation was complete the software had to be re-compiled for the 386i workstations. We also investigated X products for the DOS systems in the network.

The focus of the project shifted from integration of ANSWER, RAID, and IOIS to refinement of the IARM into the ISA 97 Compliant Architecture Model (ICAM). The ICAM will describe the components of an information system and relate them to open systems standards and technologies. It was decided that this project will focus on the definition of the Entry and Operating System layers of the model.

During the period from May 1 to May 31, 1990 several steps were taken to continue the upgrading of the AIRMICS research network. Information was collected on the costs and capabilities of several commercial hardware and software products of interest to AIRMICS including X servers for PC systems, X terminals, and diskless/dataless SPARC workstations. A report describing the information collected and considering the pros and cons of the technologies was prepared and submitted. Also, software to control the uninterruptible power supply on the AIRMICS file server was installed. A special cable was required to connect the UPS control port to the server due to the incomplete implementation of the RS232 standard on the server's ALM serial card unit. Installation of the X window system on the Sun 3 systems was completed and work began on the 386i systems. SunOS 4.1 for the Sun 3 systems arrived but could be installed until memory upgrades for the 3/50 workstations were installed.

During the period from June 1 through June 30, 1990, efforts were focused on the development of an ISA 97 architecture model and proof of concept testbed. The objectives were:

- to demonstrate the feasibility of a distributed computing system based on Open Systems standards
- to examine the current state of interoperability of Open Systems standards
- to explore the transition to Open Systems by migrating selected STAMMIS systems into the open systems environment

To do this we must first gain a better understanding of the open systems standards and environment. We therefore decided to focus on the identification of applicable standards and try to obtain documentation describing them. We will also be gathering information on commercial implementations of the standards which could be obtained for use in the AIRMICS testbed network.

During the month of July work continued on the refinement of the ISA 97 architecture model, which is now known as the ISA 97 Compliant Architecture Model (ICAM). Attention was focused on the description of the Entry, or User Interface Module and the Operating System Module. The objective was to write a high level description of these modules of the model and illustrate them with slides. The descriptions show the top-level functions of the modules and identify applicable technologies and standards.

Time was also spent planning the upgrade of the AIRMICS testbed network. It was decided that the network should be upgraded with the addition of Sun SPARC workstations, with two systems configured as servers and five systems as dataless client workstations. One of the servers will also be equipped with a second ethernet card to allow the partitioning of the network into subnets in the future. An equipment list was prepared and the order was placed to Sun Microsystems.

During the month of August the descriptions of the ICAM Entry and Operating systems modules were completed and slides illustrating the modules were developed. Suggested revisions to the material, including the separation of the narrative description from the graphics and the extension of the narrative were made. These slides are included here as Appendix 1.

The Sun workstation equipment for the testbed was ordered and expected to arrive in September. Planning for the installation was begun. We found that some conversion of data would be required, primarily for Interleaf documents. An automated script for document conversion was developed.

Orders were placed to the National Technical Information Service (NTIS) for the Federal Information Processing Standards (FIPS) publications relating to the POSIX operating systems standard and the GOSIP network standard.

We considered installing X Window System software on the PC and Mac systems so that they could be integrated into the ICAM testbed with a consistent user interface. Information on X server products for those machines was gathered. PC-Xview appears to be a good choice for the DOS systems.

During the month of September the material describing the ICAM Entry and Operating Systems modules were revised and expanded. The two modules were decomposed into com-

ponent functions and these functions described. The expanded material was formatted with Interleaf and placed on the AIRMICS file server for inclusion with other ICAM briefing material.

Two reports on the portability of GUI-based applications from proprietary environments to the X Window System were prepared. The reports are based on the experience gained from developing the WYWO application first in the SunView environment and then porting it to X using both the XView portability package and toolkit and the MIT X toolkit. These reports are included here as Appendices 3 and 4.

The FIPS publications for POSIX (151) and GOSIP (146) arrived and were studied. We hope that these publications will give us a basic understanding of the standards and will lead us to other documentation. Our goal is to develop a transition strategy or strategies for the migration to open systems.

During the month of October discussion of the ICAM model continued at weekly meetings held at AIRMICS. The Entry and Operating Systems modules were given further attention, with the intent of defining services provided by those modules to the Application module.

The Sun workstation equipment arrived and was installed in the AIRMICS network. The process of moving user accounts to the new machines, installing Interleaf and other applications software, and converting data files to the SPARC format was quite time consuming. An automated script developed for the Interleaf data conversion was helpful.

Backup of the AIRMICS file server became a problem due to the number of 1/2 inch tapes required. The process was taking 10 tapes and most of a day to perform, making the system unavailable for substantial periods. We began looking into the possibility of getting an 8 millimeter Exabyte tape drive which would back up the entire system on a single 2.5GB tape unattended.

We began gathering information on COBOL compilers for the Sun SPARC systems to use in porting STAMMIS applications into the testbed environment. We also began looking at DOS emulator software for SPARC systems and UNIX work-alike software for the DOS systems. This software would give a common user environment at the command interpreter level on both types of systems.

During the month of November the integration of the new Sun SPARCstations into the AIRMICS network was completed. User accounts are now accessible on all Sun systems and files are mounted from the AIRMICS server using NFS.

Refinement of the ICAM Entry module services continued and created interest in GUI standards and development environments. The OpenLook GUI standard is implemented on the Sun SPARCstations and is operational in the AIRMICS testbed. The MIT X Window System which was installed on the Sun 3 equipment can interoperate with OpenLook, but is not integrated - it has a different "look and feel". We wished to standardize the look and feel of the GUI across all systems. This was to be addressed in three ways: 1) install the MIT X release on the SPARCs as well as the Sun 3 systems; 2) obtain OpenLook for the Sun 3 systems; 3) obtain the Motif GUI standard software (the competitor to OpenLook) and in-

stall it on all UNIX systems. This will allow us to explore the different GUI environments and compare them.

We are still gathering information on X products for the PC and Mac. We placed an order for PC-Xview for DOS systems. The product will be installed on a PC in the AIRMICS lab and will work with PC-NFS to give X Window System compatibility for the PC.

During the month of December the exploration of GUI standards continued. The Motif software distribution was ordered through the Georgia Tech Office of Information Technology. The OpenLook software for Sun 3 workstations was obtained, but will require that those systems be upgraded to SunOS 4.1 before installation. Plans for the upgrade were discussed, and it was decided to wait until the 8mm tape backup system for the server was operational before doing the upgrade.

The MIT X11R4 release software was installed and tested on the SPARCstations. X is now available on all AIRMICS Sun workstations.

Ethernet cables and transceivers were ordered for the ICAT testbed network. This equipment will be used to install the testbed systems on a subnet separate from the AIRMICS administrative network so that testing and development activities do not affect other AIRMICS computing activities.

The 8mm Exabyte tape backup system for the AIRMICS file server was ordered. It was received and installed in March, 1991.

The PolyShell UNIX work-alike for DOS systems was installed on the AIRMICS file server and mounted on the PC systems using PC-NFS. The software provides a UNIX command interpreter (the C shell) and many UNIX utilities in the MS-DOS environment. The product is being evaluated to determine whether it can provide a consistent user environment between PC systems and UNIX workstations.

3.0 Conclusions

With the installation of the X window system software in the AIRMICS testbed we are now ready to install and integrate ANSWER and RAID. Other open system technologies such as Graphical User Interface development tools, database integration tools, and networking tools have been discovered during our migration and should be examined more closely. We believe that several of these tools may be applicable to the problem of transitioning Army information systems from the proprietary mainframe environment into the distributed open systems computing environment.

Of particular interest are CASE and reverse engineering tools such as IDE Software Through Pictures, GUI development tools such as ICS Builder Xcessory, graphical shell and migration tools such as IXI Deskterm, and PC X software such as PC/Xview. A vendor survey should be conducted and selected products obtained for evaluation in the testbed.

It should be possible to migrate a selected STAMMIS into the testbed using these technologies to enhance the user interface and integrate the application with a relational database system.

The IARM has evolved into the ISA97 Compliant Architecture Model (ICAM). Further refinement of all six ICAM modules is needed. Some detail has been provided for the User Interface and Operating System modules. Slides detailing these modules were developed for an ICAM briefing and are provided in Appendix 1.

The AIRMICS IARM testbed has evolved into the ICAM Testbed (ICAT) network. All workstations of the testbed now support the X Window System and the Motif and OpenLook user interfaces. Further evaluation of these two competing interface technologies is needed. It is not yet clear which will prevail in the commercial arena.

Using the enhanced testbed, it should now be possible to implement a demonstration of the ICAM using selected applications and networking technologies. One area of concern is networking: how will GOSIP affect the Operating System and User Interface modules? The X Window system is TCP/IP based at this time, but there are indications that a migration to GOSIP is being considered [see "Mapping the X Window onto Open Systems Interconnection Standards"; Brennan, Thompson, & Wilder; IEEE Network Magazine, May 1991].

4.0 Appendices

4.1 Appendix 1: IARM Slides

4.2 Appendix 2: Equipment List

4.3 Appendix 3: Converting Sunview Applications to X: An Introduction

4.4 Appendix 4: Converting Sunview Applications to X: Some Notes for Programmers

4.5 Appendix 5: Introduction to the X Window System (Slides)

4.1 Appendix 1

ISA 97 Briefing Slides
Operating System and User Interface Modules

William Putnam
College of Computing
Georgia Institute of Technology

This slide set was prepared for use in a technical briefing on the ISA 97 Architecture Model and Testbed.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

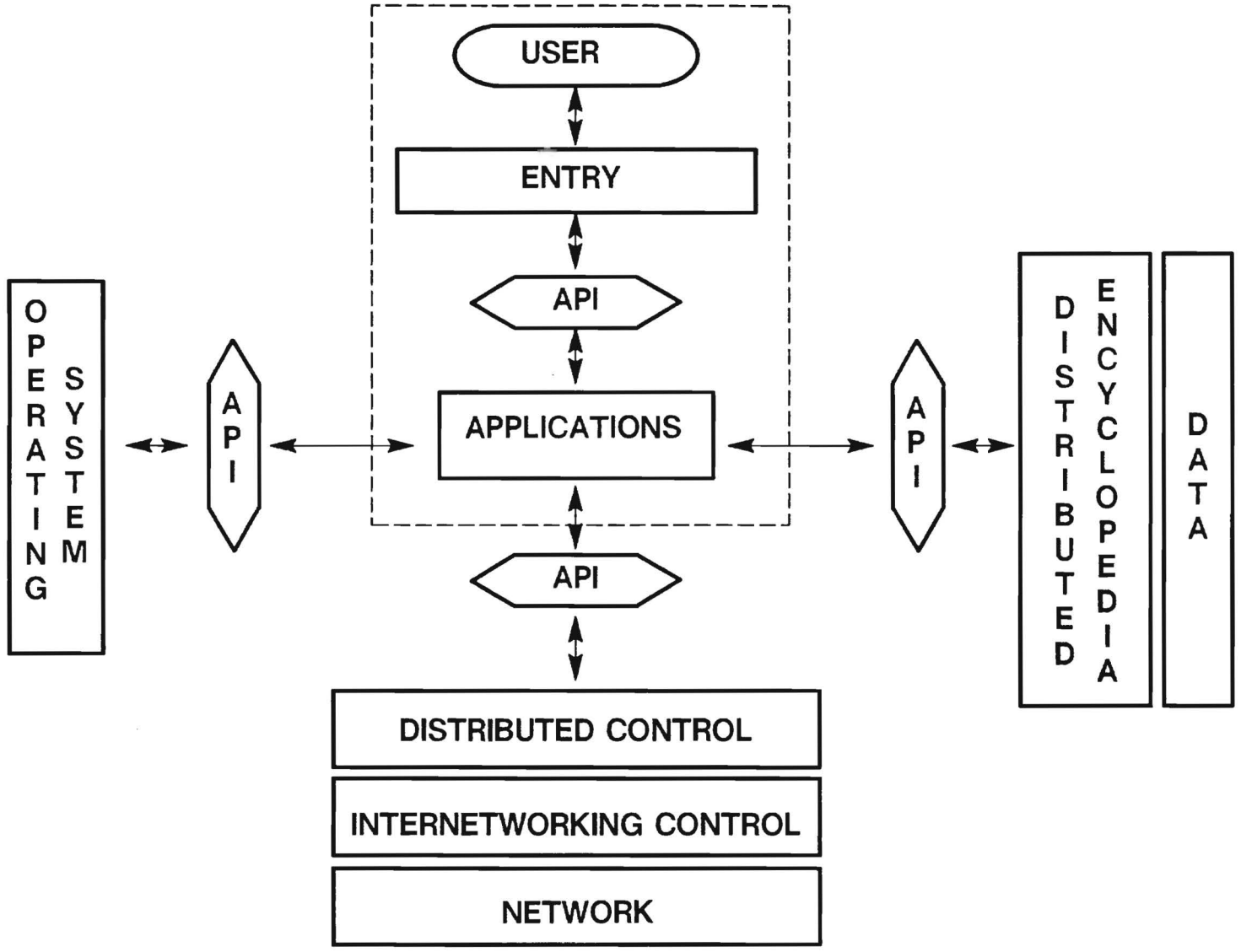
The user's view of the application is shaped by the entry module. At this level the nature of the interaction between user and application is defined. Input and output devices, such as displays, keyboards, and pointing devices, are provided and standards for their operation are defined. The user sees the devices and the rules for their operation.

The application developer sees the details of the interaction between the devices and the application software. The developer uses toolkits to build support for the devices into the application. These toolkits contain the implementation details of standard components of the user interface, such as menus, scrollbars, buttons, and so on.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards



AIRMICS

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

This slide presents definitions of the primary concepts of the User Interface and Application Standards.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

Client-to-Client and Client-to-Server Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications. The Inter-Client Communications Conventions Manual defines the standards. All X applications should comply. Compliance is voluntary: non-compliant applications may still work to some extent.

Programming Standards – X11 toolkits (Xt, Athena, HP, OpenLook, Motif) provide basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

In this view of the User-to-Application modules we show some of the existing standards and implementations of the interfaces.

The user has available a selection of input and output devices including monochrome and color displays, keyboards, mice, touchpads, trackballs, and so on.The operation of these devices is under the control of the Entry module, which implements some user interface model. The most common model today is the Graphical User Interface (GUI), which usually provides a workspace ("the desktop") containing logically distinct regions ("windows") for applications to display information and accept input. The workspace and its components are under the control of a process which is separate from the applications. Applications must cooperate with this process via standard interfaces to use system display and input resources.

The X Window system is a widely accepted GUI for many hardware platforms. It uses the desktop and window metaphor and is based on the client-server model. The "X server" manages the physical display resources. Applications are "clients" of the server and make requests to use resources.

The Application Program Interface (API) between the application and the X server consists of programming libraries called "toolkits". These toolkits supply the basic building blocks of the user interface: the windows, menus, scrollbars, buttons, etc.

The other component of the Entry module is the "look and feel" of the interface. This is defined by the arrangement of the windows, the assignment of functions to buttons, the style and operation of menus, and so on. These elements are specified in documents called style guides or interface specifications. Some of the rules and constraints in the style guide are built into the toolkits used by the application developer. Others are left to the display server or some independent process for implementation and enforcement.

In the X Window system most of these details are implemented by the "window manager", a process which coordinates the fine detail of the GUI operation, leaving the X server free to handle the raw resources. This separation of function allows users great freedom to customize their computing environments to suit their individual tastes and needs. The X system is independent of any look and feel. The applications can be made independent also. The look and feel component of the GUI is then easily changed or replaced without affecting the majority of the system.

Some examples of style guides, window managers, X toolkits, and typical application programs are shown above.

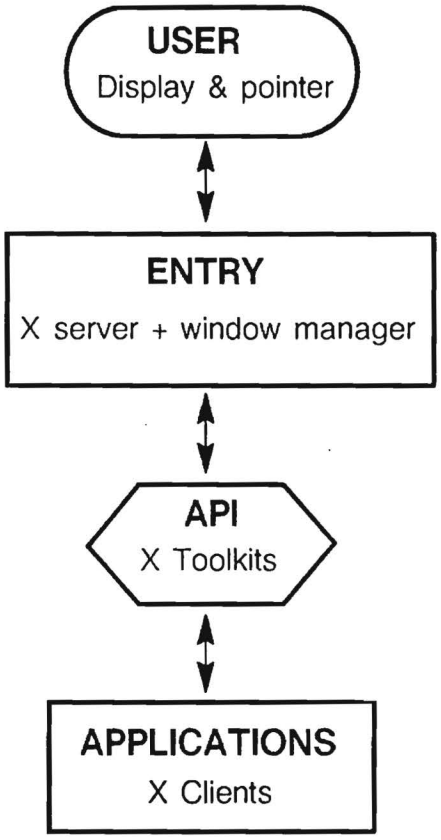
USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager



Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

X Toolkits

Xt HP OpenLook
Athena Motif

X Client Applications

xterm – terminal emulation
xcalendar – scheduling
xmail – electronic mail

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

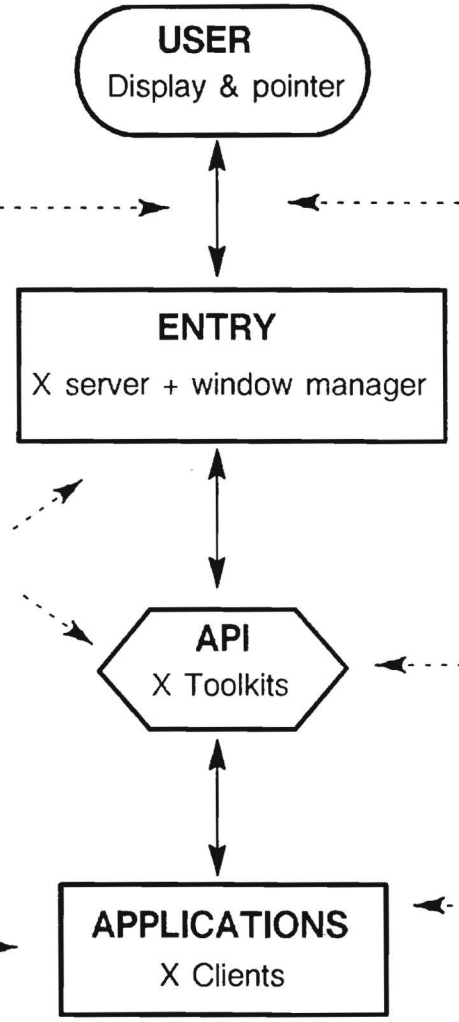
AIRMICS

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.



Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager

Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

X11R4 toolkits (Xt, Athena, HP, OpenLook, Motif) – a toolkit provides basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

Client – Client Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications.

ICCCM – Inter Client Communications Conventions Manual: new applications must comply

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

In this section we discuss the interface between the operating system and the application.

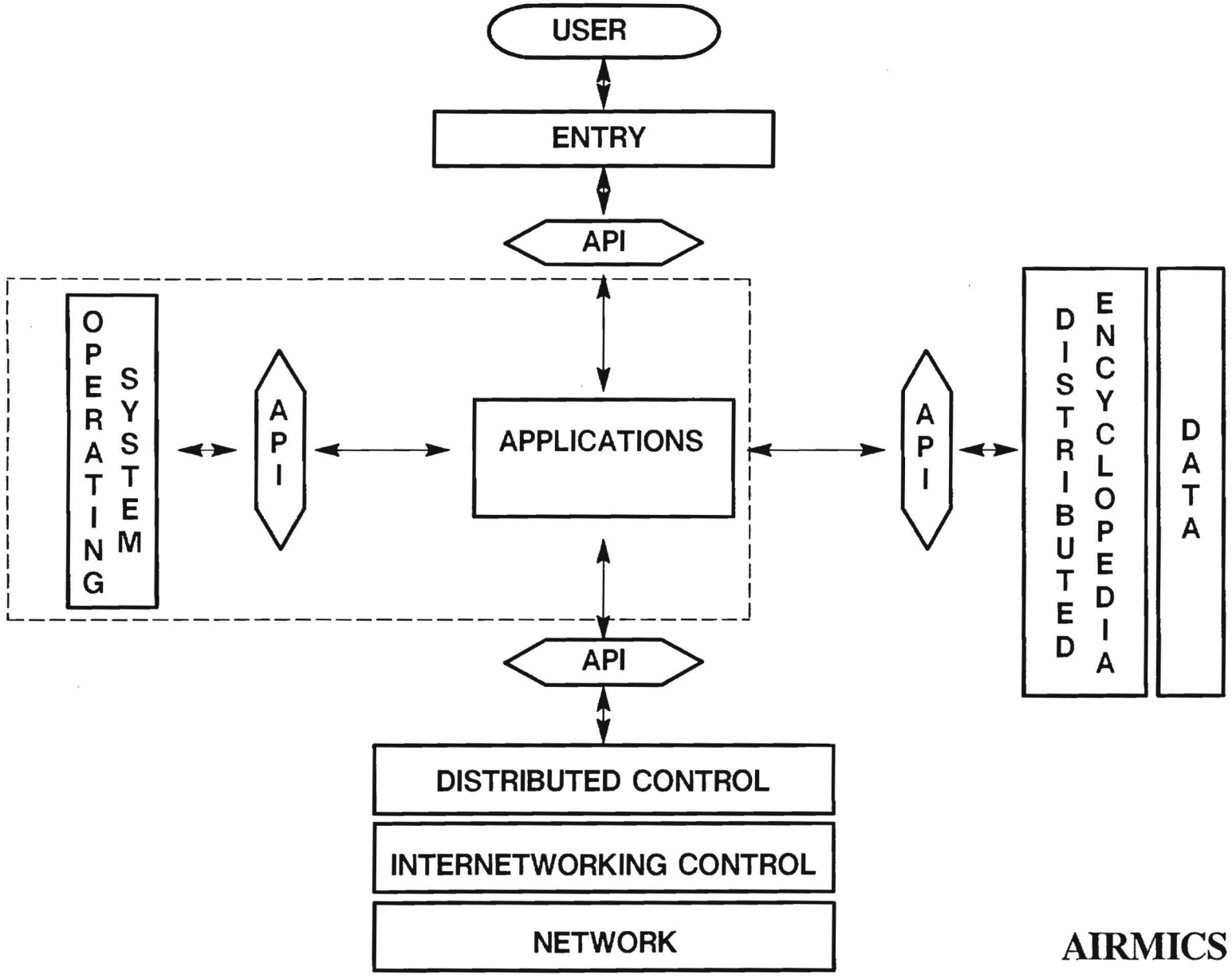
The operating system controls the allocation and usage of the computer system resources such as storage, processor time, and input/output devices. Applications interact with the operating system (API) by making requests through library functions and system calls. These are provided to the application developer in the programming libraries and toolkits in the program development environment.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

This slide presents definitions of the primary concepts of the Operating System and Application Standards.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide application developers with access to system functions and resources through standardized components at various levels.

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users. Applications must conform to the operating system standards for resource allocation, execution, and communications.

Utilities – programs used to maintain system resources

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

The desire for common operating system across all hardware platforms lead to the development of the POSIX operating system standard. POSIX specifies the methods of resource allocation and management, interprocess communication, and other details of system operation. It specifies the basic utilities to be included in the system for maintenance and operation. These are implemented by standard system function libraries and utility programs.

Many of the concepts of the unix operating system are present in the POSIX standard, and many of the available unix implementations are or soon will be POSIX compliant. Some of the POSIX compliant systems and their components are shown above. Command shells provide a direct user interface to the operating system. Libraries are used by application developers. Utility programs are used to maintain system resources.

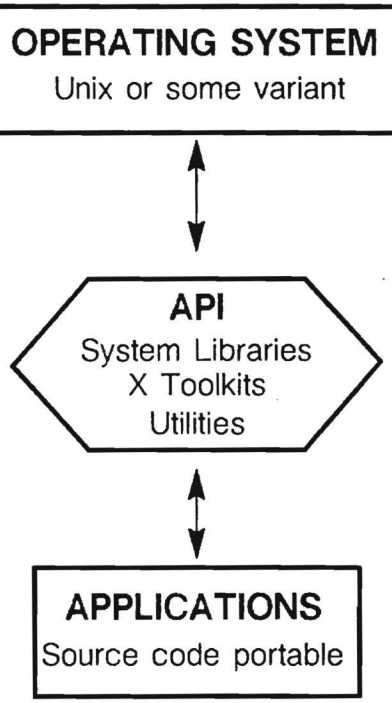
POSIX is designed to run on many different hardware architectures, including those shown above.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



Command Shells

ksh – korn shell
csh – C shell
sh – Bourne shell

CPU Architectures

MC 680x0
Intel 386/486
SPARC
MIPS

POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Libraries

math	network access & management
windowing	file & data management

Utilities

account management
user registration
backup & recovery

From the Operating System perspective, *everything* looks like an application.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

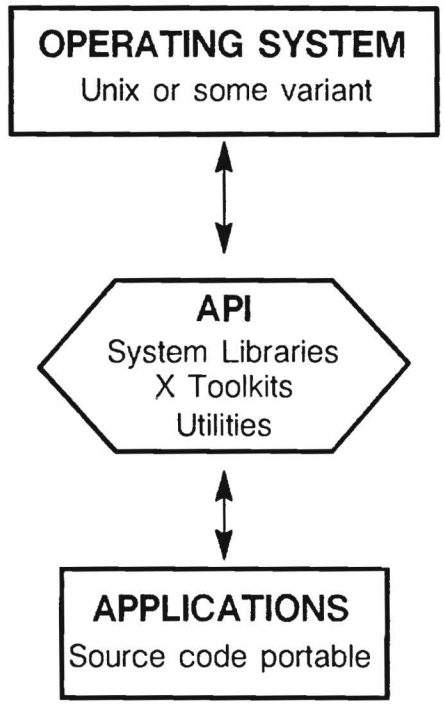
Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide access to system functions and resources for applications

Utilities – programs used to maintain system resources

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users.



POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Command Shells

ksh – korn shell
csh – C shell
sh – Bourne shell

Libraries

math
windowing
network access & management
file & data management

Utilities

account management
user registration
backup & recovery

CPU Architectures

MC 680x0
Intel 386/486
SPARC
MIPS

From the Operating System perspective, *everything* looks like an application.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Development and Evaluation Testbed

This slide shows the ISA 97 Architecture Development and Evaluation Testbed being developed at AIRMICS.

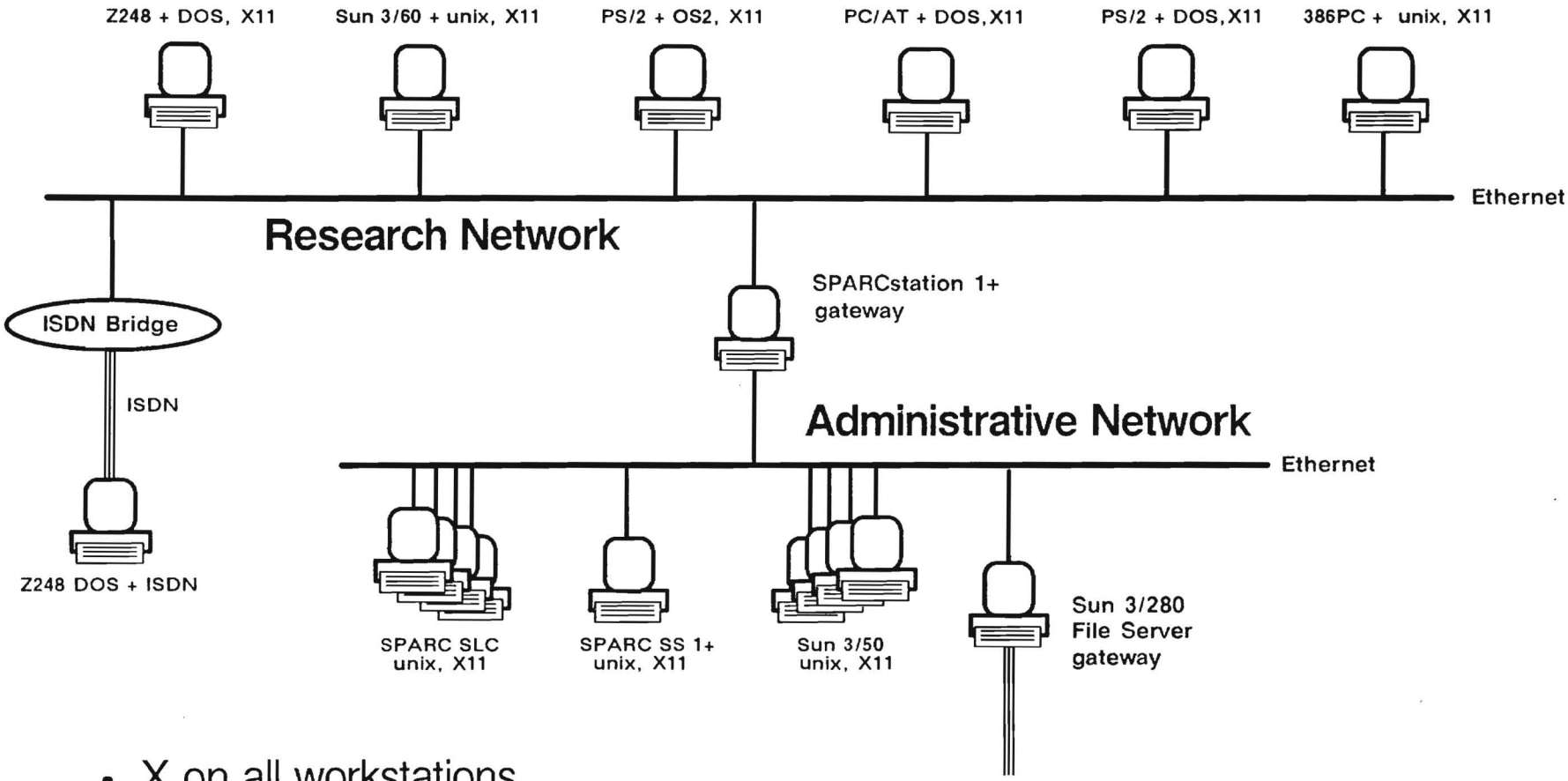
The testbed demonstrates the concepts described above in a heterogeneous, distributed, multiply connected network environment. Its features include a portable distributed GUI based on the X Window system, three different hardware architectures (Intel 80x86, Motorola 680x0, RISC), GOSIP-compliant local and wide area networking with ISDN and Ethernet, and POSIX-compliant operating systems.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Development and Evaluation Testbed



- X on all workstations
- POSIX Operating System Interface
- GOSIP communications on networks
- SQL + RDBMS

AIRMICS

BP 11/2/90

4.2 Appendix 2

Equipment List

The following equipment and software was purchased during this project:

Hardware

Two Sun SPARCstation 1+ workstations (model 4/65), with 17 inch monochrome monitor, 1/4 inch tape backup system, 104MB internal disk, and 669MB external SCSI disk.
CPU Serial Numbers: 037F0920, 037F2281
Expansion unit serial numbers: 039G0599, 039G0592

Five Sun SPARCstation SLC workstations (model 4/20), with 104MB external SCSI disk.
Serial Numbers: 029G3090, 029G3093, 029G3009, 029G3094, 029G3091

One Exabyte 8mm tape backup system model 8200S, serial number 588925.

Seven Interlan ethernet transceivers for use with workstations above.

Seven ethernet transceiver cables for use with workstations above.

Two Western Digital Ethernet cards for IBM PS/2 computers.

Software

AT&T ETI Library, Escort 3270 software and ETIP D/DB software, total cost $3,597.65.

PC/Xview - X Window server for DOS systems, $500.

Motif GUI for X Window System, obtained through Georgia Tech at no charge.

4.3 Appendix 3

Converting Sunview Applications to X

An Overview

Ian Smith
College of Computing
Georgia Institute of Technology

Introduction

This document will discuss the problems involved in converting an existing Sunview based application to X. The challenges involved in porting an application are many and varied. They range from simple correspondences to complex design issues. In this paper we will discuss several of the major issues that are evinced in this process. Principal among these problems is the one of "enforcement." X gives the programmer and user all the freedom that they want (or may not want); Sunview enforces user interface rules. To compound this problem, there are areas of Sunview that simply do not correspond to any simple type of construction in X. There is also the problem of getting high quality documentation. All of these issues will be discussed, as well as potential solutions.

Background

Before one can understand the problems in porting programs from Sunview to X, a bit of background is needed. The most significant difference in the way X applications are written versus Sunview is that X has the concept of 'Widgets.' These widgets are user interface construction blocks -- like scrollbars, buttons, and windows -- which the application programmer uses to construct the application. Sunview does not have this concept. Several widgets are usually put together into 'Widget Sets' (sometimes called 'toolkits') in which all widgets function together and use similar user interface conventions. Many vendors market widget sets and several others are available for free. Some of the more popular widget sets include (with authoring organization in parenthesis) the Athena Widgets (MIT), the HP Widgets (Hewlett Packard), the XView toolkit (Sun), the Andrew toolkit (CMU) , and the Motif Widget set (OSF). Of these, only Motif is a 'for-sale' product.

There are many widget sets available, each with their own strengths, weaknesses, and user-interface. This can be a problem if the user is confronted with several programs that function differently. This problem does not exist in Sunview. In this manner, Sunview can be thought of a window system that has exactly one widget set. We will see later that widget sets provide both a problem and a potential solution in our attempts to convert Sunview applications.

Sunview Enforces Policy -- X Does Not

Most of the difficulty involved in porting an application from Sunview to X revolves around, if not hinges upon, the fact that X does NOT enforce any type of user interface (UI) policy

on the programmer. Sunview does enforce policy. In Sunview, for example, it is impossible to generate a warning box that is not in front of the window stack. X makes no such prohibition. This canonical example may be a little far-fetched, but the point is, that under X policy decisions are left to the programmer.

This freedom creates the following problem. If you have behavior in your Sunview program that you want to duplicate, in general, you must construct this behavior yourself. This can be a great deal of work (especially since it may not be clear exactly how Sunview is producing this behavior) or worse, it may be impossible if the widget set you have chosen to use does not permit it.

The problem is compounded when you have interaction of Sunview features between programs that were not written by the same people. For example, two programs written by different persons under Sunview that both support 'cut-and-paste' with the mouse will almost certainly work perfectly well together, even though the people who wrote the program may or may not have had any contact. For this to be achieved under X, all programs and widget sets must comply with a set of standards. As you would expect, however, X does not ENFORCE that these programs be compliant- it only asks that they comply.

Unmappable Areas

There are also areas of Sunview that simply do not correspond directly to an X interpretation. One can spend a considerable amount of time attempting to get X to produce such behavior. In general, it is probably easier to redesign the interface to avoid this situation in the X environment, rather than try to get existing Sunview code ported.

It should be noted that these are usually descended from the fact that Sunview runs only on Sun workstations, so the Sunview implementers can depend on certain facts that are always true with a Sun product. Since X runs on many platforms, such assumptions cannot be made. For example, all Sun workstations run versions of Unix called 'SunOs'. There are subtle differences between normal Unix and SunOs. Sunview can depend on these differences; in contrast X runs on scores of architectures running many versions of Unix, and even on machines that are unix!

Documentation Issues

The base documentation levels of the two systems differ substantially. The Sunview system is excellently documented, and includes good indexing and tables. It also has a large tutorial section, to teach novice and intermediate programmers how to construct UI's. The supplied X documentation is reference material only. (There are several excellent third-party books available on X that document X in great detail and/or provide tutorial materials.) If the programmer does not know exactly what he is looking for, he will not be able to find this in the supplied X documentation. This can cause wastes of time and effort. If outside documentation for X is obtained, this difference can be minimized.

Widget Set Issues

The choice of widget set is a pivotal decision in the construction of any X application. We will focus our discussion on two widget sets primarily, the XView toolkit (widget set) from Sun, and the Athena Widgets from MIT.

The XView toolkit is a set of programs written by Sun in an attempt to make X programming feel like Sunview programming. The XView user interface is different than the Sunview one. XView however is not a panacea. As mentioned before, X and Sunview are considerably different and XView suffers when it tries to make X an exact match of Sunview. To its credit, XView is considerably easier to program in than most widget sets, and does in fact make X programming 'feel' like Sunview. However, its problems are considerable. I was unable in some cases to get XView to reproduce behavior of a Sunview program. The XView documentation, unlike the Sunview documentation, is poor. Additionally, XView is Open Look compliant, and it expects an OL compliant window manager to function perfectly. This is not a major drawback, but can be annoying.

Sun provides with XView a set of scripts that are supposed to convert some or all the of the Sunview code to XView. As far as I could tell these were of little value. These scripts generated XView code correctly, but what they generated differed only slightly from the Sunview code. Also, because they were automated, they certainly did not provide any assistance in areas of the code that were proving difficult to port. Converting the code by hand, with the XView conversion documentation proved at least as easy as the automated process. Additionally, this procedure allowed the programmer to use his own intelligence and knowledge of the code to produce higher quality output.

The Athena Widgets are being considered in this paper more as a representative of the normal type of widget set than anything else. Normal widget sets are those that obey the normal X conventions for widget definition and usage, which XView does not. XView follows the Sunview conventions. Porting an Sunview application to X with the Athena Widgets is difficult too. In many cases, the code has to be revamped in order to fit completely within the Athena Widgets constraints. However, due to the fact that the athena widgets comply with the X standards for widget sets, lower level calls can be invoked (if the programmer desires) so that no portion of the interface has to change from a user perspective. This basically allows the programmer to easily subvert the constraints of the widget set.

It should also be noted that much of the third party documentation on X (in fact nearly all of it) expects 'normal' widget sets, like the Athena Widgets. There recently has been a book published, however, on XView programming. I have not reviewed this book.

Conclusion

In conclusion the porting of programs from Sunview to X suffers from three major stumbling blocks:

1) Areas that do not map easily from Sunview to X;

2) Making the correct choice for the widget set in the X environment;

3) Inadequate documentation.

Of these, the second and third can be avoided almost completely if sufficient time and energy are spent prior to the commencement of the project. The first problem is more difficult. In most cases, and experienced X and Unix programmer can work around the

problems, but usually significant changes will have to be made in both the user interface and design of the program. It is this author's opinion that this problem is inherent to the nature of porting programs.

I recommend the following:

1) In all cases, prior to the commencement of the project, every effort should be spent to insure that all necessary X documentation is available. X documentation is usually supplied by third party sources, so several sources may be needed to provide adequate documentation.

2a) If the person(s) porting the program does not have significant X experience, then the XView toolkit and the "porting an application by hand" documentation provided by Sun should be used.

2b) If the person(s) porting the program do have significant X experience, then the toolkit most familiar to the author(s) should be used. The XView toolkit is not as strong as others due to its relationship to Sunview.

3) Keeping the same personnel involved with as many ports as possible will centralize the knowledge of potential porting problems. In general, porting an application is difficult only in spots, and having one person with knowledge of many such bottlenecks would be useful. Augmenting this with good record keeping of the porting process is also recommended. It should be noted that window system applications tend to have similar portions of user interfaces and code re-use is suggested whenever this is possible.

4.4 Appendix 4

Converting Sunview Applications To X

Some Notes for Programmers

Ian Smith
College of Computing
Georgia Institute of Technology

Introduction

This document will describe the problems involved in porting Sunview applications to X. It is assumed that the reader has a general knowledge of unix, X, and Sunview. It will cover specific problem areas in some detail, and will probably only be of interest to those actually doing ports. It assumes that reader has a system running unix (bsd 4.2 or greater), X11R4, and Sunview. This document should be considered as a guideline only, and your port should dictate its own individual needs.

How to write Sunview programs that are easy to port

It cannot be stressed enough how much good, original design will speed the porting process. The most import thing to keep in mind when writing a Sunview program that might be ported to X is separation. Separate the user interface from the functionality as much as possible. The following separation guidelines can be helpful when Sunview work is done.

1) Physically separate the user interface (front end) from the functionality of the program. This can be accomplished with separate files and/or directories. If you are careful to do this porting time can be reduced by large amounts.

2) Write the functionality (back end) as unix program that uses stdin and stdout then write the front end for it. You can usually accomplish this easily via the system call popen. If you require more sophisticated interfaces you can use the fork/exec/pipe sequence of system calls.

3) Document the user interface as thoroughly as the functionality. Good documentation; especially of global variables that proliferate in window system applications, can save a lot of time later on.

How to use X resources efficiently in a porting situation

In general, make all available use of the X resource database. During the process of porting the application, you should generally try to map all "attribute-value" pairs in the Sunview code to an X resource. When you converting the code, as always, try to avoid putting any constants in the X source. This accomplishes two things: First, it makes the program amenable to user customization. Second, it speeds up the porting process by avoiding the need to compile potentially large amounts of unmodified code.

In particular, pay special attention to the translation manager and its translation tables. Good use of these tables in the resource file can give you emulation of the Sunview user

interface bindings for next to no effort. It has the added benefit of letting the user customize his own environment more easily.

Nomenclature

All X library (Xlib) functions begin with an 'X' prefix. All X Toolkit functions functions begin with Xt. Toolkit functions operate at the 'widget level', ie. they operate on entire widgets rather than the lower level constructs such as lines, pixmaps and windows. In every case, however, one can use an Xt routine to retrieve the lower level constructs that underlie widgets. For example, to get the window of a widget, use XtWindowofObject.

Potential Problems

Menus

Menus are radically different in execution between Sunview and the assorted X toolkits. Many toolkits provide some form of 'pop up menu' widget, but its actions and operations vary widely. In the case of the Athena widgets, it is likely that you will want to use some thing like the Athena menu widget in conjunction with the menubutton widget. This arrangement is in general the easiest way to get menus to work, even if it means disguising the menubutton as something else (or making it invisible entirely).

tty subwindows

tty subwindows are a Sunview specific feature which does not exist in X. Any application that uses these will need to be reworked so that type of window is not needed. Generally, you can use the popen() command to give command-line-type interfaces to your X program. If worst comes to worst, you can use the fork/exec/pipe set of system calls to open a shell and read and write to it. Then use an X textwidget to do this display and input. This will require quite a bit of work.

If you want some degree of the shell functionality but don't want to support it specifically the following idea may be useful. Have a field in your interface for command line arguments and support cut buffers or selections so that users may "cut and paste" to and from XTerms.

Icons

The X icon mechanism is much more complicated than the sunview one. To make matters worse, this is also a window manager issue, so all programmer efforts to supply an icon for his application may be foiled by the window manager.

The basic steps in creating an X icon for your program are (using Xlib):

1) Create an icon using the X program 'bitmap.'

2) Use XGetIconSizes(...) to determine what the window manager prefers as the size of the icon.

3) Use XSetIconPixmap(...) to give the window manager the hint that you would like to use the pixmap you created. (Modifying the pixmap if necessary to the specification provided in step 2.)

4) Use XSetIconName(...) to hint to the window manager what you would like your icon's name to be.

It should be noted that if the window manager does not use the window property WM_HINTS for communicating with clients, this method is useless.

Cursors

Cursors are similar in X and Sunview. The only significant difference to be aware of is Sunview prefers you to define your cursor at the time a window is created and X does not. In X you use XDefineCursor to associate a cursor with a window.

There is a trade off associated with using the standard X cursors. If you use one of the predefined cursors (there are about 75 cursors distributed with X) in a manner similar to the way other applications use it, you will have an important clue for X users in how your application is operating. Conversely, if you create a custom cursor that is similar to your Sunview cursor you will have a smaller learning curve for users familiar with the Sunview version. These two sides must be weighed on an individual bases for every port.

Focus (popups and warnings)

The X focusing model is less restrictive than Sunview, in general. If your code uses the Sunview split focus model, be sure that you use the grab functions (XGrabPointer, XGrab-Keyboard,XGrabServer, and most important of all for widget programmers XtAddGrab) to control explicitly which window is getting user input.

In the most part you are going to want to force focus into a widget that has some type of urgent message or warning. If you use the athena dialog widget you can get the widget set to do this for you. Otherwise you will need to call XtAddGrab with appropriate parameters, thus forcing the user to deal with your warning message.

Constraint widgets

In Sunview there are basically two "constraint" widgets, the frame and the panel. The Athena analogues of those two widgets are the form and the paned widget. These are the two basic constraint widgets in the Xaw library and have similar functionality to their Sunview counterparts. (Note: Most other widget sets have similarly named widgets with the same functionality.) However, often X composite widgets can serve needs that are met be complex constructions under Sunview. Be sure to look carefully at the Viewport widget, which is often used to pan a window over a large virtual surface.

Do not feel limited by the constraint widgets of the Athena widgets. Many other widget sets have constraint widgets that will work successfully Athena widget children. It is often quicker to look for a constraint widget that implements the layout semantics you want than to use a widget that does not easily support your layout. The HP widgets are an especially good source of constraint widgets.

Look and feel

Look and feel can be difficult point when porting Sunview applications. In general, it most important to preserve the functionality first, then make a functioning program look and feel

as much as possible like the original. It will be nearly impossible to completely replicate the Sunview version (especially due to the problems caused by differing window managers), but these can help:

1) Give the user as many visual clues as possible to the functionality similarities in both versions.

If both the Sunview and X programs have a quit button, for example, they should both be labeled similarly and located in the same place relative to other objects in the interface. Also, touches like similar choices for fonts and shapes are fairly easy to implement, and give the user excellent visual clues.

2) Make good default choices the new version.

You do not want a situation where the user is confronted with both a new interface, and a new set of choices to make. This should be implemented in a way so that new additions or changes in the X interface should be hidden as much as possible by defaults. Ie, if you have to add a new field in the X version of the program to get similar functionality, give it a default value so that a user coming from the Sunview version does not have to understand immediately the differences in versions

4.5 Appendix 5

An Introduction to the X Window System

Ian Smith
College of Computing
Georgia Institute of Technology

This slide set was prepared for use in an introductory briefing on the MIT X Window system.

USAISEC

Introduction to X

Ian Smith

August 31, 1990

AIRMICS

USAISEC

History

- X was originally developed at M.I.T. to solve the problems of heterogenous networks of bitmapped workstations.

- The first public release of X was in 1986 with version 10R4.

- The current release of X is 11R4 with R5 expected out in 1991.

AIRMICS

USAISEC

X as a Standard

- X is the de facto standard of heterogenous networks.
- X is the standard chosen by the Open Software Foundation.
- X is the standard of research centers arounds the country.

AIRMICS

USAISEC

X and Open Systems

- X is an 'open system'. The X protocol is free and easily available.
- Anyone can modify X to suit their particular needs.
- The X sample servers are not 'supported' like a commercial product.

AIRMICS

USAISEC

Why You Might Want To Use X

- X applications generally are very user customizable; often even extensible by the end user.

- Because X is used at many research centers, there are often applications available for X that not available elsewhere.

- X applications are usually distributed in source form. This allows you to modify or learn from the source.

AIRMICS

USAISEC

Why You Might *Not* Want To Use X

- X is a large system, and has a substantial learning curve.
- X sample servers are not supported by M.I.T.
- Raw X expects user to understand significant amounts of information about the window system itself to reap the benifits of the system.

AIRMICS

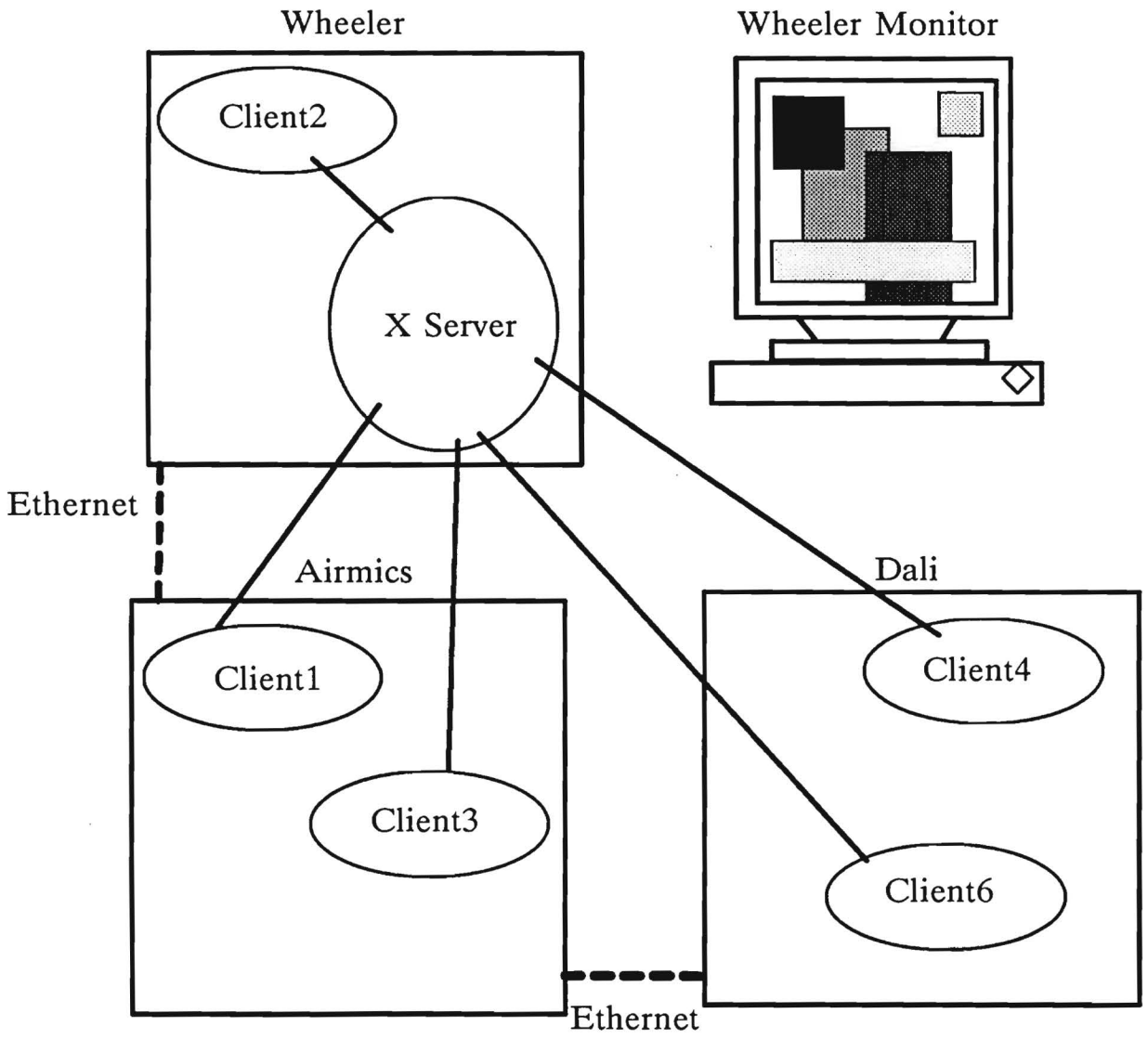
USAISEC

The X Model

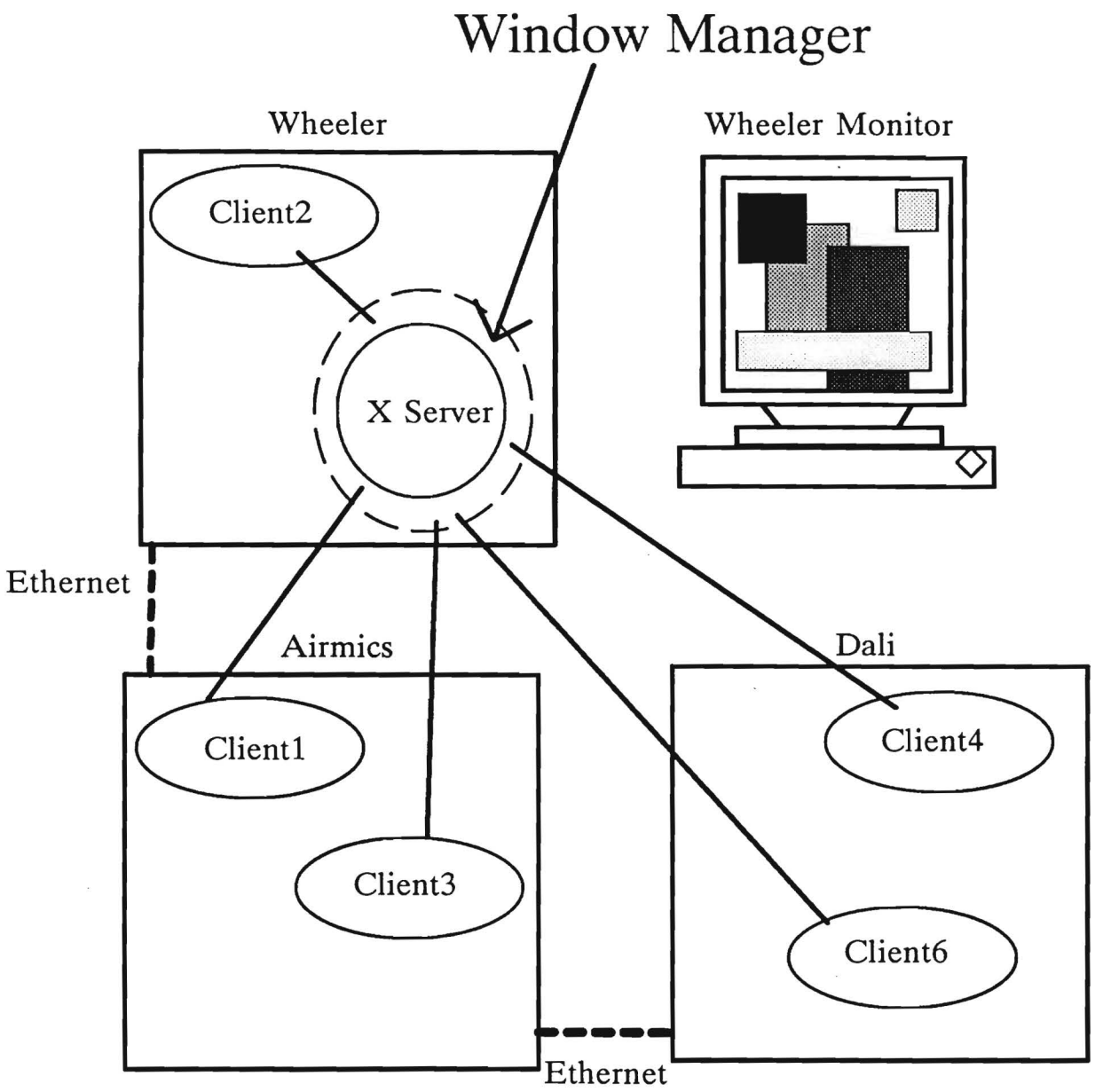
- X is based on a 'server–client' model. This model has one server per workstation with zero to many clients. These clients may be located on the same machine as the server or any machine in a network.

- X also expects one special client called the 'window manager' who is responsible for management and arbitration of all other clients.

AIRMICS



Conceptual X Model



X model with

Window Manager

USAISEC

Setting Up X On Your Workstation

- Many things in X are user customizable. To install a default configuration on your workstation use:

  ```
  % xsetup
  ```

- To start X on your workstation use:

  ```
  % startx
  ```

- To quit X on your workstation select 'quit' on the pull–down menu of the background.

AIRMICS

USAISEC

How To Run X Applications Locally

- In general, just typing an applications name should invoke it and start it running.

- All X applications live in one directory:

  /usr/bin/X11

AIRMICS

USAISEC

X Security

- The sample X server's granularity of security is coarse.

- A primitive form of security is provided by the program xhost. You use xhost to tell your server which machines may access your display. I.e.,

  % xhost +wheeler

AIRMICS

USAISEC

Running X Applications Remotely

- You must inform the client program on the remote machine where you want to display output. You can do this with:

  % setenv DISPLAY [workstation name]:0.0

- If you wish to explicitly send output to different displays use the -display [workstation name]:0.0 command line option.

- For example:

  % xmail -display wheeler:0.0

AIRMICS

USAISEC

If Something Goes Wrong...

- If the program will not run and says "Can't Open Display" check to make sure your DISPLAY environment is correct and that you have set your xhost's properly.

- If an application runs but seems to look strange (fonts are too small, colors are weird, etc.) then check to make sure that the programs "resources" are loaded. You load resources with the program 'xrdb.'

AIRMICS

USAISEC

Some X Applications

Name	Function
xbiff	Icon–oriented biff
xless	X front end for the 'less' file browser
xpostit	Simple 'Post–it' emulation for your workstation
xcalc	Puts a Calculator on display and allows cut–and–paste
xclock	Displays a clock on your screen
xman	Graphical man page browser
xmail	X front end for mail
xterm	Terminal emulator; the most used X application
ico	Draws objects onto a window and rotates them (demo)

AIRMICS

USAISEC

Where X Is Headed

- Mixed media 'displays' with audio and real–time video.
- More commercial support.
- As X gets more of a share of the end–user market, there more 'insulation' available to the user from the complexities of the system.

AIRMICS

Integration of Several Computer Systems Within a Heterogeneous Environment

Final Report

by

William O. Putnam
Ian E. Smith

Software Research Center
College of Computing
Georgia Institute of Technology

prepared for
The Army Institute for Research In Management Information,
Communications, and Computer Science
(AIRMICS)

US Army Contract DAKF11-86-D-0015-0027
GIT Project Number C-43-611

June, 1992

Contents

1.0 Introduction

The original objective of this project was to expand and evaluate the concept of the Information Architecture Reference Model (IARM) by integrating a set of independently developed applications into a heterogeneous environment consisting of several hardware systems, several operating systems, and several database management systems. Upon examination of the systems to be integrated, we found that they were to early in their development to be ported into the testbed environment and integrated at this time. We also identified elements of the AIRMICS IARM testbed environment which required modification before the integration could take place.

Once these issues had been identified, the focus of the project shifted to two areas: preparing the AIRMICS testbed for the integration, and the modification and enhancement the IARM to incorporate it into the Army's Information Systems Architecture Circa 1997 plan (ISA97). This report describes the evolution and enhancement of the testbed environment and the IARM/ISA97 model.

2.0 Summary of Activities

During the period from November 16 through December 31, 1989, background information was collected for the project. Documentation on the ANSWER, RAID, and IOIS projects was obtained and reviewed. An IPR for the ANSWER project was held at AIRMICS which provided an introduction to that system and its capabilities. A search was begun for commercial products, both software and hardware, which will help meet the requirements of the project. Sun versions of popular PC products such as Lotus 123 and Dbase have been located. A product information database was set up.

During the month of January, the documentation for ANSWER, RAID, and IOIS was examined. These systems were studied to determine whether they could be installed and integrated into the AIRMICS testbed network.

The Sun 3/50 workstation was moved to the O'Keefe building and installed on the AIRMICS local area network. Bill Putnam will be using it for the duration of the project and will work primarily at AIRMICS.

We also began examining the AIRMICS Information Architecture Reference Model (IARM) for use as a basis for the development of a testbed network to explore issues and demonstrate applications of open systems technology and interoperability. Some refinement of the IARM will be required.

During the month of February the examination of ANSWER, RAID, and IOIS documentation continued. It appears that the systems are not yet far enough along in development to be installed and integrated in the AIRMICS testbed. To ease the integration AIRMICS should install the MIT X Window System in the testbed environment and should include Sun

SPARC workstations in the testbed. The task of obtaining and installing the X Window software will be performed under this project. Plans for enhancement of the testbed network will be developed.

Study of the IARM continued. The model required changes to accommodate the Army's Information Systems Architecture Circa 1997. A need for more information on open systems standards and technologies was identified.

During the month of March the X Window System software was obtained and loaded onto a Sun 3 workstation in the AIRMICS network. The system is quite large (over 150MB of source code) and required extensive study to configure and build.

Work was also done on the development of the IARM and its integration with the ISA97 plan. We decided to merge the two into a single model of information system architecture. Information on open systems standards and technologies was gathered.

AIRMICS PC systems were integrated into the network using PC-NFS software for TCP/IP communications. Supported applications include remote printing, file transfer, electronic mail, and remote file systems mounted from Sun workstations and the AIRMICS file server. We began examining interface compatibility products to provide a consistent user interface between DOS and UNIX systems.

During the month of April the primary focus was on the installation of the X Window System on the Sun 3 workstations. The software was loaded, configured, compiled, and tested. Once Sun 3 installation was complete the software had to be re-compiled for the 386i workstations. We also investigated X products for the DOS systems in the network.

The focus of the project shifted from integration of ANSWER, RAID, and IOIS to refinement of the IARM into the ISA 97 Compliant Architecture Model (ICAM). The ICAM will describe the components of an information system and relate them to open systems standards and technologies. It was decided that this project will focus on the definition of the Entry and Operating System layers of the model.

During the period from May 1 to May 31, 1990 several steps were taken to continue the upgrading of the AIRMICS research network. Information was collected on the costs and capabilities of several commercial hardware and software products of interest to AIRMICS including X servers for PC systems, X terminals, and diskless/dataless SPARC workstations. A report describing the information collected and considering the pros and cons of the technologies was prepared and submitted. Also, software to control the uninterruptible power supply on the AIRMICS file server was installed. A special cable was required to connect the UPS control port to the server due to the incomplete implementation of the RS232 standard on the server's ALM serial card unit. Installation of the X window system on the Sun 3 systems was completed and work began on the 386i systems. SunOS 4.1 for the Sun 3 systems arrived but could be installed until memory upgrades for the 3/50 workstations were installed.

During the period from June 1 through June 30, 1990, efforts were focused on the development of an ISA 97 architecture model and proof of concept testbed. The objectives were:

- to demonstrate the feasibility of a distributed computing system based on Open Systems standards
- to examine the current state of interoperability of Open Systems standards
- to explore the transition to Open Systems by migrating selected STAMMIS systems into the open systems environment

To do this we must first gain a better understanding of the open systems standards and environment. We therefore decided to focus on the identification of applicable standards and try to obtain documentation describing them. We will also be gathering information on commercial implementations of the standards which could be obtained for use in the AIRMICS testbed network.

During the month of July work continued on the refinement of the ISA 97 architecture model, which is now known as the ISA 97 Compliant Architecture Model (ICAM). Attention was focused on the description of the Entry, or User Interface Module and the Operating System Module. The objective was to write a high level description of these modules of the model and illustrate them with slides. The descriptions show the top-level functions of the modules and identify applicable technologies and standards.

Time was also spent planning the upgrade of the AIRMICS testbed network. It was decided that the network should be upgraded with the addition of Sun SPARC workstations, with two systems configured as servers and five systems as dataless client workstations. One of the servers will also be equipped with a second ethernet card to allow the partitioning of the network into subnets in the future. An equipment list was prepared and the order was placed to Sun Microsystems.

During the month of August the descriptions of the ICAM Entry and Operating systems modules were completed and slides illustrating the modules were developed. Suggested revisions to the material, including the separation of the narrative description from the graphics and the extension of the narrative were made. These slides are included here as Appendix 1.

The Sun workstation equipment for the testbed was ordered and expected to arrive in September. Planning for the installation was begun. We found that some conversion of data would be required, primarily for Interleaf documents. An automated script for document conversion was developed.

Orders were placed to the National Technical Information Service (NTIS) for the Federal Information Processing Standards (FIPS) publications relating to the POSIX operating systems standard and the GOSIP network standard.

We considered installing X Window System software on the PC and Mac systems so that they could be integrated into the ICAM testbed with a consistent user interface. Information on X server products for those machines was gathered. PC-Xview appears to be a good choice for the DOS systems.

During the month of September the material describing the ICAM Entry and Operating Systems modules were revised and expanded. The two modules were decomposed into com-

ponent functions and these functions described. The expanded material was formatted with Interleaf and placed on the AIRMICS file server for inclusion with other ICAM briefing material.

Two reports on the portability of GUI-based applications from proprietary environments to the X Window System were prepared. The reports are based on the experience gained from developing the WYWO application first in the SunView environment and then porting it to X using both the XView portability package and toolkit and the MIT X toolkit. These reports are included here as Appendices 3 and 4.

The FIPS publications for POSIX (151) and GOSIP (146) arrived and were studied. We hope that these publications will give us a basic understanding of the standards and will lead us to other documentation. Our goal is to develop a transition strategy or strategies for the migration to open systems.

During the month of October discussion of the ICAM model continued at weekly meetings held at AIRMICS. The Entry and Operating Systems modules were given further attention, with the intent of defining services provided by those modules to the Application module.

The Sun workstation equipment arrived and was installed in the AIRMICS network. The process of moving user accounts to the new machines, installing Interleaf and other applications software, and converting data files to the SPARC format was quite time consuming. An automated script developed for the Interleaf data conversion was helpful.

Backup of the AIRMICS file server became a problem due to the number of 1/2 inch tapes required. The process was taking 10 tapes and most of a day to perform, making the system unavailable for substantial periods. We began looking into the possibility of getting an 8 millimeter Exabyte tape drive which would back up the entire system on a single 2.5GB tape unattended.

We began gathering information on COBOL compilers for the Sun SPARC systems to use in porting STAMMIS applications into the testbed environment. We also began looking at DOS emulator software for SPARC systems and UNIX work-alike software for the DOS systems. This software would give a common user environment at the command interpreter level on both types of systems.

During the month of November the integration of the new Sun SPARCstations into the AIRMICS network was completed. User accounts are now accessible on all Sun systems and files are mounted from the AIRMICS server using NFS.

Refinement of the ICAM Entry module services continued and created interest in GUI standards and development environments. The OpenLook GUI standard is implemented on the Sun SPARCstations and is operational in the AIRMICS testbed. The MIT X Window System which was installed on the Sun 3 equipment can interoperate with OpenLook, but is not integrated - it has a different "look and feel". We wished to standardize the look and feel of the GUI across all systems. This was to be addressed in three ways: 1) install the MIT X release on the SPARCs as well as the Sun 3 systems; 2) obtain OpenLook for the Sun 3 systems; 3) obtain the Motif GUI standard software (the competitor to OpenLook) and in-

stall it on all UNIX systems. This will allow us to explore the different GUI environments and compare them.

We are still gathering information on X products for the PC and Mac. We placed an order for PC-Xview for DOS systems. The product will be installed on a PC in the AIRMICS lab and will work with PC-NFS to give X Window System compatibility for the PC.

During the month of December the exploration of GUI standards continued. The Motif software distribution was ordered through the Georgia Tech Office of Information Technology. The OpenLook software for Sun 3 workstations was obtained, but will require that those systems be upgraded to SunOS 4.1 before installation. Plans for the upgrade were discussed, and it was decided to wait until the 8mm tape backup system for the server was operational before doing the upgrade.

The MIT X11R4 release software was installed and tested on the SPARCstations. X is now available on all AIRMICS Sun workstations.

Ethernet cables and transceivers were ordered for the ICAT testbed network. This equipment will be used to install the testbed systems on a subnet separate from the AIRMICS administrative network so that testing and development activities do not affect other AIRMICS computing activities.

The 8mm Exabyte tape backup system for the AIRMICS file server was ordered. It was received and installed in March, 1991.

The PolyShell UNIX work-alike for DOS systems was installed on the AIRMICS file server and mounted on the PC systems using PC-NFS. The software provides a UNIX command interpreter (the C shell) and many UNIX utilities in the MS-DOS environment. The product is being evaluated to determine whether it can provide a consistent user environment between PC systems and UNIX workstations.

3.0 Conclusions

With the installation of the X window system software in the AIRMICS testbed we are now ready to install and integrate ANSWER and RAID. Other open system technologies such as Graphical User Interface development tools, database integration tools, and networking tools have been discovered during our migration and should be examined more closely. We believe that several of these tools may be applicable to the problem of transitioning Army information systems from the proprietary mainframe environment into the distributed open systems computing environment.

Of particular interest are CASE and reverse engineering tools such as IDE Software Through Pictures, GUI development tools such as ICS Builder Xcessory, graphical shell and migration tools such as IXI Deskterm, and PC X software such as PC/Xview. A vendor survey should be conducted and selected products obtained for evaluation in the testbed.

It should be possible to migrate a selected STAMMIS into the testbed using these technologies to enhance the user interface and integrate the application with a relational database system.

The IARM has evolved into the ISA97 Compliant Architecture Model (ICAM). Further refinement of all six ICAM modules is needed. Some detail has been provided for the User Interface and Operating System modules. Slides detailing these modules were developed for an ICAM briefing and are provided in Appendix 1.

The AIRMICS IARM testbed has evolved into the ICAM Testbed (ICAT) network. All workstations of the testbed now support the X Window System and the Motif and OpenLook user interfaces. Further evaluation of these two competing interface technologies is needed. It is not yet clear which will prevail in the commercial arena.

Using the enhanced testbed, it should now be possible to implement a demonstration of the ICAM using selected applications and networking technologies. One area of concern is networking: how will GOSIP affect the Operating System and User Interface modules? The X Window system is TCP/IP based at this time, but there are indications that a migration to GOSIP is being considered [see "Mapping the X Window onto Open Systems Interconnection Standards"; Brennan, Thompson, & Wilder; IEEE Network Magazine, May 1991].

4.0 Appendices

4.1 Appendix 1: IARM Slides

4.2 Appendix 2: Equipment List

4.3 Appendix 3: Converting Sunview Applications to X: An Introduction

4.4 Appendix 4: Converting Sunview Applications to X: Some Notes for Programmers

4.5 Appendix 5: Introduction to the X Window System (Slides)

4.1 Appendix 1

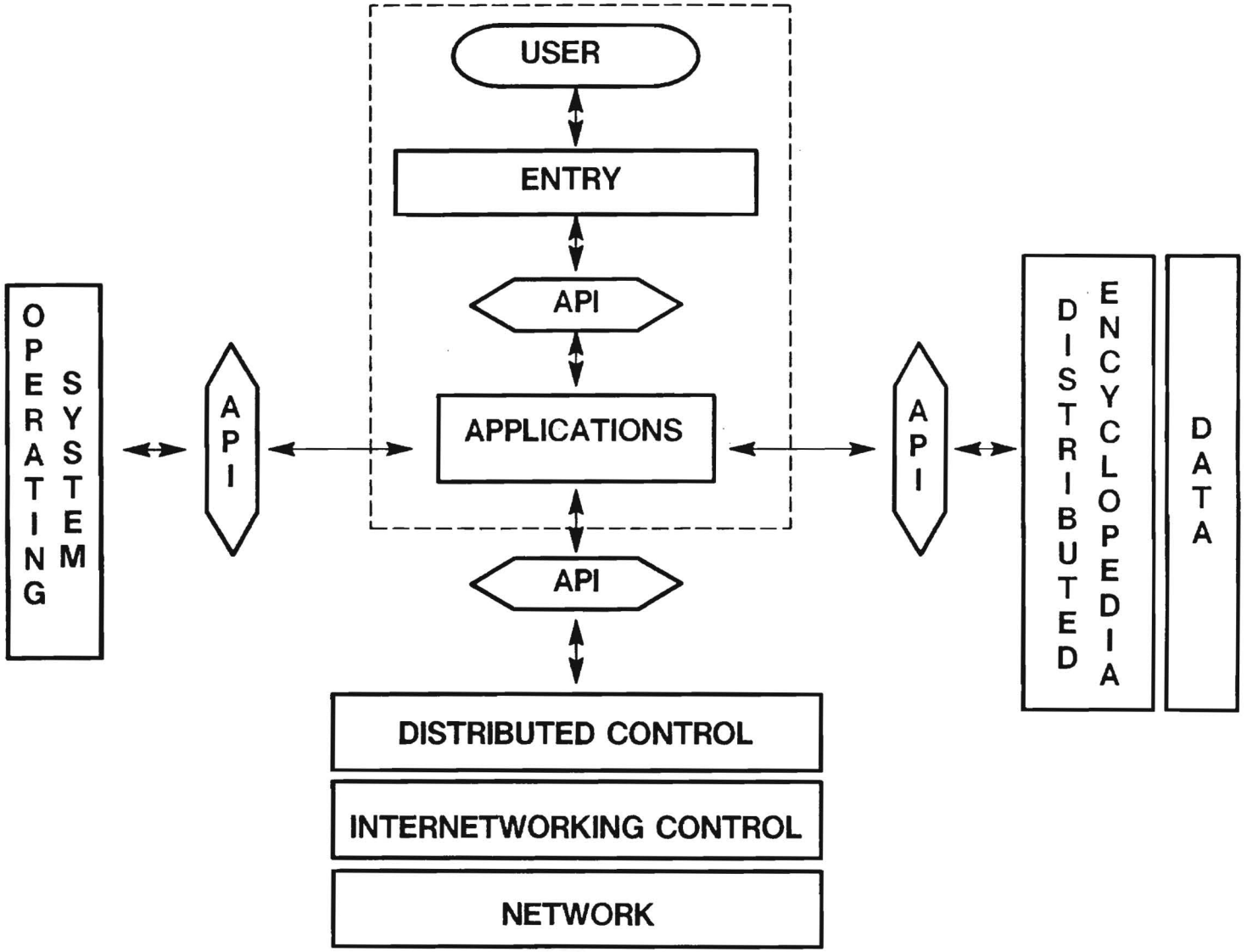
ISA 97 Briefing Slides
Operating System and User Interface Modules

William Putnam
College of Computing
Georgia Institute of Technology

This slide set was prepared for use in a technical briefing on the ISA 97 Architecture Model and Testbed.

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards



USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

The user's view of the application is shaped by the entry module. At this level the nature of the interaction between user and application is defined. Input and output devices, such as displays, keyboards, and pointing devices, are provided and standards for their operation are defined. The user sees the devices and the rules for their operation.

The application developer sees the details of the interaction between the devices and the application software. The developer uses toolkits to build support for the devices into the application. These toolkits contain the implementation details of standard components of the user interface, such as menus, scrollbars, buttons, and so on.

AIRMICS

8/90

BP 11/2/90

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

Client–to–Client and Client–to–Server Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications. The Inter–Client Communications Conventions Manual defines the standards. All X applications should comply. Compliance is voluntary: non–compliant applications may still work to some extent.

Programming Standards – X11 toolkits (Xt, Athena, HP, OpenLook, Motif) provide basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

USAISEC

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

This slide presents definitions of the primary concepts of the User Interface and Application Standards.

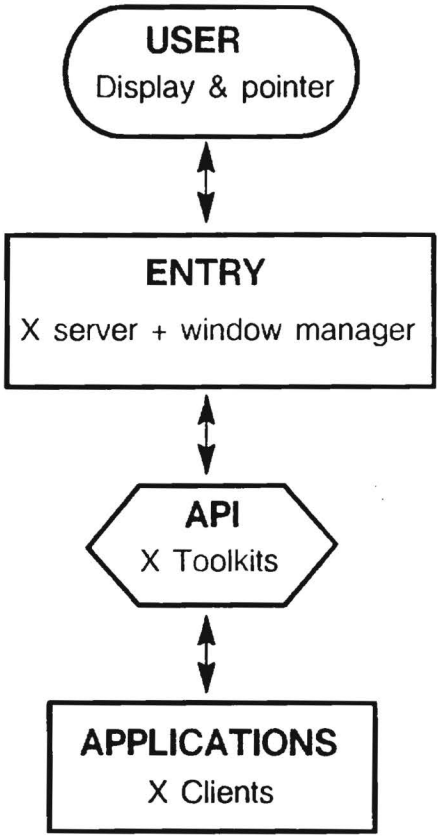
AIRMICS

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager



Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

X Toolkits

Xt HP OpenLook
Athena Motif

X Client Applications

xterm – terminal emulation
xcalendar – scheduling
xmail – electronic mail

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards

In this view of the User-to-Application modules we show some of the existing standards and implementations of the interfaces.

The user has available a selection of input and output devices including monochrome and color displays, keyboards, mice, touchpads, trackballs, and so on.The operation of these devices is under the control of the Entry module, which implements some user interface model. The most common model today is the Graphical User Interface (GUI), which usually provides a workspace ("the desktop") containing logically distinct regions ("windows") for applications to display information and accept input. The workspace and its components are under the control of a process which is separate from the applications. Applications must cooperate with this process via standard interfaces to use system display and input resources.

The X Window system is a widely accepted GUI for many hardware platforms. It uses the desktop and window metaphor and is based on the client-server model. The "X server" manages the physical display resources. Applications are "clients" of the server and make requests to use resources.

The Application Program Interface (API) between the application and the X server consists of programming libraries called "toolkits". These toolkits supply the basic building blocks of the user interface: the windows, menus, scrollbars, buttons, etc.

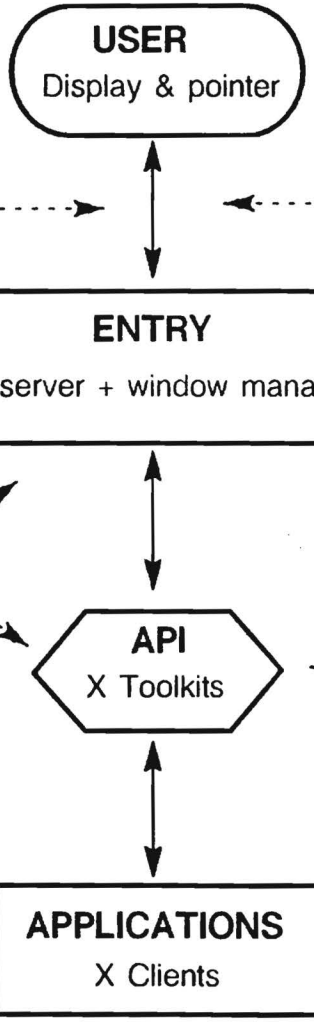
The other component of the Entry module is the "look and feel" of the interface. This is defined by the arrangement of the windows, the assignment of functions to buttons, the style and operation of menus, and so on. These elements are specified in documents called style guides or interface specifications. Some of the rules and constraints in the style guide are built into the toolkits used by the application developer. Others are left to the display server or some independent process for implementation and enforcement.

In the X Window system most of these details are implemented by the "window manager", a process which coordinates the fine detail of the GUI operation, leaving the X server free to handle the raw resources. This separation of function allows users great freedom to customize their computing environments to suit their individual tastes and needs. The X system is independent of any look and feel. The applications can be made independent also. The look and feel component of the GUI is then easily changed or replaced without affecting the majority of the system.

Some examples of style guides, window managers, X toolkits, and typical application programs are shown above.

ISA 97 Conceptual Architecture

User to Application Interfaces and Standards



Graphical User Interface (GUI) Standards – specify "look and feel" of interface and applications, including use of scrollbars, buttons, menus, etc.

Interface Style Guides for X

Motif NextStep
OpenLook Presentation Manager

Window Managers for X

twm – default standard
olwm – OpenLook
mwm – Motif

Window System Standards – specify relationships between applications and display servers. Used by developers to build applications. X11R4 is the most portable and widely available.

X11R4 toolkits (Xt, Athena, HP, OpenLook, Motif) – a toolkit provides basic building blocks and may also provide aspects of the look & feel, enforcing a style guide.

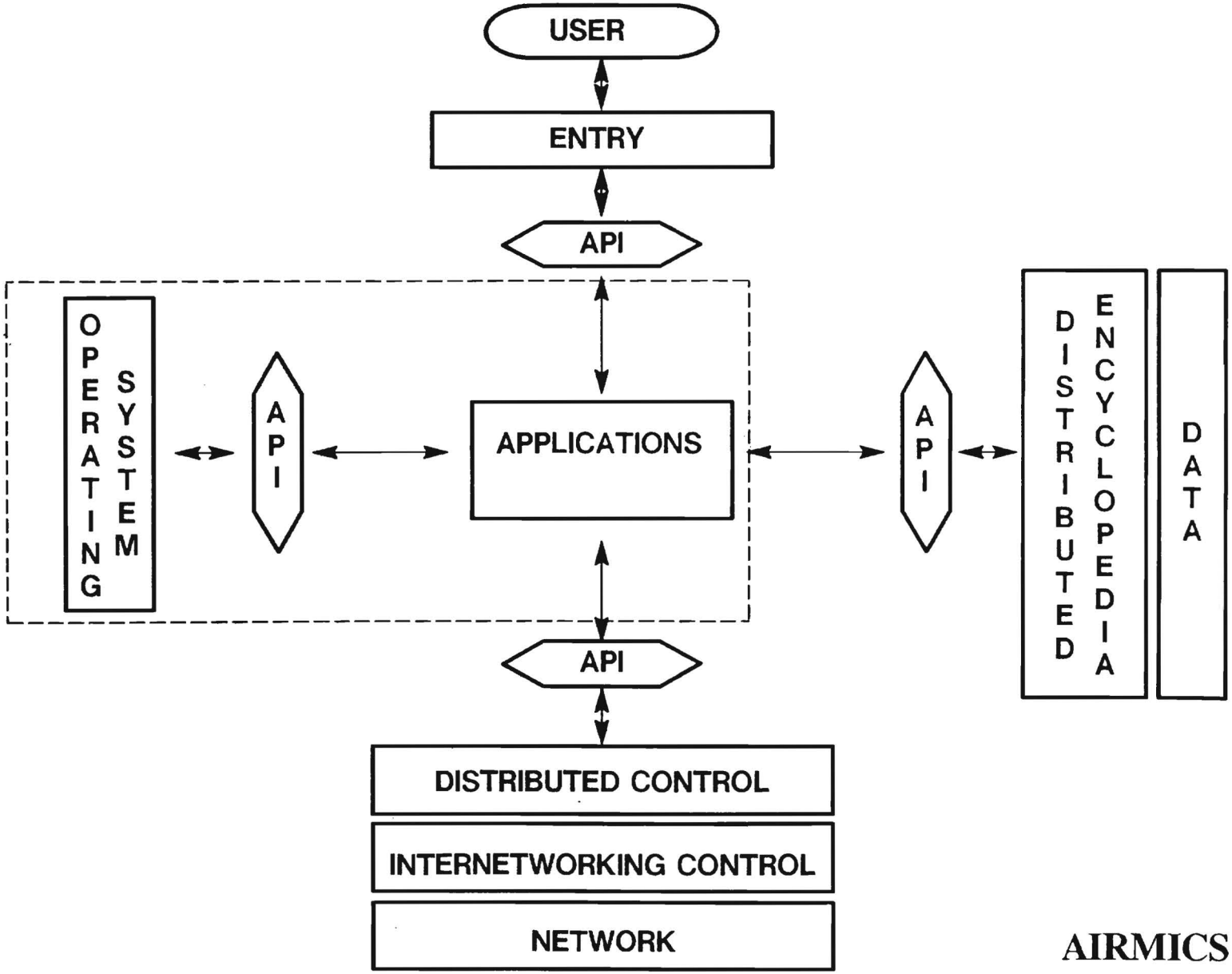
Client – Client Communications Standards – specify interactions between client applications. Used by developers to build cooperating applications.

ICCCM – Inter Client Communications Conventions Manual: new applications must comply

Applications and servers use *networking* services for remote execution and *operating system* services for resource allocation and management.

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

In this section we discuss the interface between the operating system and the application.

The operating system controls the allocation and usage of the computer system resources such as storage, processor time, and input/output devices. Applications interact with the operating system (API) by making requests through library functions and system calls. These are provided to the application developer in the programming libraries and toolkits in the program development environment.

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide application developers with access to system functions and resources through standardized components at various levels.

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users. Applications must conform to the operating system standards for resource allocation, execution, and communications.

Utilities – programs used to maintain system resources

USAISEC

ISA 97 Conceptual Architecture

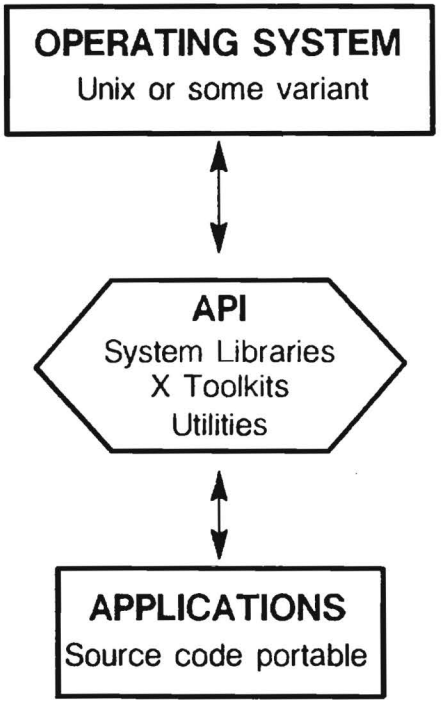
Operating System to Application Interfaces and Standards

This slide presents definitions of the primary concepts of the Operating System and Application Standards.

AIRMICS

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards



Command Shells
ksh - korn shell
csh - C shell
sh - Bourne shell

CPU Architectures
MC 680x0
Intel 386/486
SPARC
MIPS

POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Libraries

math	network access & management
windowing	file & data management

Utilities
account management
user registration
backup & recovery

From the Operating System perspective, *everything* looks like an application.

USAISEC

ISA 97 Conceptual Architecture

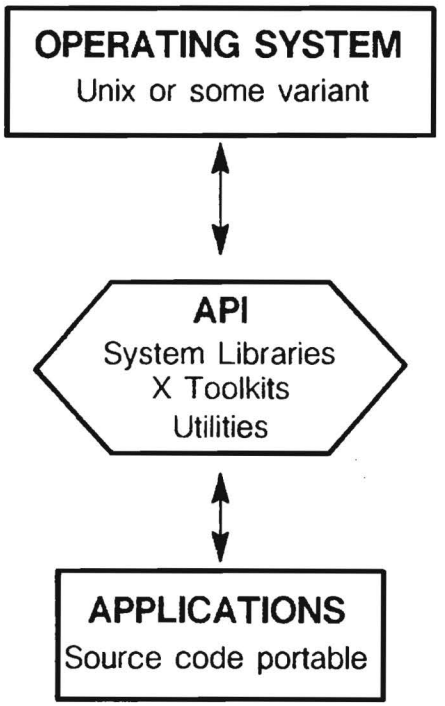
Operating System to Application Interfaces and Standards

POSIX – specify standards for characteristics of operating system environment, including system resources and management, job & process control, portability & interoperability standards

Libraries and Toolkits – provide access to system functions and resources for applications

Utilities – programs used to maintain system resources

Applications – may be DBMS, X servers & clients, networking programs & tools, or command shells for users.



POSIX implementations

SunOS 4.1	Ultrix	Mach
OSF/1	4.3BSD	SysV.4

Command Shells

ksh – korn shell
csh – C shell
sh – Bourne shell

Libraries

math	network access & management
windowing	file & data management

Utilities

account management
user registration
backup & recovery

CPU Architectures

MC 680x0
Intel 386/486
SPARC
MIPS

From the Operating System perspective, *everything* looks like an application.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Operating System to Application Interfaces and Standards

The desire for common operating system across all hardware platforms lead to the development of the POSIX operating system standard. POSIX specifies the methods of resource allocation and management, interprocess communication, and other details of system operation. It specifies the basic utilities to be included in the system for maintenance and operation. These are implemented by standard system function libraries and utility programs.

Many of the concepts of the unix operating system are present in the POSIX standard, and many of the available unix implementations are or soon will be POSIX compliant. Some of the POSIX compliant systems and their components are shown above. Command shells provide a direct user interface to the operating system. Libraries are used by application developers. Utility programs are used to maintain system resources.

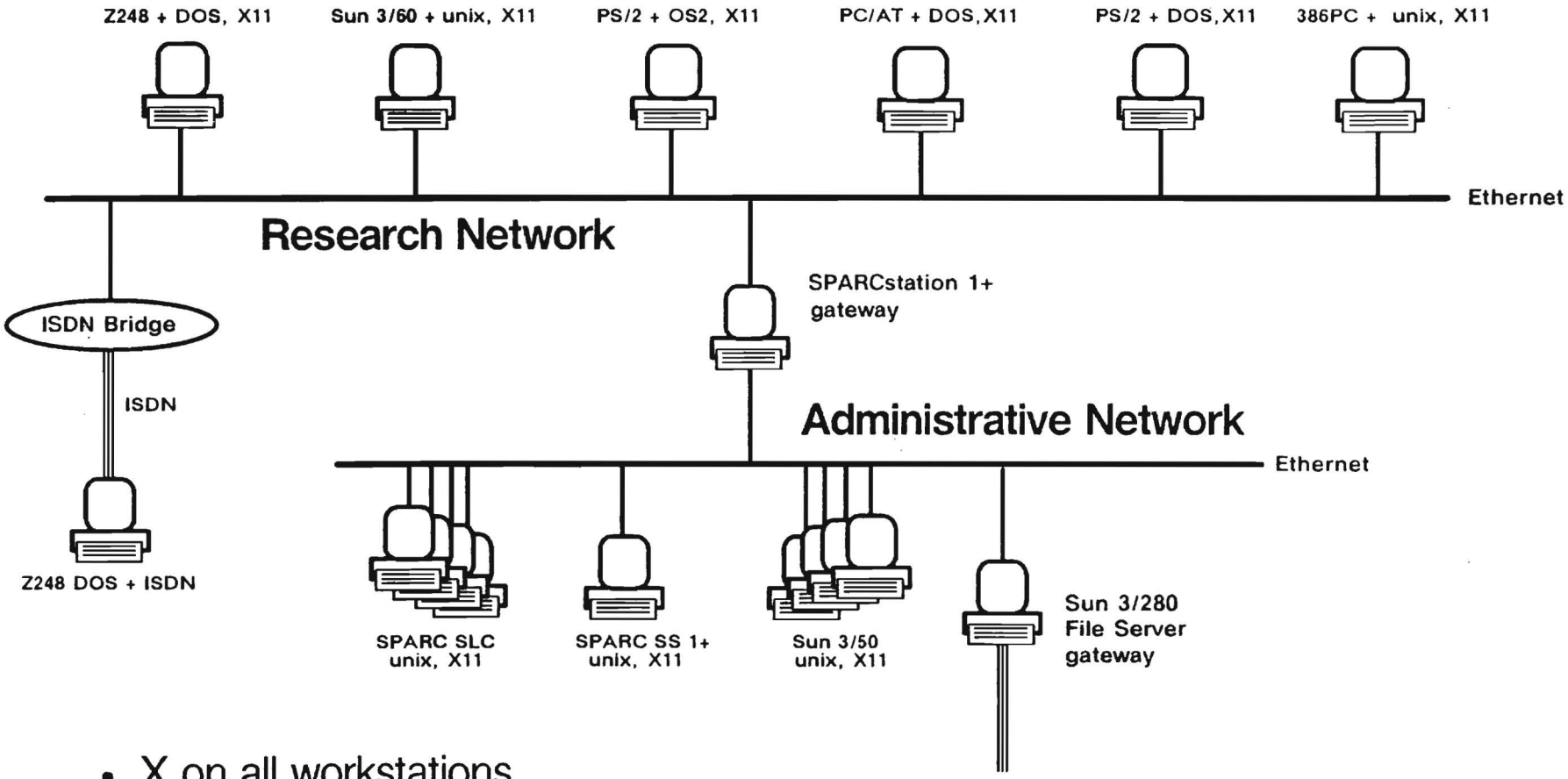
POSIX is designed to run on many different hardware architectures, including those shown above.

AIRMICS

USAISEC

ISA 97 Conceptual Architecture

Development and Evaluation Testbed



- X on all workstations
- POSIX Operating System Interface
- GOSIP communications on networks
- SQL + RDBMS

AIRMICS

BP 11/2/90

USAISEC

ISA 97 Conceptual Architecture

Development and Evaluation Testbed

This slide shows the ISA 97 Architecture Development and Evaluation Testbed being developed at AIRMICS.

The testbed demonstrates the concepts described above in a heterogeneous, distributed, multiply connected network environment. Its features include a portable distributed GUI based on the X Window system, three different hardware architectures (Intel 80x86, Motorola 680x0, RISC), GOSIP-compliant local and wide area networking with ISDN and Ethernet, and POSIX-compliant operating systems.

4.2 Appendix 2

Equipment List

The following equipment and software was purchased during this project:

Hardware

Two Sun SPARCstation 1+ workstations (model 4/65), with 17 inch monochrome monitor, 1/4 inch tape backup system, 104MB internal disk, and 669MB external SCSI disk.
CPU Serial Numbers: 037F0920, 037F2281
Expansion unit serial numbers: 039G0599, 039G0592

Five Sun SPARCstation SLC workstations (model 4/20), with 104MB external SCSI disk.
Serial Numbers: 029G3090, 029G3093, 029G3009, 029G3094, 029G3091

One Exabyte 8mm tape backup system model 8200S, serial number 588925.

Seven Interlan ethernet transceivers for use with workstations above.

Seven ethernet transceiver cables for use with workstations above.

Two Western Digital Ethernet cards for IBM PS/2 computers.

Software

AT&T ETI Library, Escort 3270 software and ETIP D/DB software, total cost $3,597.65.

PC/Xview - X Window server for DOS systems, $500.

Motif GUI for X Window System, obtained through Georgia Tech at no charge.

4.3 Appendix 3

Converting Sunview Applications to X

An Overview

Ian Smith
College of Computing
Georgia Institute of Technology

Introduction

This document will discuss the problems involved in converting an existing Sunview based application to X. The challenges involved in porting an application are many and varied. They range from simple correspondences to complex design issues. In this paper we will discuss several of the major issues that are evinced in this process. Principal among these problems is the one of "enforcement." X gives the programmer and user all the freedom that they want (or may not want); Sunview enforces user interface rules. To compound this problem, there are areas of Sunview that simply do not correspond to any simple type of construction in X. There is also the problem of getting high quality documentation. All of these issues will be discussed, as well as potential solutions.

Background

Before one can understand the problems in porting programs from Sunview to X, a bit of background is needed. The most significant difference in the way X applications are written versus Sunview is that X has the concept of 'Widgets.' These widgets are user interface construction blocks -- like scrollbars, buttons, and windows -- which the application programmer uses to construct the application. Sunview does not have this concept. Several widgets are usually put together into 'Widget Sets' (sometimes called 'toolkits') in which all widgets function together and use similar user interface conventions. Many vendors market widget sets and several others are available for free. Some of the more popular widget sets include (with authoring organization in parenthesis) the Athena Widgets (MIT), the HP Widgets (Hewlett Packard), the XView toolkit (Sun), the Andrew toolkit (CMU) , and the Motif Widget set (OSF). Of these, only Motif is a 'for-sale' product.

There are many widget sets available, each with their own strengths, weaknesses, and user-interface. This can be a problem if the user is confronted with several programs that function differently. This problem does not exist in Sunview. In this manner, Sunview can be thought of a window system that has exactly one widget set. We will see later that widget sets provide both a problem and a potential solution in our attempts to convert Sunview applications.

Sunview Enforces Policy -- X Does Not

Most of the difficulty involved in porting an application from Sunview to X revolves around, if not hinges upon, the fact that X does NOT enforce any type of user interface (UI) policy

on the programmer. Sunview does enforce policy. In Sunview, for example, it is impossible to generate a warning box that is not in front of the window stack. X makes no such prohibition. This canonical example may be a little far-fetched, but the point is, that under X policy decisions are left to the programmer.

This freedom creates the following problem. If you have behavior in your Sunview program that you want to duplicate, in general, you must construct this behavior yourself. This can be a great deal of work (especially since it may not be clear exactly how Sunview is producing this behavior) or worse, it may be impossible if the widget set you have chosen to use does not permit it.

The problem is compounded when you have interaction of Sunview features between programs that were not written by the same people. For example, two programs written by different persons under Sunview that both support 'cut-and-paste' with the mouse will almost certainly work perfectly well together, even though the people who wrote the program may or may not have had any contact. For this to be achieved under X, all programs and widget sets must comply with a set of standards. As you would expect, however, X does not ENFORCE that these programs be compliant- it only asks that they comply.

Unmappable Areas

There are also areas of Sunview that simply do not correspond directly to an X interpretation. One can spend a considerable amount of time attempting to get X to produce such behavior. In general, it is probably easier to redesign the interface to avoid this situation in the X environment, rather than try to get existing Sunview code ported.

It should be noted that these are usually descended from the fact that Sunview runs only on Sun workstations, so the Sunview implementers can depend on certain facts that are always true with a Sun product. Since X runs on many platforms, such assumptions cannot be made. For example, all Sun workstations run versions of Unix called 'SunOs'. There are subtle differences between normal Unix and SunOs. Sunview can depend on these differences; in contrast X runs on scores of architectures running many versions of Unix, and even on machines that are unix!

Documentation Issues

The base documentation levels of the two systems differ substantially. The Sunview system is excellently documented, and includes good indexing and tables. It also has a large tutorial section, to teach novice and intermediate programmers how to construct UI's. The supplied X documentation is reference material only. (There are several excellent third-party books available on X that document X in great detail and/or provide tutorial materials.) If the programmer does not know exactly what he is looking for, he will not be able to find this in the supplied X documentation. This can cause wastes of time and effort. If outside documentation for X is obtained, this difference can be minimized.

Widget Set Issues

The choice of widget set is a pivotal decision in the construction of any X application. We will focus our discussion on two widget sets primarily, the XView toolkit (widget set) from Sun, and the Athena Widgets from MIT.

The XView toolkit is a set of programs written by Sun in an attempt to make X programming feel like Sunview programming. The XView user interface is different than the Sunview one. XView however is not a panacea. As mentioned before, X and Sunview are considerably different and XView suffers when it tries to make X an exact match of Sunview. To its credit, XView is considerably easier to program in than most widget sets, and does in fact make X programming 'feel' like Sunview. However, its problems are considerable. I was unable in some cases to get XView to reproduce behavior of a Sunview program. The XView documentation, unlike the Sunview documentation, is poor. Additionally, XView is Open Look compliant, and it expects an OL compliant window manager to function perfectly. This is not a major drawback, but can be annoying.

Sun provides with XView a set of scripts that are supposed to convert some or all the of the Sunview code to XView. As far as I could tell these were of little value. These scripts generated XView code correctly, but what they generated differed only slightly from the Sunview code. Also, because they were automated, they certainly did not provide any assistance in areas of the code that were proving difficult to port. Converting the code by hand, with the XView conversion documentation proved at least as easy as the automated process. Additionally, this procedure allowed the programmer to use his own intelligence and knowledge of the code to produce higher quality output.

The Athena Widgets are being considered in this paper more as a representative of the normal type of widget set than anything else. Normal widget sets are those that obey the normal X conventions for widget definition and usage, which XView does not. XView follows the Sunview conventions. Porting an Sunview application to X with the Athena Widgets is difficult too. In many cases, the code has to be revamped in order to fit completely within the Athena Widgets constraints. However, due to the fact that the athena widgets comply with the X standards for widget sets, lower level calls can be invoked (if the programmer desires) so that no portion of the interface has to change from a user perspective. This basically allows the programmer to easily subvert the constraints of the widget set.

It should also be noted that much of the third party documentation on X (in fact nearly all of it) expects 'normal' widget sets, like the Athena Widgets. There recently has been a book published, however, on XView programming. I have not reviewed this book.

Conclusion

In conclusion the porting of programs from Sunview to X suffers from three major stumbling blocks:

1) Areas that do not map easily from Sunview to X;

2) Making the correct choice for the widget set in the X environment;

3) Inadequate documentation.

Of these, the second and third can be avoided almost completely if sufficient time and energy are spent prior to the commencement of the project. The first problem is more difficult. In most cases, and experienced X and Unix programmer can work around the

problems, but usually significant changes will have to be made in both the user interface and design of the program. It is this author's opinion that this problem is inherent to the nature of porting programs.

I recommend the following:

1) In all cases, prior to the commencement of the project, every effort should be spent to insure that all necessary X documentation is available. X documentation is usually supplied by third party sources, so several sources may be needed to provide adequate documentation.

2a) If the person(s) porting the program does not have significant X experience, then the XView toolkit and the "porting an application by hand" documentation provided by Sun should be used.

2b) If the person(s) porting the program do have significant X experience, then the toolkit most familiar to the author(s) should be used. The XView toolkit is not as strong as others due to its relationship to Sunview.

3) Keeping the same personnel involved with as many ports as possible will centralize the knowledge of potential porting problems. In general, porting an application is difficult only in spots, and having one person with knowledge of many such bottlenecks would be useful. Augmenting this with good record keeping of the porting process is also recommended. It should be noted that window system applications tend to have similar portions of user interfaces and code re-use is suggested whenever this is possible.

4.4 Appendix 4

Converting Sunview Applications To X

Some Notes for Programmers

Ian Smith
College of Computing
Georgia Institute of Technology

Introduction

This document will describe the problems involved in porting Sunview applications to X. It is assumed that the reader has a general knowledge of unix, X, and Sunview. It will cover specific problem areas in some detail, and will probably only be of interest to those actually doing ports. It assumes that reader has a system running unix (bsd 4.2 or greater), X11R4, and Sunview. This document should be considered as a guideline only, and your port should dictate its own individual needs.

How to write Sunview programs that are easy to port

It cannot be stressed enough how much good, original design will speed the porting process. The most import thing to keep in mind when writing a Sunview program that might be ported to X is separation. Separate the user interface from the functionality as much as possible. The following separation guidelines can be helpful when Sunview work is done.

1) Physically separate the user interface (front end) from the functionality of the program. This can be accomplished with separate files and/or directories. If you are careful to do this porting time can be reduced by large amounts.

2) Write the functionality (back end) as unix program that uses stdin and stdout then write the front end for it. You can usually accomplish this easily via the system call popen. If you require more sophisticated interfaces you can use the fork/exec/pipe sequence of system calls.

3) Document the user interface as thoroughly as the functionality. Good documentation, especially of global variables that proliferate in window system applications, can save a lot of time later on.

How to use X resources efficiently in a porting situation

In general, make all available use of the X resource database. During the process of porting the application, you should generally try to map all "attribute-value" pairs in the Sunview code to an X resource. When you converting the code, as always, try to avoid putting any constants in the X source. This accomplishes two things: First, it makes the program amenable to user customization. Second, it speeds up the porting process by avoiding the need to compile potentially large amounts of unmodified code.

In particular, pay special attention to the translation manager and its translation tables. Good use of these tables in the resource file can give you emulation of the Sunview user

interface bindings for next to no effort. It has the added benefit of letting the user customize his own environment more easily.

Nomenclature

All X library (Xlib) functions begin with an 'X' prefix. All X Toolkit functions functions begin with Xt. Toolkit functions operate at the 'widget level', ie. they operate on entire widgets rather than the lower level constructs such as lines, pixmaps and windows. In every case, however, one can use an Xt routine to retrieve the lower level constructs that underlie widgets. For example, to get the window of a widget, use XtWindowofObject.

Potential Problems

Menus

Menus are radically different in execution between Sunview and the assorted X toolkits. Many toolkits provide some form of 'pop up menu' widget, but its actions and operations vary widely. In the case of the Athena widgets, it is likely that you will want to use some thing like the Athena menu widget in conjunction with the menubutton widget. This arrangement is in general the easiest way to get menus to work, even if it means disguising the menubutton as something else (or making it invisible entirely).

tty subwindows

tty subwindows are a Sunview specific feature which does not exist in X. Any application that uses these will need to be reworked so that type of window is not needed. Generally, you can use the popen() command to give command-line-type interfaces to your X program. If worst comes to worst, you can use the fork/exec/pipe set of system calls to open a shell and read and write to it. Then use an X textwidget to do this display and input. This will require quite a bit of work.

If you want some degree of the shell functionality but don't want to support it specifically the following idea may be useful. Have a field in your interface for command line arguments and support cut buffers or selections so that users may "cut and paste" to and from XTerms.

Icons

The X icon mechanism is much more complicated than the sunview one. To make matters worse, this is also a window manager issue, so all programmer efforts to supply an icon for his application may be foiled by the window manager.

The basic steps in creating an X icon for your program are (using Xlib):

1) Create an icon using the X program 'bitmap.'

2) Use XGetIconSizes(...) to determine what the window manager prefers as the size of the icon.

3) Use XSetIconPixmap(...) to give the window manager the hint that you would like to use the pixmap you created. (Modifying the pixmap if necessary to the specification provided in step 2.)

4) Use XSetIconName(...) to hint to the window manager what you would like your icon's name to be.

It should be noted that if the window manager does not use the window property WM_HINTS for communicating with clients, this method is useless.

Cursors

Cursors are similar in X and Sunview. The only significant difference to be aware of is Sunview prefers you to define your cursor at the time a window is created and X does not. In X you use XDefineCursor to associate a cursor with a window.

There is a trade off associated with using the standard X cursors. If you use one of the predefined cursors (there are about 75 cursors distributed with X) in a manner similar to the way other applications use it, you will have an important clue for X users in how your application is operating. Conversely, if you create a custom cursor that is similar to your Sunview cursor you will have a smaller learning curve for users familiar with the Sunview version. These two sides must be weighed on an individual bases for every port.

Focus (popups and warnings)

The X focusing model is less restrictive than Sunview, in general. If your code uses the Sunview split focus model, be sure that you use the grab functions (XGrabPointer, XGrab-Keyboard,XGrabServer, and most important of all for widget programmers XtAddGrab) to control explicitly which window is getting user input.

In the most part you are going to want to force focus into a widget that has some type of urgent message or warning. If you use the athena dialog widget you can get the widget set to do this for you. Otherwise you will need to call XtAddGrab with appropriate parameters, thus forcing the user to deal with your warning message.

Constraint widgets

In Sunview there are basically two "constraint" widgets, the frame and the panel. The Athena analogues of those two widgets are the form and the paned widget. These are the two basic constraint widgets in the Xaw library and have similar functionality to their Sunview counterparts. (Note: Most other widget sets have similarly named widgets with the same functionality.) However, often X composite widgets can serve needs that are met be complex constructions under Sunview. Be sure to look carefully at the Viewport widget, which is often used to pan a window over a large virtual surface.

Do not feel limited by the constraint widgets of the Athena widgets. Many other widget sets have constraint widgets that will work successfully Athena widget children. It is often quicker to look for a constraint widget that implements the layout semantics you want than to use a widget that does not easily support your layout. The HP widgets are an especially good source of constraint widgets.

Look and feel

Look and feel can be difficult point when porting Sunview applications. In general, it most important to preserve the functionality first, then make a functioning program look and feel

as much as possible like the original. It will be nearly impossible to completely replicate the Sunview version (especially due to the problems caused by differing window managers), but these can help:

1) Give the user as many visual clues as possible to the functionality similarities in both versions.

If both the Sunview and X programs have a quit button, for example, they should both be labeled similarly and located in the same place relative to other objects in the interface. Also, touches like similar choices for fonts and shapes are fairly easy to implement, and give the user excellent visual clues.

2) Make good default choices the new version.

You do not want a situation where the user is confronted with both a new interface, and a new set of choices to make. This should be implemented in a way so that new additions or changes in the X interface should be hidden as much as possible by defaults. Ie, if you have to add a new field in the X version of the program to get similar functionality, give it a default value so that a user coming from the Sunview version does not have to understand immediately the differences in versions

4.5 Appendix 5

An Introduction to the X Window System

Ian Smith
College of Computing
Georgia Institute of Technology

This slide set was prepared for use in an introductory briefing on the MIT X Window system.

USAISEC

Introduction to X

Ian Smith

August 31, 1990

AIRMICS

USAISEC

History

- X was originally developed at M.I.T. to solve the problems of heterogenous networks of bitmapped workstations.

- The first public release of X was in 1986 with version 10R4.

- The current release of X is 11R4 with R5 expected out in 1991.

AIRMICS

USAISEC

X as a Standard

- X is the de facto standard of heterogenous networks.
- X is the standard chosen by the Open Software Foundation.
- X is the standard of research centers arounds the country.

AIRMICS

USAISEC

X and Open Systems

- X is an 'open system'. The X protocol is free and easily available.
- Anyone can modify X to suit their particular needs.
- The X sample servers are not 'supported' like a commercial product.

AIRMICS

USAISEC

Why You Might Want To Use X

- X applications generally are very user customizable; often even extensible by the end user.

- Because X is used at many research centers, there are often applications available for X that not available elsewhere.

- X applications are usually distributed in source form. This allows you to modify or learn from the source.

AIRMICS

USAISEC

Why You Might *Not* Want To Use X

- X is a large system, and has a substantial learning curve.
- X sample servers are not supported by M.I.T.
- Raw X expects user to understand significant amounts of information about the window system itself to reap the benifits of the system.

AIRMICS

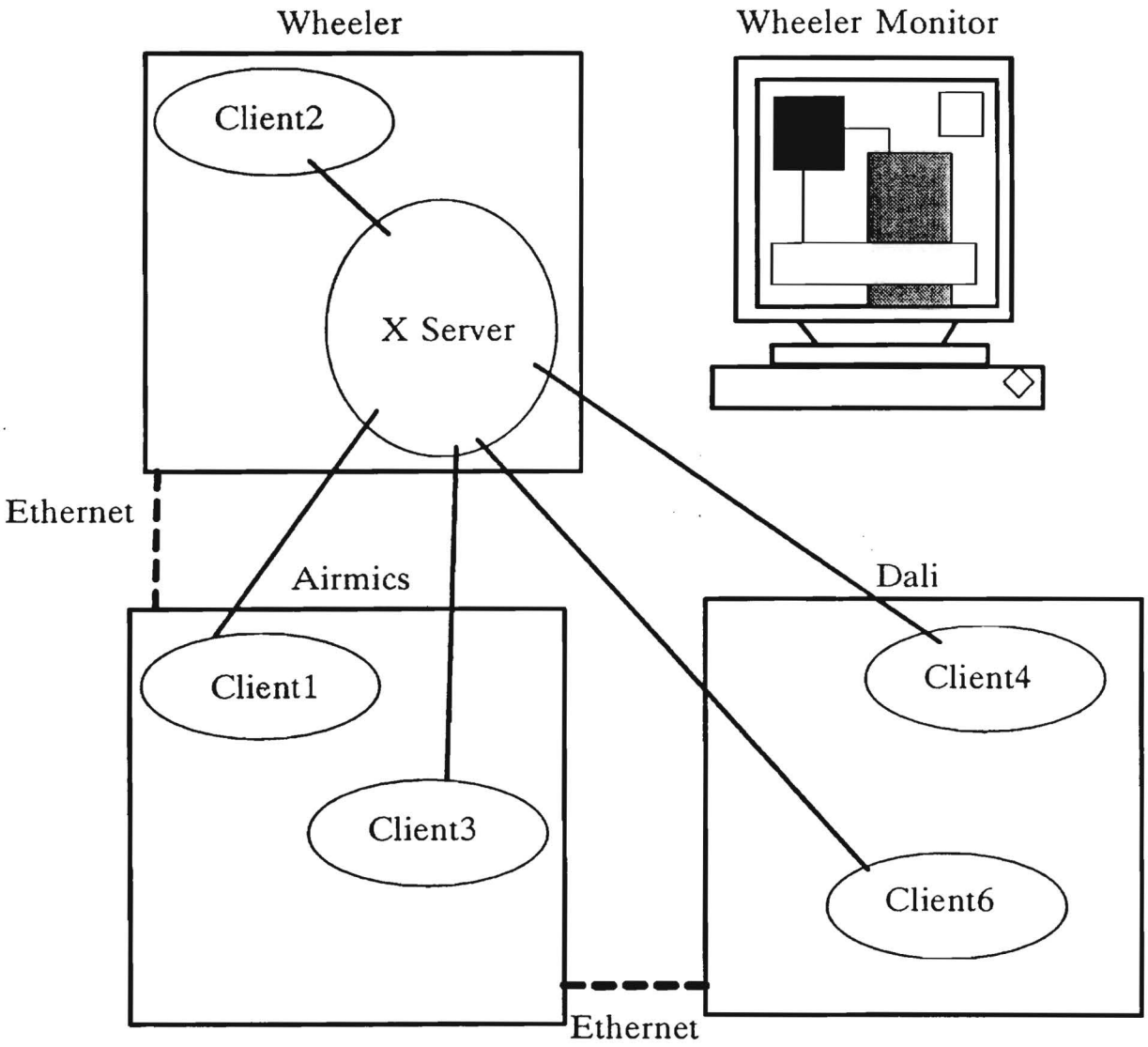
USAISEC

The X Model

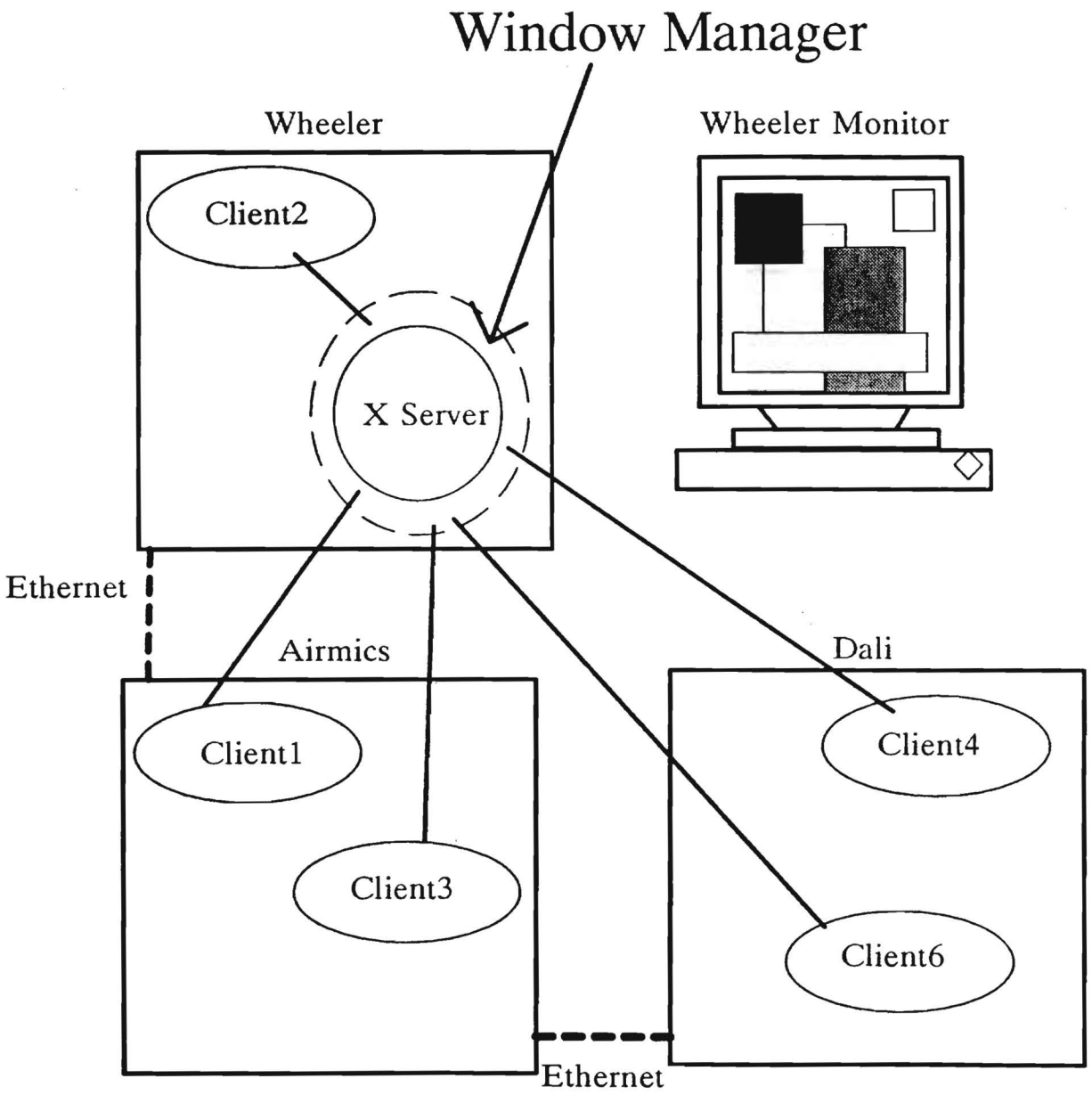
- X is based on a 'server–client' model. This model has one server per workstation with zero to many clients. These clients may be located on the same machine as the server or any machine in a network.

- X also expects one special client called the 'window manager' who is responsible for management and arbitration of all other clients.

AIRMICS



Conceptual X Model



X model with

Window Manager

USAISEC

Setting Up X On Your Workstation

- Many things in X are user customizable. To install a default configuration on your workstation use:

  % xsetup

- To start X on your workstation use:

  % startx

- To quit X on your workstation select 'quit' on the pull–down menu of the background.

AIRMICS

USAISEC

How To Run X Applications Locally

- In general, just typing an applications name should invoke it and start it running.

- All X applications live in one directory:

  /usr/bin/X11

AIRMICS

USAISEC

X Security

- The sample X server's granularity of security is coarse.

- A primitive form of security is provided by the program xhost. You use xhost to tell your server which machines may access your display. I.e.,

  % xhost +wheeler

AIRMICS

USAISEC

Running X Applications Remotely

- You must inform the client program on the remote machine where you want to display output. You can do this with:

  % setenv DISPLAY [workstation name]:0.0

- If you wish to explicitly send output to different displays use the -display [workstation name]:0.0 command line option.

- For example:

  % xmail -display wheeler:0.0

AIRMICS

USAISEC

If Something Goes Wrong...

- If the program will not run and says "Can't Open Display" check to make sure your DISPLAY environment is correct and that you have set your xhost's properly.

- If an application runs but seems to look strange (fonts are too small, colors are weird, etc.) then check to make sure that the programs "resources" are loaded. You load resources with the program 'xrdb.'

AIRMICS

USAISEC

Some X Applications

Name	Function
xbiff	Icon-oriented biff
xless	X front end for the 'less' file browser
xpostit	Simple 'Post-it' emulation for your workstation
xcalc	Puts a Calculator on display and allows cut-and-paste
xclock	Displays a clock on your screen
xman	Graphical man page browser
xmail	X front end for mail
xterm	Terminal emulator; the most used X application
ico	Draws objects onto a window and rotates them (demo)

AIRMICS

USAISEC

Where X Is Headed

- Mixed media 'displays' with audio and real–time video.
- More commercial support.
- As X gets more of a share of the end–user market, there more 'insulation' available to the user from the complexities of the system.

AIRMICS