JUL 11 1989
DATE ISSUED

ORNL
MASTER COPY

ORNL/ACP/LTR-89/4

Contract Program: Advanced Controls (ACTO) Program

Subject of Document: Advanced Control Test Operation (ACTO) Conceptual Design Report

Type of Document: Letter Report

Principal Authors: S. J. Ball and J. K. Munro, Jr.

Contributors: C. R. Brittain, C. E. Ford, R. A. Kisner, H. E. Knee, P. J. Otaduy, J. T. Robinson, L. A. Rovere, J. C. Schryver, J. D. White, T. L. Wilson, and R. T. Wood

Date of Document: June 1989

Sponsor: H. A. Alter (DOE/NE)

NOTICE This document contains information of a preliminary nature. It is subject to revision or correction and therefore does not represent a final report.

[redacted]

[redacted]

Prepared by
Instrumentation and Controls Division
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831-6010
operated by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under Contract No. DE-AC05-84OR21400

TABLE OF CONTENTS

1. INTRODUCTION AND SUMMARY 1

2. OBJECTIVES 3

3. ACTO FACILITY AND CAPABILITY DESCRIPTION 6

4. CONTROL SYSTEM DESIGN AND TESTING 11

4.1 CONTROL SYSTEM DESIGN LIFE CYCLE 11

4.2 CONTROL ENGINE 13

4.3 CONTROL DESIGN WORKSTATIONS 15

4.4 CONTROL SYSTEM ARCHITECTURES AND ADVANCED CONTROL STRATEGIES 18

4.5 TESTING AND VALIDATION BY SIMULATION 19

4.6 ARTIFICIAL INTELLIGENCE IN CONTROLLER DESIGN AND TEST PROCESS 21

4.7 FAULT-TOLERANT CONTROL DEVELOPMENT 21

4.8 SMART SENSORS AND SMART ACTUATORS 22

5. SOFTWARE DEVELOPMENT, INTEGRATION, AND SUPPORT 24

5.1 SPECIAL ACTO SOFTWARE FEATURES 24

5.1.1 Process Design Tools 26
5.1.2 Control Conceptual Design Tools 27
5.1.3 Support for System Life Cycle Document Preparation 28
5.1.4 Control Analysis and System Simulation 29
5.1.5 Implementation Tools for Control Design 31
5.1.6 Subsystem Test Environment 32

5.2 SUPPORT AVAILABLE FOR SOFTWARE DEVELOPMENT 32

5.2.1 Software Development Environment 32
5.2.2 Layered Structures 37
5.2.3 Command Line Syntax for Heterogeneous Hosts 37
5.2.4 Flexibility and Universality 38
5.2.5 Computer Aided Engineering Tools 39
5.2.6 Adapting to Rapid Technological Change 39

5.3 SOFTWARE ENGINEERING METHODS AND QUALITY ASSURANCE 40

5.4 SOFTWARE LIFE CYCLE 41

5.4.1 Technical Management Guidelines 41
5.4.2 User Requirements and Prototypes 42
5.4.3 System Specifications and Functional Description 43
5.4.4 Detailed Design 43
5.4.5 Coding and Unit Testing 44
5.4.6 Integration, Test, and Verification 45
5.4.7 User Support and Maintenance 45

5.5 DATA BASE MANAGEMENT 45

5.6 COMMENTS ON THE USE OF ADA 47

5.7 GRAPHICS 49

5.8 PLANNING FOR NEW SOFTWARE TECHNOLOGIES 50

6. HARDWARE ISSUES 52

6.1 CONTROLS HARDWARE DESIGN 52

6.1.1 Prototype Assembly and Interfacing 52
6.1.2 Guidelines for Specification of Control System Hardware 53
6.1.3 Guidelines for Design of Control System Hardware 53

6.2 COMPUTER SELECTION AND APPLICATION 53

6.2.1 Workstations and Personal Computers 54
6.2.2 Parallel Processors 56
6.2.3 Supermini and Other Mainframe Computers 57

7. DEMONSTRATION PROJECTS 58

7.1 FEEDWATER CONTROL WITH EXPERT ADVISOR 58

7.2 METAL FUEL INHERENT FEEDBACK IN CONTROL 60

7.3 SUPERVISORY CONTROL OF A MULTIMODULAR LMR 60

7.4 AUTOMATED STARTUP 61

8. COMPUTER NETWORKING AND INTERFACING CONSIDERATIONS 63

9. ARTIFICIAL INTELLIGENCE IN NUCLEAR PLANT CONTROLS . . . 66

10. DESIGN AND TEST OF HUMAN-MACHINE INTERFACES IN ADVANCED CONTROL SYSTEMS 69

10.1 INTEROPS MODEL 70

10.2 PHASED APPROACH 70

10.2.1 Preliminary Design Phase 70
10.2.2 Design Phase 71
10.2.3 Test and Evaluation Phase 71

11. PLANT MAINTENANCE TECHNOLOGIES 72

12. CONCLUSIONS 73

REFERENCES 75

LIST OF FIGURES

Figure		Page
3.1	Diagram of Network Architecture for the Advanced Controls Computer Laboratory	7
3.2	ACTO Design and Test Environment	10
4.1	Control Design Life Cycle	12
4.2	Elements of Advanced Control Engine	14
4.3	Block Diagram of Workstation	19
5.1	Diagram of Control System Design, Test, and Evaluation Environment	26
5.2	Representation of Software Development Environment for ACTO	38
6.1	Future Systems and Network Architecture	55

TRADEMARKED PROGRAMS

ADA:	registered trademark of the U.S. Government, ADA Joint Program Office
$MATRIX_x$:	registered trademark of Integrated Systems, Inc.
ProMat-Lab:	registered trademark of Mathworks
UNIX:	registered trademark of AT&T
VMS:	registered trademark of the Digital Equipment Corporation

1. INTRODUCTION AND SUMMARY

Advanced digital control systems featuring innovative hardware and software system designs can provide a high degree of automation and result in considerable benefit to the reactor industry. The chief benefits of such automation would be enhanced reliability and simplicity of operation, which would reduce operational errors and cause fewer unnecessary shutdowns. These benefits, in turn, would lead to fewer challenges to plant safety systems and reduce plant operation costs. An operational upset in today's large power plants can cost up to $1 million/d of down time, a cost that is passed to the public through higher rates for electricity. Capacity factors for plants in the United States have been significantly lower than those for foreign plants, such as in France, Japan, and Canada, at which the level of automation is greater. This comparison implies that more automation leads to more efficient operation.

Conventional (analog-based) systems are becoming increasingly less attractive and less feasible because of obsolescence and availability considerations. Moreover, the potential advantages of automated plant digital control systems outweigh analog-based systems. To realize the benefits, and to avoid pitfalls in implementing the new technologies in a complex and highly regulated environment, development work is needed in several areas. Automated control systems can provide for systematic achievement of control goals and strategies. Through the use of intelligent processors, controllers can assess current and future plant status, perform diagnostic evaluation and maintenance planning, and do signal and command validation. Until now, it has not been feasible to employ all these capabilities in conventional hard-wired, analog, centralized control systems. Recent advances in computer-based digital data acquisition systems, process controllers, fiber optic signal transmission, artificial intelligence tools and methods, and inexpensive, fast, large-capacity small computers--with both numeric and symbolic capabilities--have provided many of the necessary ingredients for developing large, practical, automated control systems.

Although some U.S. nuclear power plants currently have automated systems, a fully integrated digital control system for a U.S. nuclear power plant has not yet been designed. The task of managing the interactions among the more than 100 plant subsystems is left to the operators. Even in plants where a form of diagnostics is used to provide early warning of major changes in conditions, prompt operator interaction is required to re-establish satisfactory operating conditions.

Other U.S. industries--steel, automotive, aviation, chemical, electronics, defense, and food processing--have improved their competitiveness through automation. The U.S. nuclear industry also must employ automation in construction, control system design, plant operation, and maintenance to compete with domestic alternative power sources and foreign nuclear plant designs.

In 1985, the U.S. Department of Energy (DOE) established a task team to determine the need for, assess the feasibility of, and recommend an approach to the introduction of automation and advanced control into the nuclear power industry. The task team report, published in September 1985,[1] recommended an advanced controls program with

a control design, test, and qualification capability for current and advanced nuclear power plant designs, as well as for space-based nuclear power system designs. Quoting from the report: "The structure of the industry, the market, and competitive realities do not make it financially viable for any commercial organization to provide this capability or to combine with others to do so."

Because of the Oak Ridge National Laboratory's (ORNL's) unique strengths in control system design and analysis, DOE has provided support for ORNL to pursue research to support advanced, automated control of the proposed liquid metal reactors (LMRs). The purpose of this effort is to conduct research and to provide technical leadership in the design, test, and validation of advanced, automated control systems that would provide improved operability, reliability, and safety for advanced LMRs.

The Advanced Controls Program (ACP) effort is currently directed towards R&D and design support for the advanced LMR (ALMR). The facility and capabilities developed under the ACP are referred to as the Advanced Control Test Operation (ACTO) project. It is expected that eventually ACTO will support other current and advanced LMRs, the current and advanced light water reactor (LWR), and the high-temperature gas-cooled reactor (HTGR) designs. Other prospective applications are for research reactors [such as the proposed high-performance, high-flux research reactor (Advanced Neutron Source, or ANS)] at ORNL, military reactors such as the new production reactor (NPR) designs, and space-based nuclear power system designs.

The purpose of this report is to describe the capabilities and facilities required for ACTO to provide support to the reactor community's task of implementing advanced digital control and automation. The design, implementation, and test of such advanced automated systems for reactors requires a large, well planned and well integrated effort. Issues related to controller design, software and hardware utilization, simulation, networking, artificial intelligence (AI) and expert systems, and human-machine interfaces will be discussed. The current ACTO facility and projections for facility needs in the next 5 years will also be described.

2. OBJECTIVES

The major objective of the ACTO project is to help maximize the benefits that nuclear plants derive from advanced control technology, particularly in the areas of availability and safety. Potential benefits from the effective use of high technology control hardware and software and the use of control systems that are more intelligent, reliable, and fault-tolerant are:

1. reductions in the number and duration of outages,
2. reductions in the likelihood of "incidents" becoming "accidents," and
3. improvements in operating performance.

Operational and maintenance errors (of both commission and omission) account for a significant amount of forced outage time in operating LWRs. Operator errors were responsible for conditions that led to major accidents at TMI-2 and Chernobyl. The widespread use of highly automated systems in some foreign countries has demonstrated that advanced equipment and software techniques reduce operational errors (i.e., both operator and equipment problems) and improve plant availability. Overall safety can be enhanced significantly as well.

It is essential for the United States to increase its competitiveness in the world's reactor control field. Computer control systems are used for direct digital control of major systems in 16 full-size commercial Canadian nuclear units. Performance has been excellent, achieving over 84% availability, due in part to automation.[2] Although the U.S. record has improved considerably in the past few years (66.8% capacity factor for 1986-1988 vs 60.4% in 1983-1985 and 57.8% in 1980-1982),[3] billions of dollars could be saved (including savings on imported power from Canada) if further substantial improvements were made.

Furthermore, planning studies indicate that the use of a distributed control approach to replace most of the control signal wiring and relay logic results in a large reduction of construction work to be done on site. The planned construction schedule can be shortened by 6 to 10 months for new designs. The Japanese have apparently taken the lead in the design of high-level control systems. They have completed the design of a plant-wide, fault-tolerant, distributed computer control system and validated it on a simulator. The design will be implemented in ten new Japanese plants scheduled to become operational in the early 1990s. To remain competitive in the worldwide market for advanced nuclear units, the United States must develop such advanced, automated systems as well.

The driving force behind the Advanced Controls Program and the Advanced Control Test Operation is the idea that effective support is needed for promoting, developing, and testing advanced control systems for reactors. Such advanced systems cannot be incorporated into reactors, however, without extensive prior demonstration, test, qualification, and verification. It is our thesis that this capability could best be provided by a multiuser, multipurpose simulation and test facility such as ACTO.

The ACTO concept requires the development of advanced methods for integrated control design and testing. To be effective, these attributes must be incorporated into an integrated system architecture capable of supporting multiple users in the design, test, and qualification of advanced control systems.

To respond to these needs in the long term, the overall goals that have been set for ACTO are to:

1. Provide the capability to evaluate and design various levels of plant control automation, operator task analysis and procedure generation, expert advisor systems, human-machine interfaces (HMIs) and safety-related systems.

2. Provide the capability to perform timely test and qualification of advanced control systems.

3. Provide a vehicle for efficient development and demonstration of machine-assisted startup and other operating procedures, including cases in which the allocation of tasks between the operator and the automatic control system may vary.

4. Increase the effectiveness of nuclear power plant designers to quickly evaluate alternative control strategies, designs, configurations, and new technology options.

5. Support multiple concurrent activities involving various modes of design, test, and qualification.

6. Provide an effective test bed environment for quality assurance (QA) evaluation and environmental qualification of advanced control system hardware and software.

7. Provide an efficient interactive user interface and environmental support for management of alternative process models and interconnections, alternative process parameter data bases, and scenario analyses.

8. Provide an efficient multiuser interface that supports alternative plant configuration models from different concepts (e.g., LMR, LWR, HTGR) and various organizations (e.g., ORNL, utilities, vendors, universities).

9. Provide the information and data required for regulatory evaluation to facilitate the use of advanced control and safety system technology in nuclear power plants.

10. Provide a capability for use by plant designers/users to evaluate the response of control systems and plant components to off-normal and accident situations, and to test the functionality of the HMI.

11. Provide the capability to prove advanced control system and control room designs before committing them to hardware by performing system response checks with appropriate human environments.

Capabilities provided by the ACTO facility for LMR control system development will be applicable to other nuclear reactor programs. Potential users of the ACTO facility are designers and researchers for DOE (and DOE laboratories and contractors), utilities, universities, and nuclear reactor power plant vendors. These users are developing control systems for the Experimental Breeder Reactor-II (EBR-II), the Fast Flux Test Facility (FFTF), the modular high temperature gas-cooled reactor (MHTGR), the Advanced Neutron Source (ANS), the Advanced Light Water Reactor (ALWR), the new production reactor (NPR), commercial light water reactors, breeder reactors, and foreign reactors.

3. ACTO FACILITY AND CAPABILITY DESCRIPTION

The ACTO concept is based on the premise that advanced control system designs require an environment supportive of an integrated approach to the entire life cycle of a control system. ACTO will eventually consist of the facility, software, and capabilities required to support a control system's complete life cycle. This life cycle extends from the initial conception to final testing and installation at the eventual plant site, followed by maintenance and upgrade support.

Development of the capabilities and facility is geared to a long-range goal of providing a centralized, user-friendly design environment. This environment will be available for control system designers within the DOE community and, later, for any qualified user. The environment will consist of four parts:

1. networked, intelligent, computer workstations into which have been integrated software tools, graphics capabilities, on-line design guidance, on-line documentation and interfaces to the large-plant simulation capability at ORNL;

2. plant/component models and data bases useful for control system design and plant simulation;

3. information resources concerning control system strategies for automated control; and

4. HMI models and guidelines for designing control system interfaces with operators.

There will be a central physical location for this environment at ORNL with electronic linkage to participating universities and institutions.

At its present stage of development, ACTO's "facility" consists of two laboratories in the ORNL Instrumentation and Controls (I&C) Division equipped with modern workstations and a parallel processor "main frame." These computers are interconnected, with capabilities of accessing "the outside world." Figure 3.1 shows the complement of machines and the computer networking scheme currently in use. There are now eight workstations with a file server; four more workstations are expected to be delivered by the end of FY 1989.

ACTO software is now a mixture of commercial packages and in-house developed programs in all the areas of interest. These include several generic and special-purpose dynamic simulation packages, workstation software for simulation and control system development, and control system design packages for classical and modern controllers. Other software includes an assortment of graphics, expert system shell, data base management, and advanced computer language packages. A summary of the software now in use is given in Table 3.1.

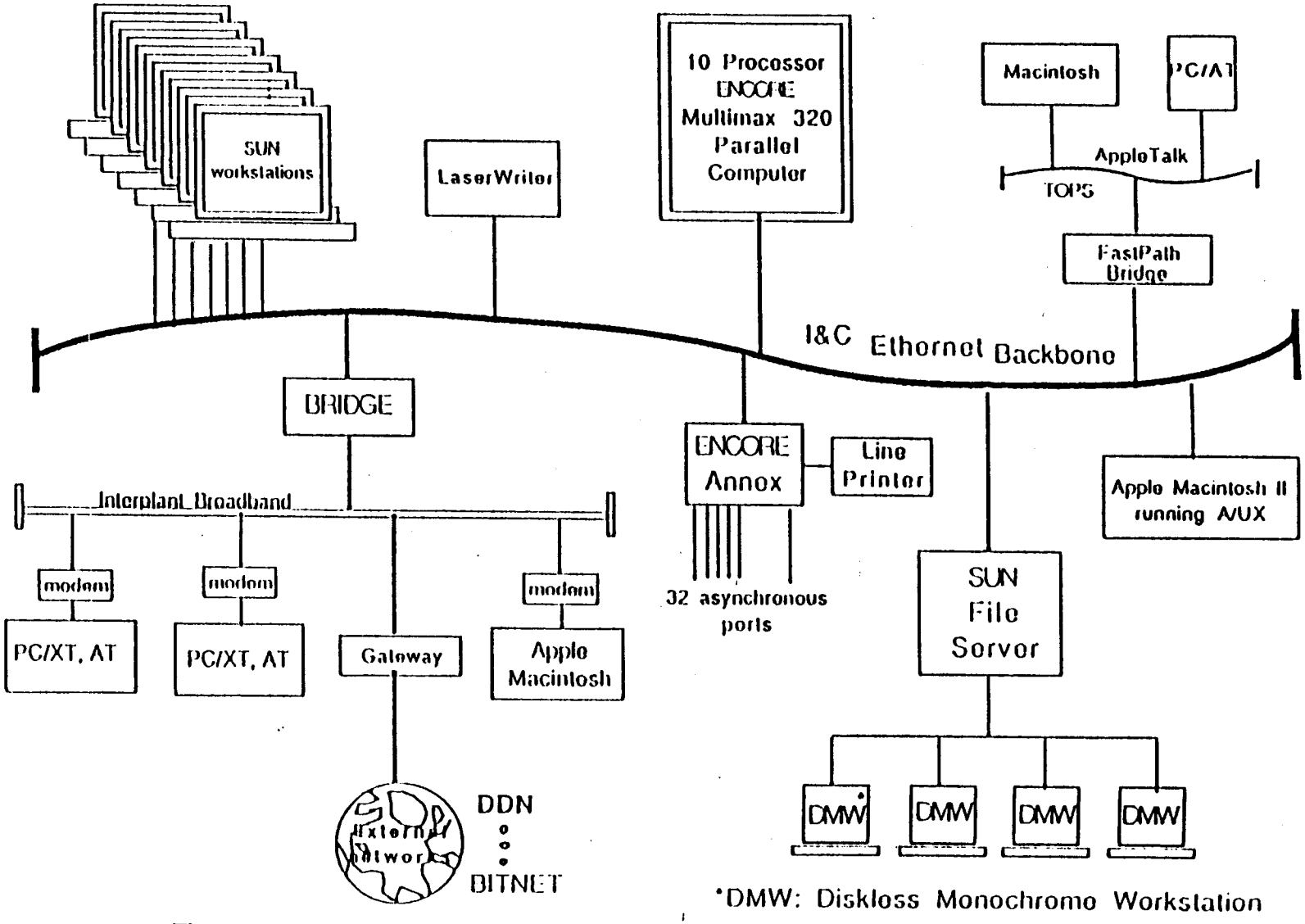

Fig. 3.1. Diagram of network architecture for the advanced controls computer laboratory.

Table 3.1. Current ACTO software

Simulation:

1. ACSL (Advanced Continuous Simulation Language) [c]*
2. MMS (Modular Modeling System) [c]
3. MMS-LMR (LMR modules) [i]*
4. Workstation object-oriented reactor simulator [i]
5. LMR multimodular simulator [i]
6. ARIES-P [General Electric]
7. SASSYS [ANL]

Control Design:

1. $MATRIX_x$ [c]
2. ProMat-Lab [c]
3. Control Laboratory [University of New Mexico]

Languages [c]:

FORTRAN, C, Ada, Basic, SmallTalk, LISP, OPS5, OPS83, Prolog, Mathematica,...

Expert system shells:

1. KES [c]
2. Oracle [c]
3. NExpert [c]

Data base management systems:

1. Unify [c]
2. DBase-III+ [c]

Special workstation software:

1. Data Views [c]

Parallel processor software:

1. Encore Parallel FORTRAN [c]
2. Encore Parallel Threads [c]

*[c] = commercial; [i] = in house.

ACTO's capabilities have been developed in part by directing the use and integration of the facilities and the software to several demonstration projects that directly support ALMR program needs. The demonstration projects are described in Sect. 7.

Future plans for additional hardware and software acquisitions and software development depend on needs and supply, such as how best to factor in the rapid improvements being made in commercially available hardware and software. A best-estimate extrapolation of the elements of the ACTO facility 5 years hence are shown in Table 3.2. Discussions of future plans for hardware, testing, and environmental qualification needs are given in Sects. 4.5, 5.1.6, and 6.2. Additional space needs are anticipated for library materials (references, reports, software documentation, and backup tapes) and conferences.

Potential expansion in HMI technology is in the areas of human interface design and test facilities. Expansion eventually would include capabilities for testing human subjects on a complete range of simulations, ranging from laboratory experiments with simple mockups or graphic representations to full-scale integrated human-system tests with a complete control complex in the final stages of design. Various support facilities with equipment for measurement of human performance capabilities would also be needed. A diagram of proposed future interrelationships of HMI development with other efforts in the design and testing environment is shown in Fig. 3.2.

Capability and technology issues for reactor control and automation life cycle design, testing, and maintenance needs are described in the following sections of this report.

Table 3.2. Elements of the future ACTO facility (through 1994)

A. Hardware and facilities (additional)

1. Super workstations
2. Data base machine
3. Supermini computer/parallel processor expansion
4. HMI/hardware integration test facilities
5. Environmental test laboratory

B. Software

1. Integrated control design/simulation workstation package
2. Generic modular subsystem and plant simulations
3. Advanced DBMS, graphics, networking, desktop publishing
4. Natural language processing software to assist with requirements tracking
5. Configuration control and other project management software

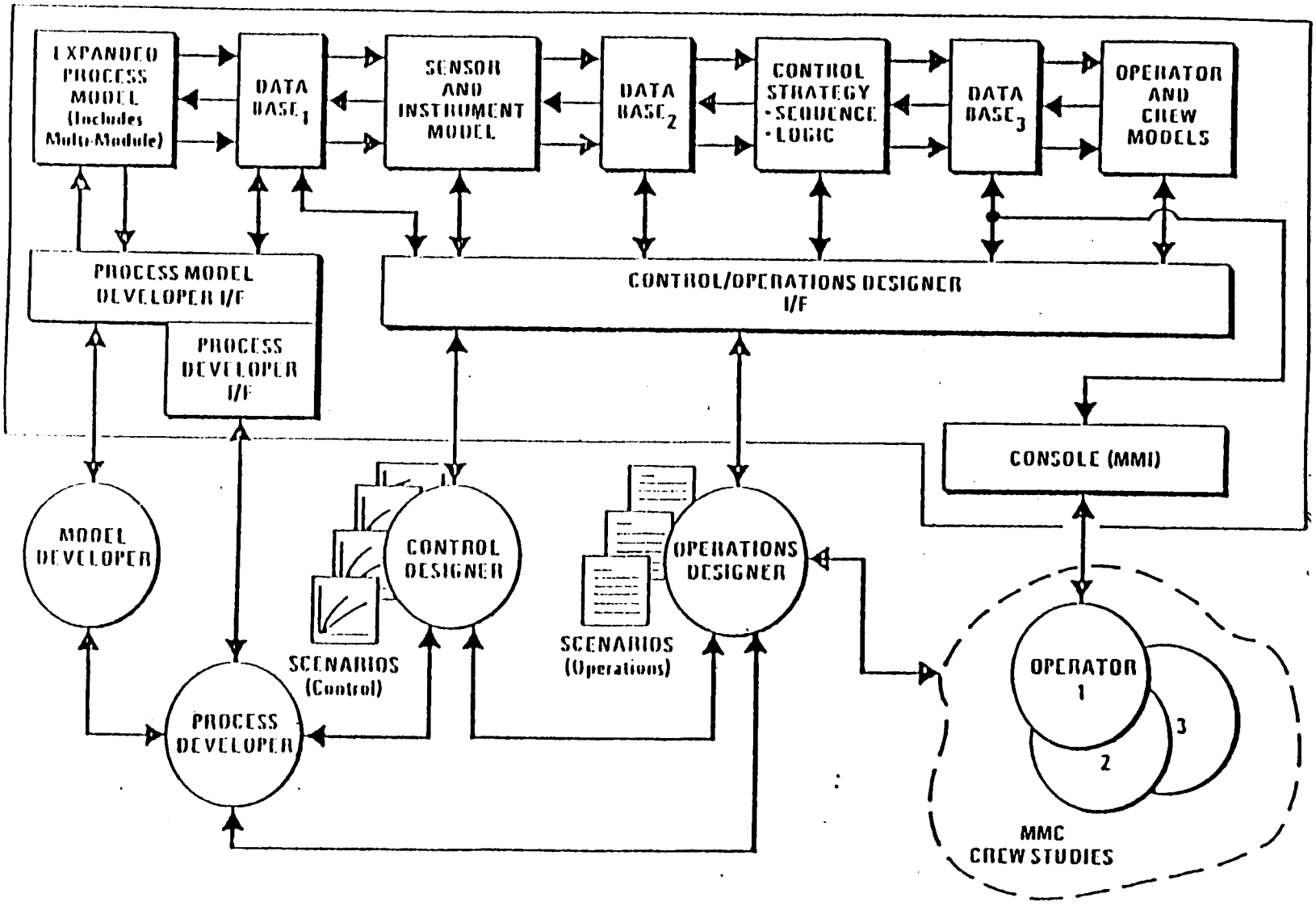

Fig. 3.2. ACTO design and test environment.

4. CONTROL SYSTEM DESIGN AND TESTING

A unique and necessary feature of ACTO will be the comprehensive and integrated nature of its development facilities, wherein the entire control system design life cycle is supported. Incorporated into ACTO will be supporting facilities for conceiving, specifying, designing, simulating, and testing control systems. These capabilities will be implemented by providing real-time interfaces to connect prototypes of control systems to computer simulations of the plant, control room mockups for testing the human-machine interface and for measuring human performance, and (eventually) environmental test facilities.

This section describes the concepts and proposed implementation of ACTO objectives with respect to control system design. Control system design concepts incorporated into the design environment include the control system design life cycle, the "control engine" concept, control system architectures and advanced control strategies, testing and validation by simulation, and applications of artificial intelligence. Two special attributes of the advanced systems, fault-tolerant designs and smart sensors and actuators, will also be discussed.

4.1 CONTROL SYSTEM DESIGN LIFE CYCLE

The control system design life cycle shown in Fig. 4.1 provides a framework for ACTO developers in which to tailor software tools to specific steps in the process. The life cycle defines inputs, outputs, and processing required for each step. It also defines interfaces for the rest of the design procedure, including prototype design and testing. These definitions—input, output, processing, and interfacing—are the essential inputs to the ACTO software tools.

One objective of ACTO is to enable the development of a top-down or step-by-step design method for controls engineering to encourage thoroughness and efficiency. The life cycle approach is a procedure that breaks the design process into tasks, each with definite objectives. Within the life cycle, ACTO can provide software tools to aid in each task and then to automatically pass results of a task to the next step. The life cycle provides milestones in the development to aid project management.

Note that the process (Fig. 4.1) is not strictly "top-down." Some feedback and iteration, represented by the back-arrows, will be necessary. Backtracking will clearly be needed to support redesigns and upgrades, and for these functions it will be essential to keep the process simulations intact (and upgraded) throughout the life cycle. The ability to readily correct errors is one of the major incentives to develop ACTO. Any experience with a word processor or spreadsheet program demonstrates that, although some productivity is gained through the initial text or problem entry, the greatest productivity gains occur when editing or revisions are required. The same is expected of ACTO. Design changes and functional specification changes will be propagated through the remainder of the design process in a well-controlled manner.

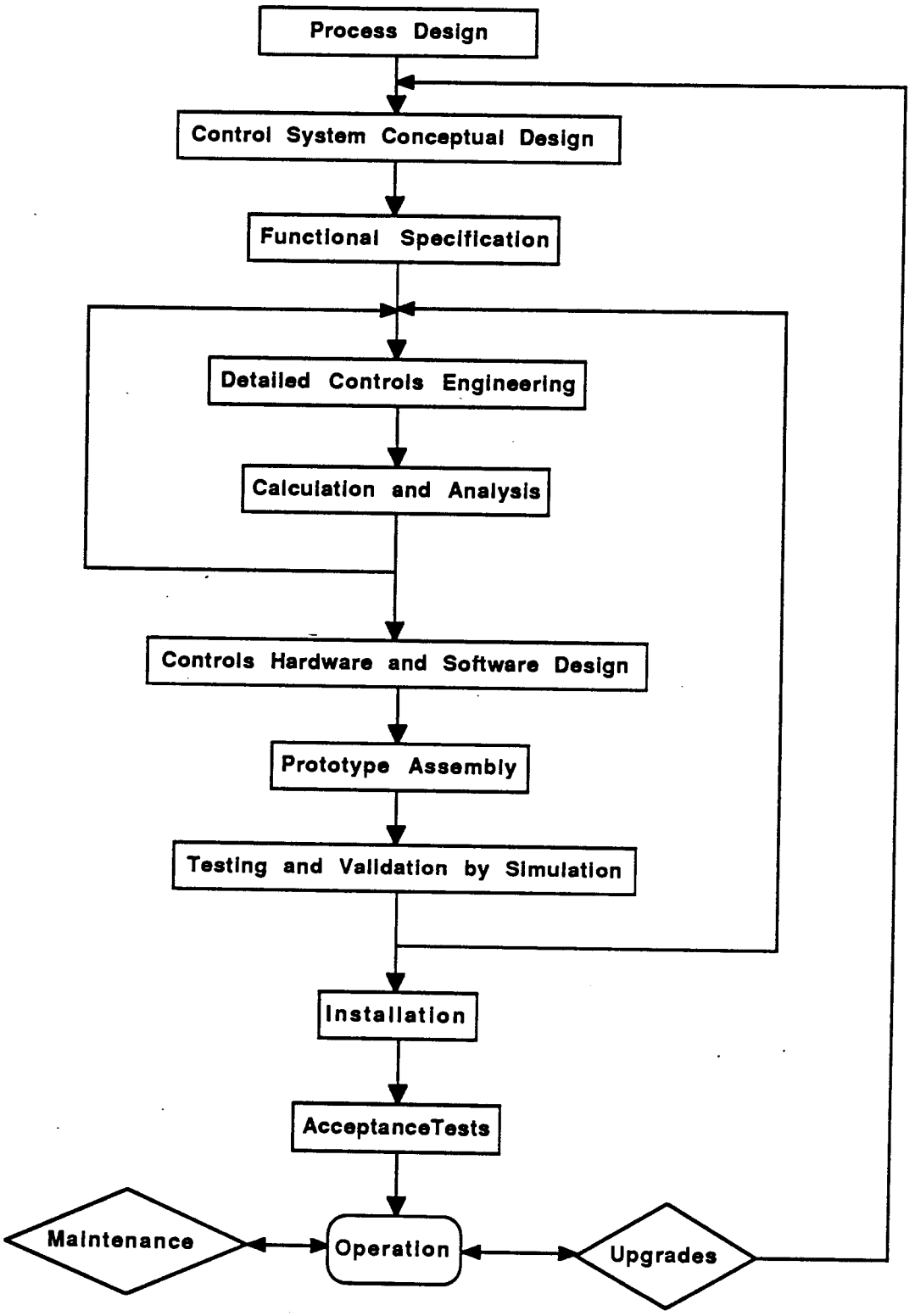

Fig. 4.1. Control design life cycle.

4.2 CONTROL ENGINE

The control engine concept is an idea developed and advocated by A. D. Alley of the General Electric Company.[4] As applied to reactor control problems, the control engine is a useful means for designing and integrating the essential elements of a comprehensive automated control system. It also presents a logical scheme for dividing (and later integrating) the tasks and the technology involved in the overall design problem.

The major elements of the control engine are shown in Fig. 4.2. The conventional part of the control system is represented by the automatic control systems block. While the control system would be all digital (vs the conventional, mostly analog, systems), many of the control algorithms in a typical control system would be similar. A significant difference would be in the ability of the automated digital control system to account for changes in control modes (such as startup, low power, and full power ranges), integrate the control of multiple reactor modules tied to a common steam system, and make allowances for various subsystems to be out of service. On-line diagnostic systems that automatically account for failures of sensors and instrumentation subsystems, known as "smart sensors," would also be included. These take advantage of the redundancy (sometimes physical, sometimes analytical) that exists in the instrumentation.

The data processors and archives block represents an automated scheme for recording, retrieving, and analyzing information that would bear on maintenance scheduling, subsystem and system fault prediction (based on operating experiences), and any other data storage chores required by system designers, QA data base managers, or regulators. The design objectives of this system deal with the efficient storing of very large data bases that must be readily accessible and have versatile display and analysis features.

The plant state analysis and decision aids block incorporates on-line real-time (and faster than real time) plant simulators that are used to provide advanced operational and diagnostic decision aids to the operators. The real-time simulation would provide information about the *expected* plant status, based on operational history and current data, assuming that there are no system malfunctions or instrumentation errors. This computed state of the plant is compared (continuously) with the measured state, and differences are computed, flagged (if significant), analyzed for possible causes, and the digested information is presented to the operator. Faster-than-real-time simulators would be used to predict the outcomes of alternative strategies to overcome operational difficulties. For example, in an emergency shutdown situation in which several alternative shutdown cooling mechanisms could be used, the simulator (which is tied in to the plant instrumentation system) could help the operators predict the outcomes of several emergency strategies, depending on the resources available, such as cooling water inventories or pumping capabilities. Other operator aid and diagnostic packages would be included here, such as alarm filtering, automatic procedures generation, and expert advisor (AI) systems.

The plant configuration and maintenance planning block refers to a sophisticated scheme to facilitate predictive (rather than simply scheduled) preventive maintenance, and includes the generation of current data base information on all significant plant

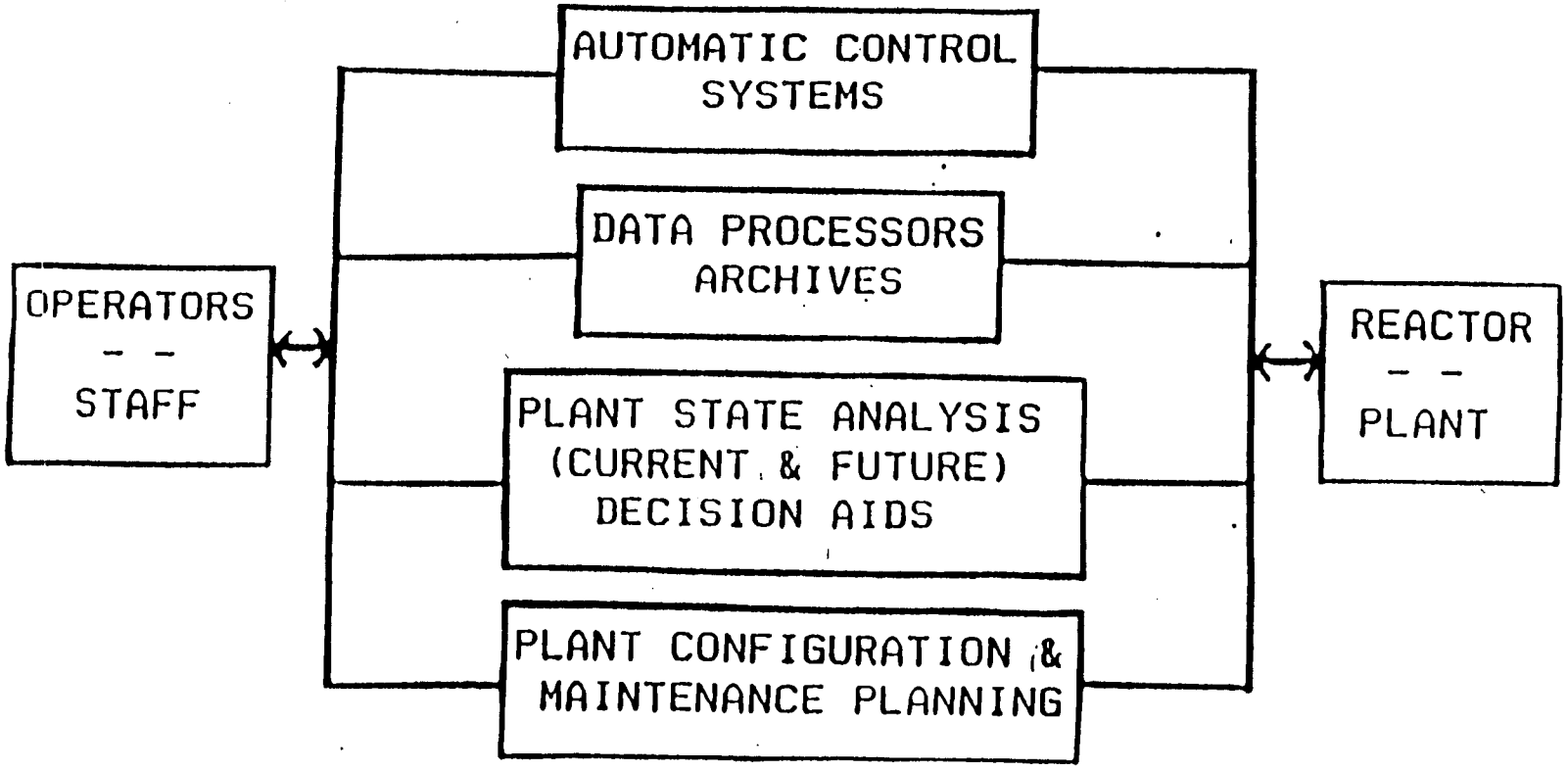

Fig. 4.2. Elements of advanced control engine.

components whose failure could affect continued operation or plant safety. The maintenance management scheme would also inform the plant maintenance supervisor of current equipment status and of the safety or operational implications of taking any piece of equipment out of service for maintenance.

Other features of the control engine concept include automatically updating the data base for the plant training simulator (based on knowledge gained from operation and maintenance/design change data) and automatic status report generation for utility and regulatory use.

4.3 CONTROL DESIGN WORKSTATIONS

The development of an intelligent workstation[5] for control system simulation and design is being undertaken to demonstrate and test some of the concepts proposed for the integrated controller design environment. The concept of an intelligent workstation is one of a desktop computer and software package that provides a control system designer with full capabilities, from design through simulation and test to control system code generation. The hardware consists of a workstation-class computer. Software consists of computer programs to organize the specification of requirements, to perform complex mathematical and logical simulations of the control design, and to illustrate the system through graphical and text manipulation software.

The need for safe, reliable, and efficient reactor operation, as well as for increased component lifetime, efficient maintenance, and improved human-machine interaction, encourages the development of intelligent control systems. By necessity, this requires high-level decision-making capabilities and coordination to integrate all facets of plant operation and maintenance. The engineering of such intelligent control systems requires a sophisticated tool set and design environment.

The controls analysis workstation is used to automate, document, and test all aspects of the design, analysis, and specification. Its advantages are:

1. productivity enhancement through improved tools and design environment,
2. error reduction in all phases of the life cycle,
3. project management improvements due to the systematic and comprehensive integrated approach,
4. automatic record keeping and report generation for the design process,
5. standardization of controls analysis methods, and
6. communications between design teams via shared resources and the linking of steps in the control system design life cycle.

All phases of control system development must be supported by the workstation. These phases include:

1. Model creation:

   a. modeling of the plant, subsystems, and components;
   b. verification of models.

2. System analysis:

   a. identification and characterization of the modes of operation for components and subsystems,
   b. analysis of the control techniques and strategy required for each operating mode,
   c. analysis of the human-machine interface.

3. Control system functional design:

   a. development of controllers and coordinators to realize the techniques and strategies,
   b. development of basic discrete-event controllers and a supervisor to control and coordinate mode switching under normal operating conditions;
   c. expansion of discrete-event and continuous controllers to accommodate higher levels of decision making and control, including degraded equipment conditions, abnormal external factors, and automatic test and calibration.

4. Control system realization:

   a. design of distributed architecture,
   b. specification of control computers and components,
   c. generation of production code for target processors.

5. Real-time interactive simulation:

   Testing and validating plant performance with actual control system equipment.

These phases of control system development will require the control workstation to support all ACTO program analysis and design activities. Three aspects of control system design are considered in this project:

1. design methods for control systems,
2. computer programs to increase productivity of designers and analysts, and
3. computer hardware to support the design and analysis software.

Improved design methods are needed to reduce engineering effort and simultaneously decrease errors. Such methods should be usable with software-based development tools, data bases, and computer workstations. In addition to simplifying the effort to specify the control system functional requirements, the methods must also document the

specifications so that computer programs can be written to implement the specifications on real-time process computers. Computer coding tasks are often performed by a software team separate from the control engineering team. The passing of the control system design between these two teams is a potential source of error and inefficiency. The design method should document the specifications in a form and vocabulary understandable by both development teams. Several concepts emerging for a useful design method are discussed later in this report.

Software packages are needed to perform mathematical and logical analyses, simulation, data base management, report generation, graphical representation, and interteam communications. While commercial software packages are available that fulfill some of these needs, they are not integrated (i.e., one company's package may not be able to call on another for data, to invoke a function, or to effect coordination towards a common goal). Computer hardware and peripheral equipment are needed to execute the software tools indicated above. The hardware must run at sufficient throughput so as not to hinder the design effort, and it must meet specific compatibility requirements.

The control design workstation software will include a graphically based software package that provides a means of assembling models of the power plant and its subsystems. The resultant model appears as a schematic of the plant. Software for automatic model generation formulates the mathematical models of the plant using the plant schematic diagram. Some customizing may be required by the designer to arrive at a final model.

The workstation environment will advise the user on the selection of appropriate control techniques and strategies, on the operation of particular plant components, and on the use of the control design workstation itself.

The designer can interact with the plant model and control system in either an on-line or an off-line mode, depending on the need. On-line interaction is when the plant model is operating (perhaps in real time) and the designer can experiment to gain a feel for the behavior of the plant system. Off-line interaction provides for batch runs. For example, parametric runs can be performed to develop a family of performance curves for a particular subsystem.

The control design workstation will be developed as an integrated group of modules. Figure 4.3 illustrates the workstation concept, including an interface to the Modular Modeling System (MMS) for using currently available plant models that were developed using Advanced Continuous Simulation Language (ACSL) and MMS. The modules of the control design workstation are:

Control System Analysis Package. This software package provides analytical tools for the development of modern (state-space) linear and nonlinear control systems. The package will have many self-contained general and specific equation solvers. In addition, it will make reference to other commercial packages for continuous and discrete-control tools.

Control System Data Base. This software package provides a centralized repository for all information relevant to those control system methods supported by the analysis

package. Categories include (1) control system design data from vendors, (2) logbooks for users of the workstation, (3) input/output files of analysis software, (4) help file and documentation for analysis software, and (5) a current bibliography related to control system methods supported by the analysis package.

Interactive Simulation Package. This software package provides a user-friendly modeling environment for testing control systems at various stages during the development cycle. Some of its features will be that it is:

1. *Interactive.* The package will allow for instant feedback on changes made to the system (either model or control). This feature encourages experimentation and shortens development time. It also provides an excellent means of demonstrating control concepts to others outside the controls development portion of the project.

2. *Hierarchical.* The package will allow detailed and course-grained models of the plant to be mixed and interchanged as required to obtain desired simulation speed and performance.

3. *Graphical.* The package will provide a graphically based display of the plant, control system, operating conditions and variables, and other parameters and features as required. The premise is that most engineers and analysts are visually oriented. The display is to be hierarchically based so that as a user needs to see more information, he may proceed into deeper layers of the system.

Plant Model Data Base. This software package will provide a centralized repository for all information relevant to plant modeling work. For example, specific design information (e.g., PRISM, SAFR, and EBR-II models) may be made available. Later, non-LMR models will be included. Also, output from simulation runs can be filed for later analysis.

4.4 CONTROL SYSTEM ARCHITECTURES AND ADVANCED CONTROL STRATEGIES

Multimodular ALMR plants require the development and demonstration of an advanced supervisory control technology. At the top level of control will be a supervisory controller that determines how demands of the grid will be met, if possible, by the modules. The level of control under the plant supervisory controller will consist of module controllers. Each module controller will try to meet the power demand of the plant supervisory controller. This hierarchy will continue down to the level of component control. Any controller unable to fulfill the goal set by its supervisor will communicate up the hierarchy. The supervisor will then try to meet its goal by another method.

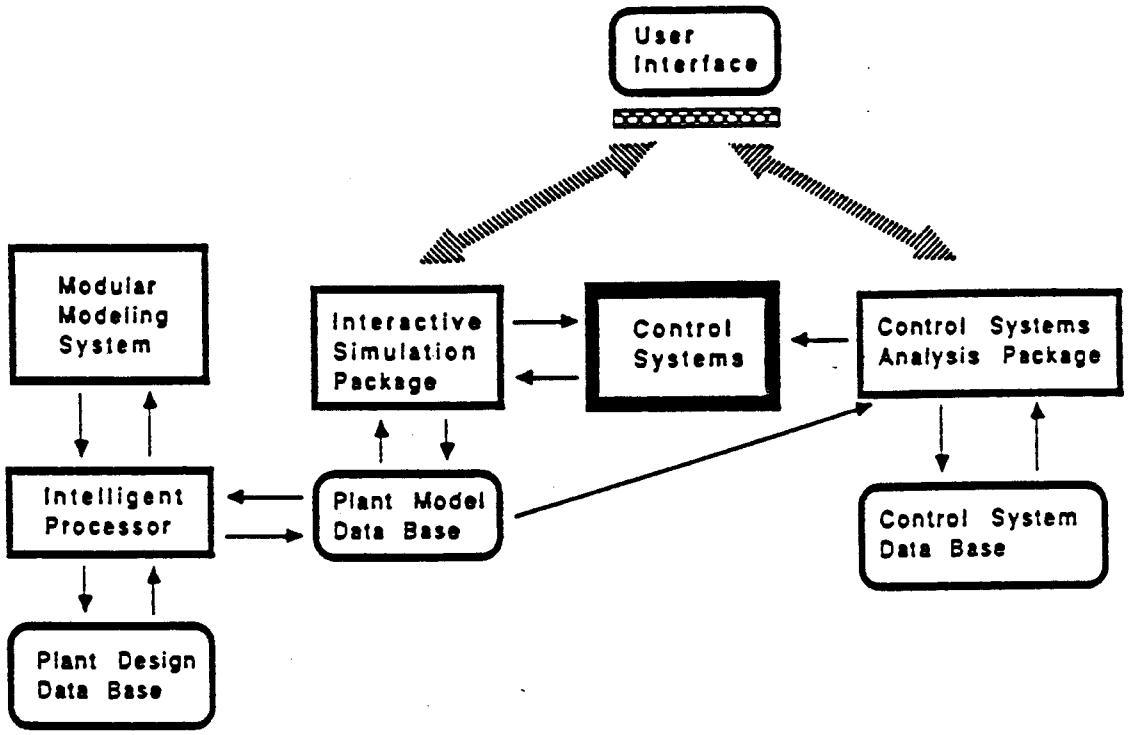

Fig. 4.3. Block diagram of workstation.

A strategy has been developed that allows the controller to follow load demands in the presence of unknown or unexpected variations in parameters and subsystem responses. A key feature of this algorithm is called parameter tracking. As a nuclear reactor goes through its normal range of operation, some plant parameters change. Also, over the life of a plant, some parameters change. The nonlinear control strategy will have the ability to track changing parameters and continue to control the reactor or reactors in a safe and efficient fashion. Further work is needed to refine and apply the strategy to an ALMR.

4.5 TESTING AND VALIDATION BY SIMULATION

Prior to installation in a nuclear plant, the ACTO facility must provide for integrated testing of control system hardware through the use of process simulation or other systematically generated test signals. Control systems can be tested over a range of process conditions from normally expected conditions to the very limits of the instruments. A hardware test facility must be able to test portions of the hardware while the remainder of the control system is simulated in software so that subsystem tests can be performed.

Hence, the conceptual design and guidelines for ACTO must factor in the use of both total plant and plant subsystem simulations. In the initial stages of the control system design and testing cycle, the simulation of both the processes and the control systems

will be combined in an integrated simulation, and not necessarily run in real time. Later in the design life cycle, however, the interfacing of separate process simulations and controller hardware will be required, as considerably higher confidence can be placed in a system that has undergone "actual operation" prior to installation. Eventually, the integrated system would need to be run in real time to design and test the hardware and operator interfaces. In all cases, the designers must be assured they are dealing with fully verified plant simulators.

Surveys and investigations of computer hardware and software capabilities for satisfying the full-plant simulator requirements, particularly the need to run with real-time operating systems, have been initiated and will continue as the hardware and software are being acquired. Discussions are being held with vendors of various types of computer hardware and software systems (e.g., workstations, parallel processors) regarding current and projected capabilities.

The potential benefits of various architectures (local and shared memory for the parallel processors), processor types (vector and scalar), operating systems, programming languages, and other features are being considered. Special attention is being given to determining the program requirements and then evaluating the applicability of current and projected availability of expert system tools, object-oriented programming systems, computer-aided software engineering (CASE) tools, data base management systems, and graphics capabilities. Because of the rapid development of both hardware and software technologies, capabilities for upgrading systems and transporting (and reusing) software are major concerns.

State-of-the-art advances in computer architectures, software engineering, very high level languages, area networking, artificial intelligence and data base management need to be integrated into the whole-plant, real-time nuclear power plant simulation capability. This capability will be linked to the design workstations described previously but will also be addressable by telecommunication networks. Design organizations will be able to use this capability to test and demonstrate their own designs and demonstrate them to the utility community. Since these designs will include actual controller software and hardware specifications, the simulator will be capable of linking to actual (or prototypical) controller hardware and running the software on it.

Methods are being developed to ensure that ACTO software development conforms to industry standards (ANSI, IEEE, NRC Regulatory Guides, etc.), both to ensure high-quality output and to ensure that the resulting controller designs are certifiable.

Simulator validation and verification are continuing efforts, and again, they will depend on the availability of pertinent data and corroborating runs from independent codes.

ACTO plans include an environmental laboratory to permit the testing of environmental effects on control systems during operation. Controllable conditions in the environmental test chamber will be temperature, humidity, and dust concentration in the atmosphere. The power supply voltage will provide a range of surges, noise, and off-normal voltage conditions. Transmitters for radio frequency (rf) interference and spark generators will also be available to test the rf rejection of the control system. A shaker

table will simulate a range of mechanical vibration frequencies. Common effects such as cabinet temperature or supply voltage can be tested and measured on the assembled, operating system. Modules can be driven to the point of failure to verify that the control system fails safely and in the modes predicted.

The ability to simulate an entire plant in real time is critical to the design of a fully automated plant. For certain types of studies, only high-level simulations (low-order models) are needed, but for test and validation of automated control systems, significantly more detailed models are needed. To this end, preliminary work has begun on evaluating computer codes for aiding the development of flexible dynamic models for typical plant components, particularly with regard to providing the user with choices between high- and low-order models.

4.6 ARTIFICIAL INTELLIGENCE IN THE CONTROLLER DESIGN AND TEST PROCESS

AI methods can be used to aid the design engineer using ACTO as well as in the control system itself. On the ACTO system, AI techniques will perform sophisticated tracking of user problems and progress. The following features are examples of aids that could be incorporated into the user interface program:

- user assistance to aid and guide novice users,
- expert teacher to tailor tutorial programs to the experience and needs of the user,
- command "spy" to recognize context or user objectives using pattern recognition based on the most recent series of user commands,
- natural language interpreter to recognize approximations to commands and instruction,
- intelligent data base management to aid in searching for information or categorizing information to be stored, and
- interactive high-level planner to specify controls design with an autonomous low-level planner using a controls analysis program (e.g., $MATRIX_x$) to create the design details.

4.7 FAULT-TOLERANT CONTROL DEVELOPMENT

The fault-tolerant control system development effort will investigate architectures for communications networks, digital processing units, and algorithms such as the component "heart beat" to detect and reconfigure for hardware failures. The study will use probabilistic risk assessment (PRA) methods to compare different strategies and will establish acceptable mean-time-to-failure limits for the control system.

The use of fault-tolerant automation can reduce the probability of a major accident through its ability to maintain complex systems within a prescribed operating envelope. Analog subsystem controllers, which constitute the essential control processes in current U.S. nuclear power plants (NPPs), have evolved over many years and have generally performed satisfactorily around a design point. Performance of these analog controllers is limited, however, in dealing with system upsets and major parameter changes. Major improvement in most aspects of subsystem control could be enabled by economical, reliable digital microprocessors.

High reliability can be further enhanced by use of fault-tolerant design techniques, previously used only in NPP protection systems. Communication among subsystems and other levels of the hierarchy is greatly improved and simplified by digital techniques. Multiplexed fiber optic data transmission and distributed architectures provide an opportunity for noise reduction and significant construction cost saving by minimizing cable lengths and interconnections. On-board memory increases the potential for self-checking for failures or decalibration.

Fault-tolerant designs of microprocessor systems promise to provide essential hardware reliability. Continuing development is needed to ensure cost-effective designs and to provide operating systems and software techniques compatible with the hardware designs. More highly automated plants will further separate the human operator from interaction with individual processes; therefore, built-in and automatic fault diagnosis and recovery techniques are needed so that timely maintenance can be performed and functions restored rapidly.

Two basic design objectives in fault-tolerant control systems are: (1) that for any single hardware or software failure, the plant will be able to continue operation without any degradation in performance; and (2) that the fault can be identified at the level of component that must be repaired or replaced. Service of the faulty part must not require taking the control system out of service or disrupting any function of the control system.

By meeting the single-failure criterion, a great many multiple failures can also be tolerated with the same level of performance. However, for certain unlikely combinations (e.g., all rod control digital processing units simultaneously fail), some degradation must be expected. The performance design objectives for multiple failures are that the plant must fail in a safe condition and that the control system must detect and indicate to the operator through alarms that it is not able to perform all of its functions. The identification requirement for multiple failures is that the possible combinations of failures must be identified. In other words, the self-diagnosis should provide all the information it can.

4.8 SMART SENSORS AND SMART ACTUATORS

Automated systems require high-quality, reliable data input from their front-end sensors, and they rely on actuators to respond properly to commands. In this development effort, smart-sensor algorithms will be developed and demonstrated for processing raw

data and generating validated data. The algorithms will be able to identify a failed sensor and lock it out of the control system. The operator interface to this system will notify the operator of the failure and the test results on which the failure was detected.

Smart-sensor standard modules will be developed and applied to measured (or simulated for demonstration) variables in the feedwater system. The control system design will be such that it can operate with any single failure in the plant instrumentation. Smart-sensor standard modules will be developed for liquid level, flow, temperature, and pressure measurements. Sensor data bus and data transmission protocol for incorporation of smart sensors into a hierarchical control scheme will also be developed.

5. SOFTWARE DEVELOPMENT, INTEGRATION, AND SUPPORT

This chapter covers the main aspects of software-related topics that will affect what can be done within ACTO. Section 5.1 describes software resources that will be available to assist with the control design process. Section 5.2 specifies the levels and kinds of support that will be available generally for whatever software development activities occur. It describes the environment an ACTO user can expect to find, if not currently, then in the very near future. Section 5.3 summarizes briefly what steps are being taken to address quality assurance requirements.

An extensive discussion of the software development life cycle is given in Sect. 5.4, tailored to the development activities of ACTO. Enough detail is provided so that this section can serve as a basic reference, and satisfy any formal requirements, for the software development plan for ACTO.

Data base management support requirements are currently evolving. Those needs that can be anticipated are identified in Sect. 5.5. This area, while acknowledged to be important, is not expected to receive much attention until more progress is made in assembling simulation models and in completing the first stage of development of the control design workstation.

Section 5.6 is devoted to the support that will be available for the use of the language Ada. A list of Ada features particularly important for control applications is provided to remind ACTO users that Ada is a higher-level language candidate and that its capabilities and deficiencies are well worth investigating. Ada is not now being recommended as the preferred language for ACTO.

Graphics support for controls design has been limited primarily to (sometimes animated) process control diagrams and to strip chart types of plots. With the recent emphasis on visualization in scientific computing, significant changes will occur in how visual images are used to convey and use information. The impact of this on the ACTO activities is still hard to discern. Section 5.7 identifies some of the technology that is leading the way.

Finally, an attempt is made in Sect. 5.8 to identify areas that need attention in planning for software support where technological obsolescence might be a problem, or where certain development trends should be encouraged to minimize long-term software maintenance.

5.1 SPECIAL ACTO SOFTWARE FEATURES

The program described in the preceding chapters will require software development, integration, and evaluation to become central activities in the Advanced Controls Program. Two major areas of focus will be (1) development of simulation software and (2) assembly of an environment in which to design control systems. Simulation software will cover physical modeling of all plant hardware and physical processes. Assembly of

the control design environment will cover the building of libraries of control algorithms and the acquisition of interface software (e.g., reusable graphical symbols for operator displays), software performance monitoring and testing tools, graphical design tools to assist with documenting designs, and expert system design advisors. This chapter addresses available support for the activities associated with developing these tools.

Both areas will require assembly and integration of many types of computer software in three main groups. The first (perhaps largest) group comprises the operating environment with various programming and design tools, higher-level programming languages, graphics libraries and tools, data base systems, routines to support communication between codes or system modules, and remote site communications. The second group includes simulators for the processes, major power plants, or reactor subsystems to be controlled. The third group will consist of the control systems or their simulations.

ACTO will provide an operating environment in which the user is, or can quickly become, comfortable and productive. This environment, the software tools for control system design testing and evaluation, and the simulators must be designed to remain usable as computer hardware technology changes. The user's view of the resources represented by this collection will be an interface (typically, the "windowing" environment of a workstation) with menus and dialogs displayed in rectangular frames on screen and a combination of user input devices that include a touch-sensitive display screen or panel, keyboard, and mouse. On-line assistance that covers all aspects of ACTO capabilities will also be available.

Figure 5.1 shows the elements of the control system design, test, and evaluation environment. Two major areas of focus--software and its development environment--will complement each other in realizing some of the primary goals of the Advanced Controls Program. The control design workstation environment shown to the right will be used to generate proposed new control systems. The real-time simulator will be a resource to this design environment. Eventually the new control system will be tied directly into the real-time simulator (through the switch) for test and qualification. Finally, the new control system will be connected to the actual hardware (demonstration or plant) shown at the lower left. In similar fashion, the control system for an actual plant or plant subsystem could be tied into the real-time simulator.

The ACTO software comprises a set of design aids and application programs to provide information, outlines for organizing work, examples, automation of tasks, and other features to improve productivity, communication, and accuracy of the controls design process. The ACTO software features presented here are intended to comply with the currently recommended Martin Marietta Energy Systems approach for managing the software development life cycle. They will also be required to conform to the software QA requirements of the Oak Ridge National Laboratory and Instrumentation and Controls Division. By the nature of its mission, ACTO will also participate in proposing and applying software development methods standards for the nuclear industry.

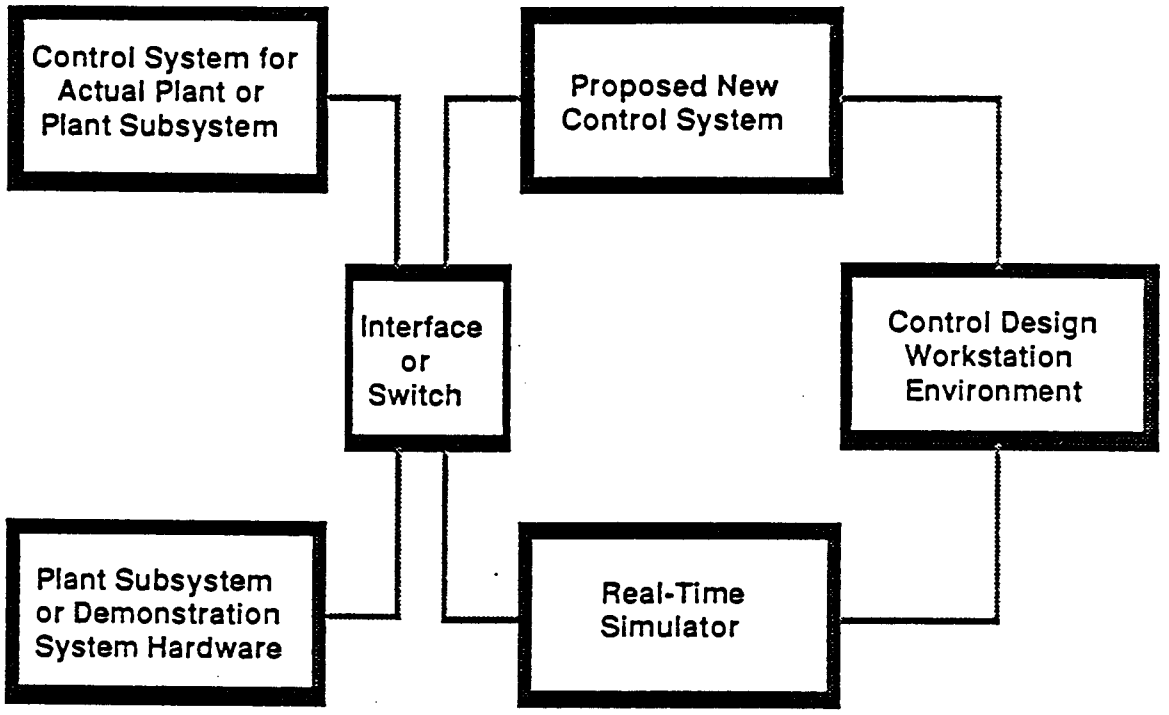

Fig. 5.1. Diagram of control system design, test, and evaluation environment.

5.1.1 Process Design Tools

Some abstract representation of the process to be controlled must exist in at least a preliminary form in order to begin the control system design. The controls designer's task is not to design the process but to describe it mathematically to the extent necessary to create the control system. The information required for controls analysis is generally the same information as required for a plant dynamic simulation. The information for process control design can be represented in two parts, the process schematic and the design data sheet (file).

For engineers, schematic diagrams form the standard representation of controls and process systems. They present the system arrangement and features in the form that is the easiest and clearest for the engineer to understand. The ACTO controls design workstation software should provide an interactive, graphical environment analogous to the drafting table for describing the process and control systems. The ACTO drafting system will provide symbols analogous to those traditionally used on process and instrumentation diagrams (P&ID) to represent the process.

Numerical data, which complete the specification of the process, are collected in design data files. The data files specify physical dimensions and operating characteristics that are necessary inputs into the controls calculations. ACTO software will provide screen-based user worksheets, in a form analogous to the design data sheet, that list the

required numerical data for dynamic simulation and controls analysis. The system will include management software to maintain a data base of modules previously specified.

The schematic and design data files contain all the information needed to construct a simulation of the process. Once a control system designer has mapped out a sequence of steps leading to a design, he can use command interpreter software (shells) to capture this sequence, expressed in procedural language descriptions (scripts) of the steps needed to go from the schematic drawings and data to a process simulation. Shells and scripts provide significant and powerful automation capabilities that form part of an integrated design facility.

Approaches other than shells and scripts will also be used. Expert systems will be coupled to data bases containing graphic (iconic) representations of collections of hardware or software components that might be needed to build a schematic diagram of a power plant or plant subsystem. These expert systems will have access to other data bases or libraries containing modules of code (in either source or object form) designed to correspond to the component icons. Suitable interfaces also will be designed to allow the modular code units to be connected according to the design in the schematic diagram. One implementation of this approach, using the expert system front end (EASE+), is being used by Advanced Controls Program staff to develop a feedwater control system demonstration. The EASE+ software and its coupling to the Modular Modeling System (MMS) library of hardware component simulation source code modules is an example of the capabilities that will be available to control system designers. A prototype for a different implementation of this approach using an object-oriented design framework is also being developed.[5]

5.1.2 Control Conceptual Design Tools

In the conceptual design step, the designer defines the qualitative objectives for the control system, outlines the basic functions of the control system, and gives the main subdivisions their shape. The designer usually works at a high level of abstraction. At the conceptual stage, the ability to construct prototypes quickly is very useful.

Creating a set of screen displays, such as those a plant operator might see, is a valuable way to represent a high-level prototype. Because the user interface is one of the most important pieces of control and plant management software, designing a prototype around what a plant operator will be working with eventually is a prudent starting point for the development process. When screen displays are developed at an early phase, it is possible for the intended end-users to begin to evaluate them and to provide suggestions for changes at a developmental stage when changes are relatively low in cost. In addition to developing screens or other displays early, the narrative that explains their use can begin to serve as the prototype the eventual user's (control system operations) manual.

ACTO will provide the designer with a collection of software tools for organizing the control design process and for designing screen displays. It will provide a library of reference designs or benchmarks from similar control systems and templates for documenting the conceptual design. The distributed, hierarchical control system (control

engine) as presented in Sect. 4.2, is a universal high-level design that shows all the functions of a fully automated control system. It serves as a base for expansion and a framework for categorization of example control systems to be stored in a data base.

Efforts will be made to capture (e.g., obtain a printed record) as many aspects of the control design process as possible. Those parts that can will be automated; those that cannot will be analyzed for use in an expert advisor system to prompt a designer. Expert advisors have been developed to assist with activities similar to those to be undertaken in the Advanced Controls Program. Logicon, Inc., a company that does independent software verification and validation work for the defense establishment, developed an expert advisor system that it uses internally to assist with tailoring the DOD-STD-2167A software development standards.[6] While access to such proprietary software may not be possible, information about its usefulness in practice and conclusions based on the experience of its users will be available.

Design information can be organized at an abstract level using data flow and state transition diagrams. Many software products are available to support these formal design methods. The best of these products include capabilities for rapid traversal of the design hierarchy from the highest level to the one with the greatest detail, a data base and file management capabilities that allow several designers to work simultaneously on the same project, the means to define links between diagrams and design specification documents, and support for configuration control so that a project manager can track and manage the software development aspects of the control design effort.

5.1.3 Support for System Life Cycle Document Preparation

A new type of computer now provides powerful assistance in automating two important areas of life cycle document preparation. With more than tens of thousands of processors, it broadcasts a single set of instructions to all the processors, which then operate in parallel (simultaneously), each on different data. The first area for which it can provide assistance is that of matching codes, standards, and regulatory agency guidelines to the user requirements or control system conceptual design. The second area is that of tracking requirements and specifications through successive phases of a project's life cycle documents.

A single-instruction, multiple-data (SIMD) massively parallel machine (MPM) could be used as follows. The text of the first document written for the life cycle set is partitioned into fundamental units having the lowest level of detail to which standards, codes, or guidelines might apply. A unique number is assigned to each basic unit associated with this document. Each unit is then used in succession as a template document that undergoes a lexical analysis. The results of this analysis are used as the basis for searching the data base of standards, codes, and guidelines to identify and retrieve all items with content highly correlated to information in the template. The selected items are then displayed to the person working on the initial document, who marks for future reference all of those items that relate appropriately to the subject matter of the template.

The requirements tracking activity will be set up and executed in a similar manner. The text in each set of documents will be partitioned into basic units, by concept or by subject matter, and numbered. The SIMD MPM will identify connections in subject matter between the fundamental text units from one document set to the next in the control design life cycle. These connections will form the basis for defining links, which can be entered into a data base, that will be the formal expression to show how original requirements are mapped into all succeeding phases of the development project.

ACTO will acquire or develop tools that allow use of massively parallel computers to assist with document preparation and management in the manner described. As this type of support is used and control designers and system analysts become more familiar with it, other possibilities will be identified for tool development. Besides boosting productivity and performance, the capability described here should also provide support for QA programs.

5.1.4 Control Analysis and System Simulation

At least two main types of computations will be required for control design. The first (simulation) determines the transient response of processes and the control systems. The second (controls analysis) calculates properties of the continuous portion of the control and process systems.

5.1.4.1 System Simulation

Simulation of processes and controls is an essential part of ACTO software development and its resources. Process and controls models will represent real plants and controls for development, analysis, and test, since such work is often impractical (perhaps also forbidden by regulations in some cases) on an operating nuclear plant system. A process model is also used to calculate a numerical approximation of the state space model for input to controls analysis programs. Models are assembled simply by connecting modules (self-contained blocks of code) from a library in an arrangement analogous to the way graphical objects are connected in a system schematic, as described in Sect. 5.1.1. The current library of (MMS) modules includes most components found in commercial nuclear and fossil power plants.

The advantages of a modular approach are significant. First, it permits modeling equations to be designed for specific conditions and configurations of a component. For example, a reactor module is divided into a suitable number and arrangement of control volumes by its designer and developer. Decisions about the best ways to design the module are then made by someone experienced with the technical issues, so that each user of that module is not required to become an expert on those details. Also, sophisticated modeling techniques can be used when needed, as for example, when a once-through steam generator simulation uses a moving boundary scheme for regimes of heat transfer for subcooling, boiling, and superheating. It is the modular structure that makes it possible to generate simulation coding directly from the process schematic, a major automation feature.

An input data file (or data base) will be prepared specifically for each module. An on-line help data base will provide clear information on input constants and other information for using the module. On-line reference notes will instruct the user on sources for the required inputs and on formulas for calculating input constants. Internally, each module will represent the dynamics of the component using first-principles modeling. A set of nonlinear, ordinary differential and algebraic equations, derived from conservation laws, equations of state, and empirical equations for heat flow and pressure drop, composes the module coding. Module complexity is defined by performance requirements. Different modules may be defined to describe the same physical component at different levels of depth and complexity, as required by the specific purpose of the simulation.

The library will contain validated modules for all components in commercial pressurized water and boiling water nuclear power plants. MMS, the library now being used, has a bonus: it contains components for fossil plants. Development is under way to add modules for ALMR designs.

In many cases, users will have a developed process model ready for use on ACTO resources. In some advanced reactor studies, the user model may in fact be the only model. Provisions will have to be made to accommodate incorporation of user models into the ACTO libraries and development environment. In some cases this will be straightforward, in others, more difficult. As time goes on, provision will also have to be made to accommodate a heterogeneous hardware environment (i.e., support for computers from vendors other than those currently represented) while the software environment becomes more uniform (at least in terms of interfacing with the user). Difficulties are anticipated in connecting the user's process simulation to other ACTO features, such as controls analysis and real-time simulation (see Fig. 5.1 and related discussion). These problems are being addressed by providing architecture as open as possible and by publishing interface requirements.

One ACP goal is to provide a facility that will let experienced control engineers develop their simulation programs with a day or less of initial learning time. Development of prototype control design workstation tools based on object-oriented design methods is one step in this direction. For already existing codes, the ACTO staff will provide assistance and expertise in converting and interfacing users' simulation codes to the ACTO system. Conversion of the process simulation for the ALMR project PRISM is well under way at ORNL.

5.1.4.2 Controls Analysis

The analysis techniques discussed in this section are primarily for the continuous features of the control system. Analyses can be done using classical or modern control techniques. Classical control techniques use various graphics, such as Bode, root-locus, Nyquist, and Nichols plots, to tune gains and other settings for stable response. Classical controls generally must be applied to single-input/single-output devices. Hence, the classical approach is best for small systems within the plant or isolated control loops that can be analyzed separately.

Modern control techniques use optimization to tune system response. Linear-quadratic-Gaussian techniques calculate optimal control based on a "cost" function. These techniques include Kalman filters for state estimators and linear regulators for calculating an optimal state feedback matrix. One primary advantage of modern controls is the capability to analyze multiple-input/multiple-output devices and thereby achieve an integrated control system.

At least two commercial sources of software products ($MATRIX_x$ and ProMat-Lab) provide comprehensive support for controls design and analysis. Tools may also exist within university engineering schools (and hence be in the public domain). One tool set has already been provided by the University of New Mexico.[7] Software will be acquired that calculates feedback gains for a system based on criteria established by the designer and that performs matrix calculations.

$MATRIX_x$ has already been used on some ACP tasks. Most $MATRIX_x$ operations require that the process and control system be represented as a linear, time-invariant, state-space system, but $MATRIX_x$ can linearize a nonlinear system for control system design and analysis. Nonlinear function blocks are available in the $MATRIX_x$ environment for use in temporal simulation. It is also possible to calculate the state-space matrix for the process using the MMS simulation model.

System identification tools that can generate a process model from experimental data may also be desirable and are readily available commercially. Use of such tools will likely require greater labor than other approaches, such as integrating a group of process and control simulation modules from the collection (e.g., the MMS library). System identification tools use regression techniques to fit a model to measured response data.

5.1.5 Implementation Tools for Controls Design

Implementation tools software for controls design will include data base management systems with appropriate user interfaces to assist with equipment specification and selection; preparation and maintenance of electrical wiring, communication cable, and optical fiber line drawings; and screen display design information.

Software to assist with equipment specification will use design information and performance requirements based on simulation results to identify and list the hardware needed to accomplish design goals. Software (to be developed) will automatically prepare a list of design constraints that affect the choice of hardware; systems for certain process control applications are already commercially available.

Automatic code generation may be possible directly from control design drawings (e.g., for programmable controllers), and some commercial control design tools can generate FORTRAN or Ada code automatically from control system functional blocks developed by using the design software. Automatic code generation capability will be acquired as needed for use with its associated design software.

The Research Instruments Section of the Instrumentation and Controls (I&C) Division has an engineering design center and development staff gaining experience in the use of

very large-scale integration (VLSI) design software for application-specific integrated circuits (ASICs) for radiation sensors, monitoring instruments, and other radiation measurement and detection systems. This group is interested in the development of smart sensors and is a potentially important resource for ACTO in this aspect of advanced instrumentation technology.

5.1.6 Subsystem Test Environment

Some software testing will have to be done under circumstances in which the controlling and controlled hardware are subjected to harsh environmental conditions, such as high temperature and humidity, low temperature, and dusty atmosphere. Harsh conditions affecting the hardware could produce undesirable side effects on the processor and hence on software performance, particularly software in embedded applications (e.g., a special purpose processor integrated into the hardware package). Software designed to detect computer hardware problems under these circumstances must be verified to do so correctly so that the system executive has reliable information for reconfiguring the remaining resources. An environmental test chamber will be required. Tests will need to include operation in the presence of power surges, noise, and off-normal voltage conditions. Transmitters of rf interference, spark generators, and perhaps even X- or gamma-ray sources may be required for some tests. A shaker table will be needed for mechanical vibration tests.

Some of these tests will be valuable for inducing failure to verify that the control system fails in a safe manner and according to the sequence predicted.

Facilities for these tests are not set up, nor are there plans for them in the near future. Limited testing can be done by making arrangements to use existing facilities in the Oak Ridge complex.

5.2 SUPPORT AVAILABLE FOR SOFTWARE DEVELOPMENT

5.2.1 Software Development Environment

While the ACP has a narrow focus in terms of its main software applications areas, the software development life cycle issues and software applications needed to support the main activities cover almost all areas of computer science and technology. This may seem somewhat surprising at first. Note, however, that the environment will have to (1) support evaluation of controls design concepts (building prototype systems), (2) provide various test bed environments such as simulators to check out new control systems, and (3) include demonstrations of new types of systems (both controls and plant simulations). Each of these three activity areas will use, probably in very different ways, a number of processors distributed across different networks or buses (data highways). Both shared and distributed memory architectures will be required; both, in fact, are already being used. Large data bases will be needed, and they will require software to tie into the activities they support. Software also will be required to support networks and communications links among the various activities with the ACTO center and

between the center and external users. All these activities will encompass a variety of integrated hardware and software.

The primary goal for software development is to provide tools that permit the ACTO user to accomplish his tasks efficiently and reliably. The environment must include not only support for a variety of higher-order languages and libraries, but also the means to manage and document development appropriately. A related goal is to create an environment that will allow the user to function as much as possible without distraction by computing system details. Some special tools available to help with this problem are described in Sect. 5.1.

In the early 1970s, developers of large software systems became acutely aware of rapidly rising software development costs because the development methods used had evolved in haphazard fashion. The primary patrons of large software systems were NASA and the Department of Defense (DOD). To contain costs, NASA and DOD began supporting efforts to discover where and how the costs arose and to see whether a change in software development methods would help to reduce them. Three areas getting the most attention were (1) higher-order language support for real-time applications (particularly embedded applications), (2) software development methods (the need for standardization), and (3) software development tools (the need for greater productivity).

The major outgrowth of effort in the first area is the DOD-developed language Ada. The major result in the second area was the introduction of new requirements for management of the software development life cycle.[6] Activities were coordinated to incorporate into the design of Ada features that support good software development practices. Large corporations that supply computer hardware and software to the commercial market learned from their experience and that of program managers of large-scale automation development projects for the federal government, and began to shift their emphasis in methods for software development as well.

With this shift came a demand for tools to support the new methods. Consequently, environments and management tools that support a formal, planned approach to the software development life cycle are being produced and marketed. Activity is particularly intense in relation to support for the Ada language because of the large economic incentive to win defense contracts that require the use of Ada and formal development methods (hence, the need for tools). Vendors are offering environments that support not only Ada, but FORTRAN 77, C, Lisp, and Pascal as well. These environments include common symbolic debugging support and configuration management tools.

Specific aspects of the ACTO software environment to be addressed include operating systems, text (file) editors, higher-level languages and associated compilers, linkers, symbolic debuggers, libraries that provide support for standard areas (e.g., numerical analysis, graphics), specialized data bases, configuration management tools, electronic mail, and interactive communications with other users. Each of these will be addressed, with comments directed toward ways to support the goals of the ACP. Support currently exists for each of these features.

5.2.1.1 Operating System

Two operating systems in widespread use from the microcomputer through the superminicomputer level are VMS (DEC's operating system for its VAX family of networked computers) and UNIX. Communities familiar with either of these are large enough that both systems warrant support to some degree. DEC also provides its own version (ULTRIX) of the Berkeley Standard Distribution (BSD) Version 4.3 UNIX.

Not yet generally available is a way to save an image of an executing code at the instant of an interrupt. The interrupt may be initiated by the user at a terminal or by a condition encountered as result of an error detected during execution. This capability is useful if execution must be terminated by the operating system when a user's time or money limit has been reached. If images and files for input and output are saved, it is possible to restart execution later. It should also be possible to use a symbolic debugger to examine the contents of the image file.

Another very useful capability would be a message-handling facility that permits one process to talk to or initiate another process. With the capability for one process to control a second, a controller-controllee hierarchy can be supported. This arrangement offers enormous power for using system resources. This capability is generally available in distributed processing environments, but has not been for serial processor systems.

The operating system will support use of multiple windows as a means of keeping several processes active in a multitasking mode. The user can switch from one window (process) to another to monitor several concurrent activities. Extensive use of multiple windows, however, will require larger amounts of memory.

5.2.1.2 Editors

Screen editors will be standard and available in a multiwindowing environment. Editors will be language-context sensitive (i.e., they will support good visual layout/formatting practices) to allow software to be developed in a paperless environment. The developer will be able to move easily and conveniently between software life cycle (system requirements and detailed design) documents and modules of code being written.

5.2.1.3 Languages

A range of languages will be supported so that controls system developers can do coding in whatever language works best for the types of applications they are developing. Supported languages will include Ada, C, FORTRAN, Common Lisp, Prolog, and Pascal. Some commonly used continuous and discrete simulation languages will also be supported. Most ACTO users will likely do much of their software development work on the systems and environments used by their own organizations. Thus, portability of source code will be important.

5.2.1.4 Compilers

ACTO compilers will meet published standards for the supported higher-order languages and, as far as possible, avoid vendor-specific extensions or nonstandard features. This accommodation will enhance the portability of source code.

5.2.1.5 Libraries

Various libraries in object code form, ready to be linked to applications programs as needed, will be available. In addition to standard mathematical functions, graphics, and numerical analysis libraries, libraries assembled and maintained to support the activities of the ACTO center include digital signal processing routines and statistical analysis routines.

5.2.1.6 Data Bases

Several data base uses have already been mentioned in Sect. 5.1. Most of the data base support required initially will be directed toward managing documents, on-line assistance for using ACTO capabilities, and sets of reusable modules of source code.

5.2.1.7 Symbolic Debugger

Compilers will be chosen that permit the use of a common symbolic debugger to display contents of various addresses and values of variables. The debugger will allow the user to execute code and set absolute and conditional break points. It should permit a user to provide instruction addresses (e.g., when an error condition is encountered) and use it with appropriate mapping tables to point back to the corresponding lines of source code.

5.2.1.8 Graphics

Convenient access to graphic displays and hardcopy is an important aspect of a productive software development environment. Features to be considered are:

1. Color capability for preparing overlays for emphasis or contrast in schematic diagrams, for highlighting modifications to documents of source code, and for indicating the status of subsystems in real-time simulations will be provided.

2. Software and its documentation will be accessible to ACTO users who need data visualization tools, structuring tools, and data analysis software from National Science Foundation (NSF) funded work at the National Center for Supercomputing Applications (NCSA) at the University of Illinois. Software from this center is intended to provide scientists and engineers with the means to identify significant features in data that are difficult to interpret, such as results from some kinds of multiparameter studies.

3. File routing capabilities will be used that permit a user to send graphics files to a hardcopy device at his site (node) or to any remote site that is part of the ACTO

network. Networks will be used to send files to spoolers (special file output managers) for plotting.

4. A variety of graphics device drivers to permit the use of Versatec printers/ plotters, film devices (e.g., DICOMED or FR80), or high-resolution laser printers to support a device-independent capability will be provided.

5. Video display technology for routine use in preparing special demonstrations of new capabilities in control or operating procedures will be provided as soon as it is readily available on the commercial market.

6. Special graphics software will be available that can use output from a process or control simulation to animate aspects of a display, such as numbers and text, object size or color, object position or orientation, and object shape (stretching or directional compression).

5.2.1.9 Computer Conferencing

When people collaborate on a software development project or when a user needs help from a consultant to track down a bug that is hard to isolate, it is convenient to set up a computer communications network that will send to specified users' terminals all information displayed on the questioner's terminal. This capability would permit not only geographically dispersed collaborators to work more efficiently, but it would also permit a controls designer to give demonstrations of new systems to groups off site. This capability could help developers, researchers, and managers to make efficient use of their time and energies.

5.2.1.10 Electronic Mail and Bulletin Boards

The electronic mail facility should allow one user to send text, binary, or graphics files to another by a simple ATTACH command. The ATTACHed file could be placed in temporary storage and the file name and location given via electronic mail to the target user, who then would retrieve the file. Bulletin boards will be maintained for important utilities, file editors, compilers, libraries, and development tools to keep users informed of known bugs and other information.

5.2.1.11 Configuration Management Tools

While many configuration management products are being marketed, they are not necessarily convenient or easy to use, especially on an occasional basis or for small- to medium-sized projects. They generally require a significant investment of time to learn, so they find use primarily when economic and productivity benefits are clear. The growing realization of need for support of management tools by software developers should stimulate improvements, even as users develop a base of experience with current versions. However, sophisticated configuration management tools technology is still developing and will not be acquired until it clearly is needed.

5.2.2 Layered Structures

Figure 5.2 illustrates the environment being implemented on all ACTO workstations and other ACTO multiuser host systems, with the operating system core (some form of UNIX, preferably BSD 4.x) at the center. Surrounding the core is a shell layer (usually with at least two vendor-supported shells such as the C and Bourne shells) that allows the user to customize the basic environment to fit the tasks. The windowing environment of choice fits approximately at the same level as the shells. While several windowing environments are currently in use by the ACTO staff, the X-Windows (Version 11, Release 3) system is recommended for all user interfaces for ACTO software because it is widely used and resides in the public domain. Compilers, utilities, and perhaps a data base management system (DBMS) are in the next layer, and software development tools, libraries, and application software are in the layer on top of that. Perhaps another layer could be added to represent system simulation models.

The X-Windows environment will probably do more to support portability of applications than any other software development feature except, possibly, Ada. X-Windows is significant because it is an integral component of the user interface. Execution of software on different hosts, even with different operating systems, is easier to manage because X-Windows defines a common medium for output display intended for a user. Software written to use the X-Windows applications interface will be portable; that is, changes to the graphics parts of the software will be minimal or none. Furthermore, in an environment in which all processor systems can communicate over Ethernet, a multitasking workstation with X-Windows support can be connected to several other hosts, each running a different program, with all display output sent to the appropriate workstation window. X-Windows software is easily accessible, and its development is guided by a consortium of major computer vendors.

Layered structure architectures provide a framework for integrating software and extending the system's capabilities. A key to their successful use is the care and thought given to the definition, design, and implementation of the required interfaces. This part of the software (application) development process is often overlooked or not given adequate attention. A windowing environment can help to support interfaces, but it can never be a substitute for proper planning and design.

5.2.3 Command Line Syntax for Heterogeneous Hosts

ACTO will initially have a three-tiered hierarchy of supercomputer, superminicomputer, and workstations (or multitasking, multiwindowing PCs). Parallel computers for the program can be classified according to level of performance, since computing power will depend on the number of processors available. While operating systems for host processors from different vendors will be chosen primarily to permit convenience of file transfer and access across the networks, another major goal will be to provide commonality of command line syntax, a capability important for user productivity. UNIX operating systems generally come with shells to help with this function. DEC provides much the same capability in the VMS operating system (DCL support) for its VAX line of computers.

ORNL-DWG 89M-13441

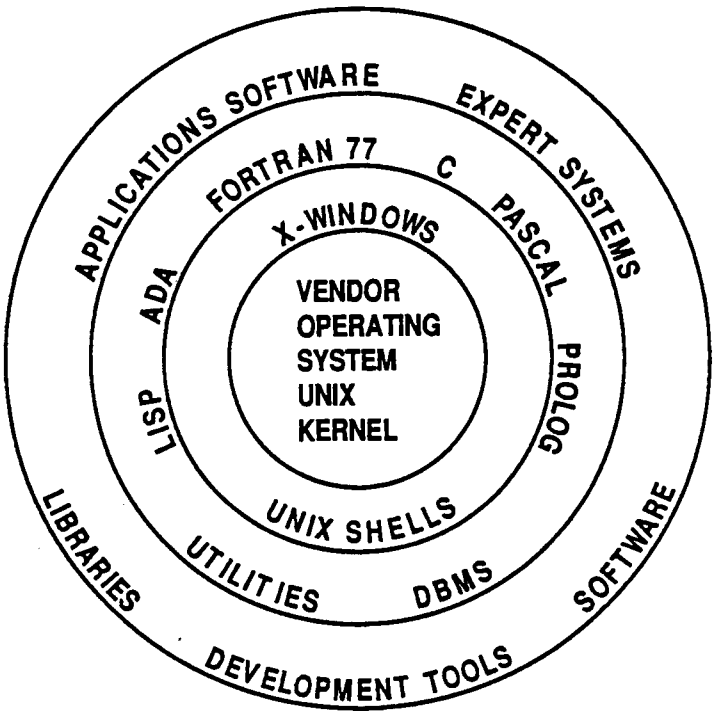

Fig. 5.2. Representation of software development environment for ACTO.

A common command syntax that works the same way on all three tiers allows the user to move from one resource to another quickly and transparently. Each user is able to define a syntax that will be his automatic default, and to redefine it at any time. This capability is especially important when developing and executing applications that use resources across networks. As networking and user interface (e.g., X-Windows) technology improves, distinctions between levels of performance of different hosts will become less significant, because ways will be developed to use the aggregate power of the networked system of processors.

5.2.4 Flexibility and Universality

Another ACTO goal is to provide a stable software development environment. A crucial and often overlooked part of the software development life cycle is its maintenance. If reactor lifetimes are 30 to 40 years, their controls systems must be able to be maintained for long periods. Many large computer centers have not had to be concerned with such long-term maintainability, so it will need specific attention at the ACTO center. The mixture of software systems and hardware arrangement must be designed so that new technologies can be incorporated while allowing support of those systems that must be maintained and used for a long time. Open-ended architectures permit this approach. Many vendors currently are committed to ready maintainability as an important design and marketing consideration. Continued vendor commitment to this capability will make it easier for the ACTO to reach its goals.

5.2.5 Computer-Aided Engineering Tools

Commercially available software now allows a person to prepare and maintain on one computer specifications for any conceivable system that is part of ACTO, examples being power plant subsystem processes, equipment, computer hardware layout and software design, control system designs, and integrated circuit designs for specialized microprocessor applications. Some commercial software comes with libraries of special symbols and tools for preparing specialized designs quickly, such as integrated circuit design or architectural drawings of a power plant. Some design tools also incorporate engineering standards for particular applications.

A growing number of design products include a capability to maintain large data bases that not only retain the definitions of design symbols used, but also store numerical data and narrative text associated with the symbols. A few hypertext products can assist with design and development activities. GUIDE, by Owl International, allows a developer to set up pointers linking items in a system specifications document to the source code modules that are based on them. If system requirements change and specifications must be modified, all source code modules that may be affected are already identified and are easily accessible for modification.

Many software products now use object-oriented design techniques first developed in the late 1970 by computer scientists at Xerox Parc (Palo Alto Research Center). These techniques are especially powerful for process control design and modeling. Tools to support object-oriented design and programming will be available as part of the computer-aided design and engineering (CAD/CAE) environment. Ada, C^{++}, Objective C, Objective Pascal, and several versions of Common Lisp can be used with this design approach. A combination language and environment, the most direct descendent of the Xerox work that has versions that will run on systems from PCs to mainframes, is SMALLTALK. Usually regarded as an AI language, this system will be available as part of the development environment.

5.2.6 Adapting to Rapid Technological Change

Computer hardware development life cycles are currently 18 months and decreasing. Memory capacities of memory systems of all kinds are increasing rapidly and costing correspondingly less per unit of capacity. Software development efforts are struggling to keep pace. Improvements in computing performance are now being pursued increasingly by use of parallelism. A number of different parallel architectures are being explored. All of this could be viewed as a planner's nightmare.

A useful principle to follow here is to choose a system that matches the performance requirements of the applications that will run on it. For ACTO, attention should be focused on the conceptual design for the *ideal* simulation environment, the *ideal* control design development environment, and the *ideal* tools, software utilities, and interfaces that support reliable and safe power plant control and management. By working from this point of view, it will be easy to identify at any given time the most cost-effective and reliable computing technology. Producing a design that will work satisfactorily for a long time in an area of such rapid technological change as computers, however, will be

difficult, because design efforts are usually more influenced by technology in use than by forward-thinking design ideas.

Primary design planning and guidance need to be based on the assumption that both software and hardware will be upgraded or replaced at least once every 5 to 7 years. Careful attention needs to be given to the use of modularity and specifications for interfaces. ACTO will provide a valuable service by establishing a data base to help track changes and costs in order to determine when in the lifetime of a plant it is economical to replace a given level of technology. Such a data base will become more useful, too, as a base of experience is built with succeeding generations of technology used solely within ACTO.

New products are changing the way computers are used. Occasionally a product is introduced or marketed in a way that changes in a very significant and unpredictable way how people use computers. The availability of the UNIX operating system to the university community for a nominal cost has had such an effect. Technologies likely to have a strong impact include (1) read/write long-term storage using compact optical disks; (2) integrated audio and video processing in desktop workstations (eventually meaning common use of voice command for process control); and (3) software to monitor, test, and diagnose system failures, and to reconfigure the system to provide optimal use of resources under new conditions. A rule of thumb is to view the top performer of a particular vendor's product line as typical, in terms of its processing capabilities, of entry-level systems for the following generation of products. Several vendors consequently have set minimum goals of doubling the processing power with the introduction of each new product line.

One new research area that appears to have potential for use in advanced control systems and that is generating high expectations is neural networks or neural computing. Neural network systems can be constructed to learn from experience and to use this "knowledge" to recognize familiar patterns presented in contexts different from those encountered previously. Neural networks can be used to deal with abstract relationships as well as with real objects.

Another area of research with potential for advanced controls is optical computing, in which bits of information, represented by trains of light pulses inside optical fibers or other optical transmitting media, are routed and combined using very high-speed optical switches.

5.3 SOFTWARE ENGINEERING METHODS AND QUALITY ASSURANCE

QA is no longer merely a goal to be stated as part of good engineering practice. All professional engineering societies are establishing guidelines and procedures for formalized QA programs; in the nuclear industry they are known as NQA-1. These guidelines are the basis of the QA program for ACTO. Software QA procedures described in procedure QA-IC-19 and proposed for use by the I&C Division at ORNL will be the basis for all software developed as part of the funded activities of ACTO.

Other guidelines and standards that will be used as appropriate are IEC 880;[8] probably the first international standard approved for nuclear power plant safety systems, NUREG/CR-4640;[9] and ANSI/ANS 7.4.3.2 (1982).[10]

5.4 SOFTWARE LIFE CYCLE

5.4.1 Technical Management Guidelines

Development of quality software requires systematic planning, especially at the beginning of the process. Before any code is written, a major planning effort is essential to meet schedules and contain costs. Experienced software developers have proposed methods to manage large projects based on their experience, and many federal agencies, especially those tied to the defense establishment, have reached similar conclusions. Wide agreement now exists on the major elements of a successfully managed development effort. These elements are described below.

Development methods have been proposed for commercial software, for military use (i.e., DOD-STD-2167A), for general use, and for operating systems, and at least one has been proposed for the nuclear industry.[11] Although they address common concerns, none is entirely applicable to ACTO, because ACTO requires many different types of software. No proposed method is intended for use in a rigid, mechanical way. The basic method will have to be modified to fit the application class for which the software will be developed. Software for critical (e.g., safety) systems doubtlessly will have to be developed according to more detailed and strict standards and procedures than that for plant management information systems, for example. Methods used by a control designer preparing a demonstration of new control strategies will be different from those used by developers of software using the same strategies for control of an actual plant.

To the degree that formal development methods are used, it will be possible to automate aspects of the life cycle development process. Some of the effort of ACTO will be devoted to automating this process where possible, for example, in matching codes, standards, and guidelines to user requirements via an SIMD massively parallel computer. Larger development efforts will need design reviews at the conclusion of major phases. Software developed for critical or costly systems should be subject to independent validation and verification.

As a minimum, even software written to try out new control strategies should have the following documentation prepared at the appropriate step in the process

- a description of the basic strategies and reasons for proposing them;
- a proposal for implementing the strategies (i.e., a description of the software design); and
- a summary of the results lessons learned, and, in the case of software to be used in demonstrations, instructions for using it.

Software life cycle documentation requirements will be designed to support effective execution of a programmatic activity. There is no reason to require any documents beyond this, even for QA. Document requirements will be justified on the basis of use as tools to aid development; they should not interfere with tasks. Documentation requirements for control engineers writing software to test new ideas will be different from those for programmers or control engineers developing software for demonstrations.

Software developed for ACTO that will need at least minimal life cycle documentation includes:

- all demonstrations of control systems,
- all plant and process simulation models, and
- a control designer workstation integrated tool set and environment.

5.4.2 User Requirements and Prototypes

High-quality software that satisfies the user depends on the attention given to a few critical areas, such as user requirements. Developing these requirements is one of the most difficult tasks in the whole software development life cycle. The user who wants the software oftentimes is not willing to give the time and energy to work with the systems analyst(s) to define the requirements. User education is usually needed, sometimes a lot of it. Getting the user to understand his needs clearly is difficult, and can be tedious when many iterations are required.

Prototypes or models can suggest alternative methods and processes. Early design of screen displays for the user interface, for example, can point out the need for certain types of information that the user must supply. Early development of the visual pieces of the user interface should be undertaken not only to help the intended user and analyst arrive at a clear understanding of the requirements, but also to give the user something to try out and to evaluate so that design changes are made early, reducing the impact on schedules and costs that would occur if they were made later.

User requirement documents should specify system capabilities, data that may be required and their location, user input and output requirements and formats, and the preferred method of operation. Any relation to other software, whether as a client or a server, must be defined. Data transfers between a client or server and others with which a processor must communicate should be identified and defined. Any specific language requirements must be specified, and any specific hardware requirements must be given.

A preliminary manual of operation written by the user, with help from the analyst, serves a role similar to that of the prototype user screen displays and increases the chances of the product's meeting the expectations of the end-user. Also, the user will have a sense of ownership of the deliverable and be less likely to feel that the developers have produced a deficient product, awkward to use or difficult to understand.

5.4.3 System Specifications and Functional Description

The system specification describes what is to be built or developed and explains what is needed to satisfy the technical requirements. A functional description describes how a system is supposed to work. It is one of the most important working documents for the development process and will evolve as changes are made to the user requirements. Change may be made because of constraints or difficulties encountered in later developmental stages. Both documents generally are considered to be the basis for developing the detailed design, for devising test plans, and for preparing system maintenance plans.

System specifications are the most important part of the development effort. They describe what the system is and what it must do. They tell what must be provided to meet the user requirements and standards for the hardware and software; they describe subsystems and components and their relation to one other; and they identify and describe interfaces. Identification and description of the interfaces is a critical but often neglected part of system specifications.

The functional description describes how the system will be used and how it must operate. It shows the source, transformation, and disposition of data. For complex or large-scale applications, tools are available to represent graphically the processes and interfaces. Data are characterized in a data dictionary according to how they are used. Data movement and transformations are depicted in data flow diagrams.

When applications are to be executed on parallel computers or multiple hosts across networks, diagrams will show the data to be moved along the network(s) or buses (data highways). Real-time applications often include timing constraints, which are more easily understood when displayed graphically.

Functional descriptions should be so detailed that a test plan can be designed along with the software. Requirements stated in a way that cannot be tested should be examined, analyzed, and restated in a form that allows testing to be done that demonstrates the requirement can be met. This may mean reworking the system specifications or revising earlier documents. In applications with large data bases (e.g., transaction processing or logging information about operation of a large plant), data management is a major task. If not addressed in the system specifications, a data base management system must be selected at this stage.

5.4.4 Detailed Design

Major activities for this phase are designs of subsystem interfaces and tests to verify software operation. Considerable effort will be required in particular for the user interface. It is not uncommon in data-intensive applications for user interface design to require 80% or more of the total development effort.

The detailed design process must also include the design of test procedures that consider

- what is to be tested,
- how it should be tested,
- why the test is needed, and
- what the test should demonstrate.

Diagrams may be helpful to show control flow, message routing among processors, process execution scheduling, data synchronization requirements, and memory management schemes.

Graphical design tools now being marketed for specialized applications can automatically generate code based on a diagram. Many transaction processing and other data base-oriented applications can be generated in this way. One reason for the interest in object-oriented design is the hope that greater automation of the software development process will be possible.

5.4.5 Coding and Unit Testing

A well planned detailed design usually makes writing source code and preparing (internal) code documentation easier and encourages use of good coding practice.

Certain standard features should be present in each coded module.

1. Instructions for using the module must be prominent.

2. The data structure should match the design, and all mnemonics (possibly excepting some loop indices) must be defined.

3. Any pseudo code should be interleaved with the actual code language.

4. Further comments related to features of the functional design should accompany the code that implements those features.

5. The coding may use "go to" statements or statement labels.

6. When statement labels are used, they should relate to the structure of the code.

7. Numeric labels should be sequential.

8. Mnemonic labels should have intelligible meaning.

As each module is produced, it should be tested if possible. Test procedure will be determined during the detailed design. Shortcomings, unexpected, incomplete, or missing requirements, and other consequences of the design that often appear at this stage must be reported immediately to the system analyst so that appropriate modifications to any supporting documentation or affected module(s) can be made.

5.4.6 Integration, Test, and Verification

After a number of units have been completed, their integration into subsystems can begin, accompanied by testing according to the procedure developed during the detailed design phase. Proper test procedures require that records be kept, certifying that tests were performed and specifying the results obtained. Any shortcomings or unexpected results must be reported immediately so that appropriate modifications can be made.

Thorough testing often requires more creativity and imagination than designing the system itself, particularly for real-time systems. Test planners must anticipate every combination of ways a user might try to do something to or with the system.

When the development team is reasonably confident that the software performs as it was designed to do, but no more, it should be reviewed by someone from outside the development effort. After this review, further testing and evaluation should be done by a "friendly user." Finally, when user instructions are near to their final form, beta testing procedures should be considered. Those testing the software should be encouraged to make written comments on the functionality of the system, the intelligibility of the instructions, and any errors.

5.4.7 User Support and Maintenance

Installation instructions and a user's guide are essential if software is to be useful as well as usable. Installation instructions should provide the name and telephone number of someone to call if help is needed. If the software is going to be maintained, the terms and conditions of support should be included in the guide.

Currently, each staff member of the Advanced Controls Program does his own maintenance on the software he uses (and, in most cases, helped to write). This arrangement will be difficult to maintain if ACP grows in size. As off-site users begin to use the resources and software developed locally, maintenance arrangements will have to be formalized. Such an effort has not been started, but models can be found at the National Magnetic Fusion Energy Computing Center, at the NSF-funded San Diego Supercomputer Center (California), and at the National Center for Supercomputer Applications (University of Illinois).

5.5 DATA BASE MANAGEMENT

Many different kinds of data and text will be needed and used by ACTO on a regular basis. To promote productivity in the use of this information, user interface display software will have to be developed so that retrieved information can be located efficiently and queries constructed conveniently. Sensitive data will be access-controlled. Finally, some documentation of ACTO activities will be required as part of the QA process. Specific groups of data illustrating the range to be covered include:

- modules of source code, both for building simulations of different plant systems and for designing controls;

- graphic symbols for constructing diagrams using computer-aided design software;
- data about specific plant designs, operating plants, and plant control hardware;
- results from benchmark simulation calculations for particular plant designs or operating conditions;
- text data (documents from early phases of a project life cycle) that need to be viewed in a convenient form or analyzed repeatedly; and
- R&D information that supports management tasks.

Data storage, maintenance, and use will vary. Some data will be managed in a structured format (e.g., data for specific hardware components) and some in an unstructured format (e.g., probably most text data). Different hardware will be needed to work effectively with different data. Large amounts of unstructured text can be managed by a massively parallel computer with thousands of processors, while structured data can be managed by DBMS software. If portability of unstructured data is important, then software having a structured query language interface and using the relational model can be used.

Individuals will use most data bases interactively by means of interface software provided with the DBMS. Much of the software will use large amounts of data directly. Interface libraries will be required to handle queries directly through subroutine calls in the application using the data. This type of support is not available for many DBMS products, especially those in widespread use on desktop computer systems. Currently, there is no known commercial DBMS with all the capabilities ACTO activities will require. The multiple systems needed will have to cover all types of data base organization, from flat files to hierarchical and relational systems, and allow for new and different methods. Insofar as possible, efforts will be made to select machine-independent DBMS software.

The integrity of data used within ACTO must be guaranteed. Integrity of data means that nothing will be lost. Care will be required to assure that the original version of a data base can be used upon demand, such as, for example, when simulations require comparisons with previously established benchmarks. Evolutionary development of data bases and the software that uses them must be possible so that newer or more accurate information can be added and more powerful capabilities incorporated. It is impossible, for example, to test adequately modifications to software, especially simulation codes, without the means to compare results with well documented cases. Thus, a large archival data base will be needed, as will software for audit, trace, and data recovery.

Security of data within ACTO must be guaranteed, meaning that data will be available only to those who ought to have it. Companies must be assured that their confidential data cannot be compromised or stolen. ACTO data base management systems will implement necessary protection procedures. Workstation and minicomputer vendors are developing additional capabilities for UNIX operating systems and for windowing environments that permit multiple applications (each sending output to a different

window) to use information from the same data base, but with access privileges properly (and separately) protected. Defense establishment requirements are pushing the development of these capabilities fast enough so that they will be available to the ACP when needed.

Some vendors are now advertising DBM machines that are supposed to provide (and often do for applications that use very specific types of data) rapid access to data. Current DBMs do not appear to be appropriate for the requirements of the ACP.

5.6 COMMENTS ON THE USE OF ADA

Should the ACTO facility be used to support development of advanced reactors for military or space applications, the use of Ada may be required to meet regulations and satisfy directives of DOD or NASA.

Currently, Ada is not used to develop control systems software except in some military applications. This situation will change when large engineering companies such as Boeing decide to use Ada for both military and commercial aircraft. *Information Week*[12] featured Ada in an article enumerating the range of applications for which the language is used and the range of institutions adopting it as the language of choice for new software. Generally, the languages of choice among control engineers (especially nonmilitary) are still FORTRAN and C, with perhaps some use of BASIC and Pascal. Among the ACP staff, the languages of choice are C, FORTRAN, and Lisp. Even though Ada compilers are now available at reasonable cost for PCs and Macintosh computers, only one attempt to use Ada for a small part of a control demonstration has been made. That experience did not show clearly an advantage in using Ada. More examples will have to be constructed to evaluate the role Ada might play in commercial nuclear power plants. The Verdix Ada Development System (VADS), available on I&C's Multimax parallel computer, will be essential for any large-scale development work using Ada.

Proceeding with development of tests of the use of Ada as one of the early activities of the Advanced Controls Program merits special attention. The language was specifically designed for embedded real-time applications, such as data acquisition systems and process control, and for support of software and system testability, reliability, and maintainability. These qualities, along with requirements specific to reactor control systems, argue for considering the use of Ada for digital control system software.

The development of Ada occurred in a manner entirely different from the development of any other language. Specifications for the language were developed and reviewed by many people before any decisions were made about who would write a compiler. After the specifications had been prepared, the contractor for writing the compiler was selected on the basis of a contest of interested vendors. Another unusual aspect of Ada was DOD's decision to register the name so that it has copyright protection. This move allows DOD to control the evolution of the language. The copyright was obtained in part to try to guarantee portability of Ada source code from one machine to another

and in part to prevent proliferation of versions of the language with vendor-added extensions.

Although the estimated number of Ada programmers in this country is only about 600, that number will have to grow dramatically in the next few years to meet the software development schedules of DOD projects for which use of Ada has been mandated. Consequently a shortage of Ada programmers is expected to last well into the next decade.

An active effort will be initiated to use Ada in a range of applications that demonstrates its possibilities and power and to identify those applications for which it is particularly suited. This effort will show the advantages or disadvantages relative to the more familiar and more widely used computer languages. A few well chosen demonstrations should be sufficient to indicate whether Ada is a good language to use for digital control systems.

Features of Ada that are intended to provide software reliability are not all simultaneously available in languages currently used in the Advanced Controls Program. They are listed below, with brief comments.

1. *Strong Data Typing* assists with validation and permits construction of a classification scheme for different types of data. This classification scheme also allows operations to be limited to use only with specified data types. A control system designer can define (abstract) data types that have meaning for his specific system. The abstract data type capability can assist in handling exceptions that occur, for example, when the value of a variable goes out of range.

2. *Data Range Initialization* assists with data validation by raising an exception flag whenever the value of a variable goes out of the defined range. For each exception, the control system designer must provide an alternative to ensure that an action plan is provided for every possible value that can be assigned to a variable.

3. *Record Data Structure* assists in identifying and grouping logically related data elements of *different* types (integer, text, real) into a structure that can be named and henceforth referred to by that name. The array structure, by contrast, is used to group data elements of the *same* type.

4. *Inheritance of Types and Characteristics* allows the use of object-oriented design techniques. General properties of a class of variables can be defined so that a generic object or algorithm is created that can be used repeatedly in different situations with different types of specific data. This powerful feature permits reuse of code.

5. *Data Flow Control* can be used to pass data between subroutines in only one direction that the programmer specifies to prevent data from being modified unintentionally by modules that need to use the data only in a calculation or a test. This feature can be of crucial importance when critical data that must not be altered under any circumstances are used.

6. *Concurrent Processing Constructs* permit writing portable code for use on hosts with different parallel architectures or with different system call libraries (even for hosts with similar architectures). Different parallel computer vendors use different naming conventions and data management schemes for their systems. Ada provides its own names and control constructs for such activities as spawning new processes, erecting synchronization barriers, locking selected data items, and instructing processes to wait.

7. *Generic Program Units* can be written to reduce the amount of code needed to implement a particular algorithm for different data types (integer, real, double precision).

8. *Specification of Precision* assists in checking the validity of sensor data that may be passed to different program units. Cases may occur in which sensor data significant to three digits must be processed in ways that depend on knowing the limits of precision. This information can be incorporated through a provision of the language.

5.7 GRAPHICS

Some of the most dramatic advances in computer technology are occurring in the area of support for graphics capabilities, especially in superworkstations specifically designed to support intensive graphics applications. These capabilities are further stimulating work under way in scientific visualization. With affordable high-performance processors, high-resolution, large color palette display systems, and an extended use of a common windowing environment (X-Windows), sophisticated visual processing of data is possible.

New possibilities for operator displays include an ability to collapse hundreds of square feet of meters, dials, knobs, colored light panels, and switches in current control rooms to two or three 19-in. color monitors with animated schematics of plant systems operator-controlled by a mouse button or an activated spot on the monitor. Automation should reduce the number of components the operator has to control in the new environment. Future displays will be able to make extensive use of color, shapes, movement of objects on a screen, video images of plant components, audio, and other advances that convey the complex streams of data from the sensors to the operators.

New technologies to consider include:

- sophisticated windowing environments that permit watching any desired combination of processes in a side-by-side display (tile format) or in a stack;

- read/write compact disk very high-density information storage hardware;

- video I/O boards that permit manipulation of video images of plant hardware components;

- touch-sensitive display panels for rapid, efficient movement between displays for plant control; and

- 3-D solids modeling and surface rendering (with light and shadow effects).

These capabilities will support a wide range of plant management functions. 3-D solids modeling will be useful for designing procedures for moving large and heavy equipment so as to avoid collisions with surrounding objects, for documenting the plant mechanical design and the as-built facility, and for evaluating proposed modifications to an existing plant. Documentation of the as-built plant using the 3-D solids model data base will be useful in designing and verifying procedures for robotic maintenance of components in high-radiation environments. In presenting large amounts of information to operators, data visualization techniques may suggest new ways that call immediate attention to potential trouble spots during unusual situations or during off-normal operating conditions.

Researchers in the areas of man-machine interface and human factors have virtually unlimited possibilities because software has opened up whole new worlds in design, use, and placement of hardware visual indicators.

5.8 PLANNING FOR NEW SOFTWARE TECHNOLOGIES

Most major scientific and technical computing center administrators do not normally have to consider one area of related issues that ACTO faces: that of hardware and software obsolescence. Power plant control systems must be supported for 30 to 40 years or longer. By contrast, most large-scale scientific experiments and engineering development projects have lifetimes <10 years. Many organizations that acquire large mainframe or superminicomputers use them for 10 to 15 years, then replace them with new systems. In that time frame, new versions of programming languages evolve, new generations of compilers are written, and significant new computer architectures are developed. A comparison of time scales suggests that obsolescence is a potential problem.

Many computer hardware and software vendors are supporting use of published standards instead of their own proprietary technology and adopting the concept of open architecture. This change may even be a necessity in order to support product lines with reasonably useful lifetimes. To achieve this, hardware is upgraded simply by unplugging the old equipment and plugging in the new. Careful thought and attention must be given to the specification and design of subsystem interfaces to support this integrated approach to incorporating new technology. Presumably this same approach can be used for incorporating new technology in control system software.

Many experienced FORTRAN programmers maintain that whatever scientific and engineering programming language is in widespread use a generation from now will still be called FORTRAN. The C language was originally developed in part to permit engineers to do technical computing, with access to system resources at the lowest levels, in an environment of limited resources. Today it is favored as a convenient language for

writing systems software. A growing number of people who develop artificial intelligence programs in one of the AI languages are writing their production versions in C instead of Lisp. Factors that influence the popularity of languages cannot always be foreseen. This is currently illustrated by the very recent growth in popularity of object-oriented design and programming methods accompanied now by a sudden interest in the C^{++} programming language.

The languages that will be in widespread use a generation from now are not certain except for those instances in which the decisions of today's planners lock in a certain set of requirements or specify a path of development for the future. The decision to use Ada for control systems software for the generation of nuclear submarines now on the drawing boards reflects the commitment of the defense establishment to use this language for many decades. These submarines will be used into the next century. Many insurance companies and banks are moving to use Ada. These decisions are important driving forces that guarantee long-term support of this language, and they will be seriously considered in making long-term plans for software development in ACTO.

New software products greatly needed, both for use in software development as well as in the types of software developed, include

- tools for debugging software (especially applications using parallel computers);
- tools for measuring performance;
- natural language processing systems to assist with preparing life cycle documents;
- libraries of features and capabilities for user interfaces that can be used with the same convenience numerical analysis libraries have provided for many years;
- and well engineered user interfaces giving convenient access to the power provided by the tools mentioned in the previous four categories.

Commercial software vendors can provide a range of products to assist with major aspects of the early phases of the software development life cycle. Few, if any, products exist to help with the middle and later phases of ACTO. Help is needed for software testing and evaluation, system integration, and user and maintenance manual preparation.

6. HARDWARE ISSUES

6.1 CONTROLS HARDWARE DESIGN

In the controls hardware design step, the process focuses on the physical implementation of the system in which the major subtasks include equipment specification and selection, wiring drawings, and panel layouts.

Equipment specification converts performance descriptions from the detailed design step and dynamic requirements from the calculation step into a description of the hardware that can perform the step. Features such as the range or capacity required by the design determine the proper commercial product for implementing the design. The ACTO design software will automatically prepare a list of design constraints from the completed design in the preceding step. A commercial product data base will be used to match the system design to available hardware and supply a list of choices to the designer.

At the hardware design step, computer-aided drafting tools will be used to create wiring diagrams, process and instrumentation diagrams, and control panel layouts. Commercial computer-aided drafting programs will be useful tools for creating working drawings for fabricating the control system.

Digital controllers will require the control algorithms to be defined by a detailed design, and calculations and analysis steps must be taken to convert the control actions into the language of the programmable controller hardware. The code generation step is roughly analogous to drawing wiring diagrams in a conventional analog control system. Translation of the completed design into coding is largely a mechanical procedure, which can be automated, thus reducing programming errors and saving thousands of person-hours over the life of a large project. Automatic code generation is realized already in a few commercial programs. For example, an add-on package to $MATRIX_x$ generates FORTRAN or Ada code from the control system functional blocks developed using $MATRIX_x$. A general-purpose automatic code generation utility is planned for ACTO using either $MATRIX_x$ or similar software.

The design of customized computer chips for control system functions is a foreseeable development in control design processes. ORNL will participate in this advancing technology through its VLSI computer chip design and fabrication facilities. CAE software for VLSI use in the ACTO computer facility will be considered at a later date.

6.1.1 Prototype Assembly and Interfacing

The assembly of a prototype of the control system depends to a great extent on the system being developed. Support for this activity will be available from hardware laboratories for assembling the control system. The laboratories will be equipped with standard electronic test equipment and some control system equipment, such as instrumentation racks, power supplies, off-the-shelf controls modules, and programmable logic devices. Currently, there are no plans for the ACTO facility to include shops for

building control panels or for wiring special purpose components; however, such facilities are available at ORNL.

The interface of the control system signals to and from the process simulation will provide a variety of connections and bus structures to suit the many types of commercial electronic control device connectors. Mechanical control devices such as turbine speed controllers and pneumatic controls such as pneumatic relays will be simulated in software.

Software support to automate the connection of simulation variables as inputs to or outputs from the control system hardware can be handled by a computer program.

6.1.2 Guidelines for Specification of Control System Hardware

The hardware technology life cycle period is often comparable to the time required to identify system requirements and produce a control system design. Therefore, it would be a mistake to specify hardware very early in the control system development life cycle; specification should probably be done while design implementation planning is under way. Hardware technology evaluation, assessment, and tracking activities, however, should be ongoing from the beginning and supported by a data base designed and maintained to permit analysis and identification of development trends in hardware technology.

After a careful study of the requirements of hardware necessary for developing control systems and the enhancements available or likely to be available in the near future, specific requirements and guidelines for the design and specification of control system hardware for ACTO will be produced.

6.1.3 Guidelines for Design of Control System Hardware

Support for the design and production of control system hardware based on VLSI technology, as well as expertise in designing and fabricating VLSI smart sensors, will be required. Particular attention will be directed to testing methods and procedures required to ensure high-quality, fault-tolerant designs.

6.2 COMPUTER SELECTION AND APPLICATION

Section 5.2.3 describes a three-tiered hierarchy of supercomputer, superminicomputer, and workstation tied together with different kinds of networks. The section before that touched on some of the features of the X-Windows display interface. Higher-speed network communications software coupled with powerful multiwindowing software will, when combined, blur many of the distinctions made in the past regarding kinds of processor environments and system management practices for the systems in each tier. Methods of computing resource use are changing rapidly and dramatically. In many research organizations, networked workstations are being used as though they were nodes of a hypercube. These networked systems are also proving to be useful for exploring ways to design and use other computer architectures. They will play a

similarly important role in the design of control, safety, and management information systems for advanced nuclear power plants.

The primary computing resources of the Advanced Controls Program are a number of networked workstations and an ENCORE Multimax parallel processor. In applications now being developed, each node of the Multimax will be given a set of instructions (task or process) different from the other nodes. A subtask executing on one node will usually have to communicate with subtasks (or processes) running on other processors. The degree of interprocess communication can vary, depending on the task. Workstations can also be coupled to the processors of the Multimax through networks. Processes running on the workstations communicate with other processes, either those of other workstations or of one of the processors in the Multimax. Processors in the Multimax communicate among themselves on a very high-speed bus, while workstation processors communicate with other processors via the slower Ethernet. Thus, tasks that have to communicate with each other frequently can be assigned processors on the Multimax for speedier processing.

In some applications, the only operationally distinguishable feature in the different types of computers is the difference in message communication speed between an internal bus and Ethernet. The use of different types of computers demonstrates the power and advantages that can be obtained with proper attention to hardware design features that support flexibility and universality, as discussed in Sect. 5.2.4. Figure 3.1 shows the current network architecture of the Advanced Controls Program computers. Figure 6.1 shows future systems that will be acquired and how they will be integrated into the current system.

6.2.1 Workstations and Personal Computers

Many of the features that distinguished workstations from PCs as recently as 1988 are fading as processor chips with 32-bit architectures become widely available for use in PCs and as memory capacities increase with accompanying decreases in cost per megabyte. Multiwindowing capabilities are being introduced into the 32-bit desktop microcomputers. Applications software for these systems is requiring ever more memory to support additional features. Central processor unit (CPU) clock speeds have increased so that, with 5 to 8 Mbytes of memory, at least 140 Mbytes of hard disk storage capacity, a UNIX operating system, and network communications, these systems become engineering workstations.

Advances in VLSI technology are occurring so fast that the first "Cray on a desktop" systems are promised by 1990, barely more than 15 years after the introduction of the first Cray-1S. These systems can now support nearly all the tools needed to develop the simulation models and control systems envisioned for ACTO. With the high-speed networks now available, additional computing power is much easier to use whenever it is needed. As more powerful systems become available, greater choices of configurations become possible. Diskless workstations were introduced to give those who needed the power of the processors in workstations access at a price much lower than that for

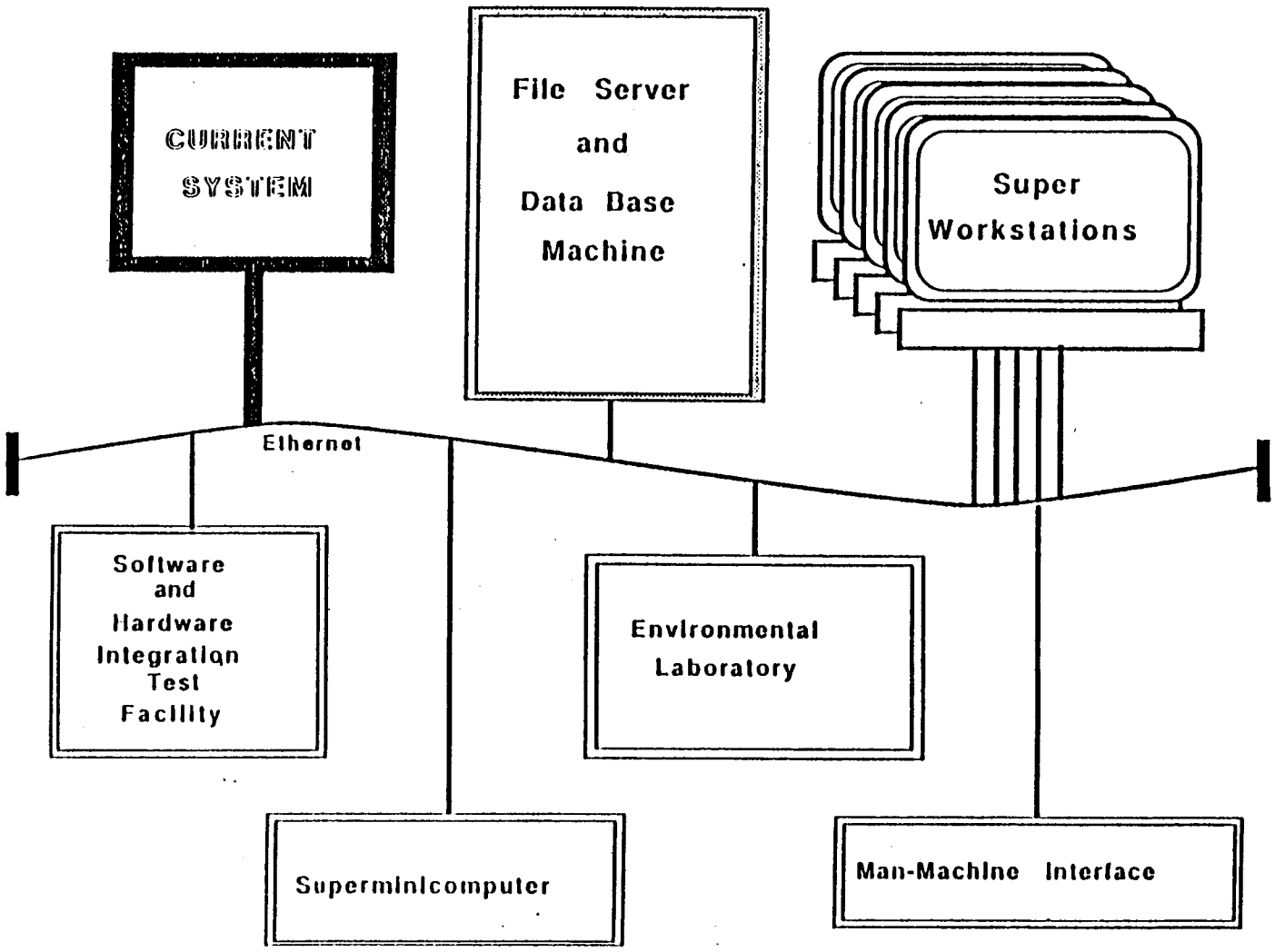

Fig. 6.1. Future systems and network architecture.

systems with disks. Desktop publishing and other aspects of software development can benefit from powerful processors without the need for large amounts of memory or local disk storage. Also, network activity for these activities is not high compared to many other uses of workstations.

The requirement for having boot services (a basic set of instructions loaded into memory to allow a processor to become usefully active) available for diskless workstations has not been satisfactory. Thus, some vendors are supplying another option: fileless workstations. These systems in fact have local files, but only those required to boot one system without having to communicate with another. Once booted, these workstations have to communicate with a file server or external disk to obtain the files needed for specific jobs. This type of system would cost nearly as much as a diskless system, but it could function easily in a heterogeneous environment. Such a system would be well suited to the needs of ACTO.

A new type of machine using a combination of DOS and UNIX operating systems is getting cautious attention by both vendors and users. It provides a bridge between the PC and workstation. Most systems of this type use the Intel 80386 microprocessor chip family. These DOS/UNIX systems are very cost-effective entry-level systems for much of the kind of basic development work of ACTO.

Workstations have an important role to play in real-time applications developments. One of the goals of ACTO is to develop the necessary interfaces and computing environment to support "plug compatibility" between computer and hardware systems simulations. In the simulation, control systems are connected first to one, then to another hardware system in such a way that, if the simulation is a good representation of the hardware, the control system cannot distinguish between systems (see Fig. 5.1 and accompanying discussion). Such a capability will be important for verification, validation, test, evaluation, and qualification of new designs and control strategies. Workstation and PC-based data acquisition and control systems with UNIX-like operating systems are now readily available and will be acquired to set up a capability to support these activities and to achieve the stated compatibility goal. When that setup is complete, moving from the conceptualization and demonstration stages to implementation of a test benchmark hardware control demonstration without having to rewrite large amounts of source code should be easy.

All the machines mentioned above are capable of working well over Ethernet and are sufficiently powerful to handle the X-Windows graphical display interface described previously.

6.2.2 Parallel Processors

A ten-node, tightly-coupled, symmetric, shared-memory parallel computer (ENCORE Multimax 320) is being used for development of simulation models, for execution of parallelized versions of existing simulation codes, and for support of control system design activities. This system is effective as a means of acquainting users with parallel processing. Programs can be moved to this machine from the familiar sequential processing environments in which they were developed, and they can be compiled and

executed essentially as they are; only a few minor changes are required. Execution occurs on a single processor. The programs can then be parallelized in stages, with their performance profiled at each stage to determine where further benefits of parallelization can be made.

The system, which has so far given very good service, has good upgrade potential. It can be upgraded either by adding more processors or by replacing existing processor boards with those having faster processors. The architecture of this system appears to be well suited for the kinds of control systems being investigated.

A very different parallel processor architecture (distributed memory) is available by using the workstations in a properly networked configuration. As a result, there is a large amount of flexibility with current resources for investigating a very wide range of system architecture and design issues.

Another parallel computer system, mentioned in Sect. 5.1.3, uses a single set of instructions running on tens of thousands of processors to operate on many different data items simultaneously (SIMD architecture). This technology is inexpensive, comparable in cost per CPU to PC clones. The large number of processors drives the price up. SIMD, however, is a technology that holds great promise for dealing with very large, unstructured text data bases. It has an important role to play in future ACTO development activities. Areas where its use is expected to be most valuable are in

1. preparing documents for managing the development life cycle of large projects (e.g., software, research reactors, and nuclear power plants),

2. maintaining plant operating procedure manuals (to keep them consistent with NRC guidelines and regulations of other government agencies),

3. monitoring plant operations logs to spot events that could indicate future problems, and

4. managing any aspect of plant operation or large development effort generally in which large amounts of text information must be accessible on short notice or maintained.

6.2.3 Supermini and Other Mainframe Computers

Need for general-purpose supermini or other mainframe computers seems less certain than it did just a year ago. The reason is the promise of new families of powerful processors that are expected to reach the market within another year. These families will permit development of systems, based on parallel architectures, that can surpass the power of the largest Crays at costs below those of the highest-performance mainframe systems. If there is to be a future role for supermini and mainframe systems, it will probably be to provide support for very specific applications, rather than to satisfy general-purpose computing requirements.

7. DEMONSTRATION PROJECTS

Current ACP demonstration activities provide timely developmental work supporting the prototypic designs of control systems for selected aspects of the LMR concept. Early initiation of demonstration projects involving realistic power reactor operation and design problems will enable development and test of the hardware and software needed for control system design life cycle support. With the benefit of this experience, final products are more likely to end up as robust, efficient, and useful systems. The tools developed and tested in the process will become a useful part of the ACTO capability.

Where possible, demonstration projects will incorporate existing reactor systems such as the Experimental Breeder Reactor-II in Idaho Falls, Idaho, the Fast Flux Test Facility in Washington, and research reactors at ORNL. Some demonstrations will be made only on the computer simulators at ORNL and other national laboratories; some will be made on prototypic controllers. These demonstrations will show how the most appropriate state-of-the-art developments in control system theory, automation, artificial intelligence, information management, human-machine interface research and modeling, and computer simulation can be integrated into viable demonstrable control system designs.

The development of expert systems has a significant potential for benefiting reactor availability and safety; however, the development of expert systems *especially* cannot afford to be naive, and our early efforts in these areas involve tie-ins with pragmatic, real-life reactor systems to allow for optimal designs of the development environment software and expert system tools and procedures used in ACTO. Expert advisor demonstration projects should begin with small subsystems, incorporating operator aids and enhanced display systems, and then proceed to more complex problems. Stringent methods must be developed for testing the reliability, robustness, and fault-tolerance of these systems to unexpected circumstances. These tests are ideally suited to ACTO, but they need confirmation by actual tie-ins to real systems.

A major part of the current demonstration effort is invested in feedwater subsystem and balance-of-plant (BOP) control, because that is a major area of concern in the industry. To a large extent, feedwater and BOP problems are common to most reactor types.

Current ACP demonstration projects described below indicate the breadth of automation and new technologies, along with the potential benefits, that ACP is pursuing.

7.1 FEEDWATER CONTROL WITH EXPERT ADVISOR

In an advanced automated control system for the LMR feedwater and steam generator systems, the prototypic control system uses a robust, multivariate, hierarchical strategy. Demonstrations include incorporation of smart sensors into a hierarchical control system and will include an expert advisor system for the operator.

The feedwater train in any steam-producing power plant is a complex system made up of feedwater pumps, valves, feedwater heaters, steam generators, turbines, turbine bypass systems, and a condenser. In U.S. light-water reactors (LWRs), a significant fraction of lost plant availability can be attributed to feedwater system incidents. LWR designs have, for the most part, analog control systems for the feedwater train. These systems are cumbersome, inflexible, and unintelligent; they are already being replaced in some LWRs because of reliability and maintainability problems. Replacement systems are primarily digital versions of the analog proportional-integral-differential (PID) control strategies previously used.

Digital control systems offer possibilities for much improved control performance and reliability. Improved strategies offer increased fault-tolerance, increased robustness, and increased flexibility to accommodate changes in hardware or software. Digital control also allows the use of smart sensors to improve fault-tolerance and robustness. Furthermore, research at ORNL on improved HMIs and AI will lead to more efficient use of operators.

The demonstration also includes a performance monitor to monitor plant readings and trends in readings to identify failures and deteriorating performance in components of the feedwater plant pumps, valves, and heat exchangers. The monitor performs observations and diagnostic analyses that previously were performed cognitively by operators or engineers. It uses tests, such as comparison to expected or previous performance, to identify failures and deterioration in performance of feedwater system components. The tests are then linked by a system of rules that relate an initiating event to the test results.

In the performance monitor and expert advisor, all digital monitoring functions require a primary and at least one confirming test to detect a "failure." Components failing only one test are categorized as "suspect." This design method gives a high level of reliability to the fault detection.

The expert advisor will analyze the information, supply to the operator the most likely causes for anomalies, advise proper corrective actions, and provide information about feedwater system status. The expert advisor provides the operator with the kind of assistance that an expert on the component might give concerning continued operation of a slowly degrading component or the operational alternatives that might allow continued operation with a particular component out of service.

The feedwater control system design employs continuous closed-loop control algorithms for the feedwater plant using both classical and modern design tools. Control loops for the feedwater plant include the level controls for the feedwater heaters, steam generator, condenser, and deaerator and flow controls for pump recirculation valves. Currently, controls for the main steam valves, turbine bypass system, and turbine generator are being added.

The mode selector recognizes changes in mode in the plant and carries out the discrete actions or sequences of actions required to stabilize the plant after a change of mode. The mode selector receives input from the operator, the performance monitor, and the

reactor protection system. It operates by starting pumps, opening valves, or resetting gains or set points in the continuous controllers in response to changes in power or equipment status.

The user (operator) interface displays the results of the demonstrated system on interactive screens. CRT displays are being designed to be similar to screens for a control room environment. The displays show the results of the demonstration and provide a mechanism for the user to control the demonstration. Special screens will also be provided to select scenarios for simulating component failures and to observe diagnostics and corrective actions.

7.2 METAL FUEL INHERENT FEEDBACK IN CONTROL

The proposed LMR concepts in the DOE program feature a metal-fueled reactor core which has significant negative reactivity feedback due to thermomechanical effects. These strong negative feedbacks with increasing temperature offer substantial safety advantages, some of which have already been demonstrated at EBR-II. These feedbacks also provide load-following characteristics that should be considered in the overall control strategy.

Preliminary work at Argonne National Laboratory (ANL) indicates that reactor core designs for EBR-II and for the Rockwell Sodium Advanced Fast Reactor (SAFR) might allow load changes in the balance of plant to be accommodated naturally by the reactors without control rod motion over a major part of the operating range. In fact, this result has been demonstrated at EBR-II for power levels ranging from 50% to 100% of full power. General Electric, however, proposes active use of control rods in load following for its proposed PRISM, which also has a metal-fueled core. The time delay for changes in load at the turbine to propagate to the reactor core are long enough that GE prefers active control rod usage to keep plant conditions within the desired operational constraints.

This demonstration of a control system strategy using the inherent feedback advantages of a metal-fueled core with minimal use of control rods for an ALMR has shown the operational limits of this control philosophy. A full-plant simulation of an ALMR utilizing modular modeling system models developed as part of the ACTO program was used to determine the limits of the "inherent" control range using no control rod motion.

7.3 SUPERVISORY CONTROL OF A MULTIMODULAR LMR

The objective of this task is to demonstrate supervisory control of a multimodular ALMR. The demonstration covers four major areas: nonlinear optimal multivariate control, control of multimodular systems, control action follow-up, and control of uncertain nonlinear systems.

As process complexity grows, the advantages of advanced automated control increase. In a process in which inability to control properly has high cost associated with it, such as

in control of nuclear power plants, any increase in complexity in control is particularly undesirable. One technique for combating complexity is the use of a hierarchical control structure, with each level of control supervising the controllers on the next lower tier of the hierarchy.

Some designs for ALMRs (and other types of reactors) incorporate multiple modules that together produce power to meet grid demand. Some designs carry the idea further to have multiple reactor cores in each module. In such designs, all reactor cores are coordinated to meet the power demand. A chief virtue of multimodular plants is increased flexibility aimed at increased plant availability. If one reactor is shut down for refueling, all others should be able to continue operation. Increased flexibility requires development and demonstration of an appropriate control strategy.

This demonstration is an example of such a hierarchical control strategy for a multimodular ALMR. At the top level of control will be a supervisory controller that determines how grid demand will be met, if possible, by the modules. The level of control under the plant supervisory controller will consist of module controllers. Each module controller will try to meet the power demand of the plant supervisory controller by coordinating multiple reactor cores. This hierarchy will continue to the level of component control. Any controller unable to fulfill the goal set by its supervisor will communicate up through the hierarchy. The supervisor will then try to meet its goal by another method.

At appropriate levels, a nonlinear, multivariate, optimal controller strategy will be used. The strategy to be used, developed as part of this program, transforms a two-point boundary value problem, which is solved off-line, into an initial value problem that can be solved on-line. This strategy allows the controller to follow a demand in the presence of unknown variations of parameters and subsystem responses. A key feature of this algorithm is called parameter tracking. Over the life of a plant, and as a nuclear reactor goes through its normal range of operation, some plant parameters change. The nonlinear control strategy developed has the ability to track changing parameters and continue to optimally control the reactor or reactors.

Advances in AI have opened many areas for innovative solutions to new problems and solutions to problems that were previously unsolvable. Control systems of the future should be able to communicate goals in forms other than numerical values.

7.4 AUTOMATED STARTUP

The scope of this task is the development of software programs, control strategies, and control system philosophies to be tested and placed into operation at EBR-II. The EBR-II staff will provide the necessary reactor facility for demonstrating these advanced control and diagnostics concepts.

The major objective of this work is to show that ALMR plants can be operated from low power to full power using computer control. This task will help solve some of the issues raised and provide proof through demonstration of automatic plant startup

control. The resulting wider acceptance of the concept may allow its incorporation into the ALMR design. A computer system to aid the reactor operator during reactor startup will be designed for placement by ANL into operation in the EBR-II control room. ORNL will implement new software programs on an actual plant and thus take the control and diagnostics software from theoretical and simulated phases to an implementation phase.

Work must be accomplished to implement a computer controlled startup capability on EBR-II in several areas. The initial task is to implement reactor startup checksheets on a computer to provide an initial interface between the reactor operator, the display screens, and the computer workstation, thus rendering a needed service to the operator. This work entails the conversion of the startup procedure, with its manual checksheets, into computer prompts to the operator. Automatic verification of plant signal values will be used. Reactor startup will be performed by manual control actions aided by computer prompts for each step. During approach to full power, the computer will perform most of the data reduction and plotting now performed by the operator. It will issue prompts for hold points while the operator reviews data, takes manual data, or makes decisions to proceed.

Subsequently, ORNL will develop startup control strategies and algorithms. The algorithms will be based on the equipment available and implement the existing startup control strategy.

8. COMPUTER NETWORKING AND INTERFACING CONSIDERATIONS

Data processing and computation for ACTO will require communications between many different machines, some of which will be physically separated by large distances. Some (perhaps many) of these will have different operational characteristics. Indeed, these operational characteristics will probably change over a short time as new computer hardware and software become available. Connecting components so that a local user can fully and easily use the capabilities of ACTO will be most challenging in terms of system administration, that is, making sure that many networking software products from a range of suppliers (commercial and university) will run compatibly on a heterogeneous set of host systems. A seemingly minor problem in a new release of an operating system for one type of host can cause major problems for networks tied to a different host. Such a problem is amplified as the number of communications packages used increases. For example, networks being used currently (connected to the Ethernet backbone) within the I&C Division include DECnet, TCP/IP, NFS, and TOPS.

Difficulty integrating the latest networking technology is expected to lie primarily in the software area. When remote users are considered, hurdles related to satisfying computer security concerns will be added. New procedures and techniques to improve security will be introduced with increasing frequency, reflecting the pace of technological change occurring generally in the areas of hardware, software capabilities, and information management tools. Increased attention that will have to be devoted to network administration will not have to be limited to an overhead activity. As time goes on, experience gained in this area will be available to apply to power plant management issues in areas such as operational reliability and safety and to use with the cumulative experience of the electric utilities industry to respond effectively and rapidly to unexpected or unusual plant behavior. For example:

Several nationwide networks now exist: ARPAnet (using TCP/IP), NSFnet (connecting the NSF-funded supercomputer centers with their users), BITNET, and the National Magnetic Fusion Energy Computing Center network. A new Energy Sciences network (ESnet) is being established to serve several programmatic areas within DOE for the basic energy and health sciences fields. It is not clear now whether future networking support for ACTO should be obtained by arrangements to use one of these existing networks or whether support should be obtained in some other way.

Current requirements for network support include the means to send electronic mail (brief messages and memos), to transfer files (data, source code, technical documents), and to do remote log-ins (in terminal access mode) to host systems at other sites. These capabilities can currently be supported by the use of modems over voice-quality telephone lines or by ARPAnet (TCP/IP). They will continue to be supported for the foreseeable future as the basic level of network access for all processor systems (workstations, file servers, parallel computer, etc.) acquired for ACTO. It is now possible for a remote user to arrange for authorization to have an account set up on one of the local systems. Access to these systems by the remote user should be at least as convenient as access from his home to his employer's computers.

Any more elaborate forms of external network access than that described above will require coordination, planning, and approvals from the Energy Systems Telecommunications external networks coordinator (unless arrangements are made for access to the NMFECC or ESnet network). The formal processes that have been set up to assist with providing communications capacity, while helping to assure reliable and stable network services, also take time to implement. Hence, requirements for this area of support for ACTO should be reviewed frequently, preferably quarterly.

As ACTO is a facility for testing control systems and not a computer research program, the network must be designed to allow users to achieve their control system design and test objectives while minimizing the amount of computer hardware knowledge (and jargon) with which they must deal. Standards for system commands and data transfer structure (protocols) must be set (or adopted) and made readily available and comprehensible to the ACTO user.

Maximum utilization of individual components of the network will require a hierarchical structure. The different levels of the hierarchy will be structured to make best use of computing resources and to provide appropriate isolation. For instance, high-performance vector processor systems will not be used for applications requiring a lot of interactive interrogation of a data base or data collection. Development of some applications will have to be isolated from division Ethernet backbones so that other users will not have their computing activities interrupted. Applications requiring intensive graphical support will have to be confined to local subnetworks so that they do not cause degradation of performance on the Ethernet segments of applications requiring high communications responsiveness.

The ACTO network architecture design will minimize the impact of obsolescence of both computer hardware and software by requiring adherence to published standards. Software (acquired commercially or written locally) necessary to drive the hardware will be designed to be as independent of the hardware as practical. This independence can be achieved by the use of standardized higher-order languages. Good quality software is important for reliable use of networks and for easily incorporating new hardware, new users, and new communication technologies into the network. Network modifications will be made in as transparent a way as possible.

Three areas in which significant changes are occurring and which current planning efforts must consider are

- use of fiber optics,
- support for high-speed graphics, and
- migration to higher-speed data transmission systems.

Data transfer protocols will be specified. Different protocols might be appropriate for different layers of the network. Many different protocols exist, and new protocols will probably evolve. Ethernet, developed by Digital Equipment Corporation, Intel Corporation, and Xerox Corporation, is a popular system for a local network that transfers low-level data at 10 Mbits/s. Transmission Control Protocol/Internet Protocol (TCP/IP) is an intermediate-layer protocol for information transfer between

heterogeneous computers. TCP/IP has been selected for the National Science Foundation (NSF) network connecting the NSF supercomputer centers to universities. Other protocols will be used, such as Sun Microsystems' Network File System (NFS) and DEC's DECnet. Hardware and software dedicated to the specific tasks of transmitting data collection and translating protocol may be necessary.

Network reliability and hardware fault-tolerance are objectives of network design. These are well known sources of lost productivity by (and annoyance for) a user. Most of the effort in these areas is expected to come from the system administrator and will include giving attention to network architecture and to communications software evaluation and testing.

The management of the network, as mentioned earlier in this chapter, will be an important ACTO development activities design consideration. Software and hardware management will include network security and optimization of resources. Network security will include internal requirements, such as access to proprietary plant design and control system data, as well as external requirements, such as restricting network access to authorized users. Optimization of resources, a goal of good management practice, might, for instance, require rerouting of information transfer in case of hardware failure or heavy traffic.

Real-time simulation, when subsecond response time is necessary, may require an entirely separate network. It may require different protocols (e.g., token ring or token-passing vs contention) from a network used for file transfers and data base interrogation in order to provide predictable scheduling of events or to avoid any undesirable interference, such as side effects from other software using the network.

9. ARTIFICIAL INTELLIGENCE IN NUCLEAR PLANT CONTROLS

Use of AI for power plant control applications covers areas that include natural language processing, knowledge engineering, pattern recognition, symbolic processing, and heuristic reasoning.

Typical applications areas in which this discipline is providing help include procedure generation, fault detection, monitoring and diagnosis, sensor validation, analysis of rules, cause-consequence tree analysis, and cognitive modeling of plant operator decision making. Most of these areas have been mentioned in other sections of this report, most notably in the description of the demonstrations being developed.

Expert systems have been found to be a useful AI application for operating and managing nuclear power plants. Table 9.1 lists 32 examples of such systems. The application titles show the range and types of uses being made of expert systems. These systems have been used primarily as off-line aids to plant operations. Integrating these applications and others like them into the plant control and operations environment is one of the current challenges to the nuclear industry and to ACTO.

Integration of natural language processing into the plant control system should include use of qualitative, declarative statements for automation of the plant operation documentation. Time and date stamped records, containing a large amount of numerical data characterizing the plant status, should be augmented with natural language statements using standard terminology to summarize the general condition of various plant subsystems and tied to the expert systems that have been integrated into the control system. A readable history of plant status, generated automatically and capable of supplying a greater amount of context information than a table of numbers, would complement comments entered in the operator's log. The statements generated automatically could be scanned by a text retrieval system and analyzed for content in much the same way numerical data always have been. Proposed here is a logical extension of tasks already being done with expert systems.

Event scheduling is an important development area covering a range of applications. It should be a fruitful area for developing a library of software modules that can be called in much the same way numerical analysis routines are called for simulations. Scheduling is needed in real-time operating systems used by the plant computers for managing nuclear fuel use and storage, for various aspects of plant maintenance, for designing control action sequences (tool support for the control design workstation), and for plant data acquisition and control.

Repetitive tasks of any sort, in any phase of plant operations, should be analyzed for possible automation, particularly those tasks in which monotony and boredom can lead to human error.

Software tools are available to support activities in the areas listed above. Many commercially available expert system products can assist with knowledge engineering

Table 9.1. List of expert system applications used in nuclear power plants

Reactor emergency alarm level monitor
Computerized tracking system for emergency operating procedures
Clones of experts at FFTF
Trip buffer expert system
Technical specification monitor
On-line generator diagnostic system
Motor-operated valve maintenance expert system
Expert systems for training
OECD Halden Reactor project expert systems
Alarm diagnosis and filtering
Alarm filtering at Idaho National Engineering Laboratory
Improving nuclear emergency response with an expert system
Reactor safety assessment system
DYSIS real-time diagnostic/control system
Residual heat removal expert system
Accident diagnosis and prognosis aid
Transient analysis of multiple failure simulations
A real-time operator advisory system
COPILOT, an expert system advisor for nuclear power plants
Handling potentially invalid sensor data
Plant status monitor system
Operational control of PWR cores
Diagnostics using model-based reasoning
Nuclear plant technical specification tracking
Fault tree analysis in expert systems
Use of PRA in expert systems

Source: R. E. Uhrig, in *Advances in Nuclear Science and Engineering*, Plenum Press (in press).

applications. Several among the current resources of ACTO have been used to develop demonstrations showing the capabilities of modern control strategies.

Software utilities to assist with natural language processing are included in the UNIX environment. They can be used to parse and analyze any text as well as commands and higher-level source code.

Pattern recognition support is available in different forms. One of the most recent to get a lot of attention is neural networks. Tools (such as special-purpose processor boards) needed to develop applications in these areas are available, either as part of ACTO computing resources or elsewhere within the I&C Division.

10. DESIGN AND TEST OF HUMAN-MACHINE INTERFACES IN ADVANCED CONTROL SYSTEMS

The human-machine interface (HMI) design and testing task is part of the Advanced Controls Program. The results of this task will provide ACTO with general-purpose capabilities for HMI R&D for ALMR and other reactor designs. Extensions of this work will also lead to the design of facilities for development and test of control consoles and panels.

The primary objective of the cognitive engineering task is to provide a means for explicitly, systematically, and quantitatively incorporating elements of cognitive/behavioral science into issues related to advanced control system designs. The cognitive engineering effort within ACP is a formal recognition of the need to incorporate human cognitive and behavioral performance elements into the designs of advanced control systems. Such recognition allows for more explicit and systematic inclusion of human performance characteristics into the design and development of manned systems. Improved use of human expertise, adaptiveness, and creativity can be expected that will increase system reliability and availability while reducing operating costs.

In the short term, new analysis tools in the form of human performance expert systems and cognitive models of the reactor operator will be developed. These state-of-the-art tools will be used within analyses that currently ignore or make relatively gross assumptions about human performance. These applications will form the basis for an experience base that can be used in the long term. Experience gained from application of the developed tools will be used to achieve a proven approach to higher levels of automation.

The design of advanced liquid metal reactors and advanced control room concepts requires an integrated approach for considering all interdependent elements that compose the system. Control strategies, HMIs, levels of automation, procedures, and control room manning are but a few of the considerations that need to be addressed early in the total system design in order to promote efficiency and effectiveness.

To a considerable degree, the design of reactor systems in the past has been viewed (and carried out) primarily as a hardware design problem. From this traditional viewpoint, a control system is essentially instrumentation and control hardware, and its design is effectively an instrumentation and control engineer's problem. Similarly, elements such as software, staff, and support facilities tend to be considered as ancillary system elements that will be addressed by someone else. This approach depends on the assumption that input from other disciplines can be obtained, when needed, from specialists in those areas. In effect, designers in different disciplines traditionally tend to be "compartmentalized" and have relatively limited interaction/communication until there is a need, much later in the overall design process, to merge all of the independent efforts into a final design.

Cognitive engineering support is being provided for the entire life cycle of advanced control systems. In particular, software tools are being developed to enhance:

1. the preliminary design phase (the identification of feasible design alternatives,

2. the final design phase (evaluation of design alternatives), and

3. test and evaluation of the final design (evaluation of the performance of the chosen design).

Additionally, these tools are being developed for eventual use within the advanced control room as intelligent operator aids. One such tool, the INTEROPS (integrated reactor operator/system) model,[13] was the focus of research and development efforts during FY 1987 and 1988.

10.1 INTEROPS MODEL

The INTEROPS model is being developed within ACP in a framework that combines the capabilities of network simulation and knowledge-based simulation. This framework, successfully employed to develop a prototype version of INTEROPS, is currently being employed to develop the full-scale version. INTEROPS prototype development was completed during FY 1987, and successful completion demonstrated a number of feasibility constructs, including:

1. the ability to link a network simulation model with a reactor plant process code,

2. the ability to have dynamic interaction between the two models, and

3. the ability to link the network simulation model with a knowledge-based model created in an expert system modeling environment in order to promote diagnostic expertise for the simulated operator.

Planned modeling activities include development, test, evaluation, and validation of a full-scale, single-operator, single LMR module version of the INTEROPS model.

10.2 PHASED APPROACH

Cognitive engineering support is being provided for three phases of the control system life cycle: (1) preliminary design, (2) final design, and (3) testing and evaluation.

10.2.1 Preliminary Design Phase

During the preliminary design phase of the life-cycle, control system designers have relatively high-level notions or design philosophies related to an envisioned control system. Expert, high-level advice to designers will aid their formulation of feasible objectives, performance specifications, and functions. Specific cognitive engineering support will be in the form of expert high-level cognitive engineering design guidelines provided through an expert system developed for this purpose. Such a tool will aid in

the specification of system design objectives, system performance specifications, and functional identifications that specifically consider the role of the operator in the system design.

10.2.2 Design Phase

During the design phase of the life-cycle, the functional descriptions of the system are elaborated into a set of potentially feasible design alternatives. In general, design alternatives will include various means of achieving the overall objectives of the system. These may include various types of hardware, software, interfaces, and levels of automation (allocation of function). During the design phase, a number of alternative designs may be generated, and each will be tested and evaluated with respect to the quality with which it meets overall system objectives. Design is an iterative process that eventually leads from a first approximation to a well engineered control system.

Multidisciplinary efforts must be integrated to allow feasible design alternatives to be subjected to simultaneous constraint evaluation. The cognitive engineering support for this phase will include development, test, evaluation, and validation of a human operator model (INTEROPS). In conjunction with other models, INTEROPS will be applied within a workstation environment to aid in the evaluation of various design alternatives within a "total system" perspective, that is, a system that incorporates all active elements including the human operator.

10.2.3 Test and Evaluation Phase

Once a final design alternative has been selected, detailed characteristics of its operation need to be assessed. ACTO will provide the capability for real-time, full-scope, full-plant simulation of the final design concept. During this phase, cognitive engineering support will be provided for assessing the performance of real operators within a real-time, full-scope simulator. Efforts will include support for the development of procedures, selection and training requirements, and training systems. Interface design will be supported through the development of requirements that emphasize

- transparency,
- model-based integration,
- hierarchial integration and data abstraction,
- usefulness and reduction of redundancy, and
- support for the operator's mental model of system structure and function.

The INTEROPS model will be directed toward use as

- part of an expert system advisor to the human operators,
- a team of real and simulated operators for training purposes, and
- an aid for the parametric examination of reactor scenarios that minimize the need for a large number of real human operators.

11. PLANT MAINTENANCE TECHNOLOGIES

Information processing and monitoring capabilities of the control system in the advanced reactor designs suggest expanding the role of the control system to include maintenance activities. Future R&D work within the Advanced Controls Program is planned in the areas of maintenance data base and expert advisor. The demonstration project will illustrate what can be done with a central computer system to automate information handling in nuclear power plant maintenance. Some of the main features planned are:

Computerized Maintenance Procedures. Tag-outs (necessary for plant safety for the highly automated plant), work order preparation, and testing procedures can be tied together in an integrated, computerized data base package.

Preventive/Predictive Maintenance. Schedules for preventive maintenance based on the health of the component can be generated by the performance monitor and expert advisor functions of the control system, resulting in early maintenance if a component is ill or delayed maintenance if it is healthy.

Service and Repair Records. Records of failure and repair automatically recorded in the computerized maintenance procedures can be formatted so that operations and engineering personnel can readily search the maintenance data for their respective information needs.

Tie-ins to National Data Bases of Failure and Servicing Data. Some national data bases for component failure data already exist for LMRs. The purpose of this task is to design interfaces in the maintenance data base to share information with a national network.

Special Software. Special software is needed to assist with updating and maintaining the maintenance data base.

12. CONCLUSIONS

The development of ACTO involves dealing with many different and difficult software and hardware planning issues. For ACTO to become (and remain) useful in the face of the extraordinarily rapid changes in technologies, the tools developed and acquired need to be flexible, robust, and universal. Obsolescence and life-cycle maintenance considerations are primary, especially considering that the life span of a power reactor is typically an order of magnitude greater than the (useful) life spans of computer hardware and software.

ACTO capabilities clearly are needed to support the development, test, and qualification of advanced control system designs in the nuclear industry. More effective use needs to be made of the high technology outside the nuclear industry as well. State-of-the-art advanced software, computer hardware, and networking need to be integrated with reactor design and control data bases so as to provide advanced control designers with an appropriate development environment.

This report has attempted to clarify the need for the R&D associated with ACTO and to present a plan for accomplishing it.

REFERENCES

1. A. D. Alley et al., *Advanced Control Test Operation - Task Team Report*, GEFR-00771, General Electric Co., San Jose, Calif., September 1985.

2. D. Chan et al., *Large-Scale Implementation of Computerized Plant Control and Monitoring in CANDU Power Plants*, EPRI-NP-4769-SR, Palo Alto, Calif., September 1986.

3. E. M. Blake, "Domestic Capacity Factors: Newer Units Join Upward Trend," *Nuclear News*, 32(7), 58-67 (see also "INPO: Performance Indicators Continue to Improve," p. 48).

4. A. D. Alley and W. R. Daniel, *A Systematic Approach to the Design of a Nuclear Power Plant Hierarchical Distributed Digital Control and Protection System*, EPRI-NP-4769-SR, Palo Alto, Calif., September 1986.

5. J. T. Robinson, "An Intelligent Simulation Environment for Control System Design," in *Advances in AI and Simulation: Proceedings of the SCS Multiconference on AI and Simulation*, R. Uttamsingh and A. M. Wildberger (Eds.), The Society for Computer Simulation International, 1989, pp. 28-31.

6. U. S. Department of Defense, Military Standard: Defense System Software Development, DOD-STD-2167A, 1983.

7. Mohammed Jamshidi, University of New Mexico, Albuquerque, 1988.

8. IEC 880, Software for Computers in the Safety Systems of Nuclear Power Stations, Bureau Central de la Commission Electrotechnique Internationale, Geneva, 1986.

9. NUREG/4640-CR, Handbook of Software Quality Assurance Techniques Applicable to the Nuclear Industry, August 1987.

10. ANSI/ANS-7.4.3.2, American National Standard: Application Criteria for Programmable Digital Computer Systems in Safety Systems of Nuclear Power Generating Stations, 1982.

11. N. P. Wilburn, *Software Quality Assurance Techniques Applicable to the Nuclear Industry*, HEDL-2810, Hanford Engineering Development Laboratory, 1986.

12. Diana ben-Aaron, "Ada Goes to Work," *Information Week*, September 8, 1986, p. 35.

13. J. C. Schryver, H. E. Knee, and L. E. Palko, *The Integrated Reactor Operator/System (INTEROPS) Model for Simulation of Advanced Reactor and Control System Concepts: Summary of Prototype Development*, ORNL/TM-10598, Martin Marietta Energy Systems, Inc., Oak Ridge National Laboratory, March 1988.

INTERNAL DISTRIBUTION

1. J. L. Anderson
2-6. S. J. Ball
7. R. E. Battle
8. C. R. Brittain
9. N. E. Clapp, Jr.
10. B. G. Eads
11. C. E. Ford
12. D. N. Fry
13. H. E. Knee
14. J. E. Jones, Jr.
15. R. A. Kisner
16-20. J. K. Munro, Jr.
21. P. J. Otaduy
22. A. W. Renshaw
23. J. T. Robinson
24. L. A. Rovere
25. J. C. Schryver
26. R. L. Shepard
27. B. R. Upadhyaya
28-32. J. D. White
33. T. L. Wilson
34. R. T. Wood
35-36. Laboratory Records
37. Laboratory Records-RC
38. I&C Division Publications Office

EXTERNAL DISTRIBUTION

39. R. J. Neuhold, Director, Division of Advanced Technology Development, NE-462
40. B. J. Rock, Director, Office of Technology Support Programs, NE-46
41. H. A. Alter, Office of Technology Support Programs, NE-46
42. DOE Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831
43. D. Carroll, General Electric Company, Nuclear Systems Technology Operation, P.O. Box 530954, San Jose, CA 95153-5354
44. Y. Dayal, General Electric Company, Nuclear Systems Technology Operation, P.O. Box 530954, San Jose, CA 95153-5354
45. W. K. Wagner, General Electric Company, Nuclear Systems Technology Operation, P.O. Box 530954, San Jose, CA 95153-5354
46. H. P. Planchon, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439
47. J. I. Sackett, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439
48. J. W. Cooke, Energy Programs Division, Department of Energy, Oak Ridge Operations, P.O. Box 2001, Oak Ridge, TN 37831-8610