American Control Conference, San Diego, CA, Session WA-5, 6 June 1984.

AN EXPERT SYSTEM SCENARIO FOR COMPUTER-AIDED CONTROL ENGINEERING

James H. Taylor
General Electric Corporate Research and Development
Schenectady, New York 12345

Dean K. Frederick and John R. James
Rensselaer Polytechnic Institute
Troy, New York 12181

Abstract

We propose using *expert systems* to create third-generation man/machine environments for Computer-Aided Control Engineering (CACE). We believe that this approach promises to provide a high-level design environment that is powerful, supportive, flexible, broad in scope, and readily accessible to non-expert users. This paper focuses on illustrating the capabilities of such an expert system via an extended transaction (sample interaction between the expert system and a user).

I. INTRODUCTION

To establish a context for discussing CACE environments, we must outline our definition of the "control engineering problem." To do this, we have enumerated a set of tasks in Table 1 that we believe comprises the major steps in any comprehensive real-world control system design problem.

Table 1

MAJOR ACTIVITIES IN CONTROL ENGINEERING

1. MODELING THE PLANT
2. DETERMINING THE CHARACTERISTICS OF THE PLANT MODEL
3. MAKING CHANGES IN THE PLANT THAT MIGHT BE REQUIRED TO MAKE IT MORE AMENABLE TO CONTROL (E.G., ADDING SENSORS TO ENSURE OBSERVABILITY)
4. FORMULATING THE ELEMENTS OF THE DESIGN PROBLEM
5. CHECKING TO SEE THAT THE DESIGN PROBLEM IS WELL-POSED; ESPECIALLY, DETERMINING THE REALISM OF DESIGN REQUIREMENTS
6. EXECUTING APPROPRIATE DESIGN PROCEDURES
7. PERFORMING DESIGN TRADEOFFS (IF NECESSARY)
8. VALIDATING THE DESIGN
9. PROVIDING COMPLETE DOCUMENTATION OF THE FINAL DESIGN
10. IMPLEMENTING THE FINAL DESIGN

These activities cover a somewhat broader scope than is generally included in the area commonly referred to as Computer-Aided Control System Design (CACSD); for that reason, we prefer the term CACE despite the fact that we may call this area "design" in the broadest sense of this term.

The development of software for CACE is well advanced at this time, with many powerful sect3control system analysis and design packages available [1 - 4]. There remains a great deal to be done, however, in terms of providing a *powerful, general environment* for the *less-than-expert user.* By the terms "powerful" and "general," we mean a CACE environment that supports the full range of control system design activities listed in Table 1, in a high-level interactive mode, with an adequate variety of design approaches available so as to provide a reasonable assurance of success in solving a broad spectrum of control system design problems. There are several factors involved in the phrase "less-than-expert user:" Such a user may not be aware of the latest theoretical developments in the field, may not have had the experience required to synthesize control theory into an effective design approach, and/or may not be a frequent practitioner of control system design or user of CACE software. In any case, many less-than-expert users find that currently-available CACE software is difficult to use effectively. Specifically, the problems may be outlined as follows:

1. Most CACE packages are specialized to treating only a part of the overall design process, to working in only one "domain" (time or frequency), and often to using one particular synthesis or design approach.
2. The problem-solving environment provided by most current design packages is generally potent but rather low-level.
3. Most packages provide little or no guidance to the user with respect to posing a realistic problem, procedures, design approaches to use, interpretation of results, et cetera.
4. Most packages provide little or no useful documentation of the design process (what was done, why it was done, et cetera).

These considerations may not concern the expert user, who can easily formulate a meaningful design problem, who is well versed in modern multivariable control system design, and who can take advantage of all of the power and subtlety of available control system analysis and design packages. However, for many real-world control system design problems the knowledge, comprehension, and attention to detail required to keep on top of the design process in existing CACE environments will surely frustrate and tax many users, perhaps beyond the limits of their capabilities.

We believe that the problems outlined above are serious in light of the current state-of-the-art in control theory and CACE. In fact, there is good reason to believe that this situation will worsen, due to the following factors:

1. New control system design approaches are constantly being added to the control engineer's repertoire, and software for their application is being developed.
2. More of the control system design problem (Table 1) is being carried out in CACE software environments.

The addition of software for new control system design approaches and to cover more of the scope of CACE will surely make existing environments even less satisfactory.

For the reasons outlined above (and discussed more substantively in [5, 6]), we believe that the need for a better CACE environment is quite clear. As indicated above, we have in mind an environment that deals with the CACE problem in very broad terms, that allows the user to work at a high level, that provides guidance and support to the user, and that provides access to a wide variety of analysis and design methods without the necessity of mastering a large ensemble of low-level commands. This need has led us to consider the use of expert systems as a way to address the deficiencies identified in the preceeding discussion. Because we consider this to be a "third-generation" concept [5, 6], the resulting software environment will be referred to hereafter as CACE-III.

Expert or knowledge-based systems are software environments designed to aid in solving problems that require high levels of *expertise*, some degree of *inference* ("reasoning"), and the use of *heuristics* (nonrigorous procedures or "rules of thumb"). Such problems are generally complicated and broad in scope, and are not amenable to clear-cut well-posed algorithmic solutions. Control system design, in the broad sense outlined in Table 1, is clearly such a task. The types of expertise that are required for CACE are: diagnosis of the plant model, setting up a realistic design problem, selecting appropriate design methods, performing trade-offs, implementing the controller, and using conventional CACE software. Reasoning capability will prove to be highly advantageous for directing and keeping track of the design process as it progresses. Heuristics are certainly a major factor in a human expert's ability to formulate a well-posed design problem. For a more detailed discussion on the subject of expert systems, or for additional background reading in the area, refer to [7, 8]. Alternatively, one may refer to [9] for a brief overview and a more extensive bibliography than that provided here.

In the remainder of this paper, we will overview an architecture for a third-generation environment that is currently under development and illustrate its "performance" with an extended example (sample transaction). We will then close with a status report, summary, observations, and conclusions.

II. AN EXPERT SYSTEM ARCHITECTURE FOR CACE

In this section, we will outline a structure of an expert system for CACE. We have developed this architecture by attempting to "model" the activity of a human expert, and will present it from the same vantage point. For a more detailed discussion, refer to [5, 6].

A central issue in the control system design process is the *complete problem formulation.* This may be represented by a "list of facts" or, in artificial intelligence (AI) terminology, "frame." The information in this frame may be organized or partitioned into three components:

1. *Plant characteristics* — linear or nonlinear, stable or unstable, minimum or nonminimum phase, (un)controllable and/or (un)observable, et cetera.
2. *Constraints* — analog or digital, (de)centralized, data rate limits, et cetera.
3. *Specifications* — rise time, bandwidth, percent overshoot, et cetera; sensitivity, robustness, et cetera.

This *problem frame* is a major focal point for CACE, as can be observed from the fact that the first five functions in Table 1 relate to this information. A major distinguishing attribute of a human expert is the ease with which these data are assembled to obtain a meaningful, workable problem. Therefore, formulating this list of facts to obtain a well-posed problem must be a central concern in CACE-III. This list of facts is established via the interaction of the user and the expert system, as set forth in Section III. We will illustrate the concept of the problem frame list of facts at the end of the transaction presented below, where its structure and meaning will be more transparent.

In the second part of the execution of a control design problem, we believe that the human expert works in a parallel construct, which we call the *solution frame.* In CACE-III, this is a list of facts that is developed as a "scratch pad" where the expert system keeps track of what has been done and what needs to be done, and where information required for decisions about the selection of design procedures and tradeoff analysis resides. Such a frame is not as clearly defined as the problem frame at this point in our study. This lack of clarity is due in large part to the fact that a set of automated design procedures has not yet been established.

The *facts* involved in the problem and solution frame concepts form one key constituent of CACE-III. The other requirement is a *rule base* that mechanizes the functions outlined in Table 1. A complete expert system for CACE must be able to carry out these tasks, and must know how to use conventional CACE software packages in so doing.

The line of thought provided by the above linking of the activities of an expert with two key lists of facts (frames) and an associated rule base has given rise to a complete functional structure of CACE-III that is depicted in Fig. 1. In particular, we have created a construct in which the rule base is partitioned into six parts, as shown. The functions of the rule bases may be summarized as follows:

1. RB1 governs interactions among the *design engineer, plant models* (both nonlinear and linear), and the *model and constraint components of the problem frame.* This rule

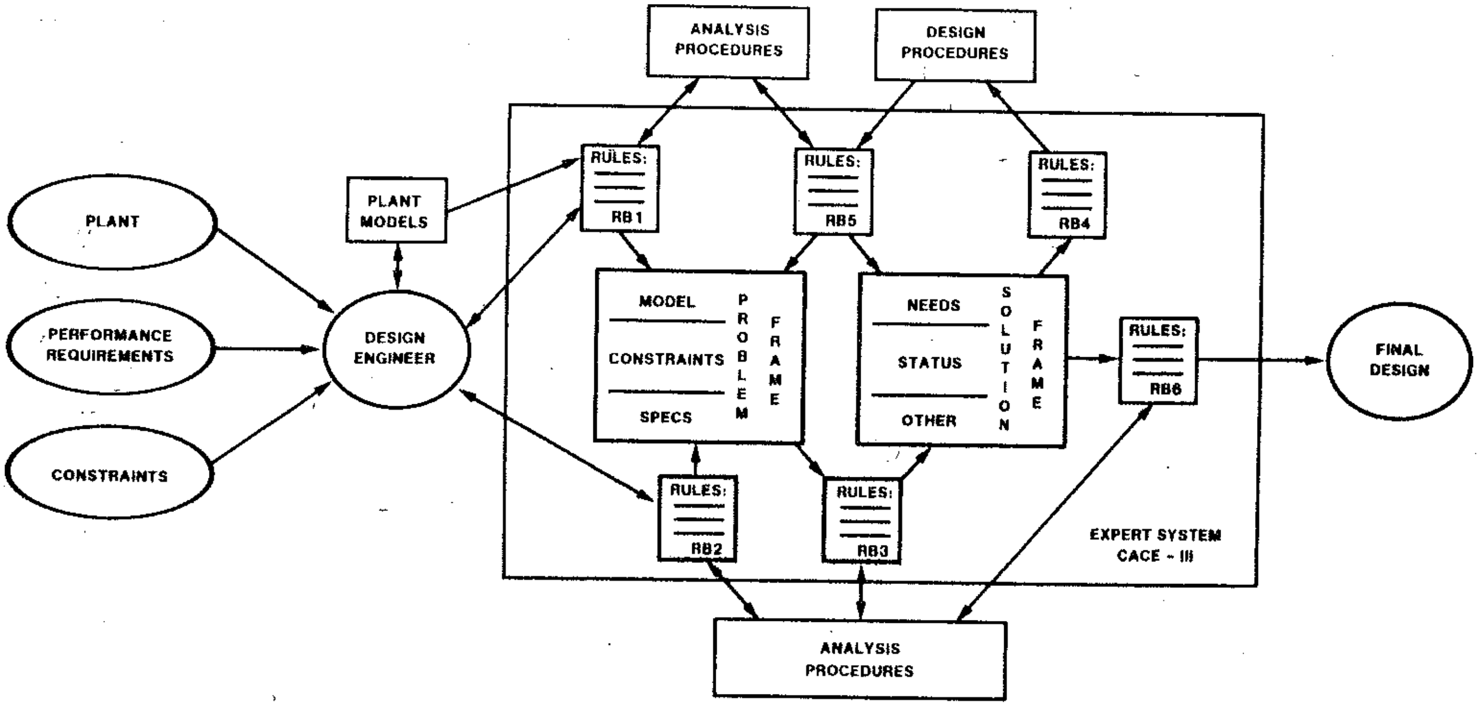

Figure 1. Complete Functional Structure of CACE-III

base provides support in model development (from both controls and numerical analysis points of view), and sees to it that all required plant data are added to the knowledge base. Constraints will be requested but are not mandatory; if supplied, these are also written into the list of facts that makes up the problem frame.

2. RB2 governs interactions between the *design engineer* and the *specification component of the problem frame.* These rules guide the user in entering design specifications and checks specifications for consistency, completeness, and workability (realism).
3. RB3 governs interactions between the *problem frame* and the *solution frame.* Rules in this rule base deal with specifications, constraints, and plant characteristics, and set up a preliminary list of facts in the solution frame describing what needs to be done to achieve design goals.
4. RB4 governs interactions between the *solution frame* and the available *design procedures.* These rules decide what design approach(es) will best solve the problem, and executes the appropriate procedure(s) and algorithm(s).
5. RB5 governs interactions between *design procedures* and the *problem and solution frames.* Every time a design procedure or algorithm is executed, the solution frame must be updated to reflect the corresponding addition/change in the open-loop transfer function. This rule base will also supervise tradeoff analysis by selecting specifications to relax and recommending that the problem frame be modified appropriately if the original specifications cannot be met.
6. RB6 governs the final control system *validation process* and conversion from idealized controller design to *practical implementation.*

The goal of the first two rule bases is to have a well-formulated problem, thus ensuring a reasonable probability of success in the design phase. RB3 serves to initialize the solution frame. Observe that RB4 and RB5 represent an iterative or dynamic "loop;" we pass through these parts of CACE-III until all specifications are satisfied, if possible. Finally, RB6 ensures that the user has a control system that will perform as desired, with as little need as possible for additional engineering for implementation. A more detailed discussion of the functions of each rule base is provided in [5, 6]; much of this will be illustrated in the sample transaction provided in Section III.

Note that these rule bases will invoke conventional CACE software in performing many analysis and design functions. Expertise regarding the use of this software will be "built-in." However, it is important that the user be able to run this software manually, if desired, so that more experienced users can expedite the design process or "steer" it according to their experience or preferences. This need can be met quite readily, as follows:

1. The support provided by CACE-III is in the form of suggestions rather than dictates.
2. A "why" facility can be invoked to determine the basis for any action, including a recommendation, so that the user can judge the advice before deciding whether or not to follow it.
3. Avoiding tedious and well-understood procedures can be accomplished by providing a simple command language that will allow the engineer to enter facts into the knowledge base.
4. The user can be permitted to invoke any underlying conventional CACE package and enter its interactive command-driven mode to do whatever is desired.

These features permit the user to operate CACE-III in "automatic," "semi-automatic" and "manual" modes, to avoid becoming frustrated and bored. The need to be able to influence the design process when the user knows how to proceed (especially if there is disagreement with the approach being taken by the expert system) and to avoid the tedium of being led by the hand through familiar parts of the task (e.g., specification entry - see the transaction below) is very clear. The features outlined above help the user to maintain control of the design process in an effective way, and thus will probably be crucial for the acceptance of CACE-III by the engineering community. Again, much of this is illustrated in the transaction below and discussed more fully in [5, 6].

We are currently undertaking the creation of a demonstration or prototype version of the CACE-III environment using DELTA, an inference engine developed at GE for a diesel-electric locomotive trouble-shooting aid [10]. We have outlined the operation of DELTA to the extent required for tutorial purposes in [5, 6].

III. A SAMPLE CACE-III TRANSACTION

An effective way to manifest the capabilities and behavior of CACE-III is to present a sample transaction. Based on the rule base development to this point in the project, we can hypothesize a quite detailed dialog. Parts of this transaction can actually take place today; portions of it, however, are based on rules and software that we have planned but not fully implemented. We have included as many concepts as we could, to illustrate what we have in mind. Perhaps some of this transaction will strike the reader as being rather simplistic — however, the primary intent of this illustration is to clarify the process involved. Also, most of the dialog (as in previous discussions) is in classical controls terminology, to make the illustration accessible to the largest possible audience. We are not espousing any particular design approach or software by the choice of procedures in this example.

A good demonstration problem for many of our ideas is provided by the task of designing a controller for a nonlinear exothermic chemical reactor model [11] which has two inputs, two outputs, and two states. A block diagram of the closed-loop system is portrayed in Fig. 2. This problem has been treated in an earlier demonstration of conventional CACE software [12], and it will be used as a basis for the sample transaction provided below. The details of this model are not important for the purposes of the present discussion.

NOTE: All CACE-III output/response is in **boldface** in this presentation, while user input is *italicized* and comments are provided parenthetically in normal type.

Welcome to CACE-III

QUERY: Is this a new or continued session [N or C] > *N*

BEGIN CONTROL SYSTEM DESIGN SESSION

QUERY: Is the model LINEAR or
NONLINEAR [L or N] > *N*

CONTROLLER DESIGN FOR A NONLINEAR PLANT MODEL

Enter filename of plant model file > *EXOR*

(Nonlinear model for an exothermic reactor.)

QUERY: Do you wish to SIMULATE or
DIAGNOSE [S or D] > *D*

(The following activities are governed by RB1.)

BEGIN DIAGNOSIS OF THE NONLINEAR PLANT MODEL

(The following section of this dialog represents our concept of a typical diagnosis of a nonlinear plant. Much of the required analysis software exists; this transaction suggests the capabilities of the rule base.)

Enter desired operating value of u,uo > *50. 33.8*

(CACE-III terminology is based on the standard notation: u = input, x = state, y = output. Exothermic reactor inputs are normalized flow rate and inlet temperature.)

Enter approximate equilibrium, xo > *15. 55.*

(This information is needed if multiple equilibria exist. The designer should have a rough idea.)

Enter approx input range, ± du (du = u - uo) > *3. 9.*

(This information is needed in order to diagnose the degree of nonlinearity.)

(CACE-III now finds the equilibrium for $u_1 = 50.$, $u_2 = 33.8$, and determines the linearized model about that point

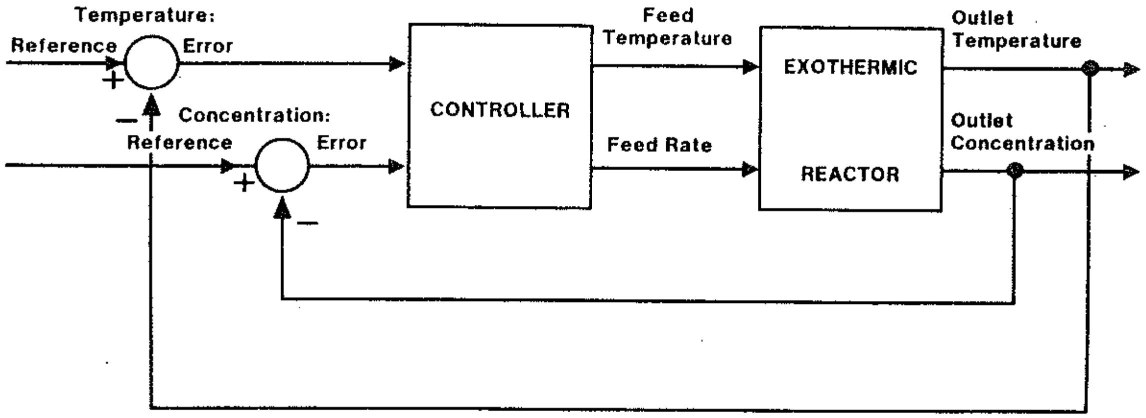

Figure 2. Exothermic Reactor Control System Configuration

[13]. It finds the eigenvalues of that model and prepares data for entry in the problem frame and a report to the user.)

(Based on the eigenvalues, CACE-III chooses ω_o and tsim and simulates both linear and nonlinear models for $o < t <$ tsim with $u_1 = 50 + 3 \sin \omega_o$, $u_2 = 33.8 + 9 \sin \omega_o t$ where the input amplitudes are given by du requested above and ω_o is the geometric mean of the linearized plant eigenvalues or a value requested from the user. CACE-III then obtains the Fourier coefficients and distortion information needed to prepare a report on the degree of nonlinearity. Based on these analyses, the following report is made:)

DIAGNOSIS OF NONLINEAR PLANT EXOR:

I. Quantitative results:

- **a. Equilibrium corresponding to specified u0:**
  xo = 9.98, 49.01
- **b. Fourier analysis results for input du * sin (2.0t):**
  maximum percent second harmonic: 0.02
  maximum percent third harmonic: 3.21
- **c. Linearized model eigenvalues at u0, x0:**
  $s_1 = 0.56 + 1.23j$, $s_2 = 0.56 - 1.23j$
- **d. Linearized model distortion analysis results for step inputs of amplitude du:**
  maximum percent rms distortion: 1.36

II. Qualitative results:

- **a. The EXOR system model has no discontinuous nonlinearities over the input range specified.**
- **b. The EXOR system model is only mildly nonlinear over the input range specified.**
- **c. The linearized model is quite accurate over the input range specified.**
- **d. The EXOR system linearized model is unstable.**
- **e. The EXOR system linearized model is minimum-phase.**
- **f. The EXOR system linearized model is controllable and observable.**

END OF DIAGNOSIS.

The linearized model has been written to file EXORL.DAT

QUERY: Do you wish further diagnostics {Y or N} > *Why*

(The user can enter "Why" at any time to obtain an explanation of CACE-III statements or actions.)

WHY/HELP facility: Specify item number > *II.c*

> **Both the nonlinear model and the linearized model have been simulated with step inputs of amplitude du. The linearized model is judged to be adequately accurate as a basis for controller design if the maximum percent rms distortion of these comparative simulations is less than 5 %. RULE RB1-0414.**

QUERY: Do you wish further diagnostics {Y or N} > *N*

Enter CONSTRAIN or SPECIFY {C or S} > *S*

BEGIN CONTROL SYSTEM SPECIFICATION DEVELOPMENT

(At this point in the transaction, RB2 takes control.)

WARNING: Without constraints the design you realize may be idealistic.

(The following dialog represents an extension of an existing specification entry procedure to allow the development of specifications for multi-input/multi-output plants.)

MENUS TO ENTER SPECIFICATIONS

Menu for sensor type

1. **Continuous-time.**
2. **Discrete-time.**
3. **Mixed**
4. **Terminate session. ... SELECT:** *1*

Menu for controller typ:

1. **Continuous-time.**
2. **Discrete-time.**
3. **Mixed**
4. **Return to preceding menu. ... SELECT:** *1*

Menu for control system structure

1. **Decoupled (diagonally dominant).**
2. **Decentralized controller**
3. **Hierarchical**
4. **Unconstrained**
5. **Return to preceding menu. ... SELECT:** *1*

Menu for defining I/O pairs

1. **Direct user select.**
2. **User select after seeing Nyquist array.**
3. **CACE-III advice based on diagonal dominance.**
4. **CACE-III diagonal dominance algorithm.**
5. **Return to preceding menu. ... SELECT:** *1*

Enter first I/O pair > *u1 y1*

Second I/O pair {u2 y2} inferred.

SPECIFICATION DEVELOPMENT FOR FIRST I/O CHANNEL

Menu for type of continuous-time spec

1. **Locations of poles in s-plane.**
2. **Time response.**
3. **Frequency response.**
4. **No more specs to enter.**
5. **Return to preceding menu. ... SELECT:** *1*

Menu for type of s-plane spec

1. **Maximum real part of s.**
2. **Minimum damping ratio.**
3. **Value of dominant pole {Re,Im}.**
4. **Damping ratio & natural freq. of dominant poles.**
5. **Return to preceding menu. ... SELECT:** *1*

Enter maximum real part of closed loop poles > *-5.*

WARNING: Your spec for max real-part of closed-loop poles is not reasonable. It should not be less than -1.4 .

QUERY: Do you wish to modify? {Y or N} > *Why*

A pole-placement algorithm has been applied to the linearized model of the plant, and it was unable to place the dominant poles to the left of -1.4 without excessive gains based on the plant input operating range you specified. RULE RB2-1022.

QUERY: Do you wish to modify? {Y or N} > *0*

Enter maximum real part of closed loop poles > *-1.4*

Your specification is reasonable.

Menu for type of continuous-time spec

1. **Locations of poles in s-plane.**
2. **Time response.**
3. **Frequency response.**
4. **Structural, e.g., no. of leads.**
5. **No more specs to enter.**
6. **Return to preceding menu. ... SELECT:** *1*

Menu for type of s-plane spec

1. **Maximum real part of s.**
2. **Minimum damping ratio.**
3. **Value of dominant pole {Re,Im}.**
4. **Damping ratio & natural freq. of dominant poles.**
5. **Return to preceding menu. ... SELECT:** *2*

Enter min damping ratio of c.l. poles > *1.0*

Your specification is reasonable.

Menu for type of continuous-time spec

1. **Locations of poles in s-plane.**
2. **Time response.**
3. **Frequency response.**
4. **Structural, e.g., no. of leads.**
5. **No more specs to enter.**
6. **Return to preceding menu. ... SELECT:** *2*

Menu for type of time response spec

1. **Rise time**
2. **Percent overshoot**
3. **Settling time**
4. **Steady-state error**
5. **Return to preceding menu. ... SELECT:** *4*

Your system is type zero.

Enter percent steady-state error > *0.25*

Your specification is reasonable.

You have now entered a set of specifications for Channel 1 that appears to be complete, consistent, and realistic.

QUERY: Do you wish to modify or add specs? {Y or N} > *N*

You may enter separate specs for the second I/O channel, or use the same specs as entered for Channel 1.

QUERY: Do you want new specs for Channel 2? {Y or N} > *Man*

(The user has asked to enter Channel 2 specifications manually rather than repeat the menu-driven procedure illustrated above.)

Enter Channel 2 specs manually >

>> MAX-STEP-SS-ERR CH2-VALUE 0.5
>> MAX-REAL-PART CH2-VALUE -1.4
>> MIN-DAMPING-RATIO CH2-VALUE 1.0

You have now entered a set of specifications for Channel 2 that appears to be complete, consistent, and realistic.

Enter specification tolerance (percent) > *10.0*

Enter CONSTRAIN or DESIGN {C or D} > *D*

BEGIN CONTROL SYSTEM DESIGN

(RB3 initializes the solution frame list of facts; CACE-III control passes to RB4 and RB5.)

WARNING: Without constraints the design you realize may be idealistic.

(The following dialog represents our concept of a typical design session. We are presently using the Cambridge Linear Analysis and Design Package (CLADP, [1]) as the basis for design, which for the multi-input/multi-output case uses the algorithms found in [14].)

Your specifications suggest using the high-frequency align (HFA) algorithm to ensure diagonal dominance at high frequencies. Recommended frequency is 5.60 rps.

QUERY: Do you wish to use HFA? {Y or N} > *Y*

Enter HFA frequency > *5.0*

HFA compensator design completed.

QUERY: Do you wish to see HFA compensator? {Y or N} > *Y*

HFA @ 5.0 rps - precompensator gain matrix:

525.91	**-0.58132**
159.89	**100.79**

Specifications have not been met satisfactorily. HFA achieved:

HFA @ 5.0 rps - CL poles:

-4.93
-5.03

HFA @ 5.0 rps - Steady-state error (percent):

-2.71 %
4.14 %

Remaining requirements suggest using the approximately commutative controller (ACC) design algorithm to ensure small steady-state error. Based on the characteristic loci, the recommended ACC frequency is 0.85 rps.

QUERY: Do you wish to use ACC? {Y or N} > *Y*

Enter ACC frequency > *1.0*

> **In order to meet the steady-state error spec, it is recommended that a lag compensator be designed, with low-frequency gain 10.8 and center frequency 0.32 rps.**

Enter lag compensator low-freq gain > *10.0*

Enter lag compensator center frequency > *0.3*

Approximately commutative controller design completed.

Specifications have been met satisfactorily. HFA + ACC achieved:

> **HFA @ 5.0 RPS, ACC @ 1.0 rps - CL poles:**
>
> **-1.30**
> **-1.39**
> **-3.63**
> **-3.83**
>
> **HFA @ 5.0 RPS, ACC @ 1.0 rps - Steady-state error:**
>
> **-0.271 %**
> **0.414 %**

QUERY: Do you wish to modify? {Y or N} > *N*

END OF CONTROL SYSTEM DESIGN

> (CACE-III control passes to RB6.)

QUERY: Do you wish to see response plots? {Y or N} > *N*

> (We would naturally expect to answer 'Y', but for the sake of space, we will refrain. The validation procedure we foresee implementing would include simulation of the continuous-time controller with the linearized plant model, then with the non-linear model; then discretization of the controller and simulation of the digital controller with the nonlinear plant would follow. This last simulation would proceed from a simple discrete-time model of the controller to a rigorous implementation that took word length and computation time into account.)
>
> **QUERY: Do wish a microprocessor implementation? {Y or N} >** *Y*
>
> **Enter microprocessor type >** *8086*
>
> **IMPLEMENTATION COMPLETE**
>
> **QUERY: Do wish a controller emulation? {Y or N} >** *Y*
>
> **QUERY: Do wish to download controller code? {Y or N} >** *Y*
>
> (Based on the current state-of-the-art, the last three steps may be fanciful in most circumstances, but this is a possible - and highly desirable - outcome that completes the "control system design problem" suggested in Table 1.)

The status of a session, or the outcome of a session when completed, is embodied in the list of facts that has been written in the course of the transaction. Such a list is illustrated in Table 2, which depicts the problem frame or list of facts that existed in the CACE-III knowledge base at the end of a system DIAGNOSE and SPECIFY session. These facts are an intermediate outcome of the transaction presented above; a reading of the transaction will provide a clear explanation and interpretation of this list of facts, so we will not comment on them further here. It should be observed that the underlying *data*, e.g., the numerical results of the nonlinear system diagnosis and the linearized model and its diagnosis, are contained in ancillary files. The expert system does not use this type of data directly, but it must know where such information can be found so that it can be provided to external analysis and design procedures as required.

Table 2

THE PROBLEM FRAME AFTER THE DIAGNOSIS AND SPECIFICATION PORTION OF THE TRANSACTION

FACT	BASIS
PLANT MODEL NONLINEAR	User
PLANT-NL-MODEL FNAME EXOR	User
PLANT-NL-MODEL TIME-TYPE CONT	Inferred
PLANT-NL-MODEL STATE-TYPE CONT	Inferred
PLANT-NL-MODEL ORDER 2	Inferred
PLANT-NL-MODEL INPUTS 2	Inferred
PLANT-NL-MODEL OUTPUTS 2	Inferred
PLANT-NL-MODEL DIAG-FNAME EXORND	Inferred
PLANT-NL-MODEL NL-BEHAVIOR MILD	Inferred
PLANT-L-MODEL FNAME EXORL	Inferred
PLANT-L-MODEL STABLE NO	Inferred
PLANT-L-MODEL CONTROLLABLE YES	Inferred
PLANT-L-MODEL OBSERVABLE YES	Inferred
PLANT-L-MODEL MINIMUM-PHASE YES	Inferred
MODEL DIAGNOSIS DONE	Inferred
SENSOR TIME-TYPE CONTINUOUS	User
CONTROLLER TIME-TYPE CONTINUOUS	User
CONTROLLER STRUCTURE DIAG-DOM	User
CONTROLLER CHANNEL1-IN U1	User
CONTROLLER CHANNEL1-OUT Y1	User
CONTROLLER CHANNEL2-IN U2	Inferred
CONTROLLER CHANNEL2-OUT Y2	Inferred
MAX-STEP-SS-ERR CH1-VALUE 0.25	User
MAX-REAL-PART CH1-VALUE -1.4	User
MIN-DAMPING-RATIO CH1-VALUE 1.0	User
MAX-STEP-SS-ERR CH2-VALUE 0.5	User
MAX-REAL-PART CH2-VALUE -1.4	User
MIN-DAMPING-RATIO CH2-VALUE 1.0	User
SPEC-TOLERANCE PERCENT-VALUE 10.0	User
CONTINUOUS-SPEC ENTRY DONE	Inferred
CONTINUOUS-SPEC ENTRY REALISTIC	Inferred
CONTINUOUS-SPEC ENTRY COMPLETE	Inferred
CONTINUOUS-SPEC ENTRY CONSISTENT	Inferred
SPEC-SESSION TERMINATION NORMAL	Inferred

IV. STATUS

We have thus far developed the the following items:

1. an architecture for CACE-III, including a specific, detailed outline of the functional characteristics of the expert system and of the rule base (Fig. 1; [5, 6]);
2. a detailed "transaction" or dialog between CACE-III and a user, to provide a more tangible "bottom-up" basis for detailing the functions of the various rule bases shown in Fig. 1;
3. working rule bases for several functions, including an automatic control system design procedure (lead-lag compensator design for a single-input/single-output

plant) and some portions of specification development assistance and plant model diagnosis (linear and nonlinear); and

4. analysis routines to implement several nonlinear system diagnostic functions.

Only item 2 is dealt with in detail in this article.

V. SUMMARY, OBSERVATIONS, AND CONCLUSIONS

A. *Summary*

The project we have described in this report is still in the concept development phase. At this point, we have created a tangible top-level description of an expert system for CACE; there is clearly a great deal of detail that needs to be filled in and implemented before we have even a prototype "real system." In the course of implementation, it is very likely that the concepts we have outlined will continue to be refined and evolve as we understand the problem more completely. We are making an in-depth presentation here and in [5, 6] of our research to date, because the application of artificial intelligence concepts to the "control system design problem" (as we have broadly defined it) has not heretofore been addressed, and because we hope that the ideas set forth are sufficiently advanced to stimulate further discussion and development.

Despite the relative infancy of this effort, there are several areas in which we believe significant progress have been made:

1. elucidating the "CACE Problem" (starting with [13] and following through Section I) and identifying substantial unmet needs of less-than-expert users;
2. showing how expert systems may be used to provide meaningful solutions to the problems identified in Section I;
3. developing a detailed architecture (Fig. 1), based on lists of facts (problem and solution frames), rule bases, and conventional CACE analysis and design software, that can support the user in the core activities of CACE outlined in Table 1;
4. developing a concept that is flexible enough that it can fulfill the needs and desires of novice and expert users alike;
5. implementing enough of the rule base to be able to provide a very detailed picture of the capabilities of CACE-III (Section III); and
6. determining some additional requirements of an inference engine for CACE-III (Section II; see [5, 6] for more detail).

B. *Observations*

We have formed a number of opinions during our research in developing an expert system for CACE that we would like to state informally as "observations":

1. Few (if any) existing CACE packages cover the full scope of the control system design problem and support the range of techniques that most users need or would like to have available.
2. Most existing broad-scope CACE software is difficult for the less-than-expert and/or less-than-everyday user to master.
3. Critical areas of user support required by non-expert users are the following: developing and diagnosing a useful plant model, developing meaningful specifications, selecting design approach(es), performing trade-off studies, validating the design, and implementing it. These services are not provided by existing CACE software environments to the extent required.
4. An expert system for CACE should be conceived with the following primary goals:
   a. expediting and removing as much drudgery as possible from the design process,
   b. reducing the probability of error,
   c. allowing the less-than-expert user to obtain better designs,
   d. adding better discipline and documentation to the design process, and
   e. enhancing rather than replacing engineering skill and judgment.
5. An expert system for CACE should benefit an engineer in another, higher-level way, by facilitating and teaching the use of new control system design methods ("technology transfer"). This can only happen if the user can take full or partial control of the design process.

C. *Conclusions*

There are a number of conclusions we have reached in the course of this study that directly pertain to CACE:

1. The application of expert system concepts would seem to be a natural choice for providing the critical areas of user support identified above which require inference, judgment, and heuristics.
2. Expert system concepts also show great promise in creating CACE environments that are flexible enough to meet the needs of a very broad spectrum of control engineers - from novice to expert designers, from infrequent to every-day users.
3. No foreseeable expert system is going to solve *every* control design problem; the case for creating such an environment must be made on the basis of observations 4 and 5 above.

We have also reached three conclusions which we believe have implications that go well beyond the field of CACE, namely:

1. Expert/knowledge-based systems show great potential for providing the basis for a vastly improved man/machine environment *for engineering design in general* [5, 6].
2. The "systems approach" (by which we mean the train of thought outlined in [5, 6] wherein the design process is modeled and reduced to a structure that can be implemented in software) *provides a powerful methodology for creating the structure ("architecture") of an expert system.*

3. *Transactions* (Section III) *can play a major role in developing an expert system,* forming a direct basis for rule base development.

Acknowledgements

The research outlined in this report owes a great deal to the suggestions and efforts of a number of people aside from the authors. The idea of applying artificial intelligence to the broadly-defined problem of CACE was an outgrowth of discussions between JHT and Professor A. G. J. MacFarlane of Cambridge University in the Fall of 1982. A major factor in the resulting research was the expertise and expert system software provided by Dr. Piero Bonissone of the General Electric Corporate R & D Knowledge Based Systems Program. Other substantial contributions were made by students from Rensselaer Polytechnic Institute, as follows: Neal Lassinger modified existing CACE software to make it possible to mechanize an automatic lag/lead compensator design procedure, and Ralph Quan helped to develop nonlinear system diagnostic software (for the analysis of harmonics and distortion). The ideas presented in this report have profitted substantially from these interactions.

References

1. Special Issue on Computer-Aided Design of Control Systems, IEEE *Control Systems Magazine,* Vol. 2, No. 4, December, 1982.
2. D. K. Frederick, "Computer Programs for the Simulation and Design of Control Systems," Arab School on Science and Technology, Bloudan, Syria, September, 1981. (To appear in *Advances in Control Systems, Theory and Application,* Edited by J. B. Cruz, Jr., JAI Press, September, 1984.)
3. Proceedings of the Second IFAC Symposium *CAD of Multivariable Technological Systems,* Purdue University, West Lafayette, IN, September, 1982.
4. *CACSD '83 Abstracts,* IEEE Control System Society Symposium on CACSD, Cambridge, MA, September, 1983.
5. J. H. Taylor and D. K. Frederick, "An Expert System Architecture for Computer-Aided Control Engineering," submitted to *Proceedings of the IEEE,* March 1984.
6. J. H. Taylor and D. K. Frederick, "Expert System Concepts for Computer-Aided Control Engineering," General Electric Report (in preparation), GE Corporate Research and Development, April, 1984.)
7. D. Michie, *Introductory Readings in Expert Systems,* Gordon & Breach, London, 1982.
8. F. Hayes-Roth, D. Waterman, and D. B. Lenat, *Building Expert Systems,* Addison Wesley, Reading, MA, 1983.
9. W. B. Gevarter, "Expert Systems: Limited but Powerful," IEEE *Spectrum,* August, 1983.
10. H. E. Johnson and P. P. Bonissone, "Expert System for Diesel Electric Locomotive Repair," *Journal of FORTH Applications and Research,* Vol. 1, No. 1, September, 1983.
11. M. Athans, M. L. Dertouzos, R. N. Spann and S. J. Mason, *Systems, Networks, and Computation: Multivariable Methods,* pp. 411-413, McGraw-Hill, New York, 1974.
12. J. H. Taylor, "Nonlinear Functional Aspects of CACSD," *IEEE Control System Society Workshop on CACSD* (published as a compilation of viewgraph hardcopy), Berkeley, CA, April, 1982.
13. J. H. Taylor, "Environment and Methods for Computer-Aided Control Systems Design for Nonlinear Plants," Proceedings of the Second IFAC Symposium *CAD of Multivariable Technological Systems,* Purdue University, West Lafayette, IN, September, 1982. (Issued as General Electric Report No. 82CRD193, GE Corporate Research and Development, August, 1982.)
14. A. G. J. MacFarlane and B. Kouvaritakis, "A Design Technique for Linear Multivariable Feedback Systems," *International J. of Control,* Vol. 25, pp. 837-874, 1977.