

An Intelligent Front End for Control System Implementation

James H. Taylor & Pavol Šereš
Department of Electrical Engineering
University of New Brunswick
Fredericton, NB CANADA E3B 5A3
Internet: jtaylor@unb.ca

Abstract

We present a new expert-system “front end” or Design Advisor for Implementing Systems (DAIS) for use in conjunction with a commercial digital control system environment, e.g., the Elsas Bailey INFI 90 System. The objective of DAIS is to make it substantially easier for applications engineers to make effective use of the broad spectrum of capabilities of this and similar hardware and software systems for industrial controls implementation. This concept is of quite general applicability for industrial controls implementation environments.

1 INTRODUCTION

One of the main goals of computer-aided control engineering (CACE) is to facilitate the design and implementation of control systems for practical applications. While control-theoretic considerations are important, they do not provide all the answers needed by field engineers in carrying out this task. This leaves a substantial gap between the capabilities of well-known control-theoretic software environments such as MATLAB [1] and MATRIX_X [2] and more practical problems associated with choosing algorithms, tuning them, and implementing systems. We emphasize that there is little or no gap between the systems that can be implemented on systems such as the INFI 90¹ and those that can be designed using modern control theory and packages such as MATLAB and MATRIX_X – what is missing is support for more down-to-earth concerns such as those mentioned above.

The specific difficulty faced by both Elsas Bailey and their customers is that many line engineers lack the knowledge and experience to take full advantage of the advanced capabilities of industrial control systems equipment such as their INFI 90 system. From a vendor's perspective, it seems that many customers exploit

only a small percentage of the algorithms available; from the customers' perspective, either they are buying a system that seems to provide a lot of unnecessary functionality, or else there is a frustration that they can't take advantage of functionality that they need but cannot use effectively.

Based on these considerations, we decided to create a Design Advisor for Implementing Systems (DAIS), using an “expert-aided CACE” paradigm [3, 4] and a suitable CACE environment that combines numerical and symbolic evaluations, namely Pang's expert-system framework and software MEDAL [5, 6]. More specifically, DAIS has been conceived to elicit a definition of the problem (e.g., simplified or more detailed form of process model, qualitative and/or quantitative performance objectives, constraints), and then recommend a solution or outline a decision-making process so that the user could reach his/her own conclusions. The “solution” is in the form of one or more controller designs, with simulated performance plots and support for implementing it using the INFI 90. It presently incorporates rules of thumb for determining controller type and simple tuning rules for parameterizing the compensator (e.g., see [7, 8]), as illustrated in the examples that follow; in the next phase, we plan to add similar but extended approaches for multivariable systems (e.g., [9]) and expert-aided strategies for autotuning and autosynthesis (e.g., [10, 11, 12, 13]).

This project has reached the end of its first prototyping phase. The “alpha” version of DAIS provides the basic functionality outlined above. In the next phase, the knowledge base for this system will be heavily based on Elsas Bailey's expertise, obtained both from the organization that creates and implements algorithms for use in the INFI 90 and that which provides customer applications support. In addition, the INFI 90 documentation and application guides will be a valuable source of control design and implementation knowledge. The system (alpha and beta versions) will also be tested at the University in conjunction with an industrial control

¹We would like to acknowledge the recent gift of an INFI 90 system from Elsas Bailey (Canada) Inc., Dartmouth, Nova Scotia.

systems course, where DAIS will suggest how to design a controller, and, in the associated laboratory, where the students will be faced with a realistic industrial problem, i.e., "here is a process, here are some requirements and specifications, here is the Elsas Bailey hardware and software for implementation; use DAIS to solve the problem". During this next phase, we will design and build the "beta" version that will be more integrated with the INFI 90 and also tested in the field.

2 GENERAL CONCEPT

The preceding discussion motivates the usefulness of a **Design Advisor for Implementing Systems (DAIS)** for the Elsas Bailey INFI 90 and other digital control implementation environments of similar class. The following conceptual outline provides the basis for this "front end" environment. DAIS is designed to support the industrial applications engineer in the following areas:

1. Definition of plant characteristics (description of the process to be controlled)
2. Definition of performance objectives for the controlled process
3. Definition of implementation, operational and other constraints
4. Selection of control scheme(s)
5. Design / tuning of controller(s)

These activities form the core of computer-aided control engineering (CACE), and involve problem-solving approaches that combine knowledge of both theory and "heuristics" or experience-based "rules of thumb". In light of this, it has been observed that artificial intelligence (AI) can provide useful contributions in "diagnosing the plant model, setting up a realistic design problem, selecting appropriate design methods, performing trade-offs, validating the design, implementing the controller, and using conventional CACE software" [3]. This is because: "Heuristics are certainly a major factor in a human expert's ability to formulate a well-posed design problem." (same citation).

A high-level description of this idea will be outlined using two approaches: working through a partial "scenario" using DAIS, and then sketching the knowledge and decision-making framework of the environment. This paper will then conclude with a discussion of implementation plans and approach.

2.1 Preliminary DAIS Scenario

DAIS has been implemented using a simple graphical user interface that gives the applications engineer rea-

sonable flexibility in carrying out the activities enumerated above. One direct way to proceed in developing this idea is to display some screens that arise in a scenario using such a system:

In Figure 1 (top half)² we see a typical start-up screen from the DAIS system. It provides the primary menu, supporting the five areas of activity enumerated above, plus a standard set of options stop, skip, and why. In the future, our conceptual design includes Library (access to a database of previous applications), Archive (support for inserting the present application into the Library), and Document (support for documentation of the present application). To begin work, a user would select a menu option and proceed.

Next we assume that the user wishes to execute Step 1 in the process, definition of plant characteristics; responding to the first menu with a 1 produces the second menu shown in Fig. 1, which supports several exact and approximate approaches for accomplishing this crucial step. At present, DAIS supports entering a plant model in transfer function or state-space form, or supplying less detailed plant characteristics in either the time- or frequency domain, or providing input/output data for model identification.

The most practically-oriented option (especially for process industry applications) is Option 3, "Semi-quantitative time-response form". This raises the screen depicted in Fig. 2, which illustrates a direct approach for capturing simple (minimal) characteristics that should suffice for selection of more basic control schemes and their design (parameter tuning). As a preliminary measure, we suggest that frequency-response and time-response data are natural choices for these more qualitative aspects. More comprehensive support, and access to more sophisticated control schemes, may be available if the user supplies a more detailed plant model by choosing Options 1 or 2 on the screen for defining plant characteristics (bottom half of Fig. 1).

The display produced by Option 3, as depicted in Fig. 2, simply represents a set of common response characteristics with an arbitrary parameterization. If the user selects "Possible Step Response 1" (a first-order lag with time delay), for example, then DAIS will follow up by requesting estimates for the delay time, rise time (after delay) and steady-state gain. This will complete DAIS's "internal model" of the process, which in turn will influence future suggestions for control scheme and parameter settings, as discussed below.

Elicitation of performance objectives and operational and other constraints (if any) is accomplished in a similar fashion, working down from Steps 2 and 3 on the

²Figures 1-5 can be found at the end of the paper.

first menu (Fig. 1) and using a similar menu/screen-based interface. The user is then ready to progress to control scheme selection (Step 4) and tuning (Step 5).

If the applications engineer has supplied a suitable problem definition (plant characterization, performance specification, constraints), then the rule-based system (RBS) will select a suitable control scheme or set of candidate schemes and support the user in selecting one (if more than one candidate exists) and completing the design (e.g., tuning controller parameters). If the problem specification is not adequate for controller selection and design, then DAIS will provide guidance on how to rectify the situation (see Section 2.2 for further detail).

Once a control scheme is selected, the design/tuning step can be undertaken. The exact nature of this part of the process will depend on the detail and quality of the internal model elicited from the user. (We are considering asking for some rough uncertainty measure in addition to the parameterization outlined above; in the simple scenarios implemented so far (e.g., Fig. 2) this has not been a pressing need.)

To continue the example from Figs. 1 and 2, and assuming that the user provided the following information:

- parameters for the qualitative model 1 are: delay time = 6 sec, rise time (after delay) = 4 sec and steady-state gain = 10.5;
- desired closed-loop 1% settling time (after delay) is 5 sec and tolerable % overshoot is 0%: and
- the constraint on the input to the plant is that u should not exceed 0.24 "units" in magnitude,

the Smith Controller is recommended (Block 160 in the INFI 90) and the parameters are determined by a simple tuning algorithm (first three lines in Fig. 3). Using the internal plant model and parameterized Smith controller, simulated step-response plots are generated; in this case a warning is raised that the constraint is violated (continuation of Fig. 3) and the step-response plots are displayed as in Fig. 4. A recommendation is made that the user relax the settling time specification to 7 seconds (bottom of Fig. 3); of course, the user is free to take that advice or make any other change to the problem definition that might alleviate this problem.

Once the preliminary controller behavior is displayed, as in Fig. 4, the user is allowed to tune the controller by perturbing its parameters by either $\pm 40\%$ ("coarse tuning") or $\pm 10\%$ ("fine tuning"); comparative step-response plots are displayed for the user to select the final design.

That completes an illustrative scenario using DAIS. We also provide a number of other options for defining the internal plant model, as mentioned; here we merely illustrate the use of model identification from user-supplied

step-response data in Fig. 5. Our approach uses a heuristic scan of the step-response data to determine the time delay and then the usual least-squares data fitting via matrix pseudo-inverse to obtain the time constant and steady-state gain.

2.2 Preliminary DAIS Framework

The knowledge and decision-making framework of DAIS is organized along the lines suggested in [3]. The basic idea is that there are two foci of information, called the *Problem Frame* (PF) and the *Solution Frame* (SF). The information in the PF is to be elicited from the applications engineer, as suggested in Section 2.1, or derived from information thus supplied. At the present time, the contents of the SF are dictated by the functions and capabilities of the INFI 90 system, and the corresponding problem-definition information needed to apply these functions and capabilities effectively.

The information in the PF is gathered through the straight-forward use of menus and screens. Once the user signals that problem-solving is to commence (by choosing Option 4 on the main menu, Fig. 1), this data is processed to see if it is an adequate problem definition, and if so what implementation options are available. If we think of the information in the PF as being stored in "slots", then each INFI 90 function or capability can be represented in the SF by a "template" defining the PF information needed (or slots that must be filled appropriately) to permit its application. The more basic control algorithms such as PID may have a few basic information slots specified in their templates; more advanced schemes or functions may have more extensive PF data requirements (templates). Each template, then, would be comprised of a set of slot labels specifying the PF information that must be provided for a successful application of the corresponding function or capability. Additional factors, such as ranges for data in the PF may also be included in the template; these would capture quantifiable constraints such as "if the controller order must be less than 4 then H-infinity control is not available".

The decision-making in DAIS thus involves several components:

1. checking that the data in the PF is consistent and well-posed, e.g., seeing if the performance requirements are sensible given the plant characteristics specified, seeing that the constraints and performance requirements are not incompatible, etc., and
2. matching the entries in the PF with the templates specified in the SF - if a template is satisfied by the user's information in the PF, then the corresponding function or capability is said to be "available".

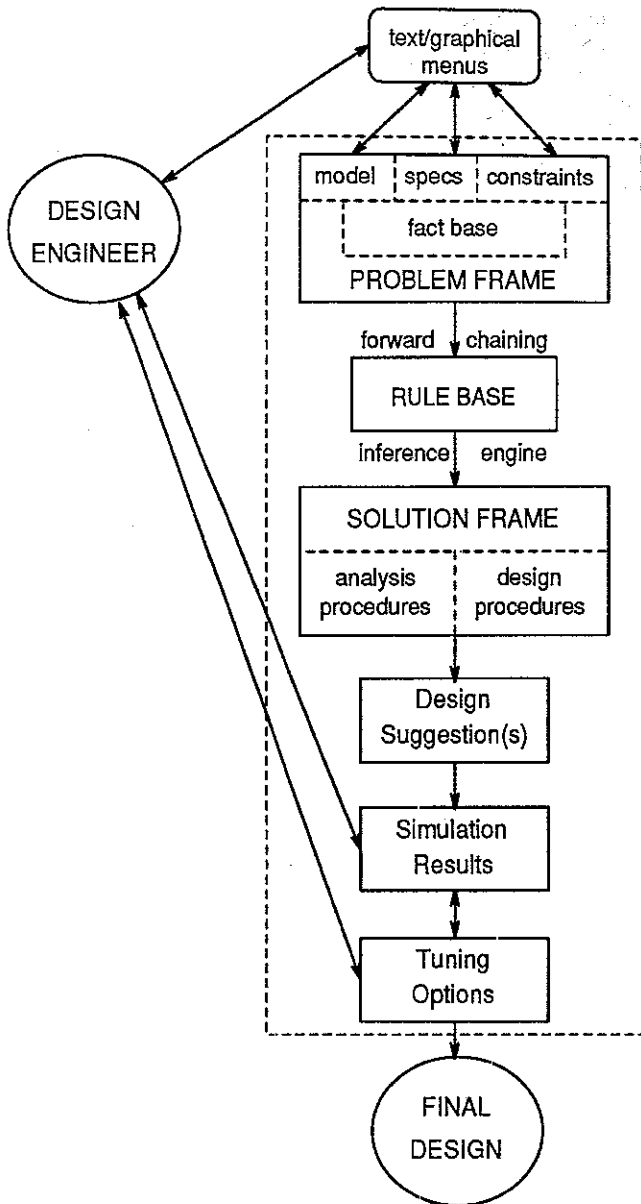


Figure 6: DAIS Knowledge Framework

In the alpha version of DAIS these features are quite limited. However, it is our belief that the system should not stop with a simplistic 'yes' or 'no' assessment with respect to availability; rather it would be far more helpful if the reasoning system would also check templates that are *nearly* satisfied and advise the user that additional functions or capabilities would be available if further information were provided or if the plant characteristics, performance requirements and/or constraints were modified slightly. DAIS should be able to tell the user how to obtain and supply additional information or make modifications in many instances.

Based on these ideas, the DAIS framework can be depicted as shown in Fig. 6, which is a simplified version of that from [3]. In describing this scheme, we speak

of our plan for the next phase of development; the alpha version of DAIS represents a useful and illustrative subset of the concepts that follow.

It is assumed that the design engineer has available a "plant model", either in detail (e.g., a transfer function or ABCD model) or in simple/qualitative terms (e.g., standard metrics such as DC gain, delay-time and significant time constant). The user will interact with DAIS via a graphical interface as suggested in Figs. 1 and 2 to get the appropriate plant model "slots" in the PF filled in. If higher-level model information is supplied, then DAIS will infer the lower-level metrics, e.g., qualitative model parameterizations as described in Section 2.1. The outcome of this dialog is an *internal model* which will be used as the basis for design, simulation and tuning. Upon completion of the plant model component of the PF, the user proceeds to constraint and specification entry, which is done in a similar manner, via various menus.

At this stage the RBS is activated to start the process of comparing the filled slots of the PF with the templates in the Solution Frame (SF). First, DAIS runs some quick heuristics to check if closed-loop specifications are reasonable. The final word is left to the user; however, should the user choose contradictory or unachievable specifications in light of the plant model and constraints, then all DAIS can do is make a few checks and issue warnings. At the end of this step, the PF is ready for the initiation of control scheme selection and design. Then, any templates that are immediately satisfied are designated "available". Other templates which are not completely satisfied are tagged with the designations of the PF slots which correspond to missing data or unsatisfied requirements. If there are available controllers, then the user is supported in selecting one; in either case (whether or not there are available controllers), it will be possible to request help in increasing the number of available controllers by revisiting the Problem Frame.

The process of adding to the list of available controllers would proceed as follows: the RBS would work through the "unavailable" controllers' missing-data tags, to select those that appear to be easiest to satisfy. The result of this analysis is a prioritized list of PF slots to try to fill, which is used to initiate a dialog with the user to increase the number of "available" controllers in a systematic manner. The interaction with the user to accomplish this job might take several steps. An important part of this process would be reasoned requests for additional information, e.g.,

- "If you can supply a higher-order transfer function model you will be in a position to apply <list of additional control schemes>."
- "If you can state that the plant is stable you will

be able to ...”

- “If you can state that the plant is not conditionally stable you will ...”

Once a control scheme has been selected, the RBS attempts to evaluate controller parameters to achieve satisfaction of the performance criteria and constraints. This involves execution of known tuning strategies where they are conveniently available, or making suggestions to the user where automatic tuning is not feasible. The latter interactions might entail suggesting parameter values, telling the user what process to carry out (e.g., inspect Bode plots to determine various characteristics), and outlining what to look for in the results.

Finally, after the controller scheme and parameters were designed, simulation results are provided using the internal plant model. At this point the user may either accept the controller suggested by DAIS or run Option 5: Tuning of the controller (Fig.1). Once the controller structure and parameters are finalized, an auxiliary rulebase is run, to transform the controller gains into INFI 90 parameters.

2.3 Knowledge Representation and Processing in DAIS

A rule-based system approach provides the best structure for implementing DAIS. The preceding discussion illustrates its power and applicability in addressing the problem at hand. Such an environment should be capable of both forward and backward chaining modes of inference. Forward chaining would be an effective way to fill PF slots, especially where a single piece of information (e.g., a plant transfer function) may be used to establish a number of lower-level facts (e.g., to determine steady-state gain, that a plant is stable, that it has right-half-plane zeroes, etc.). Backward chaining would be an effective way to manipulate information in the SF, e.g., check the “Fact Template” for a control scheme, see if the facts are established or, if not, determine if they can be established from available information, etc. This is not an absolute requirement, however; there are approaches available to solve the problem using either forward or backward chaining alone, as we have done with MEDAL.

3 BUILDING DAIS

Based on this preliminary development effort, there arise several major areas of focus to be considered in building the next, more comprehensive version of DAIS:

- Knowledge capture: We need to develop a more comprehensive knowledge engineering plan that includes further UNB input and (most importantly) experience from Elsas Bailey and other “ultimate consumers” (commercial users of the INFI 90). As a starting point, we have a large amount of documentation from Elsas Bailey on the INFI 90, including detailed application bulletins that will allow us to proceed well beyond the high-level “alpha” version presented here.
- “Narrow-slice” development: Several local companies have expressed interest in working to add a meaningful depth of capabilities for their application areas. These include a heating, ventilating and air-conditioning contractor, a power system consulting company and a paper plant. Creating capabilities for carefully limited scenarios will provide a deeper understanding of the problem and the requirements of the end user community.
- “User friendliness”: We will continue to take utmost care to make DAIS flexible and easy to use. Primary concerns are: minimizing to the greatest extent possible the need to “bother” the user for information, suggesting alternative approaches but not dictating a course of action, and providing more direct connections to implementation using the INFI 90 (e.g., writing the parameters and configuration files directly; compare with Fig. 3).

4 SUMMARY AND CONCLUSION

Our intention in creating an “intelligent front end” for the Elsas Bailey system is to increase the efficiency and satisfaction of industrial controls engineers in using powerful, state-of-the-art control systems implementation environments such as the INFI 90. The effort in developing the present “alpha” version of DAIS represents a feasibility study and an attempt to realize our concepts to serve as the basis for the next stages of development, as outlined above. Therefore, the details are tentative and subject to revision as we work in collaboration with Elsas Bailey personnel and their customers to understand requirements and strategies for implementing such a system.

We do believe, based on the concepts outlined above, that the benefits of DAIS will be substantial. From an industrial point of view, making it easier for an applications engineer to employ a broader range of INFI 90’s capabilities has a clear and significant pay-off. From an academic standpoint, the research is important in its own right (advancing the state of the art in computer-

aided engineering of control systems), and the potential for using DAIS in classroom and laboratory settings to bring industrial approaches and solutions to the fore is an additional attraction.

5 REFERENCES

- [1] MATLAB *User's Guide*, The MathWorks, Inc., Natick, MA 01760.
- [2] MATRIXx *User's Guide*, Integrated Systems, Inc., Santa Clara, CA.
- [3] J. H. Taylor and D. K. Frederick, "An Expert System Architecture for Computer-Aided Control Engineering" (invited), *Proceedings of the IEEE*, Vol. 72, December 1984.
- [4] J. H. Taylor, "Expert-Aided Environments for CAE of Control Systems", Plenary Lecture, *Proc. CADCS '88 (Fourth IFAC Symposium of CAD of Control Systems)*, Beijing, PR China, 23 August 1988.
- [5] G. K. H. Pang, "An Intelligent Front End for a Control System Design and Analysis Package", *Proc. CADCS '88 (Fourth IFAC Symposium of CAD of Control Systems)*, Beijing, PR China, 23 August 1988.
- [6] G. K. H. Pang, "A Matrix and Expert System Development Aid Language", *Proc. CACSD'92, IEEE Computer-Aided Control Systems Design Conference*, Napa, CA, pp. 218-224, March 17-19, 1992.
- [7] K. Åström, *Ziegler-Nichols Auto-Tuners*, Department of Automatic Control, Lund Institute of Technology, May 1982.
- [8] K. Åström and B. Wittenmark, *Computer Controlled Systems - Theory and Design*, Prentice Hall, 1984.
- [9] S. Engell and R. Müller, "Fast and Efficient Selection of Control Structures", *Proc. ESCAPE-1, Computational Chemical Engineering, Vol. 16*, pp. 157-164, 1992; "Multivariable Controller Design by Frequency Response Approximation", *Proc. ECC*, Groningen, June 1993.
- [10] D. W. Clarke and P. J. Gawthrop, "Implementation and Application of Microprocessor-Based Self-Tuners", *Automatica*, Vol. 17, 233, 1981.
- [11] K. J. Åström and T. Hägglund, "Automatic Tuning of Simple Regulators", *Proc. IFAC 9th World Congress*, Budapest, Hungary, 1984.
- [12] J. H. Taylor and K. J. Åström, "A Nonlinear PID Autotuning Algorithm", *Proc. American Control Conference*, Seattle, WA, 18-20 June 1986.
- [13] J. H. Taylor and P. G. Stringer, "An Auto-Synthesizing Nonlinear Control System Using a Rule-Based Expert System" (invited paper), *International Journal of Adaptive Control and Signal Processing*, Vol. 5, No. 1, January 1991.

```

Welcome to DAIS, a Design Advisor for Implementing Systems on the INEI-90 (R)
-----
      University of New Brunswick, Fredericton NB Canada
      P. Seres and J.H. Taylor - v 0.5 alpha - 18 January 1996
-----
DAIS' expert system environment supports the industrial controls engineer in:
  1. Definition of plant characteristics
  2. Definition of performance objectives
  3. Definition of implementation and other constraints
  4. Selection of control algorithm
  5. Tuning of controller
  6. stop
  7. skip
  8. why

Answer: 1
You may provide plant model information in the following formats:
  1. Transfer function
  2. State-space
  3. Semi-quantitative time-response
  4. Semi-quantitative frequency-response (Nyquist)
  5. Semi-quantitative frequency-response (Bode)
  6. Plant identification from step response
  7. stop
  8. skip
  9. why

Answer: 

```

Figure 1: DAIS' Start-up Menu

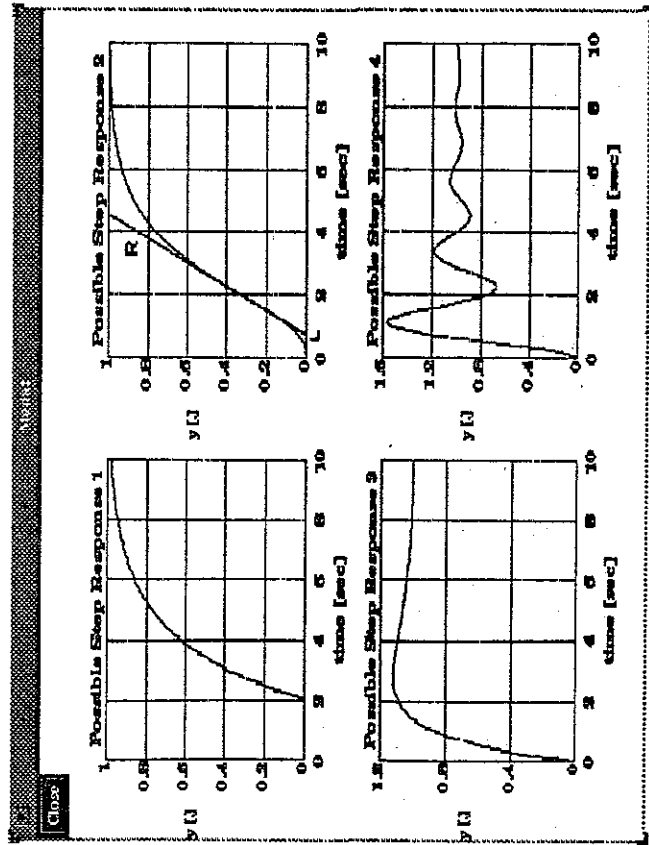


Figure 2: Screen for Picking Plant Time-Response Characteristics

```

Use a Smith Controller Block #160:
Set S7 to 10.6072, S8 to 6.0000, S9 to 3.7632 and S10 to 1.000
0
Inference engine ran to completion.
<MEDAL>
<MEDAL>
<MEDAL>
<MEDAL>
I am plotting time response data for Smith Predictor
Plotting ...
Plotting ...
Plotting ...
You are exceeding your constraints.
I suggest that you modify your specifications.
Try to increase settling time to: 7.0000
  
```

Figure 3: Screen with Controller Recommendation

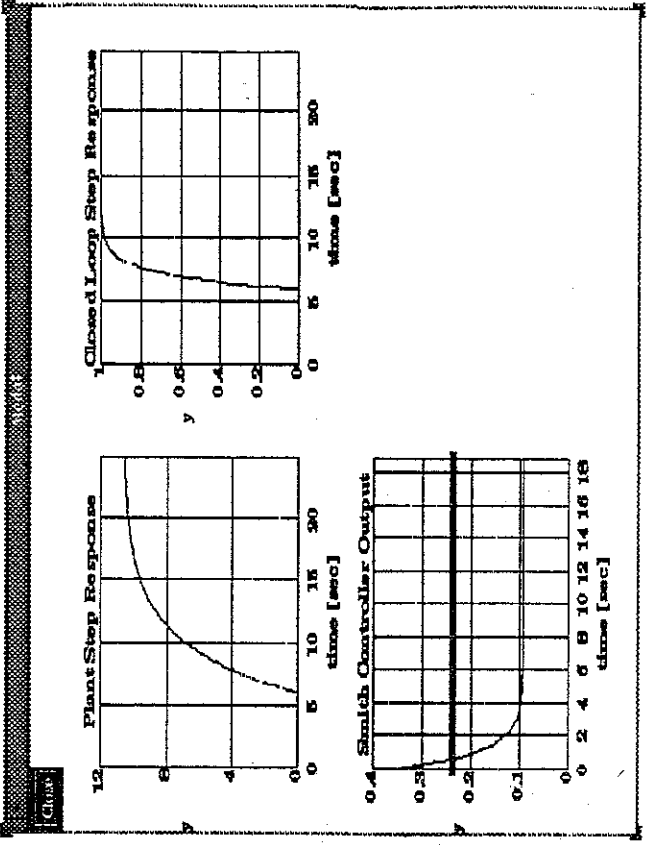


Figure 4: Predicted Closed-loop Time-Responses

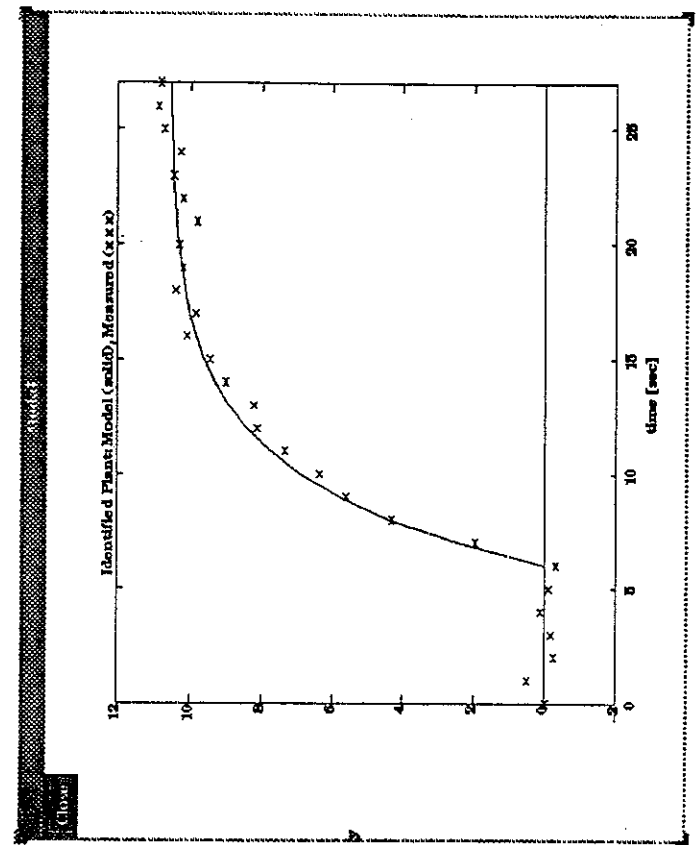


Figure 5: Model Identification from Step-Response Data