An Expert-Aided Implementation Interface for Industrial Process Control Systems

James H. Taylor & Cheney Chan
Department of Electrical & Computer Engineering
University of New Brunswick
PO Box 4400
Fredericton, NB CANADA E3B 5A3
email: jtaylor@unb.ca & l2gu@unb.ca

Abstract: We present the design and implementation of 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 Elsag 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 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 chosing algorithms, tuning them, and implementing systems. We emphasize that there is little or no gap between the systems that can be implemented on a modern distributed control system (DCS) such as the INFI 90[1] 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 outlined here.

The specific difficulty faced by both DCS vendors 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 the 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 believe they need but cannot use effectively.

Based on these considerations, we decided to create a Design Advisor for Implementing

[1]We would like to acknowledge the recent gift of an INFI 90 system from Elsag Bailey (Canada) Inc., Burlington, Ontario.

Systems (DAIS), using an "expert-aided CACE" paradigm [3, 4, 5] and a suitable CACE environment that combines numerical and symbolic evaluations, namely Pang's expert-system framework and software MEDAL [6, 7]. More specifically, DAIS was 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 implementation using the INFI 90. A preliminary prototype [8] incorporated rules of thumb for determining controller type and simple tuning rules for parameterizing the compensator (e.g., see [9, 10]); in the current phase, we are adding similar but extended approaches for multivariable systems (e.g., FASTER [11, 12]) and expert-aided strategies for autotuning and autosynthesis (e.g., [13, 16, 17]). In addition, we are implementing a more seamless interface between DAIS and the INFI 90 and demonstrating it using real industrial process control equipment (a liquid flow, pressure and temperature control laboratory test stand built from commercial components contributed by Honeywell Canada). Finally, we will beta test the DAIS concept and implementation with selected industrial customers, starting in the summer of 1999.

The prototyping phase of this project is complete [8]. The "alpha" version of DAIS provides the basic functionality outlined above. In the next phase, the knowledge base for this system is being heavily based on Elsag 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 are valuable sources of control design and implementation knowledge. The system (alpha and beta versions) is also being tested at the University in conjunction with an industrial control systems course, where DAIS suggests how to design a controller using MATLAB, and, in the associated laboratory, where students are faced with a realistic industrial problem, i.e., "here is a process, here are some requirements and specifications, here is the Elsag Bailey hardware and software for implementation; use DAIS to solve the problem". At the end of the current phase, we will have refined, built and tested a substantially more functional "beta" version of DAIS and applied it in the field.

2 General Concept

The preceding discussion motivates the usefulness of a **Design Advisor for Implementing Systems** (DAIS) for the Elsag Bailey INFI 90 and other digital control implementation environments of a 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)
6. Implementation of the control system

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). From these considerations, the use of a rule-based expert system is a natural choice for implementing DAIS. Pang's expert-system package MEDAL [6, 7] is particularly well suited, since it seamlessly integrates a rule-based expert system for analysis and design heuristics with MATLAB-like numerical capabilities.

A high-level description of this idea is outlined below using two approaches: working through a partial "scenario" using DAIS, and then sketching the knowledge and decision-making framework of the environment. The scenario presented deals with a typical single-input / single-output (SISO) case, for simplicity, with significant extensions representing beta-version capabilities. This paper will then conclude with a more detailed discussion of implementation plans and approach.

2.1 Illustrative DAIS scenario

DAIS has been implemented using a simple graphical user interface that gives the applications engineer reasonable 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:

The DAIS system start-up screen is depicted in Fig. 1. It provides the primary menu, supporting the six areas of activity enumerated above, plus a standard set of options **stop**, **skip**, and **why**, with obvious meaning. To begin work, a user would select a menu item and proceed.

Normally, a user will begin by executing 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.

Options **3** and **4** represent the most practically-oriented approaches, especially for process industry applications, as long as the problem is SISO. Option **3**, "Semi-quantitative time-response form", raises the screen depicted in Fig. 2, which illustrates a direct approach for capturing a simple (minimal) characterization that suffices for selection of more basic control schemes and their design (parameter tuning). More comprehensive support, and access to more sophisticated control schemes, may be available if the user supplies a more detailed plant model by choosing the transfer function or state-space options on this menu. Finally, the alpha version of DAIS also supports option **5**, the loading of input / output data sequences for a SISO plant. DAIS then carries out a simple least-squares model identification procedure to generate a plant model, of the form $K\ \exp(-s\tau)/(1+sT)$ or a

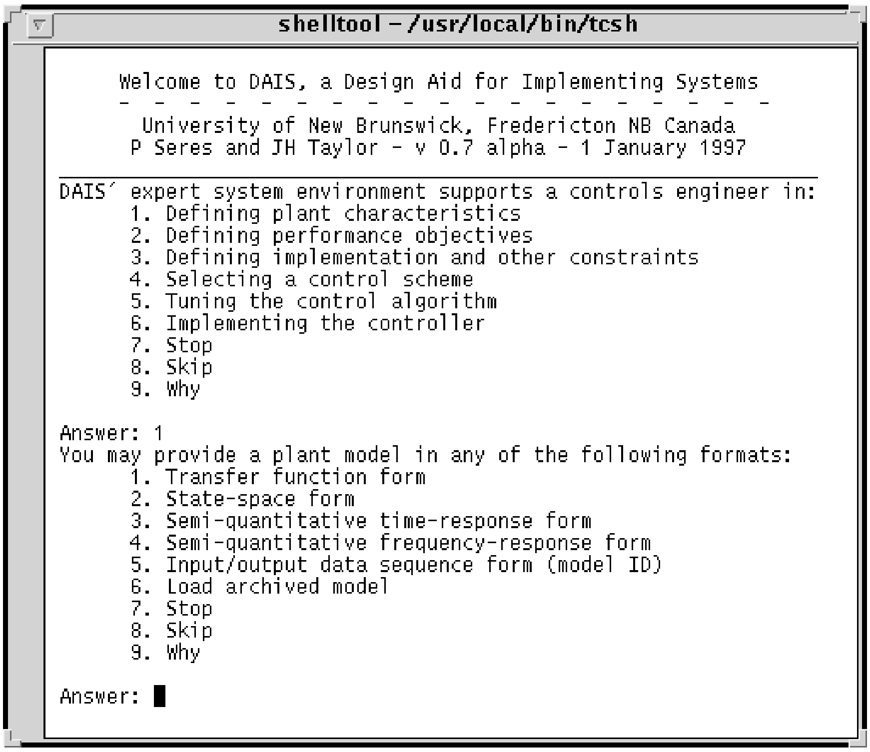

Figure 1: DAIS' Start-up Menus

first-order lag with time delay. For MIMO problems we are planning to incorporate ADAPTx, a commercially proven package for identification of time-series models for multivariable plants [14, 15].

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 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 design (Step 5).

If the applications engineer has supplied a suitable problem definition (plant characterization, performance specification, constraints), then the rule-based system 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

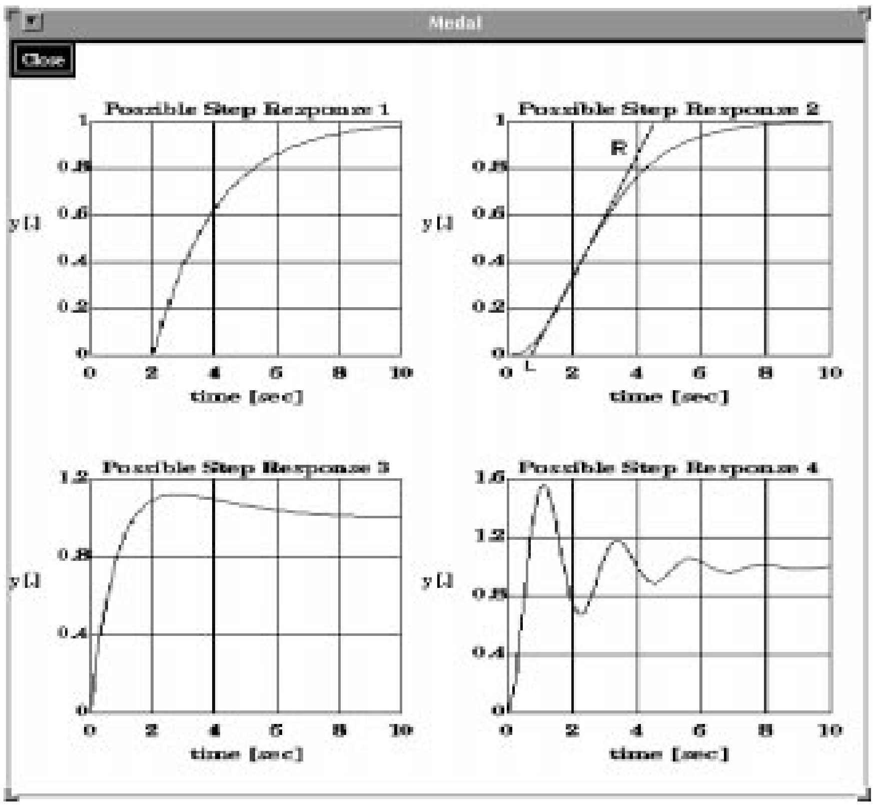

Figure 2: Screen for Picking Plant Time-Response Characteristics

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 is not a pressing need.)

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

- parameters for the qualitative model **1** are: delay time = 6 sec, rise time (after delay) = 2 sec and steady-state gain = 10.5;
- desired closed-loop 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 (see the first two 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 remove this violation, or simply decide that the controller is "good enough".

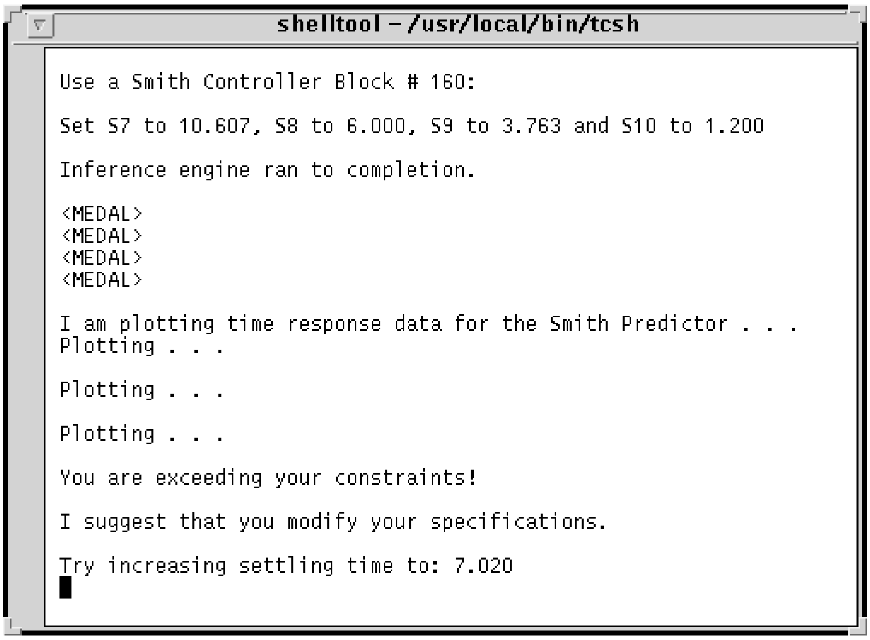

Figure 3: Screen with Controller Recommendation

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 ± 40% ("coarse tuning") or ± 10% ("fine tuning"); comparative step-response plots are displayed for the user to select the final design. Finally, the beta version of DAIS will automatically generate the INFI 90 configuration file for the control system.

2.2 DAIS knowledge 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 attention, called the ***Problem Frame*** (PF) and the ***Solution Frame*** (SF). The information in the PF is 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, screens and prompts, as illustrated. Once the user signals that problem-solving is to commence (by choosing item **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) 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

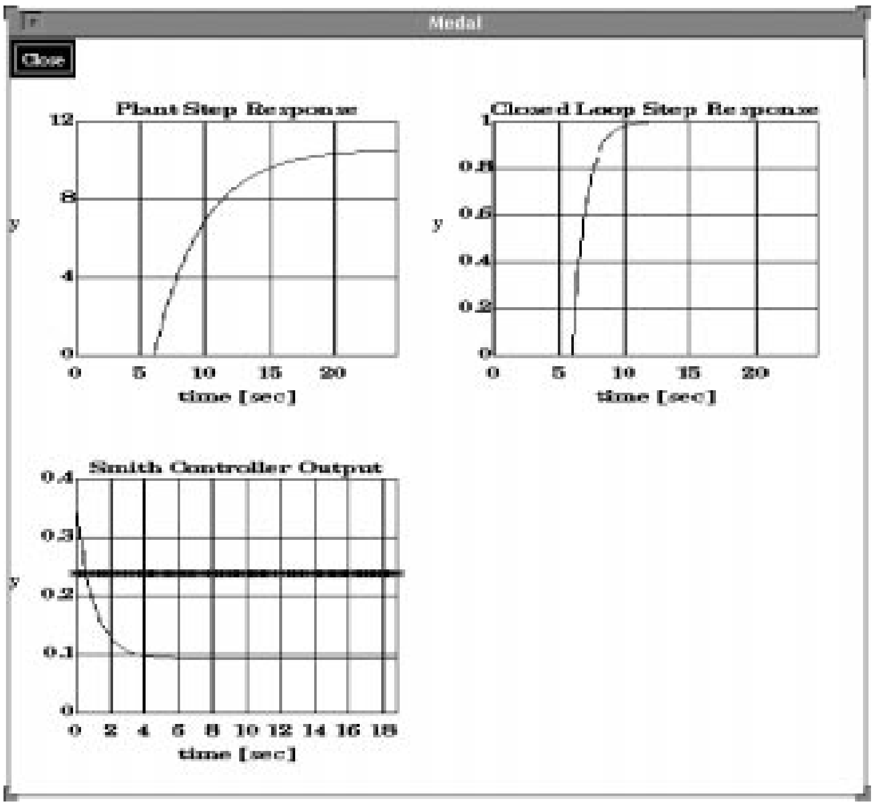

Figure 4: Predicted closed-loop Time-Responses

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 for use in rules such as "if the controller order must be less than 3 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".

In the alpha version of DAIS these features are quite limited. However, the beta system will not stop with a simplistic 'yes' or 'no' assessment with respect to availability; rather it will 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 will tell the user how to obtain and supply additional information or make modifications in as many instances as possible.

To illustrate this, the process of adding to the list of available controllers will proceed as follows: The expert system will work through the "unavailable" controllers' missing-data tags, first selecting those with one item of information missing, etc., to determine those that appear to be easiest to satisfy. The result of this analysis – a prioritized list of PF slots to try to fill – will be used to initiate a dialog with the user to increase the number of PF slots that are filled, in a systematic manner. It will analyze the missing-data tags, see which of them are most likely to be easily ascertained, and interact with the user to accomplish the job. The outcome of such an iteration will usually be a request for an additional piece of information, e.g.,

- "If you can state that the plant is stable you will be in a position to apply <list of additional control schemes>."
- "If you can determine that the plant gain margin is infinity you will ..."

Once a control scheme has been selected, the expert system 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, the expert system will aid in the validation of the final design produced under DAIS's guidance and in moving into implementation and testing. Design validation might involve setting up and executing simulation studies using a more detailed nonlinear model of the plant, for example – however, this is beyond the scope of the beta version currently being developed and tested. Presently we are focussing on how DAIS will aid the user in interfacing the INFI 90 with the process, setting up the selected controller, tuning it on line, and activating it – beyond automatic generation of the INFI 90 configuration file, we have to work out the details with Elsag Bailey applications engineers.

3 Implementation Plan and Approach

Two functional aspects of DAIS are of particular importance in this phase of the project: extensions to handle, at least in a limited manner, multi-input / multi-output (MIMO) plants, and extensions to achieve a better integration with INFI 90 software and hardware. The necessity of dealing with MIMO plants needs no explanation – aid in implementing SISO loops is of relatively minor importance in most process industries. The value of a more seamless interface between DAIS and an industrial DCS system can be appreciated by noting that the recommendation shown in Fig. 3 leaves the user facing the significant task of manually transferring the control system definition from DAIS to the DCS. Not only do the parameters need to be entered, but the configuration (DCS blocks and their interconnection) has to be transferred as well. Plans have been made to handle both of these problems.

One of the difficulties encountered in dealing with MIMO plants is model entry. The simple options **3** and **4** in Fig. 1 are not applicable, and it is often infeasible to supply MIMO models in transfer function or state-space form. Therefore, we are focussing on model identification from input/output data sequences as the primary approach for model specification. The elementary least squares algorithm used in the alpha version of DAIS is too cumbersome to extend to the MIMO case, and expecting the user to apply sophisticated methods such as those in the MATLAB system identification toolbox seems unreasonable, given our goal of supporting nonexpert users. Therefore, we are investigating the use of ADAPTx [14, 15], a higher-level and robust model identification package layered on MATLAB as the analytic engine.

A detailed study of MIMO controller synthesis approaches was also conducted, to find a suitable technique that will fit into the DAIS paradigm. Based on industrial applications and preferences, we focussed on frequency-domain methods that can be used to synthesize simple controllers. In addition, such a technique should be applicable to simple MIMO process models, including those with time delay. The best candidate we found was the multivariable controller synthesis method of Engell and Müller implemented in FASTER [11, 12].

To implement this approach in DAIS, we are limiting the controller type to PID and the performance objective to be nearly decoupled loops with simple frequency-domain characteristics derived from specifications in the time-domain, using classical heuristics [18] to make this translation. In brief summary, this technique uses this ideal closed-loop response specification to determine an ideal (but generally unrealizable) controller, then defines an optimization problem to determine the actual controller parameters (e.g., PID gains) that approximate the frequency response of the ideal controller, using a frequency-dependent weighting scheme that accounts for the sensitivity of the closed-loop system to changes in the compensator. This approach is systematic, with simple heuristics, and is thus ideal for implementation in DAIS using MEDAL's expert system and numerical capabilities.

4 Summary and Conclusion

Our intention in creating an "intelligent front end" for the Elsag Bailey system is to increase the efficiency and satisfaction of industrial controls engineers in using powerful, state-of-the-art distributed control systems (DCSs) such as the INFI 90. The effort in developing the present "beta" version of DAIS represents a major step towards achieving this goal. The overall framework, functionality and approach are well defined, but a few details concerning the DAIS-DCS interface are still somewhat tentative and subject to revision as we try a realistic industrial application and work in collaboration with Elsag Bailey personnel to better understand requirements; these areas will be resolved in the next few months, prior to beta test this coming summer.

We do believe, based on our progress to date and the reception from Elsag Bailey Canada and a few industrial users, 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 a DCS'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 control engineering), 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] MATRIX$_X$ ***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] J. H. Taylor, J. R. James and D. K. Frederick, "Expert-Aided Control Engineering Environment for Nonlinear Systems", ***Proc. IFAC World Congress, Vol. 6,*** pp. 363-368, Munich, FRG, 31 July 1987.

[6] 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.

[7] 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.

[8] J. H. Taylor and P. Šereš, "An Intelligent Front End for Control System Implementation", ***Proc. IEEE International Symposium on Computer-Aided Control System Design,*** Dearborn, Michigan, September 1996.

[9] K. Åström, ***Ziegler-Nichols Auto-Tuners***, Department of Automatic Control, Lund Institute of Technology, May 1982.

[10] K. Åström and B. Wittenmark, ***Computer Controlled Systems – Theory and Design***, Prentice Hall, 1984.

[11] S. Engell and R. Müller, "Fast and Efficient Selection of Control Structures", ***Proc. ESCAPE-1, Computational Chemical Engineering, Vol. 16***, pp. 157-164, 1992.

[12] S. Engell and R. Müller, "Multivariable Controller Design by Frequency Response Approximation", ***Proc. ECC***, Groningen, June 1993.

[13] D. W. Clarke and P. J. Gawthrop, "Implementation and Application of Microprocessor-Based Self-Tuners", ***Automatica, Vol. 17,*** 233, 1981.

[14] W. E. Larimore, "The Optimality of Canonical Variate Identification by Example", ***Proc.*** 10^{th} ***IFAC Symposium on System Identification,*** Copenhagen, Denmark, July 1994.

[15] W. E. Larimore, "Optimal Order Selection and Efficiency of Canonical Variate Analysis System Identification", ***Proc.*** 13^{th} ***IFAC World Congress,*** San Francisco, CA, June 1996.

[16] K. J. Åström and T. Hägglund, "Automatic Tuning of Simple Regulators", ***Proc. IFAC 9th World Congress,*** Budapest, Hungary, 1984.

[17] J. H. Taylor and K. J. Åström, "A Nonlinear PID Autotuning Algorithm", ***Proc. American Control Conference,*** Seattle, WA, 18-20 June 1986.

[18] J. J. D'Azzo and C. H. Houpis, ***Feedback Control System Analysis and Synthesis***, McGraw-Hill Book Co., New York, NY, 1960.