The Knowledge Engineering Review, No. 5:3, 1990, 181–214

AI in control systems engineering

D. A. LINKENS

Department of Control Engineering, University of Sheffield, Sheffield, UK

Abstract

Decision-making is an integral part of any consideration of the discipline of control systems. This implies feedback of knowledge in some format, whether it is mediated via human or machine-oriented methods. The application of AI methodologies to control systems engineering is described in this review under two broad categories of design and implementation. The design of control strategies is itself divided into phases of modelling and simulation, identification, and algorithm selection and tuning. The aspects of implementation which are covered include real-time considerations such as knowledge-based control and fuzzy logic, multi-sensor data fusion, fault detection and Human–Computer Interaction (HCI).

1 Introduction

By its very nature the discipline of control systems involves the processes of measurement, decision-making and actuation. The human-being is capable of performing all of these functions in certain circumstances and in so doing demonstrates real intelligence. The human body is itself made up of complex interacting biological control systems, some of which can be described, analyzed, or even replaced using quantitative numerical and engineering methods. Thus, the use of on-line computer-based drug therapy systems for alleviating body malfunctions is now a demonstrated reality. Such developments have come about because of advances in the design and implementation of feedback control algorithms.

Design methods for control systems owe their advances to two main application thrusts. These are industrial process control and military systems. The latter area was strongly progressed during World War 2 and has been steadily advanced subsequently in the aerospace industry. The numerical, or quantitative, approaches of control design have been successful in many such applications where detailed mathematical models are commonly available. In contrast, the industrial process control scene has seen little algorithmic development beyond the classic, and widely utilized, three-term controller. A major reason for this lack of progress is that the quantitative design methods currently available do not suit situations where detailed mathematical models are not available for the processes to be controlled. Such issues are being addressed currently via two approaches. The first is the introduction of robust design techniques which allow for process uncertainty. The second is the use of qualitative techniques related to AI, and which form the subject of this review.

The available techniques for control system design are themselves heavily reliant both on computing and human interactive skills. The mechanization of these techniques via AI principles is considered in each of the major sub-divisions of systems design. These are modelling and simulation, identification and estimation, and strategy and algorithm design. Section 2 of the review deals with these aspects. The implementation of systems design involves the concepts of real-time control, which is covered in Section 3. Again, three areas are addressed. The first part of this section is a review of real-time direct expert control, especially considering the applications in process engineering. The second part is devoted to Supervisory Expert Control, where overseeing of more conventional control is performed by a Knowledge-Based System (KBS). The important

subject of fault detection and diagnosis is also briefly discussed. The third part is that of Human–Computer Interaction (HCI) which is particularly important for safety-critical aspects of operator/control room displays.

The current state-of-the-art is summarized in the conclusions, together with remarks on the challenges which lie ahead in this particular field of AI in Engineering.

2 AI in control system design

For many years control design has been heavily dependent upon computing because of its intensive mathematical basis and its reliance on graphical interpretations. It has thus been a fruitful field for CAD utilization. However, the background skills necessary to understand the algorithms involved in the design, and to interpret the displays are very extensive. As a result, there are relatively few people who can adequately utilize the CAD facilities, particularly considering the rapid rate of development in these design techniques. The incorporation of knowledge-based techniques into the CAD tools is thus attractive for all branches of systems design. The major concept will be that of an expert-advisor, since human-centred design techniques should always predominate in the field of systems engineering.

2.1 AI in modelling and simulation

This forms a major component, since feedback design techniques cannot proceed without some knowledge of a mathematical model suited to the process to be controlled. Modelling here means a mathematical structure which adequately represents the "physics" or real-life functionality of the system. Simulation means the solution of such model equations via a computer, since analytical solutions are seldom feasible. Such simulations are commonly undertaken using high-level simulation languages. Conventionally, dynamic system simulations are either *continuous* (i.e. governed by differential equations) or *discrete* (i.e. governed by difference equations). In surveys conducted at industrial sites in Holland and Japan, it has been concluded that software for simulation is the most widely used for dynamic system design (Van den Bosch & Van den Boom, 1985; Araki, 1985). The range of simulation software that is currently available also supports this conclusion (Linkens, 1988).

Modelling and simulation is a multistage, interactive process embodying several levels of experties. Figure 1 illustrates these levels in a framework cast in AI techniques as suggested by Lehmann et al. (1986). The layers move from general to specific from top to bottom, commencing with general problem formulation, objectives and specification. This leads to conceptual modelling, where specific methodologies are chosen, followed by construction of particular models in the chosen domain. Further stages involve verification, experiment planning, validation, model interpretation, and ultimately evaluation of the whole exercise. Many skills are involved in these different levels, and Lehman envisages expert systems techniques to be applicable widely in this infrastructure. The importance of HCI aspects and global databases is also evident from Figure 1. These concepts are prototyped in a PC-based system INT^3 (Lehman, 1987). The Modelling language is RESQ, which is primarily a network queuing simulator useful for consideration of reliability problems and Petri networks. RESQ runs on an IBM mainframe, and the remaining framework links to it via a PC, utilizing the expert system shell MI.

Lehman suggests the following points as motivation for AI in modelling:

- users are not specialists in all areas of system evaluation,
- have not an extensive overview of modelling methods,
- not familiar with details of model construction,
- have not sufficient practical experience with model verification, model validation, experiment planning or interpretation.

In considering the merging of qualitative and quantitative techniques he distinguishes between

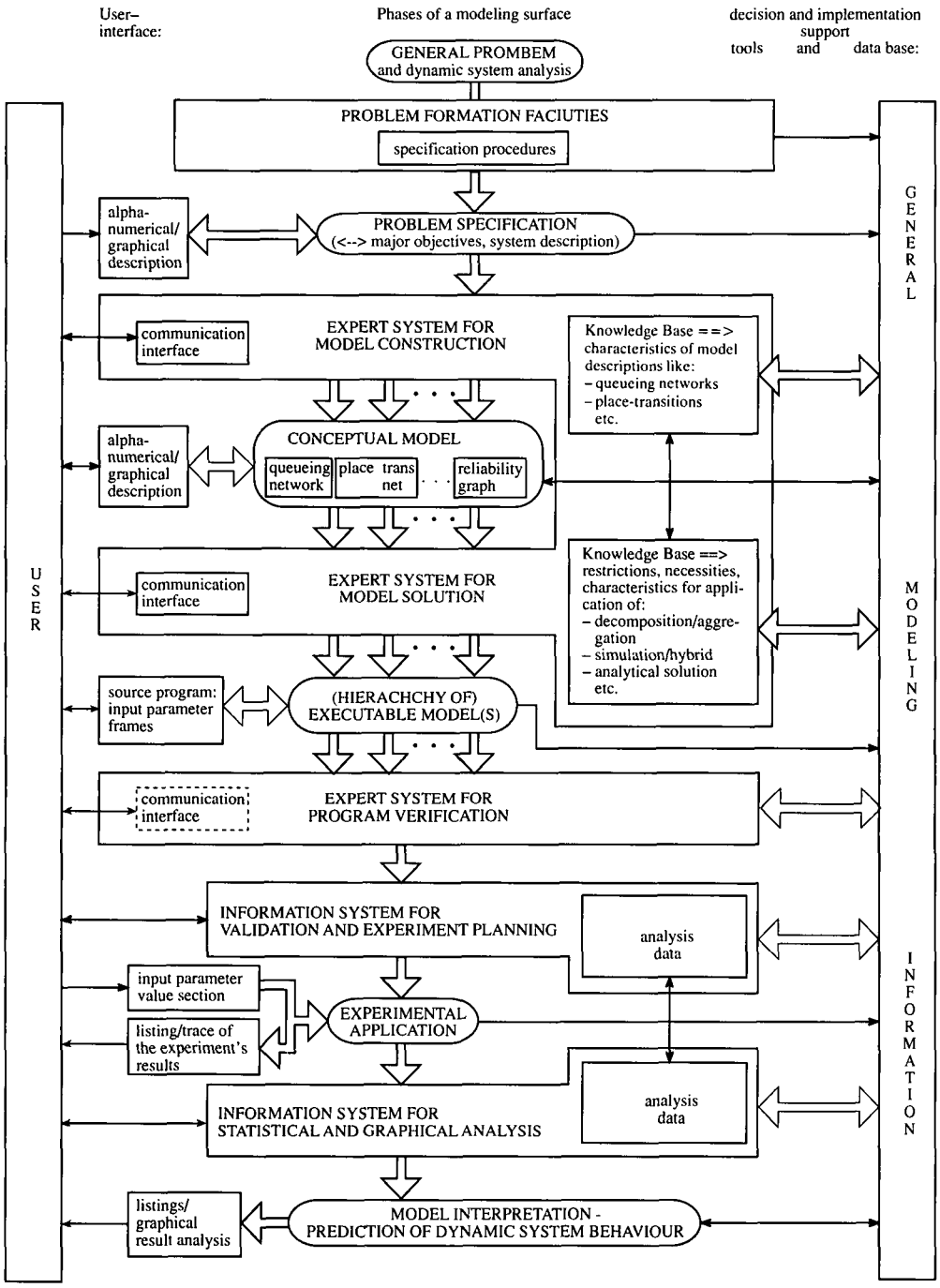

Figure 1 Functional concepts for knowledge-based and hierarchical modelling (from Lehmann et al., 1986).

assistance of expert systems by simulation (e.g. for representation of time-dependent knowledge of inferencing), and assistance of simulation by expert systems (e.g. representation of heuristics throughout the modelling/simulation process). Knowledge about modelling is dispersed among a wide variety of disciplines, both in terms of its methodologies and applications. As a result, Ziegler & de Wael (1986) suggest that this has led to a "throwaway" attitude towards simulation models, giving little appreciation for the benefits of "modelling in the large"—adhering to methodologies that enhance the reuse of models and hence the cumulative development of model encoded knowledge. The analogy with current trends in Software Engineering for reusable code is obvious. To overcome this, Ziegler (1976) proposed a "multifacetted" methodology, which is claimed to be suitable for AI related implementation. The initial prototype concepts emphasize decomposition

of systems into objects using a frame-based inheritance mechanism. This is extended to include experimentation aspects of modelling.

2.2.1 Continuous system simulation

The problem of achieving an adequate skill base for simulation is clearly seen in modelling for biology (Pave & Rechenmann, 1986), where additional abilities in mathematics and computer science are required of the experimentalist. To overcome this, Pave considers a modelling knowledge representation based on fames, in which the only inference is via inheritance and procedural attachment. He also discusses the use of specific languages (i.e. description languages) which may aid both the elaboration and interpretation of models. Examples of these include engineering block diagrams, compartmental analysis, and bond graphs. Direct translation from schematic representation to mathematical formula is proposed, together with an "inverse" translation. The example studied is a chemical-like language associated to models of population dynamics where a mathematical expression can be associated with a "functional scheme" (i.e. a pseudo-chemical reaction). This leads to a hierarchical organization of these Lotka–Volterra models, which can be explored in the framework of a structured knowledge-base. Many of the ideas mentioned above have been incorporated into a modelling infrastructure and called KEMS (Knowledge-based Environment for Modelling and Simulation). The modality is that of continuous dynamic system simulation, and the motivation is the need to provide software tools for both expert research users, and also practising engineers on site. The particular scenario used for prototyping has been pipeline pressure reduction systems as used by British Gas. A functional specification for each environment indicated the need for multiple co-operating expert systems with different architectures (Rahbar et al., 1987). The central component involving model selection and construction uses frame-based representation, for which a special purpose inference mechanism was written in Prolog (Tanyi & Linkens, 1987). This also provides automatic translation from the model equations into simulation code (validated for each component). Particularly for novice users, the importance of good graphical user-interfaces is paramount both for model construction and simulation outputs. To achieve this a graphics toolkit EASE+ has been used, which also includes certain database facilities. The storage and retrieval of dynamic model structures together with their experimental details is a vital component in the drive for reusable software modules, as noted by Ziegler. To achieve this, browser and archiver modules have been developed using data compression techniques of "reverse-delta" storage and retrieval (Bennett et al., 1989). In this paper futher consideration is given to the need for providing global databases within such environments, showing the present limitations to both conventional relational databases and tentative object-oriented approaches to information retrieval. Figure 2 shows the basic modules in the infrastructure, which is further described in Bennet et al. (1989). Although the frame-based structure of KEMS gives powerful generation of validated simulation code for systems constructed from a model-base repository, it does not provide an easy mechanism for basic component generation. To enable an expert dynamicist to enter his knowledge into KEMS, an additional facility known as KAM (Knowledge Acquisition Module) has been built using a semi-natural language format for modelling input. The syntax of this language allows for a powerful blend of frames and production rules together with incorporation of scripts and facets (Tanyi & Linkens, 1989). Work is currently under way to incorporate further tools into the environment. In particular, model validation and experiment planning and interpretation are important and are being constructed as additional, and different, expert system modalities. To increase the usefulness of KEMS to a wide range of users, the improvement of schematic to mathematical translation, as mentioned by Pave, is being studied using bond-graph methodology. Further work is also being performed on AI aspects of dynamic databases for modelling, and human factors aspects of HCI design for such computer-aided design environments.

2.1.2 Discrete system simulation

The work just described concentrates on continuous systems simulation and is relevant to process

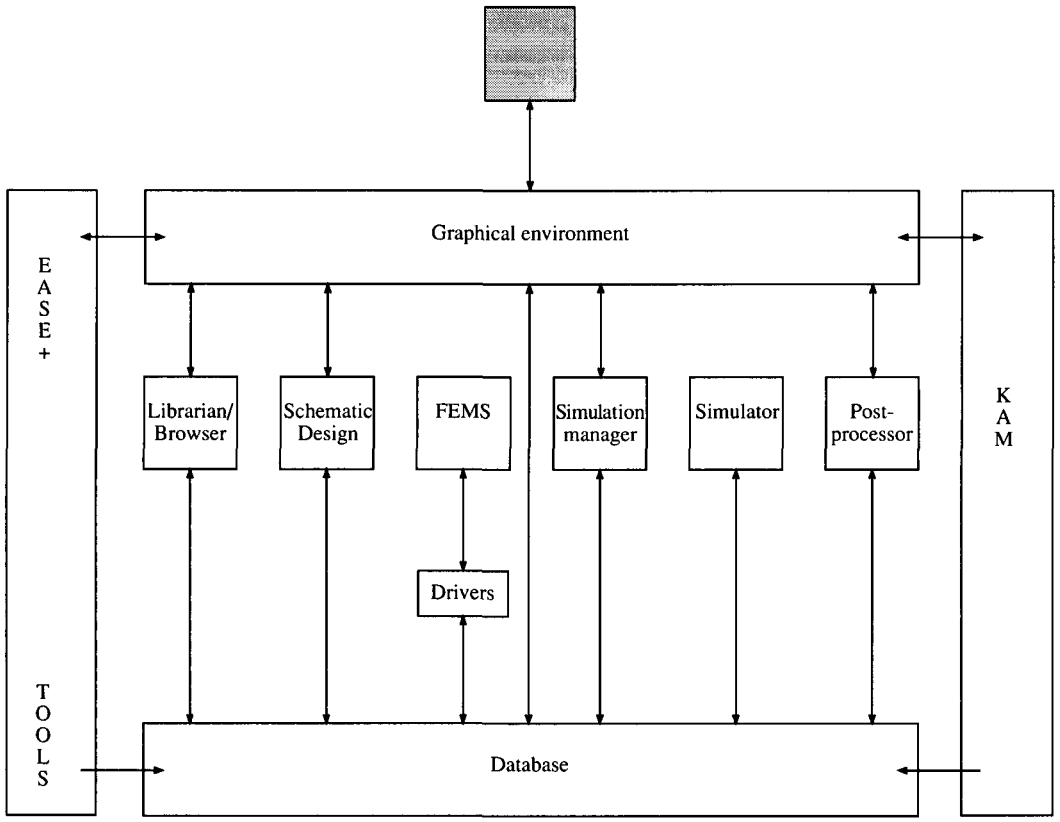

Figure 2 Schematic diagram of the software environment KEMS.

control and the aerospace industries. In some senses, AI methodologies have found more impact on discrete system modelling, since an object-oriented paradigm fits naturally into this area, in contrast to the continuous system field. An overview of an add-on module, SIMKIT, to the hybrid AI-toolkit KEE is given by Neilson (1987) in the context of design of control systems for automated factories. Advantages claimed for this approach were much reduced time required in production of the simulator, simplified training for its use, and improved modifiability. These correlate with other studies using AI toolkits, indicating usefulness for rapid prototyping, but limitations caused by slow run-time and large memory resources.

Two approaches to the intelligent interface systems for systems modelling have been described. The first is called EZSIM and provides a front end to a discrete simulation language SIMNET. It is written in Common Lisp and runs on a PC. Systems are built up graphically using nodes and arcs. A rule-based module is used to analyse the connections to check for logical errors and advice is given as to what components are needed logically to complete an engineered system.

The second approach, NATSIM, creates models of production/distribution systems solely from a natural language description of the problem. It uses a semantic-based parser PHRasal Analyser (PHRAN) which is composed of three major parts: a database of pattern-concepts pairs, a set of comprehensive routines, and a programme which suggests appropriate pattern-concept pairs. The basic methodology of NATSIM decomposes the systems analysis into identification and analysis of material flow, information flow, and secondary (auxiliary) information flow. The natural language input is first parsed and then processed again to generate a formal system representation which can be directly transformed to its equivalent DYNAMO (discrete system language) code.

SEE (Simulation and Expert Environment) aims to integrate different equation simulation with object-oriented programming and rule-based reasoning (Lounamaa & Tse, 1986), thus providing

symbolic and numerical reasoning in one environment. In common with other software environments, SEE can be conceptualized at three levels. The first level consists of a set of tools, such as deduction, simulation and analysis, each of which could be used independently. The second level consists of a uniform data structure, which provides communication between the tools. The third level compromises specific models to be run. The particular problem for which models were constructed is that of a multi-agent behaviour. It was recognized that applicability to other domains was limited by the slowness of SEE, and that "its strength lies in flexible study of small scale demonstrative cases".

2.1.3 Continuous/discrete system simulation

Dynamic simulation languages usually emphasize either continuous or discrete modalities. This is reflected in the AI approaches mentioned so far. One language which attempts to provide balanced features for both descriptions is SYSMOD, produced originally at RAE, Farnborough, England. One example of a mixed simulation system constructed using the hybrid toolkit, KEE, is given by Langen (1987). In this example, rapid prototyping is again emphasized for the production of a simulator for Heating, Ventilation and Air Conditioning (HVAC). Initially, an air handler was modelled considering steady state conditions only, and providing animated graphics to illustrate temperature profile, and air flows. Subsequently, other models were built including a boiler, a chiller and a zone model. A frame structure was used for model description, to which object-oriented concepts were added via procedures activated by events. Rules were incorporated to provide optimization of control strategies. An example of this was the tuning PID of controller parameters to minimize overshoots. In the boiler model, transient behaviour was introduced by first calculating the steady state energy flows and temperatures, and then applying time constants to these values to predict the dynamic response of the boiler.

2.1.4 Steady state simulation

The simulations considered so far have been mainly of a dynamic nature, whereas process plant models are often of a steady state nature. Thus, flowsheet models involve the calculation of steady state heat and material balances in chemical processes, and numerous programmes exist for them. Fjellheim (1987) considers knowledge-based augmentation of such models. His intelligent front end to the flowsheet programme PROCESS is called KIPS (Knowledge-based Interface to Process Simulation). It uses a bottom-up strategy starting as a graphical input tool and leading into knowledge bases for checking, guiding and automating the model construction task. The procedure is considered to be knowledge-intensive having the following requirements:

- understanding of the plant, its goals and specific properties
- purpose of the model, i.e. which plant features are the focus for study
- strategies for using PROCESS to achieve the modelling goals
- particular stylized ways of using PROCESS facilities
- how to select one of several possible alternatives for expressing a plant feature
- particular details of specific PROCESS units, parameters etc.

Thus the aim is to provide an *intelligent assistant* for the process engineer. KIPS is written in LOOPS using an object-oriented approach for the knowledge-base. This comprises a set of model objects for the unit processes, a taxonomy of which is shown in Figure 3. The need for hierarchical design is stressed, with possibilities of submodel creation, editing, zooming etc. Future aims, therefore, are to provide libraries of flowsheet modules (i.e. multi-unit sub-assemblies), design procedures compromising "canned" sequences of well understood design steps, and goal driven synthesis of flowsheet models. Further recommended goals are planning of simulation experiments, "coarse" simulation for pruning initial designs, fault diagnosis (i.e. model validation) and interpretation and evaluation of modelling results.

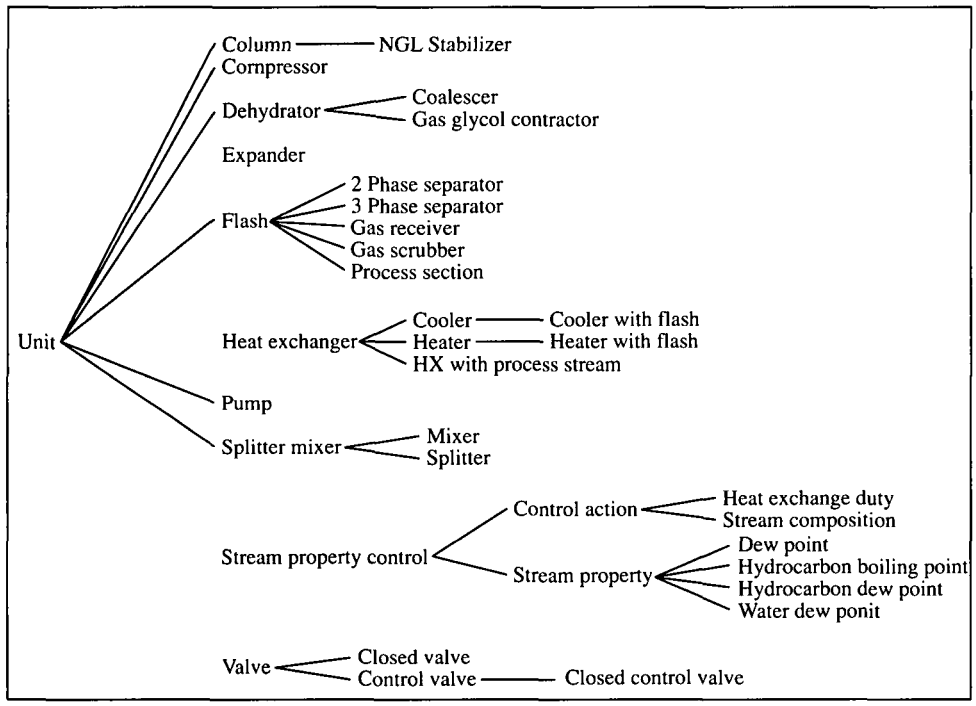

Figure 3 Taxonomy of the PROCESS unit in the KIPS process flow-sheet programme (from Fjellheim, 1986).

2.1.5 Symbolic/numeric modelling and simulation

The final papers referred to in this section are concerned with the merging of symbolic and non-symbolic concepts in hybrid AI applications involving conventional procedural languages rather than general or special purpose simulation languages. The topic of qualitative modelling as understood in the AI community is not discussed here because of its vast literature and relative immaturity of development. Reference can be made to survey material in this field via papers such as that of Rajagopalan (1986).

Gladd & Krall (1986) consider the use of symbolic methods to aid experimentation with large-scale numerical computations. In particular they consider systems which involve eigenvalue calculations and physical models characterized by scalar state variables. Extension to vector state variable problems would be feasible, but problems of data storage and manipulation would be difficult to handle using their implementation. Unusually for this type of AI related work, their system called ANLYZ is targeted onto a CRAY supercomputer using Lisp for symbolic work and FORTRAN for the numerical computations. The system is loosely coupled using disk file transfer for communication between the two components.

They note that even when modelling simple physical systems, the amount of labour and computing required to obtain insightful understanding is so great as to preclude any systematic or exhaustive method. In practice, computational physicists bring many heuristics to the tasks of exploring the parametric wilderness of a complex mathematical model. They make use of analytically tractable subcases, and mathematical intuitions about the nature of the modelling equations and solution algorithms. They form realistic goals about reasonable expectations from particular models, and analogies to previous research. They also recognize patterns emerging from a computational model. Such patterns include monotonicity, oscillation, extrema existence, erratic or pathological behaviour, and they can spot the existence of key parameters. Thus, effective use of computational models involves pattern recognition, planning under uncertainty, correlation of different forms of information, qualitative reasoning about quantitative systems and the effective use of heuristics. ANLYZ concentrates on the experimentation phase, which uses as a basic unit of knowledge a "run", which is defined as the results obtained by linearly varying one of the

parameters in the model. Knowledge is stored in frames, which also include slot information on time and date of run, version of model code, reasons for making the run, initial values of state variables, indications of pathological behaviour and physical irrelevancy etc. From the vast amount of data obtained in such experiments ANLYZ uses search heuristics which imitate human behaviour in searching computer output for relevant past cases. Rules are used in terms of qualitative description of the underlying data, similar to a fuzzy logic style. An example of this is that ANLYZ can allow very accurate initial contitions to be interpolated from frames of knowledge corresponding to previous runs. Thus, a distinction is made between "deep" reasoning and long causal chains, used in the analytical science of model building, and the computer solution of such models involving descriptive and experimental science. The infeasibility of exhaustive search techniques for large-scale mathematical models leads to the concept of the intelligent assistant which reduces the cognitive load and alleviates the labour intensive tasks of users of such large models.

The problem of communication of data between dissimilar languages involved in hybrid AI applications is addressed by Borchardt (1986). A language STAR (Simple Tool for Automated Reasoning) is described and is implemented as an interpreter written in "C". It comprises two components, the first of which is the possibility of intermixing data structures defined in different languages. The second component is the availability of basic manipulation utilities for use by external routines. Sharing of data values in STAR is bidirectional. STAR was used to prototype a KBS for the interpretation of imaging spectrometer data. Thus, a symbolic level combining algorithmic and rule-based operations was defined in STAR to manage a set of analysis routines and numerical data structures defined in "C". At the symbolic level two general tasks are performed. The first concerns the overall control of operation of the application system. The user and the system share the task of steering the analysis, with the system acting as a "co-ordinator" by listing available options, allowing backtracking to recompute values, and serving as a source for retrieval of partial results. The second task is a qualitative interpretation of the results of the analysis. The system correlates results from the numerical computations obtained by "C" functions with symbolic information contained in the STAR knowledge base.

2.2 Systems identification and estimation

This branch of control systems engineering has conventionally been concerned with the determination of parameter values in dynamic mathematical models suitable for subsequent design of control strategies. The structure of a mathematical model may be obtained from the equations describing the physical phenomena taking place in the system. However, in many cases the system being studied is either a "black box", or at least a "grey box", where the causal chain is not completely known. In such cases, a generalized structure is obtained from measurements and observations on a system, as well as the experimenter's background and experience. The experienced investigator examines input–output data from a system, and deduces from them such structural information as linearity, non-stationary, existence of non-linearities or minimum system order. This process is often referred to as *identification,* and involves a great deal of experienced knowledge, which could be codified in a KBS.

Knowledge-Based Identification System (KBIS) by Bekey (1988) is an expert system which uses process and identification knowledge to guide the construction of models and their subsequent parameter estimates. It also aims to advise the experimenter in matters of data acquisition, model formulation and interpretation. Bekey also contends that a system of production rules to describe a process is, in fact, a model of the process, and that to obtain a valid set of production rules is system identification. Obviously, the methods used to identify such rules are quite different from those used to estimate parameters in differential equation models. As examples of this he cites the reflex control of human walking, and the control of grasping in human and robot hands. Bekey also considers a neural network as another way in which the behaviour of complex systems can be

identified. A similar approach has been adopted in the identification of a model for the control of depth of anaesthesia in operating theatres (Linkens et al., 1986). Clinical knowledge has been encoded in a C-based program called RESEAC (Real-time Expert System for Advice and Control), thus providing a symbolic model of an anaesthetist's actions. Such a knowledge-base is large and difficult to elicit, extend and maintain. Hence investigations are under way to replace this qualitative approach with a neural network model.

The development of an intelligent front-end to CAD package for system parameter estimation (IDPAC) is described by Larsson & Persson (1988). It uses a command-spy strategy based on scripts in order to understand what the user is doing. The scripts represent different command sequences which might be used in an IDPAC interactive session. By matching scripts against the actual command history, the expert interface is able to guess what the user wanted to do. Via a query mode, a context-sensitive defaulting facility, the user is helped with remembering trivial things such as file names, parameter values etc. This provides a useful "on-line notebook facility". Diagnostic reasoning within an identification session is provided by production rules associated with each script. In standard dynamic model estimation fashion the input–output data are divided into two sets; one for identification and the other for cross validation. The user is then guided through a session, firstly performing some preliminary tests on the signals. After the model parameters have been estimated, various validation methods are suggested for usage by the experimenter. The knowledge base comprises about 20 pages of script code, including 130 production rules, and represents only a small part of the knowledge needed for systems identification. Reilly & Timberlake (1987) provide another example of an intelligent front-end for the well-known Box and Jenkins forecasting techniques used for time series prediction in many applications.

A more ambitious approach than the intelligent front-end is to construct an Intelligent Tutoring System (ITS) for training and instruction in the whole field of system identification (Betta & Linkens, 1988; 1989). The system which is calle EASI (Expert Advisor for System Identification) is written in the expert system shell ADVISOR-2. The large amount of textual knowledge required has made the project of an encyclopedic nature. A user-model-based ITS structure was adopted for this project (Brown, 1986). A user-model is established at the beginning of a session from information on the user's background in control engineering, system identification, familiarity with related topics and frequency of use of the techniques. No attempt is made to provide adaptive dialoguing concepts, but the user may change his model status manually. The user-interface provides conventional explanation facilities of an expert system shell, with the detailed knowledge presented in small chunks with highlighted key concepts, and the ability to browse through the knowledge base. Bibliographic material is provided at the end of a session by way of a detailed list of references relevant to the particular path through which the consultation has proceeded. Further, annotated bibliographical material is provided to guide the novice user via directed study and reading. This gives a type of "graceful degradation" of the system when the user cannot perform the identification studies being advised at the conclusion of the session. An additional feature is the incorporation of an on-line dictionary and glossary of terms, which also functions as a Thesaurus for the novice user.

The sequence of tasks typically performed in dynamic system identification is shown in Figure 4, which illustrates the iterative nature of the process, common to most engineering design methodologies. The structure of the knowledge base modules in EASI is closely related to the elements in Figure 4. The system acts as intelligent tutor for 3 large in-house identification packages. Two of these are powerful general purpose non-linear single-input single-output estimation suites—NLI (Billings et al., 1984) and FASTNOI (Tsang & Billings, 1987). The third is a linear multivariable identification package—LMI (Hamiane & Morris, 1985). The knowledge base is structured into various modules as follows:

(1) Experiment design

This provides advice on issues such as environmental conditions, input design, sampling rate

determination, experiment length, and a number of preliminary classical identification methods such as step responses.

(2) Model structure determination

Two classes of knowledge are covered, being *a priori* and *a posteriori* methods. The *a priori* methods are used before performing estimation, and include Determinant Ratio Test, Impulse Response and other classical methods. In contrast, the *a posteriori* methods are performed on the results of estimation, and include Loss Function Behaviour Analysis, F-ratio Test, Akaike Information Criterion, and Pole-Zero Cancellation Effects.

(3) Parameter estimation methods

Advice is given on a wide range of estimation techniques, each of which is contained in a separate module for use in any of the cooperating knowledge bases in the expert system. The methods include Ordinary Least Squares, Weighted Least Squares, Extended Least Squares, Generalized Least Squares, Maximum Likelihood and Prediction Error Methods, and Instrumental Variables. In the non-linear case these are augmented by numerous techniques such as Sub-optimal Recursive Least Squares and Nonlinear Prediction Error methods etc.

(4) Model validation tests

This covers corelation-based techniques which determine the whiteness of various residuals in the estimation process. Also, techniques of data comparison and model prediction tests are dealt with.

(5) Other modules

These cover topics such as Data checking, Data handling, Data anomalies, Bibliographic material, Dictionary/glossary of terms, and Model representation.

The modules are kept to less than 100-kbyte each, to aid editing and improve the speed of interactive response. EASI runs on an IBM PC, and is intended to be used alongside another terminal running the identification software. Work is currently underway to provide an integrated identification environment provided both qualitative and quantitative components on a SUN

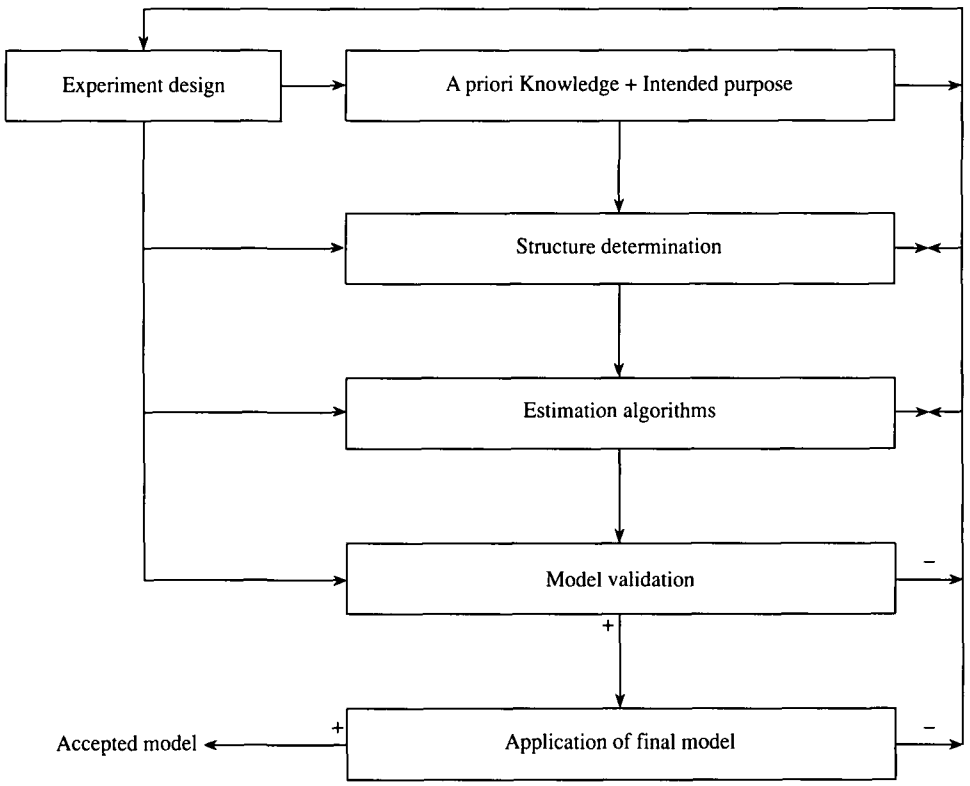

Figure 4 The system identification cycle.

workstation. EASI has been validated using both graduate research students and taught Master's students operating under case study conditions.

An entirely different approach to the intelligent advisor role is that of Haest et al. (1987), who attempt a totally automated identification strategy. Thus, an expert system coded in OPS is integrated with an in-house identification package, SYSID. Again, there is a significant difference from the EASI project which is aimed at an intelligent tutorial system where the human is expected both to learn and make decisions regarding the adequacy of the identification results. ESPION deals only with discrete linear parameter estimation, and with MISO structures.

The data are divided into two subsets: the first for parameter estimation, and the second for model validation. They define an *identification exercise* as one entire exercise model search, including the possible consideration of several different sampling periods. Also, an *exploitation* is defined as a model search at constant sampling period, whose aim is to find an optimal model structure for that condition. At constant model dimension and constant sampling period either a fast or a slow search can be performed. Under the fast search mode, one has a laboratory-type environment which it is claimed can be exploited by the expert system to acquire real learning abilities.

The knowledge base can be divided into the following sets of rules:

- Fast search
- Slow search
- Initial guesses for delays
- Investigated model counts
- Analysis of model structure
- Quality index computation
- Sampling period validation
- Classification of models
- Exploration to determine best model order
- Identifiation exercise control

After some initial structure determination, including delay estimation, the expert system takes over and iterates around various model orders attempting to find the best model fit to the data. It uses a quality index, with 10 criteria, to determine the adequacy of a model. The main criterion is a minimum in the predictor errors with increasing model complexity. Failure to find such a minimum leads the expert system to alter the sampling rate for the data. The system has been evaluated on both simulated and industrial process data, for which it claimed that performance equivalent to a human operator was obtained. It is admitted, however, that the final choice of a model must ultimately be left to human judgement. This corresponds with the viewpoint behind EASI, which considers that the expert advisor should primarily be intended to increase the skill level of engineers in the identification process.

2.3 Control system design

The design of feedback control strategies is the third of the enabling systems methodologies considered in this section. Originating from the classical graphical design techniques of Nyquist and Bode, a whole toolbox of methods now exist for construction of feedback algorithms. Most of the methods have a deep mathematical basis to the algorithmic design, but have a graphical re-interpretation which then requires considerable conceptual skill for execution of the design. This is particularly true for multivariable systems which use extension matrix manipulation for the design procedures. Feedback design was one of the earliest engineering disciplines to make extensive use of computer graphics, but the development of CAD tools has been hindered by the complex conceptual skills required for their successful utilization by the majority of control system designers. It is because of this that the incorporation of AI techniques into this area is particularly challenging.

2.3.1 Classical control CAD and AI

Some of the earliest serious work in this field has been conducted by Taylor, Frederick and their co-workers, mainly under the auspices of General Electric of America. In their early work (Taylor & Frederick, 1984) they laid down the aims and objectives for a software environment which became known as CACE-III (Computer Aided Control Engineering—Third Generation). As in modelling and simulation, the requirements are to co-ordinate symbolic and numeric computation using expert system concepts. To do this they used a GE inference engine DELTA (later upgraded to DELPHI) written in Lisp running on a Symbolic workstation. This was coupled to a VAX running FORTRAN-coded design algorithms operating under VMS. In laying down the objectives for CACE-III they observed a number of limitations to existing second generation CACE packages:

- specialized to one aspect of the overall design process, whereas a whole "toolkit" is necessary
- problem-solving environment is potent, but driven in a relatively low-level command style
- little guidance in the detailed and often complex design procedure
- little, if any, documentation about trade-offs and constraints in arriving at final controller designs
- little, or no, useful support in validating and implementing the final design
- difficulty in incrementing the package with the ever-increasing new design techniques being developed.

In attempting to incorporate the control designers expertise involving judgement, heuristics and inference in a knowledge base, the methodology should include diagnosis of the model, setting up a realistic design problem, selecting appropriate design methods, performing trade-offs, implementing the controller, and actual use of conventional CACE software. These tasks are elaborated in Table 1, and these concepts were represented initially in a *problem frame* and a *solution frame.* The problem frame provides a well-posed problem formulation in the form of a set of facts, relating to issues such as plant characteristics, constraints and specifications. The problem frame had two rule-bases devoted to model development and design specifications. In contrast, the solution frame forms a type of scratch pad, whereby the designer and the expert system work in parallel to achieve a satisfactory solution. A further four rule-bases are involved in the integration of the complete system. These rule-bases cover the establishment of needs, design, and validation and implementation.

To avoid the danger of the control designer becoming frustrated with a CACE environment, the process should have the following features:

- the support should be suggestive, and not constraining
- a *why* facility can be invoked to give the user advice prior to judgemental decisions
- a command language should be available for direct entry of facts into the knowledge base

Table 1 Control engineering activities

1.	Modelling the plant to be controlled
2.	Determining the characteristics of the plant model
3.	Making changes that might be required to make the plant more amenable to control (e.g. adding actuators or sensors)
4.	Formulating the elements of the design problem
5.	Checking to see that the design problem is well-posed; especially, determining the realism of specifications in light of the given plant model and design constraints
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

- permission to invoke the underlying conventional package directly via its command-driven mode if required.

The structuring of the rule-bases along functional lines within the discipline of CACE enables switching between the modules with the following benefits (James et al., 1985a).

- Top-level goals can be re-evaluated continually by the supervisor, based on results from the operational rule-bases.
- Incremental design is supported, since partial sets of facts can be stored for subsequent use.
- Rule bases can be written and debugged separately, prior to integration.
- The supervisor is used to limit the scope of the search, thus keeping reasonable run-times.

In an evaluation of CACE-III, James et al. (1985b) claim that the rule system, which has over 300 rules, had demonstrated a high level of competence in automatically designing lead–lag compensation for a single-input, single-output linear plant (a common control design requirement). Later implementations integrated the rule-bases together, and also the two data structures (i.e. the problem and solution frames). Certain forms of non-monotonic reasoning used by a design engineer were achieved by using a forward rule interpreter and a small set of Truth Maintenance Rules.

The coupling of symbolic and numeric computations was achieved via the VAX mailbox. Procedure commands allow VAX/VMS sub-processes to start and stop external programmes, such as CLADP (a large control design package), which provide data for the expert system. In a similar fashion, two-way interchange between the representations is feasible. Expansion of the expertise within CACE-III includes access to SIMNON (a non-linear simulation package) and optimal controller design methods. Extensions considered for the updated DELPHI inference engine include improvements in the representation of uncertainty by incorporation of linguistic likelihood (cf. fuzzy logic). This would provide linguistic assessments, and combinations which would facilitate trade-off aggregations typical of the redesign process. This work has led to the MEAD (Multi-disciplinary Expert-aided Analysis and Design) project initiated by the US Air Force in collaboration with General Electric. This aims to provide a CACE environment for Integrated Flight, Propulsion and Structural Control (IFSPC) in aircraft design. The MEAD architecture compromises a Supervisor, a Database manager, an Expert System and a User-interface. The packages integrated into the architecture comprise MATRIXx (for linear design), GENESIS (for nonlinear simulation), ALLFIT (for providing low-order equivalent system models) and AUTOSPEC (for MIL specification verification).

An expert system to aid students in learning control design principles is described by Fortuna et al. (1988). Their system, ECOSYN (Expert Control Synthesis), provides assistance for compensator design for linear systems in the frequency domain. It handles lead–lag compensation, Guillemin–Truxal synthesis, pole-placement design and regulator tuning principles. Problem formulation is provided by way of closed-loop performance parameters, constraints etc. Facts and rules are established to determine which design methodology suits the problem specification. The overall structure of ECOSYN is shown in Figure 5, and each design tool has its own knowledge base devoted to its usage. The system is PC-based and uses an inexpensive expert system SEM written in Pascal.

2.3.2 Advanced control CAD and AI

A more ambitious project called MAID (Multivariable Analytical and Interactive Design) covers CAD for multivariable systems (Pang & MacFarlane, 1987). The motivation behind this work is that design tools for multivariable control systems are very complex and difficult to learn and use. MAID acts, therefore, as an instructor and guide through the design processes. The expert system is implemented using Xi running on a PC, while the design tools are coded in the CAD package MATRIXx running on an IBM mainframe. The design is broken down into a number of sub-problems, by dividing it into separate frequency regions: high, intermediate and low frequency

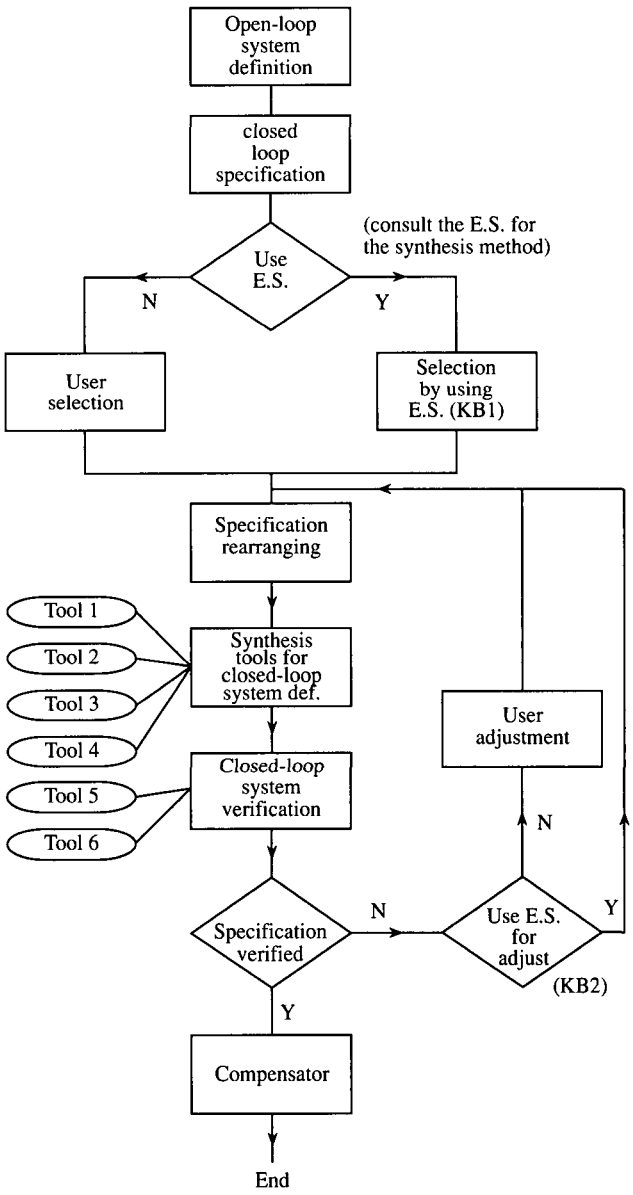

Figure 5 Functional structure of the ECOSYN expert system for educational use in control system design (from Fortuna et al., 1988).

(Pang & Boyle, 1986). Also the design knowledge is divided into different levels of complexity and organized into a hierarchy as follows:

Level One:
Simple Design Technique
Level Two:
Reverse Frame Approximation Technique
Level Three:
Complex Design Technique.

The advantages of organizing the design knowledge into different levels of complexity are stated below:

- The designer can be taken through the lower levels of complexity systematically before he tries to tackle his problem with a more complex approach.
- The manner in which the design "fails" at each level can give important clues about the possible success of greater levels of complexity.

- The explanation of how a design solution has been reached can be presented in a manner reflecting the level of complexity required by the design.
- The designer can rapidly determine whether his problem has reached an unacceptable level of complexity and thus decide whether he should modify his specifications rather than continue his search with a more complex design approach.

This general technique of problem partitioning (decomposition) should embody the following two rules:

- minimum coupling between components, and
- maximum cohesion of functions within a component.

This partitioning enables well-defined sub-procedures to be built. This has the significant advantage that the knowledge base is extremely modular. Thus, it is the one key to addressing the complexity issue. The following are the main advantages of rule-base partitioning:

- Rule-bases can be written, debugged, updated separately and then integrated into the overall structure. This will save time and errors can be identified more quickly.
- The scope of search is limited.
- One potential danger of production rule systems is that the system may get into infinite loops when the knowledge base gets large.

The use of a modular structure can help locate the problem. The structure of the knowledge base is shown in Figure 6. The following is a brief description of each frame. Each of them except the Supervisor can advise on the use of the appropriate functions in the control system design package MATRIXx.

Pre-design Analysis Frame:

The main functions of this frame are to analyze the model of the plant and check the open-loop stability of the plant.

HF-frame and HF Design Checking Frame:

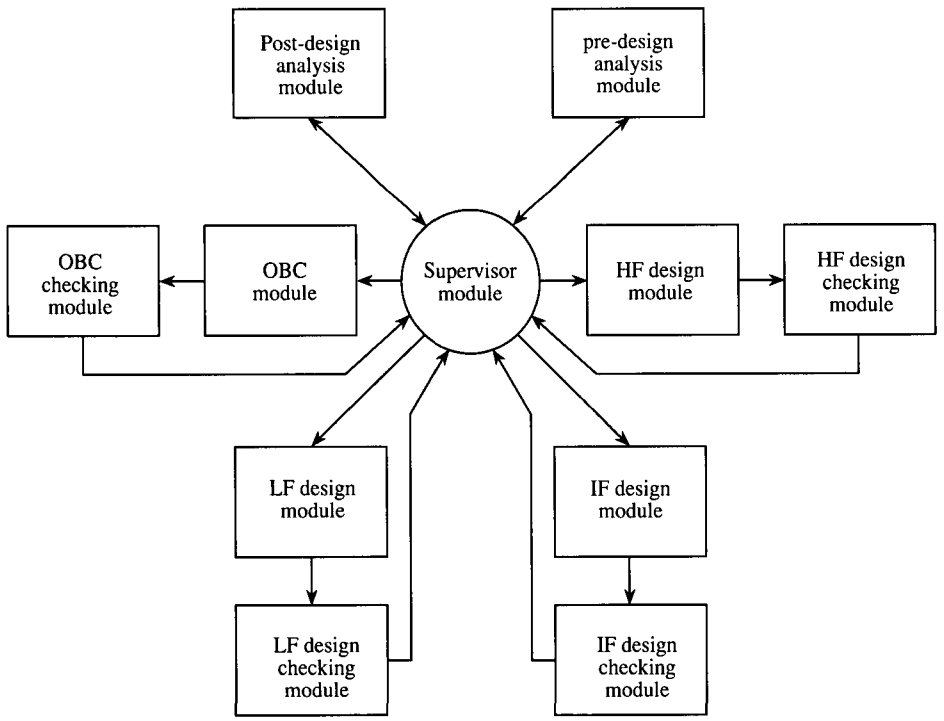

Figure 6 Structure of the knowledge-base for the MAID expert system advisor for advanced control system design (from Pang & MacFarlane, 1987).

The objective of these frames is to obtain good phase properties in the high frequency region.

IF-frame and IF Design Checking Frame:

The objective of these frames is to obtain good gain-phase properties in the intermediate frequency region.

LF-frame and LF Design Checking Frame:

The objective of these frames is to obtain good gain properties in the low frequency region.

Post-design Analysis Frame:

This frame will advise the designer on the analysis of the compensated plant (e.g. closed-loop step response both for linear and non-linear plant models).

In validating MAID using students and industrial engineers it became apparent that a lot of the detailed terminology used in the complex design techniques was unfamiliar. To alleviate this, an on-line thesaurus was used to assist in the learning phase. They used the expert system to provide what they call the 3T's of CAD: Transfer, Teach and Tailor. One should transfer knowledge by guiding a designer through techniques with which he is unfamiliar. One should teach the designer via explanation and guidance, and also tailor the knowledge in large engineering design packages by providing a supportive and flexible intelligent front-end to large software packages.

While the above considered off-line techniques, expert systems have been considered for on-line use. Sanoff & Wellstead (1985) describe a system which would facilitate the installation and maintenance of self-tuning adaptive controllers by non-specialized personnel. Numerous self-tuning algorithms are available requiring several design parameters to be chosen, particularly the sampling rate for the process. The expertise of self-tuning design is encapsulated in two expert systems collectively called CORTEX (Configuration and Run-time Expert). The first system is responsible only for the initial configuration of the controller. It expects the user to have a reasonable knowledge of control and the process, but none of self-tuning theory and practice. The result of this first interaction is the setting up of a data file which is read by a run-time expert system, composed of two parts run as foreground and background tasks. The foreground task operates in real-time and makes decisions such as forgetting factor settings, identifier disablement etc. The background task buffers the user from details of controller operation, providing explanations and reports. Its main function, however, is to provide high-level decisions, such as changes in sampling rate when required. To achieve necessary speeds the knowledge base for the foreground task is compiled into a decision tree.

2.3.3 Knowledge representation and databases for CAD

The use of an object-oriented approach to CACE design is suggested by Zygmont (1987). Beginning with Simula the oriented-oriented approach has had simulation in mind. This can be utilized by viewing CACE as a process of simulating discrete events (i.e. stages in the design process). Thus a class hierarchy is established starting with a top-level Manager, and branching into Models, Controllers etc. A major class would be Interfaces, which would allow message passing to external packages, as in the case of the CACE-III software. The question of knowledge representation is addressed by Rimvall (1988), who noted the need for a more flexible information-retrieval system than the classical help-function. To achieve guidance through a complex set of software tools Rimvall & Bomholt (1985) developed the IMPACT query facility, coded in ADA. This provides a command language diagnostic assistant, augmented with a Prolog-based data-organisation and retrieval section. Taylor (1987) uses an expert system to help the user manage the data. This expert system will also be responsible for the integrity/consistency checking necessary in the management of models and data for large industrial projects. Another issue addressed by IMPACT is the multi-dimensional matrix data structure required in modern control design techniques, which is not adequately supported even by well-known design packages such as MATLAB.

A core data model for control design is proposed by Maciejowski (1985) who noted the need for much more flexible data types than those provided by conventional databases. The core model has

as its highest level entity the object SYSTEM with attributes DEFINITION (covering subsystem decomposition), CONNECTIONS, CLASSIFICATION, PROPERTIES and NAME. The use of object-oriented concepts in database retrieval for CACE models is further developed by Maciejowski (1988), who prototyped a special data language for interaction with MATLAB. These themes are also considered by Astrom & Kreutzer (1986), using the underlying notion of system representation. They note that systems can be described via matrices, polynomials and transfer-functions, but suggest that even more flexible data abstractions are desirable. The problem with matrix descriptors is that it is difficult to implement operations which are naturally viewed as operations on systems. Thus, particularly when dealing with large systems, it would be desirable to have a hierarchical interconnection of subsystems. They divide the object classification into systems structure and system behaviour. Graphical representations, such as block diagrams, signal flow diagrams and bond graphs are commonly used to portray systems. To capture such descriptions one used the abstraction of a box with input and output terminals. It can be represented as an object with the variables Name, Inputs, Outputs, Subsystems, Connections. The methods attached to a system object would include functions such as creation, addition, deletion, loops, aggregations, conversion etc. The system behaviour can also be described via the object paradigm. Categories include Static, Qualitative, State space, Transfer Function, Describing Function etc. These would have methods attached to them, for example the Static object might have Linearize, Maxgain, Mingain, Degree of linearity, Operating range etc. A small prototype, written in Lisp, was developed using the above concepts, and illustrated via a model-reference adaptive control structure.

2.3.4 Planning, specification and configuration

Work is described by Trankle & Markosian (1985) who use a planning expert system to provide a controller which can adapt to catastrophic plant changes. Three knowledge-base modules are provided, being a System Identifier, a Control System Designer, and a Control Implementation Supervisor. The expert planning system uses a tree data structure, whereas the design process comprises a sequence of operations that recursively traverses successive nodes of the tree. Expert design rules are represented as operators. Control design differs from traditional planning problems in a number of ways. Most domain-independent planning has concentrated on providing general frameworks, whereas control design has many specificities and detailed algorithmic design knowledge. Also, control is a continuous-valued domain, whereas planning is viewed as a discrete-event process. Further, uncertainty exists in control design, partly as a result of approximations in the plant models, and also because some of the design effects cannot be predicted before the decision is made. Thus, the STRIPS paradigm (Cohen & Feigenbaum, 1982) is inadequate because of multiple side effects caused by design operations. Extensions were proposed which incorporated multiple heuristics embedded in a global database accessible to the planner. The CACSD planner was coded in Lisp on an AI workstation, and coupled to a VAX for the control design suites. The system has successfully designed a flight control system for a fighter aircraft operating in a terrain-following mode. This mode has stringent requirements for regulation of altitude and rate of climb, and poses a particularly difficult problem for the designer.

A very different design domain for control is that of Programmable Logic Controllers (PLC), which are now very frequently used in industry (Vingerhoeds et al., 1988). The hardware for PLC's has kept up with modern technology, but the principles of operations and peculiarities of programming have not. The major feature of PLC's is implementation of control actions via boolean functions. The first basic method comprises combinational applications, with static data. The second group occurs in sequence control systems, in which the logic travels through a number of states. Programming is via simple, but tedious, ladder diagrams, and in this work automatic programming mediated via Prolog is explored. The expertise required is divided amongst three types of engineer: the process engineer who has a detailed knowledge of the plant, the installation engineer who sets up the PLC parameters, and the service engineer who inserts test rules and extra logic to assist in fault detection. The expert system then provides guidance on hardware

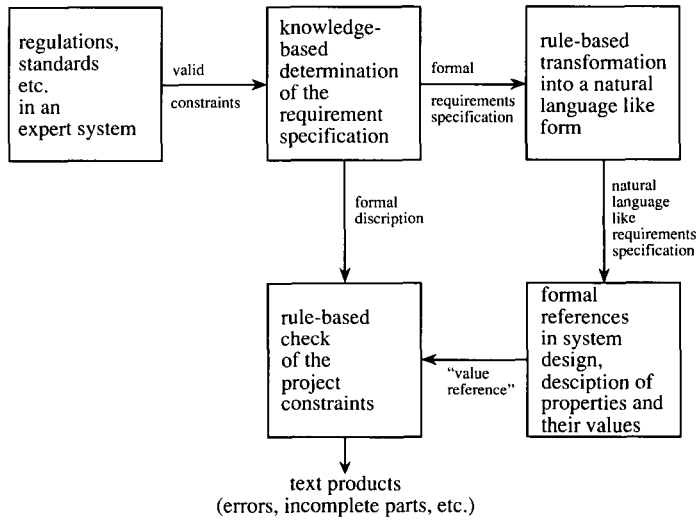

Figure 7 Knowledge-based structure for configuring and checking constraint specifications for process automation systems (from Wagner, 1988).

configuration, and produces the software code necessary for both runtime control and service diagnostics.

A higher-level of abstraction is required when the generation of specifications for the process automation systems is considered (Wagner, 1988). In this work, a computer tool is described which helps to determine a complete and consistent specification of constraints, such as performance, reusability, security, legal regulations, and quality standards. This is described as "requirements engineering" whereby problem statements are defined and translated into functional specifications. Different approaches to constraint specification include Checklists, Requirements/Properties Matrix and formal languages such as RML (Borgida et al., 1985). To overcome limitations in these methods, Wagner prototyped a knowledge-based method in connection with an Integrated Project Support Environment (IPSE) called EPOS, which is being used in numerous European industrial projects. The tool is realized as a collection of different expert systems, with an overall structure as shown in Figure 7. The emphasis is on automatic translation of a formal constraint description of the problem into a natural-language-like requirements specification.

A software engineering style is demonstrated in the work on knowledge-based retrieval of reusable project results by Beutler (1988). The reuse of software code is an important component in increased productivity, better maintenance, and improved reliability and quality. IPSE tools now provide project databases, which contain technical and managerial information of the whole life cycle of projects. This information includes concepts, control systems and pieces of real-time software. Beutler has produced a prototyped knowledge-based tool, PROSA, which uses a "descriptive search" approach as a query language. The tool is integrated with a database library of obtained reusable project results. This contains short project descriptions, a classification system and indexing rules for extracting and concentrating the information. The classification scheme uses a poly-hierarchical structure based on a multifaceted approach due to Ranganathan (1957). This is basically a frame-based approach using inheritance between the facets. The query formulation is via a "descriptive search" which looks for similarity between the project descriptors (stored project results) and the search descriptor (investigator's interest). The search produces a ranked listing, with the most relevant projects at the top. Automatic indexing is required to produce project descriptors from many different project databases. This is done by evaluating indexing rules from the databases to concentrate project results towards the search descriptors. The indexing rules which contain "terminological" knowledge are represented by a special description format called CONTEXT. Thus, taxonomy is specified via FACET objects, results are stored via PROJECT

objects, and terminologies via CONTEXT objects. The object SEQUENCE then allows for whole series of search strategies to be executed. PROSA contains search procedures for accessing ASCII-files and EPOS project databases.

Little has been said in this section about HCI and graphics aspects for CACSD. The knowledge-based manipulation of signal flow graphs via Prolog has been extensively explored by Barker and his co-workers (1986), and utilized for the common block diagram approach for control systems design. Adaptive dialoguing features have been described by Munro et al. (1986).

3 Real-time control

Having considered in the previous sections aspects of AI related to the off-line design of control systems, we shall describe in this section the impact of AI-related concepts on the implementation of AI-related concepts on the implementation of on-line control techniques. The real-time nature of such systems poses particular problems which are not evident in off-line expert systems. After a review of requirements in real-time control and consequent architectural details, we shall consider details of a number of implemented systems.

In a survey paper, Taunton (1989) describes a programme of experimental AI applications initiated by Sira (Scientific Instrument Research Association) in collaboration with the process industry. This programme covered a range of topics from supervisory control and troubleshooting to planning and scheduling. The overall theme in these areas is that of Process Management and Operation, where benefits from knowledge-based strategies might accrue from: higher yield from input resources; higher utilization of resources; reduced production losses; increased levels of automation. Experiences in the field of Process Monitoring show four problems in this demanding area which are not commonly met in off-line expert systems. These are: integration of vast amounts of numerical instrument data with symbolic knowledge; reasoning about events sequenced in time; prescribed time decision making; system validation for both inference engine and knowledge-base components. The second area is that of Troubleshooting, or Fault Diagnosis. In this area, several levels of criticality may be defined which regulate the speed at which decisions must be reached. Thus, low criticality (in terms of time) matters could be those of control, and high criticality could be safety-related problems. The third area is Supervisory Control, which is primarily concerned with tuning the process to obtain maximum product yield. Although algorithmic optimization of some processes in the petrochemical industry can be successful, usually supervisory control is heavily dependent on the expertise of the process operating teams, which is itself often variable. The fourth area is Planning and Scheduling, which is a management function concerned with ensuring that fluctuating demand is met via the most economic utilization of production resources. This requires knowledge about technical and economic constraints, and is more of a creative than an analytical task. It is suggested that rule-based systems are inadequate for this function, and that a model-based reasoning approach is necessary. Thus, to cover all the above topics a powerful knowledge representation and reasoning framework is required, which should include the ability to deal with casual, temporal, spatial and taxonomic knowledge. Taunton suggests that the Process Management System of the 1990's will include a number of reasoning systems cooperating via a common knowledge base.

The theme of architecture for real-time expert control is considered by several authors. The common thread is that of the global knowledge-base, giving rise to a *blackboard* architecture. The blackboard model for industrial control was proposed by Haton (1983) based on concepts from image understanding and speech recognition AI applications (Erman et al., 1988). To provide an Integrated Intelligent System for real-time control, Rao & Jiang (1989) suggest that integration is required at the following levels: knowledge of different disciplines; various objectives, e.g. research and development, engineering design, process operation and control; different expert systems; symbolic and numeric computation; different information systems. To provide these features they propose the concept of a Meta-System, which should have the following functions:

- Coordinator to manage all symbolic reasoning and numeric computation.
- Distribute knowledge into separate expert systems and numeric routines, ensuring modularity for good knowledge management.
- Integration of new, easily acquired, knowledge.
- Provide an optimal solution for the conflict solutions and facts among different expert systems.
- Provide for the possibility of parallel processing.
- Communicate with measuring devices and final control elements.

The blackboard approach is also advocated by Arzen (1988), based on experiences with a prototype system which will be described later. For real-time control, such a framework involves the activation of different knowledge sources both in sequence and in parallel. A typical case when knowledge sources act in parallel would be during steady state control of a process. One knowledge source takes care of the actual control algorithm, while other knowledge sources handle different monitoring aspects. Three different strategies for combining knowledge sources into sequences have been implemented. The first is to use primitives that let knowledge sources activate or deactivate each other. The second is to have a number of pre-stored sequences, such as initial tuning protocol, and return to steady state of control under certain alarm conditions. The third, and most complex, method is to dynamically generate sequences. In this strategy, each knowledge source can be viewed as an operator that transforms the system from its initial state to its goal state. A sequence is recursive generated by comparing the desired goal and the current state with conditions of the operators.

Voss (1988) considers architectural concepts which are strongly influenced by the German joint industrial/academic venture TEX-I (Technical expert system for interpretation, diagnosis and control of technical systems). To mediate between expert system and process control components, he proposes a Process Interface (PI), whose main function is that of data abstraction. The general tasks of the PI fall into the following categories:

- simple data abstraction or filtering
- detection of situations leading to expert system activation
- detection of diagnoses on process signals
- mediator between process control and expert system sections
- adjust control actions proposed by the expert system to suit the process control system.

In addition to the PI, Voss notes the need for a global manager which he calls the monitor, which is analogous to the Scheduler of Arzen and the Meta-system of Rao and Jiang. The knowledge sources are considered to be modular processes with priorities assigned by the Monitor. In this way parallel processing and other distributed problem solving schemes can be utilized. It is noted that this blackboard style has additional constraints, in that the individual knowledge processes may be interrupted at any point of execution. These concepts have been incorporated into the TEX-I project using the hybrid tool BABYLON (di Primio & Brewka, 1985). Two other themes are addressed by Voss, one of which is model-based reasoning. The need for deep modelling is noted, with the comment that every intelligent autonomous system must have or develop a model of its environmental world. In particular, it should have a model of itself, if it is to be able to manipulate its own environment. TEX-I provides a model of a system via a hierarchical representation of its processes and components. The other theme is that of Temporal Reasoning and Truth Maintenance. In real-time control this must provide an interrupt mechanism allowing suspension of process activities. This requires histories of values involving time stamping of lower level signals, and qualitative temporal logic descriptions for higher level signals. Extensions are necessary to provide for: efficient computation in hierarchical and mixed quantitative and qualitative relations; focusing onto particular temporal episodes; garbage collecton to reduce older parts of histories. Multiple contexts are required for aspects such as fault diagnosis, and to achieve these goals a powerful and efficient Reason Maintenance System (RTS) is required.

In the following sections, AI in real-time control applications will be considered under the two categories of Direct Expert Control (DEC) and Supervisory Expert Control (SEC). In DEC

applications AI concepts are utilized to provide the primary control functions, whereas in SEC the control functions are provided by conventional numerical techniques while AI methods are used in the hierarchical supervisory mode. The DEC section is further sub-divided into the areas of fuzzy control, rule-based control, and real-time toolkits.

3.1 Direct expert control

3.1.1 Fuzzy control

For many years, fuzzy control has been studied based on the seminal work of Zadeh (1965) on fuzzy logic theory. The basic concept behind this is that humans reason imprecisely and not via numerical calculation. Thus, early work by Assilian & Mamdani (1975) used fuzzy logic to replicate human operator behaviour on a steam plant. A survey of early control applications is given by Tong (1977), and extensions to self-organizing fuzzy logic control (SOFLC) were described by Procyk & Mamdani (1979). A wide range of industrial processes have been investigated via fuzzy control, including heat exchangers, blast furnaces, waste water treatment, train transportation, and numerous other fields described in Sugeno (1985). More recently it has been applied to biomedical systems involving muscle relaxation control in operating theatres (Linkens & Mahfouf, 1988) and blood pressure management via drug infusion. Self-organizing systems have also been further investigated both for satellite attitude control and muscle relaxant administration. In this later application one is dealing with the human body which has massive inter-patient variability in the dynamic behaviour resulting from drug infusion. Linkens & Hasnain (1989) show that SOFLC can provide good transient response to a single demand at the commencement of an operation, starting with a totally empty fuzzy rule-base. It is also shown that SOFLC is a robust strategy in terms of dealing with different drug pharmacokinetics and additive noise contamination.

Fuzzy control has also been studied for motion control of autonomous guided vehicles (AGV). This is a classic control problem for robotics research, whereby a hierarchical strategy is commonly implemented. Three levels are often considered, being the planner which operates on a global terrain map, a navigator which utilises a detailed local map, and a motion controller for obstacle avoidance. Harris & Read (1988) describe a knowledge-based fuzzy motion controller, and refer to earlier fuzzy decision rule methods for path determination of AGV's which progressively discover the environment, for car parking and for hierarchical control of unmanned robots. The lateral motion control was approximated by a linear second order system, and consideration given to necessary fuzzy operations required with commensurate sampling rates. They concluded that for a natural frequency of 10 Hz, a sampling rate of 50 Hz would be attainable, and 19 quantization levels would be acceptable for the error and error rate signals. In simulation studies they found good robust properties under large changes in system parameters, additive noise, and either ramp or sinusoidal input demands.

Concepts of fuzzy logic are also in expert systems which attempt to reason with uncertain data. This was true of early systems such as MYCIN in medicine and PROSPECTOR in mineral exploration. Similar ideas are incorporated for real-time systems, an example being that called PATHOS (Permantier, 1988). In this case the coupling of symbol-oriented knowledge bases with numerically-oriented technical processes is accomplished via linguistic variables and fuzzy logic. PATHOS is a frame-based representation language allowing for uncertainty propagation. Standard fuzzy operators are used, together with multiplication for combining certainties for propagation, and linear interpolation for accumulation of uncertainties. The link between the process and the expert system is made via "fuzzy comparison operators" in the condition parts of the rules. PATHOS has also been used for optimization problems where exact mathematical models do not exist, or large numbers of parameters are involved. To perform optimization, one can introduce the states of a system as "versions" and represent these in chronological order, and thus consider the inclusion of time. The concept of "versions" was added to PATHOS without changing its original semantics, and new constraints added to analyse the memory of versions for optimization.

In this way, one can study Modelling problems where time is included in the problem solving process.

The well-formulated theory of fuzzy logic inference and control means that it can be implemented using hardware. Togai & Watanabe (1985) describe a fuzzy inference chip which would give a performance of 80 000 Fuzzy Logical Inferences per Second (FLIPS). The VLSI inference engine consists of three major parts: a rule set memory, an inference processing unit, and a controller. An initial implementation contains 16 rules, with each rule having 124 bits of antecedent and 124 bits of conclusion. Application areas could include intelligent robot systems and real-time command and control systems.

3.1.2 Rule-based control

Arising out of the early work on fuzzy control by Mamdani and his co-workers, the use of rule-based control for cement manufacture has been widely cited. Thus, Blue Circle Industries have developed the LINKman system for both control and optimization of their cement kilns with the aim of reducing production costs (Haspel, 1989). The cement kiln dynamics are very difficult to model mathematically, and are an ideal candidate for rule-based rather than algorithmic control. The problem is multivariable in that the four independent parameter setpoints of coal, feed rate, fan and kiln speed all strongly influence the three key process measurements of excess oxygen, burning zone temperature and exhaust gas temperature. The rule-base for control is obtained by observing experienced operator behaviour in successfully controlling kilns manually and translating this into rules such as:

IF Temperature A is small positive
AND Oxygen is medium negative
THEN fuel rate change is medium negative.

In experiences with LINKman the following benefits are claimed for cement manufacture: reduced fuel consumption (4%); increased clinker outputs (4%); improved product quality (4%); reduced milling costs (12%); reduced refractory temperatures; improved refractory life (30%); kiln exhaust gas NOX levels reduced (25%). Other qualitative benefits accruing from the rule-based control are claimed to be: increased kiln process knowledge; reduction of normal shift variations; added management data collection and logging facilities. Via a process engineer's interface the rule-base can easily be altered or incremented, and it is claimed that cost benefits have been markedly better when works engineers are actively involved in the rule-base refinement during post-commissioning periods.

LINKman has also been used by BP at a lubricating oil refinery (Thomson, 1988). This project was sponsored by 10 industrial partners and the Department of Trade and Industry. The particular process chosen was that of furfural extraction. The furfural plant removes unwanted aromatic compounds from oil using a solvent (furfural) extraction process. Furfural is first mixed with oil in a contactor, and subsequently removed by means of a series of distillation columns. The benefits resulting from the project were: improved level control in 4 unit processes; fuel saving via heater pass balancing; improved specific gravity control and yield optimization which provided a payback period of less than 12 months. In acquiring the knowledge for the rule-base it was found that the bulk of the work was in fixing the linguistic descriptors such as HIGH from the process engineer and operator experience.

Under an Alvey community club project, RESCU was developed as a real-time system on an industrial plant. The club comprised 23 industrial partners, 3 universities and a systems contractor (Leitch & Dulieu, 1987). The application chosen was that of quality control of a chemical process, which is not susceptible to conventional, algorithmic process control methods. RESCU was rule-based, and contained two major components. The first was the knowledge base itself, developed in its own representation language, and consisted of both passive and active (i.e. time-dependent) knowledge. The other component was a set of utilities, written in POP11, providing reasoning

mechanisms, and executive functions. The inference engine provides single and multiple forward and backward chaining mechanisms, and coordinated the knowledge flow via a shared working memory or blackboard. To cope with the temporal nature of process information RESCU provided time stamped data items, giving both specific event occurrence information and also time interval validity indicators. In drawing conclusions from the project, the need for a frame-based representation was noted, and also the desirability of early prototyping and well designed and engineered HCI emphasized.

Other real-time KBS include ESCORT (Sacks et al., 1986), EXTASE—an alarm processing expert system (Jacob et al., 1986), PROP—an expert system for monitoring water pollution in power plants (Gallant et al., 1985). The problem of temporal reasoning is handled by RESCU in a modular approach. ESCORT uses a graceful degradation approach where events are assigned a priority rating. EXTASE uses best first conflict resolution to investigate the most important hypothesis first. Uncertainty is handled in RESCU by attaching "certainty factors" to facts and rules. PROP keeps a "credibility value" for each hypothesis, whilst EXTASE attaches factors to causal couplings and ESCORT uses a three valued logic with an explicit "unknown" value. In terms of knowledge representation, RESCU and ESCORT use shallow explicit process models, whilst EXTASE has a deeper explicit model representation. In contrast, PROP's model is implicit with the event graph formalism used by the system.

ESCORT has recently been used on a butadiene plant at BP Chemicals, Grangemouth. It is a relatively small plant with about 60 control loops, and does not have any problems of information overload on operators (Thomson, 1988). The system runs on two computers, with a Gateway machine (DEC MicroVAX) connecting to the Process Control Computer, and a Lisp machine (Symbolics 3620). Knowledge representation uses an object-oriented style of class inheritance. A diagram interface allows the P & I (Piping and Instrumentation) diagram of the process plant to be entered into the system (Mason, 1989). This provides connectivity information to ESCORT. Further knowledge is input on process fluid data, principles of loop operation, transducer and control valve failure modes and operation of heat exchangers, providing information on causal couplings. The project is of a monitoring nature, aimed at identifying the first sign of a plant problem, diagnosis of the problem, and advice to the operator on suitable remedial action.

Although the above examples of KBS for real-time have concentrated on industrial applications, similar developments have taken place in medicine. Thus, the control of on-line drug therapy in operating theatres and intensive care units is an example of clinical engineering. In areas where direct measurement of physiological variables is difficult, if not impossible, the use of AI techniques is appealing. An example of this is the control of unconsciousness (or depth of anaesthesia) for patients undergoing operations. The anaesthetist uses a mixture of clinical signs (i.e. qualitative measures) and physiological measurements such as ECG (i.e. quantitative data) to decide a drug regime. An attempt to provide an expert advisor for this scenario uses multiple signs and real-time measurements together with an internal model of pharmacokinetics of drugs to recommend dosage rates (Linkens et al., 1986). Uncertainties are propagated using Bayesian inference, and the whole system is written in "C" running on a low cost Atari machine. The knowledge base has been refined and subsequently validated via a number of clinical trials involving several surgical procedures.

Returning to the theme of direct rule-based control, Krigsman et al. (1988) consider a real-time system targeted onto Atari ST machines using Real Time FORTH as the host language. Generation of the rule knowledge involves phase plane segmentation (i.e. error and error-rate signals), which is commonly used in fuzzy logic control. They propose five knowledge layers, four of which are for direct control action, and the other is for supervision. The first layer divides the phase plane into 8 regions with max and min control values. The second layer has 24 area classifications and provides finer tuning on the control action. The third layer has 48 area classifications, and again provides parameters for the control actions (different to layers 1 and 2). The fourth layer contains a set of rules which attempt to force the system to follow a prescribed time trajectory, and thus provides a form of MRAC (Model Reference Adaptive Control). The fifth

layer provides supervisory functions such as determination of speed of the system, adaptation of boundary regions for classification of the layers, and recognition of entry into the steadystate area of the phase plane. The whole knowledge base comprises 170 rules, with a maximum calculation time of 180 ms.

Fault diagnosis is implicit within much of the work already described, and many expert systems are explicitly targeted for this rea of industrial importance. A system has been produced to provide early warning of possible failure in helicopter rotor blades by Stewart Hughes (Karwatzki, 1987). The functional structure of an intelligent alarm system is described by Ogard & Woods (1988) and illustrated via a gas compressor example. Many fault diagnosis systems exist, and the whole field is of great importance in industry.

3.1.3 Real-time toolkits

In an early prototype, a real-time shell for intelligent feedback control called ARTEFACT was developed in Prolog (Francis & Leitch, 1984). Knowledge is represented by a set of conditional goal/subgoal pairs. These are used to represent subsystem relationships within a complex system, and taken together from a relational model. Uncertain inference is used to deduce applicable relations consisting of goal/subgoal pairs with associated truth values. These are combined by a planning routine to deduce paths involving changes in the control inputs of the system that satisfy a prescribed objective. The method was illustrated on a fluid level control rig comprising 2 coupled tanks, with extension to an n-coupled tank case. This extension showed the considerable reduction in the number of heuristics achieved with the use of a relational model. Also, relations between subsystems are relatively easy to obtain, and do not depend on system complexity.

Under an ESPRIT project a KBS Toolkit, called QUIC, has been produced, and provides advanced development tools for a range of control, fault diagnosis and simulation functions. This utilizes concepts of qualitative modelling for complex phyiscal systems and the toolkit has been validated on realistic industrial demonstrators. These include diagnosis for the thermal cycle of power plants, data interpretation from a satellite, and regulatory control and automatic start-up/ shut-down of a cement manufacturing plant.

The MUSE real-time toolkit (Clare, 1989) claims to be flexible in how knowledge is represented, open in its interconnection with other software and processes, and efficient in terms of execution and development effort. It utilizes both rule and model based representations with an object-oriented language. By using multiple knowledge sources which are scheduled as required it provides run-time efficiency. A maximum performance of 1600 rules per second on a SUN 4 under UNIX has been achieved. An example of its use of causal model reasoning is that of alarm handling and fault diagnosis of helicoptor engine systems. To tackle the difficult area of failed sensors, a model was developed of the relationships between various systems and their dependencies. From this knowledge certain sensor states can only be the result of sensor failure. Other applications include predictive maintenance of marine diesel engines, and the control of highly viscous glues for holding surface-mount components during wave soldering (West et al., 1988).

A real-time expert system development tool with a long pedigree is G2 (Moore, 1989). It originated in PICON which ran on Lisp machines, but has been refined and ported onto SUN workstations more recently. Current applications of G2 include telemetry monitoring of flight data, manufacturing processes and assembly, a biochemical process, a variety of chemical and petrochemical processes, and closed-loop robotics. Prototypes include network monitoring, financial transaction monitoring, office workflow planning, life support systems, semiconductor manufacturing and nuclear reactor safety products. G2 enables knowledge to be entered via graphical icons, where individual objects have attributes and connectivity. To achieve real-time efficiency G2 uses a focusing approach, whereby the inference engine continually scans the scene, and then uses meta-knowledge to determine which knowledge source to employ as the focussed area. Truth maintenance involves changes in inference even when no new data are available, since time is a factor in validity or certainty of inference. This is achieved by attaching an expiration time to each value and then propagating this forward through multiple levels of inference. G2 provides a

simulation facility for investigating consistency and completeness of knowledge. Failure scenarios are a powerful way of validating knowledge bases, and fault conditions can be tested by simulation.

3.2 Supervisory expert control

Although it is clear in the foregoing section that supervision is an important component in all KBS, this section primarily considers the situation where supervisory rule-based assistance is attached to control systems which are basically algorithmically-based (e.g. classic PID controllers). Krigsman et al. (1988) refer to a number of schemes of this type. These include scheduling of PI parameters depending on error boundaries, adaptation of PID parameters via heuristic reasoning, adaptation of PID parameters via pattern recognition (an example being that of the Foxboro EXACT system), and supervision of adaptive control schemes.

The field of intelligent auto-tuners for PID controllers is now well populated with industrial manufacturers (Epton, 1989). The auto-tuner mimics the practical procedures which might be carried out by an engineer during commissioning. They work by monitoring and analyzing the process error signal either in response to a start-up transient or to injected changes in set-point or to load disturbances (Figure 8). Once analyzed in terms of rise time, overshoot period etc, heuristic rules are used to modify the controller coefficients using techniques often based on tuning concepts due to Ziegler and Nichols. An example of this is the Foxboro EXACT system (Dearden, 19898) which uses a pattern recognition system with the claimed advantage of: no mathematical model needed; no disturbance to the process; operation familiar to instrument engineers; simple to implement; provides continual adaptation; handles varying dead-time. The main candidates for auto-tuning are loops involving pressure, temperature and composition, whereas level, flow and cascade control are less likely to benefit. An alternative scheme due to Satt Control, via Astrom and co-workers at Lund, provides for gain scheduling and relay auto-tuning (Tetley & Ulff, 1989). The auto-tuner works by temporarily replacing the PID algorithm with a function which corresponds to a relay with hysteresis. This induces a deliberate oscillation in the process output, from which by measurement of period and amplitude it is possible to derive suitable PID parameters. This system was prototyped using the expert system architecture described by Arzen (1988).

The wider issue of supervisory expert control is addressed by Astrom et al. (1986). They noted that a good industrial PID regulator already has heuristic logic which deals with issues such as switching smoothly between manual and automatic operation, transients due to parameter changes, effects of non-linear actuators, wind-up of integral term, etc. Further, they observed that algorithmic-based adaptive control schemes have for a long time included "safety nets" which are

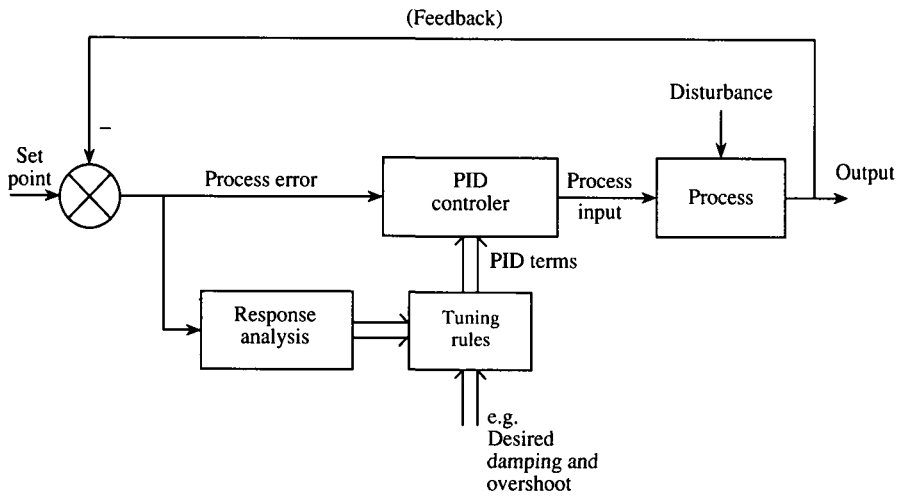

Figure 8 Schematic diagram for a response analysing auto-tuner for process control (from Epton, 1989).

heuristic rules for providing safe control systems. Necessary functions to perform these tasks can be grouped into knowledge sources as follows:

- Main control: minimum variance control; minimum variance supervisor; ringing detector (i.e. unwanted oscillations); degree detector (for correct order of the algorithm)
- Estimation: parameter estimation; estimation supervisor; perturbation signal generator; jump detector
- Self-tuning: self-tuning regulation
- Learning: set regulator parameters; smooth-and-store regulator parameters; test scheduling hypothesis
- Main monitor: stability supervisor; compute means and variances.

The database for such an architecture should include event lists to deal with time-varying aspects, and hypotheses lists with different levels of abstraction. In addition it should store processing history in a manner suitable for automatic learning.

The above ideas have been replicated in medical applications. One example is that of supervisory control of artificial ventilators for newly born babies (Zhang et al., 1989). The primary control functions are performed by a numerical self-tuning algorithm called Generalized Predictive Control (GPC), while the overall supervision is provided by a rule-base elicited from clinical experts. Similar concepts are being developed for muscle-relaxant anaesthesia which lends itself to algorithmic control, unlike that of depth-of-anaesthesia mentioned earlier.

3.3 Human computer interaction (HCI)

HCI requirements have been implicit in all of the off-line and on-line systems engineering activities mentioned so far. In this section, therefore, only one specific area of particular importance to real-time control is considered. This is the design of computer interface pictures for Operational Control Rooms.

A KBS for generation of control room views is described by Elzer et al. (1988) under the ESPRIT project PS56 GRADIENT. This has led to prototype software called MARGRET (Multi-functional All-puRpose GRadient Experimental Test environment) to provide an Intelligent Graphical Editor (IGE). Knowledge acquisition via interviews with picture designers and control room designers revealed a complex multi-disciplinary design, as shown in Figure 9, which can be explained by applying models developed by Rasmussen & Goodstein (1985). It was concluded that actual drawing of pictures is only a small part of the task, and that the whole design constitutes an engineering problem, albeit not well formalized. A number of types of knowledge are involved, which can be classified as:

- knowledge about the specific application;
- knowledge about design procedures;
- knowledge about user behavior (e.g. possible operation sequences);
- ergonomic knowledge.

The designer extracts information from sets of pictures used successfully from previous related applications, and merges this with both static and dynamic details for the present project. This requires searching through large files and databases and coping with inconsistencies in the knowledge obtained. MARGRET uses a knowledge representation scheme comprising three parts: application model; function model; picture pyramid. The application model is obtained from plant details such as P & I diagrams and S & C (Supervision and Control) diagrams, and is component-based. In contrast, the functional model describes substructures, using "Functional Scripts" (Zinser, 1988). Functional scripts are similar to frames and consist of a set of nodes representing process and control equipment or subsystems. The scripts contain both "functional topology" and "structural topology", and together they form a multi-level semantic network. The

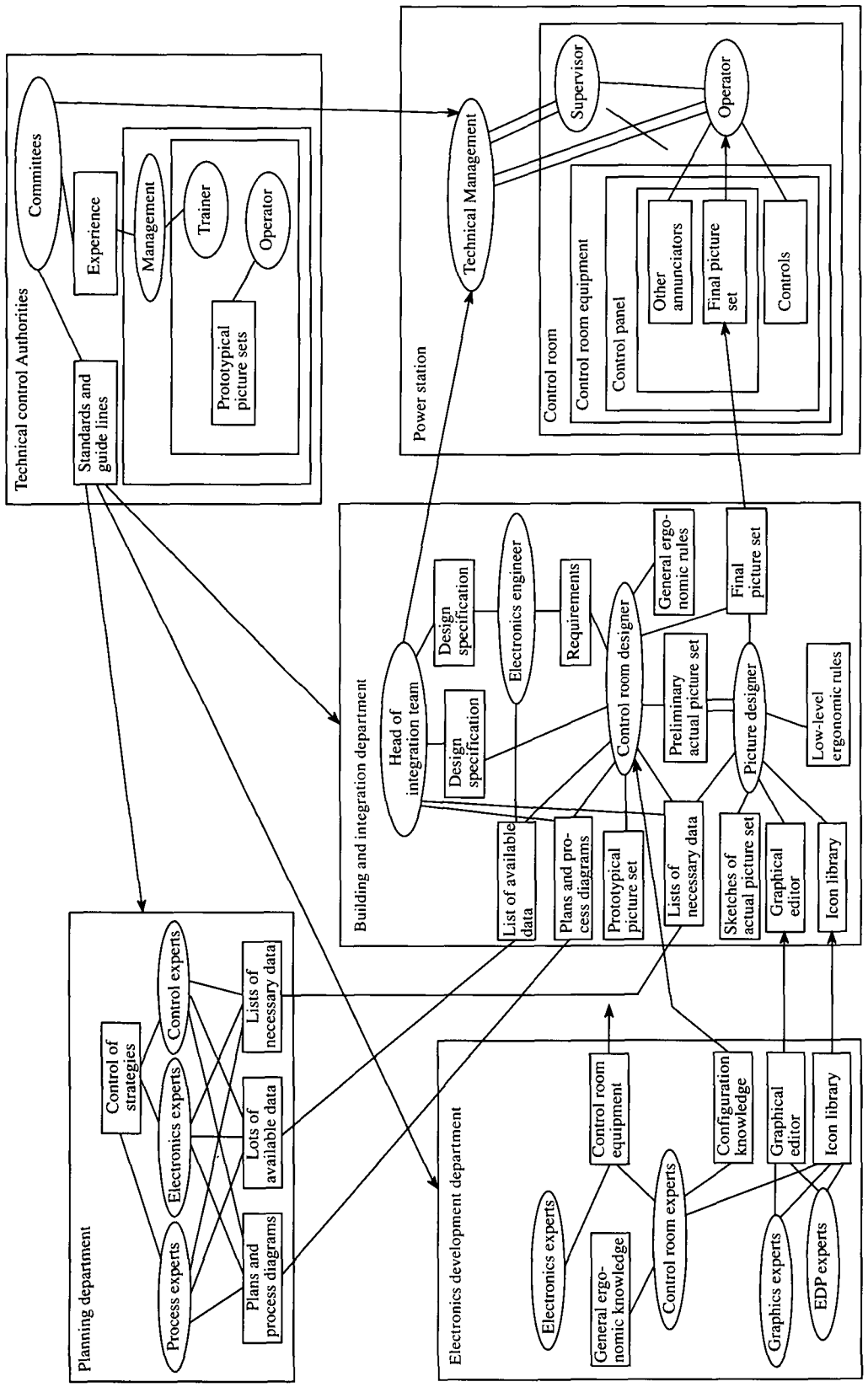

Figure 9 Inter-disciplinary influences on the ergonomic design of control room pictures (from Elzer et al., 1988).

"picture pyramid" is an object-oriented graphic structure where each icon is directly accessible for information retrieval, process control interaction, and change in graphical representation.

Using the IGE as a tool, the principal steps of the picture design process are as follows:

- the designer selects appropriate subsystem and operator tasks;
- the system automatically generates a technically correct graphic display;
- the designer improves the layout and aesthetic quality of the picture;
- the designer finishes the picture by filling in vountary details etc.

The IGE provides facilities for moving, rotating and scaling icons, and provides the binding of picture elements to process variables, thus producing a dynamic picture. Each picture can have alarm limits associated with it, together with a function description attached to it. These facilities are very similar to those described under the EASE+ environment in Section 2 where the modelling system KEMS was discussed.

Related work by Kolseki et al. (1988) deals with the additional concept of an expert system for the static ergonomic construction and evaluation of industrial control views (called SYNOP). It is well known that control room operator perception is affected by factors such as the use of colours, contrasts, size of characters, density of information etc. SYNOP is written in Lisp and uses the graphics package GKS for creating views. Its use comprises three phases:

Phase A:
the graphic view is interpreted into a semantic network;
Phase B:
the ergonomic knowledge rules are inferred from the semantic network. Meta rules are used to focus attention onto particular aspects of the ergonomic evaluation. Some of the sub-rules are executed automatically (e.g. change of colour); whilst others merely give advice to the designer (e.g. graphics layout style);
Phase C:
the system builds the graphics files and provides ergonomic advice files.

SYNOP has been evaluated via operator picture construction for an experimental power plant. It is intended to integrate SYNOP into an IGE having the following steps in an "ergonomic design loop".

Step 1: define the control views specification via task analysis. A KBS called Decisional Module of Imagery (DMI) is being developed for this.
Step 2: use SYNOP to define the graphic objects, after which each view must be dynamically evaluated. This will lead to reconstruction of the views and probable modification of step 1.

The overall design cycle is shown in Figure 10, which illustrates the feedback nature of HCI design via an engineering approach.

4 Discussions and conclusions

4.1 Summary of work to date

Not surprisingly, many of the current issues in AI methodology have appeared in this review of control system utilization of such techniques. It is particularly noticeable that industry has paid a lot of attention to expert systems and invested considerable resources in their evaluation, especially for real-time control applications. Presumably this is because industry perceived cost/efficiency/competitive benefits which might accrue from computerised systems in controlled processes.

Styles of KBS vary considerably between applications from intelligent assistants to totally automated feedback systems. Particular points in this spectrum are determined by the degree of intervention that is required by the human decision-maker. Thus, direct knowledge-based control

of systems at a low level is feasible, while supervisory functions can also be performed using expert system technology. In all cases, however, the need for human intervention is essential. Although knowledge representation has been predominantly that of rule-based systems, there are numerous examples emerging which are frame-based or object-oriented. There are many general purpose simulation languages available commercially, but few have any AI concepts integrated into them. The object-oriented paradigm fits most naturally into discrete event simulation. Commercial products in this area, such as STEM, are now becoming available. Most of the modelling packages referred to in this paper currently exist at a research level, although some have both extensive and advanced concepts built into them.

In the area of control design, the package which has reached the greatest stage of development is MEAD. It is presently undergoing extensive validation trials, and has sufficient resources behind it to achieve software integrity and maturity.

For real-time process control, the use of fuzzy logic, or rule-based control, has been possible for many years. Thus, the LINKman software has been in use on cement kilns for some time, and is being extended presently into other process engineering applications. The RESCU rule-based system remains at the demonstrator phase, but other real-time toolkits are available commercially, examples being MUSE, ESCORT and G2.

Auto-tuners are now commonly available for industrial 3-term control, but they do not really exhibit concepts of AI, and examples of supervisory control are rudimentary. The systems developed for HCI construction remain currently at a research level, and much development appears to be necessary before practical exploitation can be achieved.

4.2 Current trends

In modelling and design software for control systems, the trend appears to be attempts to produce more integrated environments which contain elements of AI to provide better utilization and greater flexibility of use. This is encouraging the consideration of two important themes, being those of database design and open architecture. The latter concept is particularly important for large software environments, which require modularity in construction to facilitate easier addition and maintenance of innovative components.

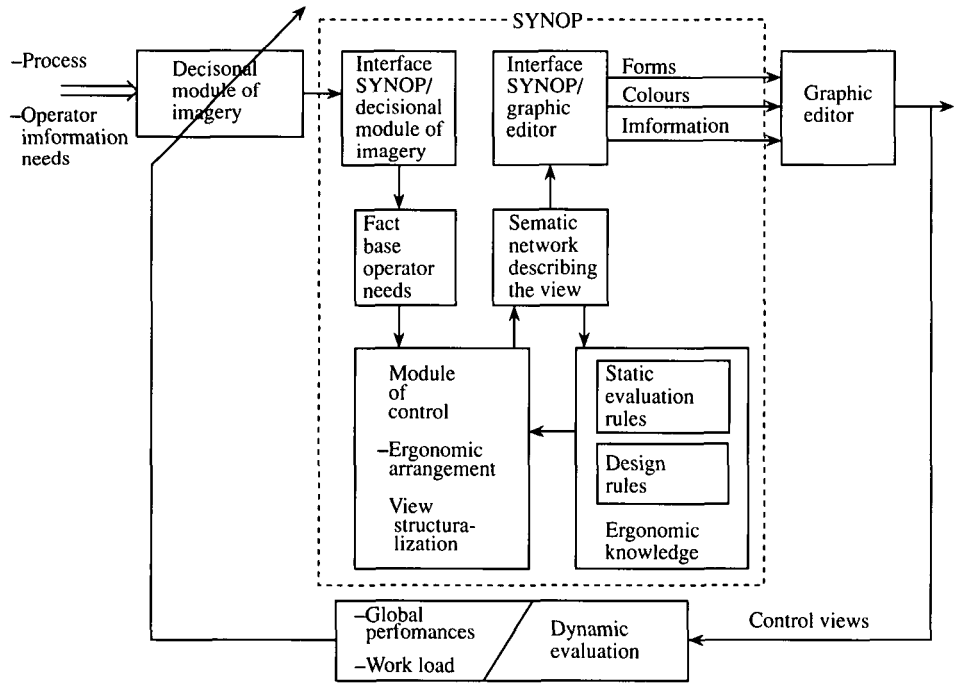

Figure 10 Integration of an expert system SYNOP in a package for design and evaluation of industrial control views (from Kolski et al., 1988).

The needs of real-time systems include common features like multiple types of knowledge such as causal, spatial and taxonomic together with the need to reason with uncertain evidence. The primary requirement is the ability to reason over time using deeper knowledge than mere production rules, and of course the time critical nature of the reasoning. Fast response is essential giving rise to implementation considerations of architecture and software. These problems become evident in the realm of multi-Sensor Data Fusion (MSDF) which is relevant to many fields including military and process engineering. In this case, vast amounts of information from different sensors and knowledge sources have to be merged and abstracted for display to human decision-makers who must respond rapidly to emergency situations.

The difficulty of knowledge acquisition has encouraged the concept of rapid prototyping, sometimes using hybrid toolkits. Although this may help to overcome scepticism/hostility barriers when attempting to penetrate a particular applications area, the bottleneck of speed soon becomes apparent. This conflict between rapid development and run-time speed mirrors the evolution of high level languages and assemblers where modern compilers can give high efficiency code. A similar situation has already arisen in AI with compiled Lisp and PROLOG code. The insertion of knowledge into a system is an important subject, and can be strongly helped via pseudo natural language interfaces such as those introduced for modelling. These are capable of automatic code generation producing validated, efficient modular segments. This approach is similar to the Programming Specification Languages used by software engineers as an intermediate step in proceeding from problem statements to computer code. A conflict in these knowledge interpreters is the balancing of flexible and powerful knowledge representation schemes with ease of use by novice or infrequent users.

4.3 Future research and development themes

Current challenges to the application of AI to systems engineering are common to many other areas and include problems of real-time requirements, knowledge acquisition and validation, and the "brittleness" of many KBSs.

A commonly recurring theme in most systems areas is that of complexity, whether from a mechanistic algorithmic viewpoint or from behavioural symbolic styles. Faced with such complexity, the almost uniform approach is to decompose situations using a hierarchical structure with multiple sub-goals. This hierarchical approach leads naturally to the concept of diffeent knowledge bases for each subproblem. Hence, multiple cooperating expert systems (with different styles) have emerged as a framework for the production of integrated KBS. At this point, the degree of coupling between the knowledge bases becomes an important issue. Tight coupling may produce fast run-time systems, but is lengthy to develop and difficult to debug and maintain. Loose coupling gives slower response and lack of integration which is soon mediated to the user. Medium coupling thus emerges as a suitable paradigm whereby federated systems are constructed from several packages/knowledge bases centered around a blackboard architecture. Crucial to such a viewpoint is the need for a powerful and flexible database representation. Engineering systems require much more flexibility in data expression than those of the relational style used in commerce. Considerable research is still needed to provide this expressive richness with fast retrieval, essential for interactive AI systems either for design or control. The blackboard architecture lends itself to the commonly recurring theme of the need for an overall supervisor, planner or manager. In many practical situations, such global management concepts are embedded in the system via complex heuristics.

Complexity in systems uncovers the richness of types of knowledge which needs to be considered in AI applications. Simple classifications indicate the presence of both continuous (analogue) and discrete (digital) elements of representation and processing. The clearest taxonomy emerging at present, however, is separation into quantitative and qualitative components. Control systems design has emphasized the algorithmic (i.e. quantitative) aspects of design, and very elegant but complex techniques have become available. There are limitations, however, in the

usefulness of algorithmic methods in real-world situations where available models may only be at a conceptual rather than a mathematical level. AI methods tend to stress the symbolic (i.e. qualitative) levels, and hence offer the complementary aspect to system design. It is the merging of these quantitative and qualitative techniques which provides both a stimulus and challenge for the future. Each approach has its strengths and different areas of maturity, and synergetic coupling between them is an exciting future prospect. It is in this area that major advances in the development of an integrated systems approach is envisaged. The importance of databases is apparent again in such a vision.

It would be wrong not to remark on the importance of HCI in integrated systems design, since the subject is not always addressed adequately, even in large demonstrator projects. HCI is, itself, a complex multi-disciplinary subject, which must be approached in a properly considered engineering methodology. Modes of interaction with the user should be constructed carefully, with an emphasis on steering rather than regimenting. The possibility of "command spies" and "adaptive dialogue" is fascinating, but much progress will be needed before these become useful reality. These thoughts cannot be divorced from the need to produce safety-critical systems. For this purpose, it is essential to formulate specifications adequately, and the concept of re-usable knowledge modules such as designs, models, projects and software code become apparent. Within this contect the vital importance of validation and ultimately evaluation is clearly evident.

The overall conclusion becomes that to introduce "AI in Engineering" one must emphasize the engineering of AI systems. If we are to develop integrated systems for control then the very nature of complexity and diversity in real-life demands that we apply well-established engineering principles to all stages of their development. In other words we must give attention to the whole engineering design cycle of project formulation, specification, conceptualisation, implementation, verification, experimentation and validation, interpretation and evaluation, together with reiteration throughout the whole process.

References

Araki, M, 1985. "Industrial applications in Japan in CAD packages for control systems" *IFAC Symp.* Lyngby, pp 71–76.

Arzen, KE, 1988. "An architecture for expert system based feedback control" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 15–20.

Assilian, S and Mamdani, EH, 1975. "An experiment in linguistic synthesis with a fuzzy logic controller" *Int. J. Man–Machine Systems* **7** pp 1–13.

Astrom, KJ, Anton, JJ and Arzen, KE, 1986. "Expert Control" *Automatica* **22** (3) pp 277–286.

Astrom, KJ and Kreutzer, W, 1986. "System representations" In *Proc. IEEE Control Systems Soc., 3rd Symp. on CACSD,* Arlington, Virginia, pp 24–26.

Barker, HA, Chen, M, Grant, PW, Jobling, CP and Townsend, P, 1986. "A graphical pre-processor for computer-aided control system design. Inst MC Workshop" *Computer-Aided Control System Design,* Salford, pp 15–20.

Bekey, GA, 1988. "Knowledge based systems in modelling simulation and identification" *IFAC Symp. on Sys. Ident.,* Beijing.

Bennett, S, Rahbar, M, Linkens, DA, Tanyi, E and Smith, M, 1989. "A Knowledge-based Environment for Modelling and Simulation (KEMS)" *MULTI '89 Simulation Conference,* San Diego.

Betta, A and Linkens, DA, 1988. "EASI (Expert Advisor for System Identification): A prototype package for linear and non-linear system" *Control '88,* Oxford.

Betta, A and Linkens, DA, 1990. "Intelligent knowledge-based system for dynamic system identification" in *Proc IEE Pt D,* Vol 137, Pt D, pp 1–12.

Beutler, KP, 1988. "The *Descriptive search* to realize a knowledge-based retrieval of reusable project results" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 119–122.

Billings, SA, Fadzil, MB, Voon, WSF and Leontaritis, IJ, 1984. *User Reference Manual of NLI: Nonlinear Identification Package,* Dept. of Control. Eng., University of Sheffield.

Borchardt, GC, 1986. "STAR: A computer language for hybrid AI applications" In *Coupling Symbolic and Numerical Computing in Expert Systems* (ed. Kowalik), Elsevier: Amsterdam, pp 169–177.

Borgida, A, Greenspan, S and Mylopoulos, J, 1985. "A knowledge representation as the basis for requirements specification" *IEEE Computer,* pp 82–91.

Brown, PJ, 1986. "Interactive documentation". Software Practice & Experience, 16, No 3, pp 291–299.

Clare, J, 1989. "Real-time application of advanced control in industry" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 10.1–10.9.

Cohen, PR and Feigenbaum, E, 1982. "MOLGEN" In *The Handbook of Artificial Intelligence,* Vol. 3, Los Altos, CA: William Kaufman Publishing, pp 551–556.

Dearden, H, 1989. "Application of self-tuning control for process plant optimization" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 4.1–4.14.

di Primio and Brewska, G, 1985. "BABYLON: kernel system of an integrated environment for expert system development and operation" In *Proc. 5th Int Workshop on Expert Systems and their Application,* Avignon, pp 573–583.

Elzer, P, Borchers, HW, Weisang, C and Zinser, R, 1988. "Knowledge-supported generation of control room pictures" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 129–136.

Epton, J, 1989. "Adaptive and self-tuning controllers: an overview" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 2.1–2.11.

Erman, LD, Hayes-Roth, F, Lesser, VR and Reddy, DR, 1988. "The Hearsay-II Speech-Understanding System: integrating knowledge to resolve uncertainty" In *Blackboard Systems* (ed. R Engelmore and R Morgan), New York: Addison Wesley.

Fjellheim, RA, 1986. "A knowledge based interface to process simulation" In *AI Applied to Simulation* (ed. Kerckhoffs, Vansteenkiste and Ziegler), Simulation Series, Vol. 18, no. 1, Feb, SCS: California, pp 97–102.

Fortuna, L, Nunnari, G, Vassallo, F and Zuccarini, P, 1988. "An educational tool for control system design: Why an expert system? In *Trends in Control and Measurement Education,* IFAC Symp., Swansea, pp 207–212.

Francis, JC and Leitch, RR, 1984. "ARTIFACT: A real-time shell for intelligent feedback control" In *Proc. British Computer Soc. Conf. Expert Systems '84,* also in *Research and Development in Expert Systems* (ed. Bramer), Cambridge University Press: Cambridge, pp 151–163.

Gallant, M, Guida, G, Spampinato, L and Stefanini, A, 1985. "Representing procedural knowledge in expert systems: An application to process control" In *Proc. 9th Int. Joint Conf. on AI,* pp 345–352.

Gladd, NT and Krall NA, 1986. "Artificial intelligence methods for facilitating large-scale numerical computations" In *Coupling Symbolic and Numerical Computing in Expert Systems* (ed. Kowalik), Amsterdam: Elsevier, pp 123–136.

Haest, M, Bastin, G, Gevers, M and Wertz, V, 1988. "An expert system for system identification *IFAC Workshop on AI in Real-time Control,* Swansea, pp 101–106.

Hamiane, D and Morris, AS, 1985. *Linear Multivariable Identification Package,* University of Sheffield: Dept. of Control Eng.

Harris, CJ and Read, AB, 1988. "Knowledge based fuzzy motion control of autonomous vehicles" In *Artificial Intelligence in Real-time Control,* IFAC Workshop, Swansea, pp 149–154.

Haspel, D, 1989. "The use of an expert system for cement manufacture" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 8.1–1.9.

Haton, JP, 1983. "Knowledge-based and expert systems in industrial applications" *IFAC Workshop on Artificial Intelligence,* Leningrad, pp 83–89.

Ionescu, D, 1988. "An expert system with a learning mechanism for real time process control" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 27–40.

Jacob, F, Suslenschi, P and Vernet, D, 1986. "EXTASE: An expert system for alarm processing in process control" *Proc. 7th Eur. Conf. on AI,* Vol. 2, pp 103–108.

James, JR, Frederick, DK, Bonissone, PP and Taylor, JH, 1985a. "A retrospective view of CACE-III: Considerations in co-ordinating symbolic and numeric computaton in a rule-based expert system" *2nd Conf. on AI Applications: The Eng. of Knowledge-based systems.* Washington: IEEE Computer Soc. Press, pp 532–538.

James, JR, Taylor, JH and Frederick, DK, 1985b. "An expert system architecture for coping with complexity in computer-aided control engineering" *3rd IFAC/IFIP Int. Symp. CADCE '85,* Lyngby.

Karwatzki, JM, 1987. "An expert system for monitoring helicopter airworthiness" *Measurement and Control* **20** pp 112–114.

Kolseki, C, Van Daele, Millot, P and de Keyser, V, 1988. "Towards an intelligent editor of industrial control views using rules for ergonomic design" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 137–142.

Krigsman, AJ, Verbruggen, HB and Bruijn, M, 1988. "Knowledge-based real-time control" *IFAC Workshop on "AI in Real-time Control",* Swansea, pp 7–13.

Langen, PA, 1987. "Application of artificial intelligence techniques to simulation" In *Simulation and AI* (ed. Luker and Birtwistle), *Simulation Series,* Vol. 18, no. 3, SCS: California, pp 49–57.

Larsson, JE and Persson, P, 1988. "The knowledge database used in an expert system interface for IDPAC" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 107–112.

Lehman, A, 1987. "Expert systems for interactive simulation of computer system dynamics" In *Simulation and AI* (ed. Luker and Birtwistle). *Simulation Series,* Vol. 18, no. 3, SCS: California, pp 21–26.

Lehman, A, Knodler, B, Kwee, E and Szczerbicka, H, 1986. "Dialogue-oriented and knowledge-based modelling in a typical PC environment" In *AI Applied to Simulation* (ed. Kerckhoffs, Vansteenkiste and Ziegler), *Simulation Series,* Vol. 18, no. 1, SCS: California, pp 91–96.

Leitch, R and Dulieu, MRW, 1987. "RESCU REVISITED—A review of a practical real-time expert system" In *Research and Development in Expert Systems* (ed. Moralee), Cambridge: Cambridge University Press, pp 267–276.

Linkens, DA, 1988. "Control system design: a survey of personal computer software" *Trans. Inst. MC,* **10**(5) pp 240–257.

Linkens, DA, Greenhow, SG and Asbury, AJ, 1986. "An expert system for the control of depth of anaesthesia" *Biomed. Meas. Info. Contr.,* **1**(4) pp 223–228.

Linkens, DA and Hasnain, SB, 1989. "Self-organising systems and their transputer implementation" In *Parallel Processing and Artificial Intelligence* (ed. Reeve, M & Zenith, SV), Chichester: Wiley, pp 249–247.

Linkens, DA and Manfouf, M, 1988. "Fuzzy logic knowledge-based control for muscle relaxant anaesthesia" *IFAC Modelling and Control in Biomedical Systems,* Venice, pp 185–190.

Lounamaa, P and Tse, E, 1986. *The Simulation and Expert Environment in Coupling Symbolic and Numerical Computing in Expert Systems* (ed. Kowalik), Amsterdam: Elsevier, pp 83–100.

Maciejowski, JM, 1985. "A core data model for computer-aided control engineering" *Cambridge University Eng. Dept. Tech. Report CUED/F-CAMS/TR257.*

Maciejowski, JM, 1988. "Data structures and software tools for CACSD: A survey" *IFAC Symposium,* CACSD, Beijing.

Mason, A, 1989. "Practical implementation of large real-time expert systems for process and plant management" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 11.1–11.16.

Moore, RL, 1989. "Application of the real-time expert systems in the US" ibid.

Munro, N, Palaskas, Z and Frederick, DK, 1986. "An adaptive CACSD dialogue facility" In *IEE 3rd Symp. on Computer-Aided Control System Design* (ed. Lineberry), New York, Arlington, VA: IEEE, pp 146–149.

Neilson, NR, 1987. "The impact of using AI-based techniques in a control system simulator" In *Simulation and AI* (ed. Luker and Birtwistle), *Simulation Series,* Vol. 18, no. 3, SCS: California, pp 72–77.

Ogard, O and Wood, E, 1988. "Intelligent alarm handling" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 69–74.

Pang, GKH and Boyle, J-M, 1986. "An expert system for analytical and interactive design of control systems" *Expert Systems '86,* Cambridge.

Pang, GKH and MacFarlane, AGJ, 1987. "An expert systems approach to CAD of multivariable systems" In *Lecture Notes in Control and Information Sciences,* New York: Springer-Verlag.

Pave, A and Rechenmann, F, 1986. "Computer-aided modelling in biology: An artificial intelligence approach" In *AI Applied to Simulation* (ed. Kerckhoffs, Vansteenkiste and Ziegler), *Simulation Series,* Vol. 18, no. 1, SCS: California, pp 52–66.

Permantier, G, 1988. "Representation of inexact engineering knowledge about real time systems" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 41–46.

Procyk, TJ and Mamdani, EH, 1979. "A linguistic self-organising process controller" *Automatica,* **15** pp 15–30.

Rahbar, MT, Bennett, S and Linkens, DA, 1987. "Functional specification of an intelligent simulation environment" *UKSC '87 Conf. on Computer Simulation,* Bangor, pp 187–192.

Rahbar, MT, Bennett, S, Linkens, DA and Thompson, B, 1989. "A integrated database: its role in a knowledge-based environment for modelling and simulation (KEMS)" *ESC '89 3rd Euro Congress,* Edinburgh.

Rajagopalan, R, 1986. "Qualitative modelling and simulation: A survey" In *AI Applied to Simulation* (ed. Kerckhoffs, Vansteenkiste and Ziegler), *Simulation Series,* Vol. 18, no. 1, SCS: California, pp 9–26.

Ranganathan, SR, 1957. *Colon Classification,* London: Madras Library Association.

Rao, M and Jiang, T-S, 1989. "Real-time intelligent control" *Engineering Application of Artificial Intelligence.*

Rasmussen, J and Goodstein, LP, 1985. "Decision support in supervisory control" *IFAC Congress on Analysis, Design and Evaluation of Man–Machine Systems,* Varese, Italy.

Reilly, DP and Timberlake, AI, 1987. "Intelligent front-end to Box and Jenkins forecasting" In *Interactions in AI and Statistical Methods* (ed. Phelps, B), Technical Press, Unicom Seminars Ltd.

Rimvall, M, 1988. "Interactive environments for CACSD software" *IFAC Symp on CACSD,* Beijing, China.

Rimvall, M and Bomholt, L, 1985. "A flexible man–machine interface for CACSD applications" Preprints of *3rd IFAC/IFIP Int. Symp. on CAD in Control and Eng. Syst.*, Copenhagen, Denmark.

Sacks, PH, Patterson, AM and Tunes, MHM, 1986. "ESCORT: An expert system for complex operations in real-time" *Expert Systems,* **3**(1).

Sanoff, SP and Wellstead, PE, 1985. "Expert identification and control" *IFAC Conf. Ident. and Syst. Parameter Estimation,* York, pp 1273–1278.

Sugeno, M, 1985. *Industrial Applications of Fuzzy Control,* Amsterdam: North Holland.

Tanyi, E and Linkens, DA, 1987. "A frame-based modelling and simulation environment" *UKSC '87 Conf. on Computer Simulation,* Bangor.

Tanyi, E and Linkens, DA, 1989. "Addition of a knowledge acquisition facility to a knowledge-based environment for modelling and simulation (KEMS)" *ESC '89 Advanced Control Congress,* Edinburgh.

Taunton, C, 1989. "Expert systems in process control: an overview" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 5.1–5.5.

Taylor, JH, 1987. "Conventional and expert-aided database management for computer-aided control engineering" In *Proc. Amer. Cont. Conf.*, ACC.

Taylor, JH and Frederick, DK, 1984. "Expert systems architecture for computer-aided control engineering" In *Proc. Amer. Cont. Conf.*, ACC.

Tetley, B and Ulf, A, 1989. "Self-tuning and gain scheduling for process control" *ISA '89 Advanced Control Conference,* NEC, Birmingham, pp 5.1–5.5.

Thomson, AC, 1988. "Real-time artificial intelligence for process monitoring and control" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 89–94.

Togai, M and Watanabe, H, 1985. "A VLSI implementation of fuzzy inference engine: toward an expert system on a chip" New York: IEEE, pp 192–197.

Tong, RM, 1977. "A control engineering review of fuzzy systems" *Automatica,* **13** pp 559–569.

Trankle, TL and Markosian, LZ, 1985. "An expert system for control system design" In *Control '85 IEE Conf.*, Cambridge, pp 495–499.

Tsang, TTC and Billings, SA, 1987. *FASTNOI: Fast Nonlinear Orthogonal Identification,* University of Sheffield: Dept. of Control Eng.

Van den Bosch, PPJ and Van den Boom, AJW, 1985. "Industrial application in the Netherlands for CAD packages in control systems" *IFAC Symp.*, Lyngby, pp 58–61.

Vingerhoeds, RA, Delbar, P and Boullart, L, 1988. "Expert systems for process control using automatic knowledge acquisition" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 57–62.

Voss, H, 1988. "Architectural issues for expert systems in real-time control" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansa, pp 1–6.

Wagner, B, 1988. "Knowledge-based constraint specification in the development of process automation systems" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 113–118.

West, AA, Chandraker, R, Williams, DJ and Mulvaney, DJ, 1988. "Hybrid representation of real-time control rules for manufacturing process control in electronics manufacture" *IEEE Symp. Intelligent Control,* Arlington, Virginia.

Zadeh, LA, 1965. "Fuzzy sets" *Info. and Control I,* pp 338–353.

Ziegler, BP, 1976. *Theory of Modelling and Simulation,* New York: Wiley.

Ziegler, BP and De Wael, L (1986). "Towards a knowledge-based implementation of multifaceted modelling methodology" In *AI Applied to Simulation* (ed. Kerckhoffs, Vansteenkiste and Ziegler), *Simulation Series,* Vol. 18, no. 1, SCS: California, pp 42–51.

Zhang, L, Cameron, RG, Ammour, K, Dugdale, RE and Lealman, G, 1989. "Rule-based ventilator management and control for neonates suffering respiratory distress" *IFAC Workshop on Decision Support for Patient Management: Measurement Modelling and Control,* London.

Zinser, K, 1988. "Design issues and knowledge representations for modern control room interfaces" In *Artificial Intelligence in Real-Time Control,* IFAC Workshop, Swansea, pp 123–128.

Zygmont, A, 1987. "Object-oriented programming and CACSD" *ASEE Annual Conference,* Reno, Nevada.