The Knowledge Engineering Review, No. 5:3, 1990, 167–179

Applications and prospects for AI in mechanical engineering design

IAN M. CARTER

NEI Parsons, Heaton Works, Newcastle, Tyne and Wear, NE6 2YL and CAD Centre, University of Strathclyde, 131 Rottenrow, Glasgow, G4 0NG

Abstract

Mechanical engineering design is a broad subject area covering many topics and has influences upon many other engineering disciplines and activities. Computer support for mechanical engineering design activity has been in draughting systems and analysis packages, but there has been little in conceptual design assistance. This paper presents a number of areas of work in which AI techniques and developments are being used, sometimes in conjunction with traditional methods, to improve the support of design. The approaches to design and design systems are covered, along with some techniques that are used. Specific design systems illustrate progress, and integration issues and simultaneous engineering systems indicate the way research is moving. Finally, discussion of the trends and future topics indicates where and how effort may be applied in the future.

1 Introduction

As an engineering discipline, mechanical engineering is a very broad subject, covering such topics as thermodynamics, stress analysis, mechanics, fluid mechanics, dynamic analysis and control. Different facets are used in other disciplines, such as board and connector strengths in VLSI designs, pipe stresses and fluid flow in process plant design, bending stresses and material strengths in electrical machines. In some ways, mechanical engineering is the jack of all trades of engineering, but support for mechanical engineering activities would benefit all these other areas as well.

In traditional terms, drawings are the language of mechanical engineering, be they of components, assemblies, systems or layouts. Computer support has been given to drawing production, through the use of CAD systems, and to the analysis of structures (the applied mathematics required for such analysis being another facet of mechanical engineering).

The skill in engineering is converting concepts into modules which are fit for analysis and drawing. This might be in terms of choosing suitable analysis methods and applying them correctly for a given structure or concept, or it might involve recognizing and choosing the correct analogy with which to model the concept. For example, electrical circuit analogies are often used in dynamic systems analysis (e.g. torsional stress analysis). There is, however, little, if any, computer-based support for these types of activity.

The activity described in the preceding paragraph can be attributed to the design stage of an artefact's lifecycle. Although mechanical engineering is involved in many of the other stages (such as production engineering, process planning, process monitoring and diagnosis), this review will concentrate on the design stage.

2 Mechanical engineering design

Much of the work which involves support for mechanical engineering is being done in the field of design. The computing support provided at present is through CAD drawing systems. These vary

depending upon the type of item being considered, for example, piping and electrical circuits, as well as the "standard" mechanical components. These systems are, however, often used merely to computerise existing manual methods of working. Little assistance is given to the designer (as opposed to the draughtsman) in forming the concepts that are to be drawn.

Design theories and design process models have been developed (Boyle, 1989; Mostow, 1985; Pugh, 1985; Pugh & Morley, 1988) as a means of representing what a designer does, with the aim of developing a rigorously based system which can support him. The difficulty in producing a theory or process model is that it is impossible to make it cater for many different scenarios. In addition, it is difficult to test completely.

An alternative approach to finding out about design practices is to watch a designer at work (Ullman et al., 1988). Studies can be made either of engineers in their everyday tasks, or by conducting experiments in which one or more engineers are given one or more design tasks and asked to provide solutions (sometimes constrained by other factors, such as being a design for volume production). This type of study can lead to a better understanding of some parts and types of design activity, which may in turn lead to better design aids, but the path is long and hard.

Along with others, Chandrasekaran (1986) has identified three types of design: conceptual, innovative and routine (or parametric). In general, development work has concentrated on the last of these, as it is the simplest and best understood. It is also already possible to perform considerable parametric design using conventional CAD systems. In this light, it is disappointing that there has been relatively little investigation of the viability of linking knowledge-based systems to CAD systems. Perhaps it is not perceived as enough of a research challenge, merely implementation detail. This is certainly not the case, and is the first step on the road to supporting mechanical engineering design.

Reviews of both mechanical engineering and AI work are available, but Finger and Dixon's recent articles on research in mechanical engineering (Finger & Dixon, 1989) are comprehensive in their coverage and can be recommended. However, as the authors indicate, there are few examples of research which have passed into industrial use. The reasons for this are undoubtedly complex, but must include the approach to research and the way in which it is presented. This topic will be discussed later, but should be borne in mind in the following sections which deal with specific aspects of design and design systems, illustrated by a number of pieces of work.

2.1 Aproaches to design systems development

It is possible to identify two approaches to building design systems, both as a means of supporting designers and the design process, and as a means of studying, and hence understanding, design. The first approach is to develop systems which are able to produce design solutions, either by replicating human activity or by some other means. The alternative is to develop systems which provide support for the human design activity. These two approaches have been described as "expertise centred" and "model centred" (MacCallum, 1989).

Expertise-centred sysems are developed for a particular application, and often a particular situation within that application. This has the advantage that "a working application" is often developed, but requires detailed design knowledge for the application. Also, it can be limiting in the opportunities it provides for expansion or further exploitation. This may not be a problem for an industrial application, but does seem a disadvantage for research systems, where spawning of ideas is important.

Model-centred systems are concerned with providing the necessary tools, environment and decision aids for human designer. This may require the encoding of common sense knowledge as well as different levels of design knowledge. A significant problem with this approach is the interaction and communication between the designer and the system, which must take place at a higher level of understanding than for expertise-centred systems. As research systems, model-centred systems provide an ongoing opportunity to extend capabilities and experiment with new ideas. However, from an industrial perspective they can look rather open ended unless specific test

case applications are also developed. Smithers et al. (1989) discusses the options and why the AI in Design programme at Edinburgh University has followed a model centred approach.

2.2 General approach to design

Vernon (1987) describes the systematic method of designing systems as opposed to the "ad hoc approach of gradually closing in on a design until it 'looks' about right, or until one has used up one's allocated time". He gives a diagram of the anatomy of a system, breaking it into *decision making, action and performance monitoring subsystems,* and then puts it all into a *wider system,* and that into an *environment,* creating a three tier model of disturbances and influences. Although a system can be divided into many levels, at any one time it is only practical to consider four: *environment, system, subsystem* and *element.*

The essence of his argument is that a systematic approach to defining a problem will ensure that the definition is complete and structured, which will allow a structured, and hence well-controlled, solution.

A structured approach to analysis is also presented by Ross & Brackett (1986) in their SADT system (Structured Analysis and Design Technique). This technique centres on a top-down, modular, hierarchical structure, in which a problem is decomposed into a number of subproblems, each of which can be expanded in a similar way. The problem is given two representations, in terms of activity and data decompositions. In this respect, their approach is not unique, but they do introduce a symbolism for representing this process. The method does seem to have its limitations: a module can only have six sub-modules; the decomposition and interconnections of the model are purely hierarchical when a network of heterarchy might be better; the approach considers "things" (objects or data) and "happenings" (activities) as different entities, unlike the object oriented approach in which an object can represent any entity. In using SADT (which is not necessarily computer-based) various procedures, protocols, management and checking features have to be implemented.

The methods in which a system can assist in the design process are tackled by Cholvy & Foisseau (1985). Two different approaches are described, which can be used for different situations. The first is for parametric design, where "the design process is a sequence of choices made among the set of possible partial solutions, driven by the requirements". For this type, the problem is divided into subproblems by a problem reduction strategy. The difficulties of this approach are centred about the general lack of a data model to represent the information being manipulated, so that it is difficult for the system to ask questions about its own knowledge.

The alternative approach is "requirements checking at will", which supports creative design and is not specialized. It checks and controls the user's actions with integrity constraints, i.e. a type of truth maintenance system. However, this method cannot make decisions for the user, nor advise him.

A mixed approach is also discussed, in which the system will attempt as much as its knowledge will allow, and then ask the user to choose between options. Suggestions are made for a realistic architecture which could consist of "different expert systems aided by a database". For this type of system a supervisor will be required, and blackboard techniques are put forward as a possibility, with decomposition of the problem and optimization of the final design. The mixed approach will have more general application, and will be better suited as an intelligent assistant type of system.

2.3 Requirements of a design system

Hatvany (1985) discusses the computer aids that are desperately needed, but unavailable. The aids include recognition of, and recovery from, a failure in the design process, and coping with unprecedented and unforeseen problems. The shortcomings of systems design is illustrated by comparison of the house-building architect and the electronics designer. The former designs the

house before buying any bricks, whereas the latter does exactly the opposite, in the belief that "all you have to do is write a few lines of clever code and you can stick together anything". What is required is an overall plan, giving total synthesis, even if the implementation is bottom-up (he recognizes that a top-down approach is not entirely satisfactory for design).

The principles which might be useful to design systems include database management systems, truth maintenance and game playing strategies. Problems should be divided and strata added, in this case three—operational, functional and physical. This is similar to Ross and Brackett's activity and data decompositions (Ross & Brackett, 1986). It is in the physical strata (the implementation of the functions) that a bottom-up approach must be combined with the top-down approach. To cope with unprecedented problems requires "situation recognition", which includes pattern, visual pattern and speech recognition, but goes further than these by integrating and interpreting them.

The classical scheme of an expert system is difficult to apply to a manufacturing situation because it is open, having no feedback. Hatvany suggests nesting this scheme into a more global framework which introduces the feedback required. It also includes "process evaluation", which is used to learn and thus augment the systems' knowledge.

Tomiyama & Yoshikawa (1985) present the designer's creative or intelligent support requirements with a view to increased productivity and decreased errors. They divide design into conceptual, basic, detail, production and prototyping and test, but suggest that conceptual design is the part most needing attention, as conventional CAD systems have already affected the others.

There are problems with conventional CAD: the systems are not intelligent (there is sometimes no checking of information, and it cannot answer the designer's questions); the interface is bad (an image of the solution must be fed in, and this then has to be checked by hand, which means that errors are likely with large amounts of data); the environment is non-integrated (there are interface problems between different CAD systems, mostly because there is no unified data description and hence translation is required).

Any future CAD system must: support designers in all stages (system and knowledge integration); have a unique and common data description (model integration); be intelligent, to capture errors, answer questions, suggest alternatives; reduce time and cost of designing as a whole. To meed these requirements a CAD system will comprise a workstation connected via a LAN (Local Area Network) to other computers and systems, with a very intelligent supervisor to control the system. The Designer's Workbench is suggested to fulfil these needs.

Machinery and operations must be represented before inference and control mechanisms are considered. Also, intensional and extensional knowledge is required, to describe an entity and relationships of entities respectively.

They have built a production rule system, but found it difficult to write good knowledge modules, as there was no way to structure the knowledge and the system was too sensitive to small changes of rules.

Marechal (1985) shows the need for management of a design team or project, meeting the different needs at the different levels. Use of computer-supported tools gives new possibilities but enforces the whole process to be more explicit and formal. Before the design of a design management system can begin, the design process must be analyzed and modelled to show its structure and the responsibilities. There must be a standard terminology for a product or object life cycle in terms of products, materials, processes, corpus and contexts. The life cycle can be divided into three stages: *design, production* and *usage.* These stages have similarities and the same model could be used as the basis for each, with the parts definition, realization and utilization. He goes on to discuss closed, open and integrating systems and static and dynamic behaviour, also escape mechanisms such as conflict resolution.

CIM (Computer Integrated Manufacturing) needs transparent communications as different systems handle the same object using different representations, thus requiring translation. Then, is the translation sufficient (containing enough information for the receiver) and complete (will not lose information when translated back)? This raises questions about network connection types, the number of transformations required and the control of such a network.

David (1987) discusses the requirements of an Intelligent CAD system, and introduces a system called Ciao (Computer Intelligently Assisted Design).

Intelligent design requires the integration of technological expert systems (e.g. calculations), I/O techniques (e.g. natural language interfaces) and database models. An Intelligent CAD system will combine these with design theory or a design process model.

Ciao uses a multi-expert system approach. Expert systems are better at limited but deep problems, so it is easier to build a system of cooperating systems than a single "super-expert system".

There are two concepts: the cooperation strategy (how the tools will be used); and the design process model (how to combine different approaches into one formal expression).

A comprehensive design process model (CDPM) is used to define all authorized design processes, and to support and manage the applied design process. The "object to be designed" represents the product specification, the "designed object" is the possible answer. The CDPM manages design activities with processes based on induction, deduction, experience, creativity and intuition. The object-oriented approach seems most suitable for the Intelligent Knowledge Management System to assure coherence and evolution of information.

The information space is organized as a hierarchical blackboard to give several views of and selective access to the information. The precise organization depends on the design process. This approach allows extension, assures coherence of the managed information and takes into account the design process.

Kitzmiller & Jagannathan (1987) highlight the problem with design being that of its size and complexity, so that it cannot be handled by a single individual, organization or design environment. There is a need for a cooperative distributed environment, for which they suggest a blackboard is an appropriate development framework.

The design problem is decomposed into a hierarchy of manageable subproblems, and the system will support distributed, multi-disciplinary, team-oriented approach to design. Design is a blend of creative, innovative and routine. The designer must strike a compromise between performance, operation, cost, producibility, reliability, maintainability, supportability, etc. This requires a balanced top-down, bottom-up approach.

Some difficulties are found in using the components of the product as the method of structuring the design knowledge due to the replication of this knowledge in some instances. The approach they suggest is to use a hierarchy of blackboards, or a blackboard system composed of blackboard modules, each of which can run as a separate process.

2.4 The design model

Pugh (Pugh, 1985; Pugh & Morley, 1988) advocates the need for a model of design, and gives some alternative forms. Such a model must be relatable, understandable, enhance effectiveness and efficiency of practice, be comprehensive and have universal application for disciplines and products. This is quite a tall order, and none of the models he gives (including his own) can meet all these requirements.

The model of design covers marketing, specification, conceptual and detailed design to manufacturing and selling, because in the past a universal complaint has been that designers work in an isolationist mode, and do not take the needs of the users (or makers) into account.

Pugh highlights some shortcomings that he has found, which are not unique to the marine industry: "It would appear that there is a distinct lack of systematic front end activity in the marine field and certainly lack of reference to systematic specification formulation".

Mostow (1985) also looks at models of the design process and why they are needed. He decides that "design is largely a process of integrating constraints imposed by the problem, the medium, and the designer". A comprehensive model should address the following aspects: the state of the design, the goal structure of the design process, the design decisions and their rationales, control of

the design process and the role of learning in design. Each of these is expanded using issues and ideas for solving them.

2.5 *Constraint-based design*

Design can be viewed as a constraint satisfaction problem, with the requirement to apply and relax constraints in an unconstrained manner. Work has already taken place on constraint-based systems and constraint satisfaction, and this is now being applied to design problems.

Serrano & Gossard (1988) present a graph theoretical approach to design, with constraint networks being represented in directed graphs, where design parameters are nodes and relationships are the arcs connecting the nodes. Matching representations are used to generate automatically the parameter dependencies, and algorithms evaluate the network for over or under constraint, redundancy and conflicts.

The *concept modeller* is the implementation of these ideas, linking abstract conceptual models to constraints and providing automated evaluation and modification of the models. Constraints are used to represent the relationships between parameters involved in the design; for example, the parameters used in an equation to calculate the value of another parameter. A constraint manager is used to provide four main functions:

- *Evaluation,* the process of examining the constraints for given values of parameters;
- *Minimization of computing,* to limit the evaluation only to those of relevance to the operation, thus speeding up evaluation;
- *Consistency maintenance,* to identify inconsistencies and provide advice for resolving the conflicts interactively;
- *Qualitative reasoning,* to provide information about the dependencies between the parameters, which is used to search for a better solution or to give a better understanding of the solution space.

Sapossnek (1989) argues the case for constraint-based design over parametric design. He claims that the major difference is that the former separates the problem statement from the problem solution techniques, whereas the latter mixes the two together. From this distinction consequent differences can be drawn.

Parametric systems depend on the causal ordering of relationships in a form suitable for their solution, i.e. in a directed acyclic graph, which is essentially a procedural representation and thus specifies at least part of the solution method.

In contrast, constraint-based systems are declarative, having no causal ordering of the relationships in an undirected graph. Some variables must be fixed before the system can be used to calculate others, but the sets of fixed and calculated parameters can be changed, so that different starting points and default values can be used. Parametric systems have a more explicit representation of fixed and calculated parameters, thus not permitting such flexibility. In addition, simultaneous equations cannot be expressed in a directed acyclic graph, but they can be handled in a constraint-based representation.

2.6 *Specific design systems*

Brown and Chandrasekaran (Chandrasekaran, 1986; Brown & Chandrasekaran, 1985; Brown, 1985) have produced a number of papers concerning mechanical design. The papers generally discuss an example system for the design of an air cylinder to illustrate how their system works.

Chandrasekaran (1986) discusses the differences between diagnostic and design systems, and that he expects the knowledge structures and control regimes for the two areas to be different. However, he recognizes that the present languages are somewhat limiting in their representational abilities, by expecting a uniformity across tasks and a mixing of domain knowledge and program-

ming devices. Backward and forward chaining are not necessarily the most appropriate methods of control in some situations. The need for good knowledge organization is clear, and they have chosen a hierarchy.

It is suggested that there are three types of design: totally new, partially new and redimensioning (i.e. parametric). It is this last area which is considered. The general structure of the product is known, but the actual dimensions and material types are not. The design expertise is contained in a hierarchy of design specialists which mirrors that of the components, so each component has a set of knowledge which is used to design it. The choices are made by the specialists using design plans, i.e. alternative procedures that can be used for a particular component. The choice of plan is made by the specialist's selector, and will depend on the requirements of the design, e.g. for cost or weight. However, if alternative plans are available, it suggests that they have been pre-planned, so the system's behaviour will not be totally adaptive.

The control is top-down, with one specialist making some decisions, and then passing tasks down the hierarchy to other specialists. When a plan fails, local recovery will be attempted and, if unsuccessful, control will be passed up the hierarchy as far as necessary.

There is a need to represent knowledge in a form that is suitable for its solution, rather than in one way for the whole system (including the control); i.e. that there should be multiple representations and organizations in a single system. There should also be a generic representation of knowledge and control at the appropriate level of abstraction.

Brown & Chandrasekaran (1985) and Brown (1985) expand on the description of "Air-Cyl", giving examples of the code for each of the parts of the system. The system has been built using DSPL (Design Specialists and Plans Language). The parts of the system are:

Specialists	the design agent for a given part
Plans	alternative plans for specialist to use
Tasks	a sequence of steps
Constraints	to check the design, and suggest remedy on failure
Failure handlers	to recognize and react for failure messages
Redesigners	to recover from failure by a suggested redesign
Sponsors	to estimate the suitability of a plan, given the state of the design
Selectors	to take plan suitabilities from the sponsors and decide which the specialist should execute

This is an object-oriented approach, with standard routines being used throughout the system's structure. The failure handling mechanism is good, giving localized error handling. The suitability of the plans seems to be limited towards one parameter, i.e. just to cost or performance, so it may have difficulty optimizing between several parameters.

Presumably the approach can be used to handle any product that is well structured, but it is not obvious if alternatives in the structure can be handled or if it is only dimensions and material types that can be changed. The system handles redimensioning design, but whether it can be expanded to encompass any part of the other two types of design is difficult to say. If a different problem were to be considered, it may come down to rebuilding the system in DSPL, using the same techniques, rather than having ready built building blocks.

"Dominic" (Howe et al., 1986; Orelup et al., 1988) is an attempt to produce a general system (like Cholvy and Foisseau's "requirements checking at will" type of system) which can give a design in any domain. However, it has a very small domain size at present, handling a maximum of ten goals and variables. Design is treated as a best-first search, using weak methods. Unlike Brown and Chandrasekaran (Chandrasekaran, 1986; Brown & Chandrasekaran, 1985; Brown, 1985), who are described as "working nearer the other end of the (power/generality) spectrum", the problem is not decomposed into subproblems, but instead they focus on iteratively redesigning a single object or simple system. The problem to be solved is represented by a fixed finite set of design variables for the physical parameters. A dependency table shows how changes in each variable affect each goal, and is used to select the best change to make next, until a satisfactory solution is reached.

The initial design starting point is chosen by the user (or from user-supplied routines), as are the goal priorities and need for attention. This seems to indicate that much of the initial preparation has to be done by the user, and that the system then carries out iteration based on a best-first selection of a change to make.

For a new domain, the expert provides the basic knowledge, and Dominic learns the rest (the dependencies). A domain is described by the problem specifications (fixed requirements or constraints), the design variables (dimensions) and the design goals (parameters measuring the quality of the design).

Dominic has been tested with two domains: v-belt drive systems and aluminium extruded heat sinks. To illustrate the size of a domain, the v-belt drive system has six specifications, five variables and five goals.

Corby (1986) describes blackboard architectures and looks at them from an engineering point of view. He sees them as a common database for cooperating expert systems, giving uniform access to a uniform representation. This is important in an engineering context when many different people and systems may have to interact with any design system. "Viewpoints" are discussed; these allow several experts to handle the same object, but to see different properties (e.g. the designer, the user, the finance expert). It is the objects themselves which should know who can see what.

Multiple hypotheses and solutions are of great interest in CAE. Modularity is necessary and should be incorporated. The usual data structure implementations for blackboards are similar to the frame notation, so frame representation languages and object oriented programming are often well suited to develop blackboards.

Design is considered as an expert (source) cooperation process, with different knowledge representation experts. The blackboard data structures are protocols allowing experts to understand each other. The challenge of such a system is the generality of this protocol to enable many different experts to cooperate. There will be several different control strategies required, and the design will proceed by the interpretation of an imperative design plan. It is necessary for the system to be open to other external software and to be able to cope with alternative testing, i.e. choosing which design is best to proceed with. The control aspects cannot be "hardwired", but must be adaptive.

CAE systems need to provide complex planning and control flow, to allow the design to proceed top-down, increasing the level of detail and obtaining agreement at each level. The system should be able to tackle any sub-task "on-line", and be able to recover from most failures.

2.7 Computer-aided simultaneous engineering

There have been an increasing number of references to Computer Aided Simultaneous Engineering (CASE). CASE endeavours to allow the consideration of many influences upon the design stages so that, for example, manufacturing constraints and tolerances are taken into account at the design stage.

Work at Carnegie-Mellon University has developed a framework for CASE and implemented it in a computer system. Rehg et al. (1988) describe the need for CASE, in particular due to the shortcomings of traditional CAD. They say that CASE concerns the coordination of the various "projects" (i.e. phases) of the product lifecycle, such as design, manufacturing or assembly. This is in order to eliminate the problems that are due to lack of communication between functions and the incompatibility between the information and design solutions of the different functions. CASE is thus being used to provide integrated supporat for product development.

At this point it is worth reproducing the issues for new generation CAD that are presented, as these reflect the major concerns of many systems developers (Rehg et al., 1988):

- Development of multiple, integrated representations;
- Development of distributed, cooperative problem-solving approaches to design, with multiple problem-solving mechanisms;

- Incorporation of "downstream" concerns, such as manufacturing, material selection and assembly issues, into the design process;
- Development of new sets of tools to support the kinds of problem-solving activities that are essential to mechanical design, such as spatial and geometric reasoning;
- Automation of a variety of non-creative design decisions, to relieve the designer from low level drudgery and free him to pursue creative design choices and explore design alternatives;
- Encapsulation of expert design knowledge in design agents and critics.

The framework and system presented in Rehg et al. (1988) is based on the use of *design agents* and *design critics* (embodying the design expertise) and the use of multiple *design representations* (the way in which design information is represented in different contexts). The work has addressed three main types of tools: agents, critics and translators. An agent is a module which develops a design by manipulating a design representation in response to a specification or requirement. A critic is a tool that evaluates a design with respect to certain criteria, and feeds this evaluation back to the design agent. Critics can give feedback about the local consequences of a decision (i.e. within the same project) or the consequences to another project. It is the latter type of feedback that provides the mechanism for achieving simultaneous engineering. Translators are used to map one design representation onto another, so that the output of one agent can be used as the inut to another.

Design representations have been developed separately for synthesis and analysis activities. The synthesis representations comprise design elements in a semantic network, with the design variables embedded in a constraint network which expresses the limitations on their values. The analysis representations, in contrast, are unstructured, being in whatever form seems most suitable.

The system developed in this work has used the design of manual car window regulators as its example domain. Although this is a relatively simple artefact, it requires sufficient design expertise to test a system and also has effects on other aspects of the overall car design, such as the door shape. A description of the system and its capabilities can be found in Rehg et al. (1988).

A companion paper (Tulakdar et al., 1988) develops some of the concepts further, and investigates the coordination of routine and nonroutine activities in the design process. Routine activities are those which follow defined methods, whereas nonroutine activities are the development of new methods and the introduction of new technologies.

The issues in coordinating projects are illustrated by the relationships between window regulator design, door geometry design and the design of the necessary manufacturing processes. The first and second projects are normally undertaken concurrently, whilst the third takes place after the first two. With concurrent projects, interconnections must be made at high levels of abstraction to help prevent the production of incompatible results, which would occur if the only link was once the two designs had been virtually completed. With projects in series the difficulty is ensuring the later project's concerns are taken into account in the earlier project, without requiring the complete representation of its activities. In this example, it is the requirement of manufacturability of the regulator and door designs which must be considered during their design.

As a brief comparison with the above work, Ishii et al. (1988, 1989) have developed a CASE framework and system which is based on calculating the compatibility of different elements of possible solutions with each other and with the design requirements. Much of the effort has been to define how to undertake the compatibility analysis which, as well as rating a design element, also produces justifications for that rating and suggestions for improvement—in some ways similar to the critics of Rehg.

A notable difference in this work is, however, that the authors are considering more of the product lifecycle than design and manufacturing, also including maintainability and aesthetics. The complete list of typical issues given is *aesthetics, primary functions, reliability, producability, assemblability, testability* and *serviceability*. This is an improvement on the above, as it shows the wider scope of the product lifecycle. However, it is disappointing that this list, in common with

many engineering perspectives, does not include financial and commercial influences. The notion that these issues are outside the concern or influence of the designer or engineer is out of date, as they play an important role in modern engineering design. Reliance on a designer's innate feel for costs or on some "magic formula" is unlikely to b successful, so that the costing process must be brought into the design equation at the earliest possible stage, just as manufacturability is. Similar arguments can be made for consideration of the legal aspects of a design decision, whether they relate to national laws and regulations or to penalty clauses attached to a contract.

2.8 Integration of systems

Computer systems are already used extensively in design, mostly applied to analysis and documentation. Even without the sophisticated design aids and tools that are discussed above, there is a requirement for existing systems to be integrated so that information flow can be speeded up and the quality of decision making improved.

One of the major concerns in industry is to make better use of current facilities and resources, enhancing their capabilities rather than making them redundant through the introduction of something totally new. Thus, methods of integrating systems (analysis, CAD, CAM, etc), firstly to pass data and secondly to allow genuine interaction, are required. Few systems directly address this issue, but some work has been carried out.

Chalfan (1986, 1987a,b) describes the situation at Boeing where a number of design programs (belonging to and maintained by different departments) are used in preliminary aircraft design, usually in a specific sequence determined by their I/O data requirements and order of execution. There are also (in Boeing's case) different "configurations" of the individual programs, depending on the application (e.g. for tactical or commercial aircraft). The "first level" solution to the problem is to automate the transfer of data from one progam to the next, thus reducing the errors and time involved in manual transcription. The "second level" solution is to add design expertise to give the integrating system an "understanding" of the objectives of the desired design analysis.

Three alternative approaches were tried before a successful method was developed. These involved a procedural program which subsumed the individual programs; a database management system to store a library of software primitives; a procedural program calling the original programs and finally a knowledge system, also calling the original programs.

The earlier approaches were unsuccessful principally due to maintenance and upgrading problems, but the indications from the knowledge system are that this will not be a major difficulty. The benefits arising from this type of approach are to expedite the computational process and provide more design alternatives, without making all present systems redundant. Chalfan indicates that the time taken for the knowledge system can be between 1.5 and 10 min for the controlling system, plus the sum of the run times for the individual programs, which compares very favourably with the normal design cycle time of several weeks.

The indications are that, while the original programs must be integrated, it must be done in such a way that leaves them independent and still usuable in their original form (i.e. only have one copy). It will be most successful if the designs produced by the resulting systems are stored globally, so that they can be accessed uniformly. The integration knowledge should be flexibly and modularly encoded, thus making it easier to modify.

3 Discussion

The scope for support of mechanical engineering is very broad, partly due to the breadth of the subject, and partly due to its complexity. These would seem ideal criteria for advanced computing support. For research topics, there are those concerned with mechanical engineering itself and its practices, and there are the computing techniques which need to be devised or improved to cater for realistically sized problems. The subsequent transfer of research results into industry must be

addressed and, indeed, must be considered at the outset of the research. Before discussing the research topics and trends, it will be beneficial to summarize the extent of current work.

3.1 Summary of work

Work to date has varied from individual methods and techniques through design theories to complete design environments. Investigation of the design process and development of models and theories to represent it have produced much information. In this area there seems to be general agreement on the need to have a structured approach to the analysis and description of the process, but there is considerable divergence in the final models or theories that are produced. This may be due to the background of the authors (engineer, computer scientist or psychologist, industrialist or academic) or due to the type or example of design that is being considered. Indeed, much effort has been put into investigation of the different types of design (e.g. conceptual, innovative and routine), while the immediate differences between design for one-off manufacture and design for mass production need hardly to be explained.

The definition of the requirements of a design system has identified the need for a more interactive approach, with built-in checking, evaluation, failure handling and explanation. Modern design cannot usually be undertaken by one individual, so a design system must tackle the problems of design team management and its integration and interaction with other functional teams. Most current design system support is in the downstream areas of draughting and analysis, with little assistance for conceptual design. In this ara much work is still to be done, and it will make use of techniques such as contraint-based reasoning to enable the modelling of concepts and the addition, manipulaton and deletion of contraints in a flexible manner before a final design is chosen.

Some AI techniques have been evaluated by the development of design systems specific to one domain example. These give useful information about the modelling of design, but do not necessarily permit the easy reuse of the developed system. This has lead to a debate about the benefits derived from the alternative development approaches of expertise centred (as in these specific design systems) and model centred (which are more of a general design environment).

The integration of systems into a design environment is of direct concern to industry who want to make better use of existing resources and enhance their capabilities rather than make them redundant. The aims are to speed up information flow and increase information and expertise availability so that the quality of decision making can be improved. An extension of system integration is the complete design or product management environment, where many factors are represented and considered during the design process, whether it is taking place at the conceptual stage or during redesign or decommissioning. The total environment depends on all the previous areas of work—developing theories, process models and techniques—and brings them together in what is beginning to be the start of the next generation of CAD systems.

3.2 Research topics and trends

Engineering techniques and methods of problem solving are not well understood, so there is ample ground for improvement. However, the difficulty may be in getting the results of this type of work accepted by practising engineers. Tools to support engineering activity may be easier to think about and develop, but still rely to a certain extent on the understanding of what is required and how it might be used. Tools also need to be accepted, not only for their function but also for their usefulness and their usability. A cautious approach in this area may be gradually to extend the capabilities of already accepted tools, such as CAD and analysis packages rather than to take a leap in the dark with a radically new approach.

On the purely computing side, many techniques have been developed and proven in the computing laboratory, but have yet to stand up to the test of a large problem, with vast quantities of data, as will be found in most engineering problems. Even a single turbine blade can have several megabites of information associated with it to define all its characteristics of size, contours,

stresses, process plans, shipping dates, etc. There is thus much work to be carried out to improve methods for storing, retrieving and manipulating large quantities of complex data in a flexible manner, so that concepts can be juggled and constraints applied and relaxed at will.

There will continue to be a requirement for work on the theories and techniques which support advanced work, but perhaps three research areas can be identified in particular as those in which most effort is likely to be applied, and where the benefits can arise.

Firstly, conceptual modelling, although already addressed to some extent, has still to come of age with the production of truly useful systems. Efforts will be needed to develop the theories required for systems to support this type of activity, as well as the improvement of computing techniques to handle large amounts of imprecise data. Current constraint management and truth maintenance systems have yet to prove themselves satisfactory for practical size designs and to be able to handle flexibly several alternative designs.

Secondly, integration of existing and future systems into a flexible and global environment for product lifecycle management. This area includes the issues of global and multi-user data modelling, information interchange, cooperative working and task management and scheduling, quite apart from the possibilities of parallelism. The challenges of incorporating old and new technology in one system is enormous, but the benefits could be equally large, with better information availability and interchange, speedier and more accurate designs, the designer having more time to consider all the relevant information and hence better quality decisions being made.

Thirdly, human–computer interfaces must continue to be developed so that they meet the requirements of the users, and show the information in the most easily assimilated form. This requires that they are very flexibile, open when necessary, expressive, helpful and can check the user's responses or even anticipate his requirements.

3.3 Technology transfer

The transfer of research results to industry requires more than the listing of suitable research topics, or the development of the tools envisaged. In order to foster interest in the results (or in the research effort itself), there must be genuine products which address particular problems with which a company can identify. As well as addressing the problem area, the product must be able to cope with true examples of it, with all of the complexity and size problems.

In addition to these criteria, the product must fit into a company's method of working and culture, and must have a finished appearance. This last point, which implies an engineer-friendly interface, is vital if a new tool is to be accepted in everyday use and is the reason for interfaces being one of the above research topics. Finally, research products must address the requirements of practising engineers, not the perceived requirements of others. Does an engineer want a system that will automatically design for him, or one that supports him while he designs? Does an engineering manager want a system that solves all his technical problems or one that assists him in the areas in which he is not an expert, such as budgeting and marketing? It is only by addressing these types of question that any research effort will bear fruit and fulfil its significant potential for benefiting Mechanical Engeering activity.

References

Boyle, J-M, 1989. "Interactive engineering systems design: A study for artificial intelligence applications" *International Journal for AI in Engineering,* **4**(2) pp 58–69.

Brown, DC, 1985. "Capturing mechanical design knowledge" *Proceedings ASME International Computing Engineering Conference,* pp 121–129.

Brown, DC and Chandrasekaran, B, 1985. "Expert systems for a class of mechanical design activity" in *Knowledge Engineering in Computer-Aided Design,* ed. JS Gero, Amsterdam: Elsevier, pp 259–282.

Chalfan, KM, 1986. "A knowledge system that integrates heterogeneous software for a design application" *AI Magazine,* Summer, pp 80–84.

Chalfan, KM, 1987a. "An integration tool for life-cycle engineering" *IJCAI '87,* August, pp 592–595.

Chalfan, KM, 1987b. "A generic tool for integrating software components" *CIPS,* Edmonton, pp 264–267.

Chandrasekaran, B, 1986. "Generic tasks in knowledge-based reasoning: High-level building blocks for expert system design" *IEEE Expert,* Fall, pp 23–30.

Cholvy, L and Foisseau, J, 1985. "Encoding requirements to increase modelisation assistance" in *Knowledge Engineering in Computer-Aided Design,* ed. JS Gero, Amsterdam: Elsevier, pp 205–214.

Corby, O, 1986. "Blackboard architectures in computer aided engineering" *International Journal for AI in Engineering,* **1**(2) pp 95–98.

David, BT, 1987. "Caio: an intelligent CAD system" in *IFIP WG 5.2,* October.

Finger, S and Dixon, JR, 1989. "A review of research in mechanical engineering design. Parts 1 & 2" *Research in Engineering Design,* **1**(1–2).

Hatvany, J, 1985. "The missing tools of CAD for mechanical engineering" in *Knowledge Engineering in Computer-Aided Design,* ed. JS Gero, Amsterdam: Elsevier, pp 393–400.

Howe, AE, Cohen, PR, Dixon, JR and Simmond, MK, 1986. "Dominic: A domain-independent program for mechanical engineering design" *Artificial Intelligence,* **1**(1) pp 23–28.

Ishii, K, Adler, RE and Barkan, P, 1988. "Knowledge based simultaneous engineering using design compatibility analysis" *Proceedings of the Third International Conference on AI in Engineering, "Artificial Intelligence in Engineering: Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 361–378.

Ishii, K, Goel, A and Adler, RE, 1989. "A model of simultaneous engineering design" *Proceedings of the Fourth International Conference on AI in Engineering, "Artificial Intelligence in Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 483–501.

Kitzmiller, CT and Jagannathan, V, 1987. "Design in a distributed blackboard framework" in *IFIP WG 5.2,* October.

MacCallum, KJ, 1989. "Does intelligent CAD exist?" invited talk, *Fourth International Conference on AI in Engineering.*

Marechal, G, 1985. "The ARCADE and OSA contribution to the theory of integrated CAD systems" in *IFIP "Design Theory for CAD",* pp 333–363.

Mostow, J, 1985. "Towards better models of the design process" *AI Magazine,* Spring, pp 44–57.

Orelup, MF, Dixon, JR, Cohen, PR and Simmons, MK, 1988. "Dominic II: Meta-level control in iterative redesign" *Proceedings AAAI'88, Seventh National Conference on Artificial Intelligence,* Vol. 1, pp 25–30.

Pugh, S, 1985. "Systematic design procedures and their application in the marine field—an outsider's view" presented at *Second International Marine Systems Design Conference,* May.

Pugh, S. and Morley, IE, 1988. "Total design. Towards a theory of total design" Monograph, Design Division, Univeristy of Strathclyde.

Rehg, J, Elfes, A, Talukdar, S, Woodbury, R, Eisenberger, M and Edahl, R, 1988. "CASE: Computer-aided simultaneous engineering" *Proceedings of the Third International Conference on AI in Engineering, "Artificial Intelligence in Engineering: Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 339–360.

Ross, DT and Brackett, JW, 1986. "An approach to structured analysis" *Computer Decisions,* September, pp 40–44.

Sapossnek, M, 1989. "Research on constraint based design systems" *Proceedings of the Fourth International Conference on AI in Engineering, "Artificial Intelligence in Design",* ed. JS Gero, Amsterdam: Elsevier/ Computatonal Mechanics, pp 385–403.

Serrano, D and Gossard, D, 1988. "Constraint management in MCAE" *Proceedings of the Third International Conference on AI in Engineering, "Artificial Intelligence in Engineering: Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 217–240.

Smithers, T, Conkie, A, Doheny, J, Logan, B and Millington, K, 1989. "Design as intelligent behaviour: An AI in design research programme" *Proceedings of the Fourth International Conference on AI in Engineering, "Artificial Intelligence in Design",* ed. JS Gero, Elsevier/Computational Mechanics 1989, pp 293–334.

Talukdar, S, Rehg, J and Elfes, A, 1988. "Descriptive models of design projects" *Proceedings of the Third International Conference on AI in Engineering, "Artificial Intelligence in Engineering: Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 379–392.

Tomiyama, T and Yoshikawa, H, 1985. "Requirements and principles for intelligent CAD systems" in *Knowledge Engineering in Computer-Aided Design,* ed. JS Gero, Amsterdam: Elsevier, pp 1–23.

Ullman, DG, Dietterich, TG and Stauffer, LA, 1988. "A model of the mechanical design process based on empirical data: A summary" *Proceedings of the Third International Conference on AI in Engineering, "Artificial Intelligence in Engineering: Design",* ed. JS Gero, Amsterdam: Elsevier/Computational Mechanics, pp 193–216.

Vernon, 1987. "A systematic approach to the design of systems" *IEE Electronics and Power,* June, pp 371–375.