

From Functional Specification to Concept Design - Strengths and Weaknesses in Some Current European Approaches

by Daniel E. Whitney

Summary

Most commercial computer aided design (CAD) is not really design software but instead either supports two dimensional drafting or three dimensional geometric modeling, with added text representing dimensions, tolerances, process notes, and so on. The most advanced commercial CAD software permits geometry to be parameterized by numerical or symbolic arguments, with some equality and inequality constraints on these variables. However, such software mainly supports creation of geometry at a detailed level and does not directly permit engineering, exploration of rough concepts, or discovery of conflicts and tradeoffs. When commercial CAD companies and their customers speak of "concept design," moreover, they usually mean exploration of geometric shapes, often rather creative shapes. In other cases they mean exploration of space allocation, typically involving packing components inside a given skin. [Brown]

Researchers have taken note of these gaps and are trying to create CAD that will help designers explore non-geometric concepts and to create rough realizations from loosely stated design goals and required functions. This article reports on several European and UK labs that are working in this area. Their approaches include one or more of the following:

- ° scripts or design procedures that guide the designer from the specifications to classes of realizations in terms of known elements
- ° diagramming methods that permit the designer to hook elements together in different data views
- ° engineering-oriented approaches that permit the designer to think functionally about groups of predetermined elements, with the computer providing some engineering knowledge or constraints

Behind all of the research projects reported on here (and most others in this area that the author is aware of) is the assumption that "product design" consists more or less of a set of steps that one engages in and passes through successively, while establishing and refining information. This relatively clean view contrasts sharply with what many in industry actually experience: superimposed on (and often dominating) the set of steps, there occurs sharp conflict, wide gaps between specifications and possible realizations, and a constant struggle to predict future problems and costs, understand the needs of other designers, and so on. No lab visited or known to me represents design as a struggle to identify and resolve conflicts or bases its research on such a view. Yet this view describes real design better than existing research or teaching paradigms and implies great needs for computer aids of a type that no one is trying to create.

A summary sense of the state of these efforts can be gained from the following generic anecdote. Each of these research groups showed me points in their software at which a conversion was made between a more abstract representation and a more concrete one (from the statement "convert energy" to a diagram of a motor-generator set, for example). In every case, when I asked "Did the computer do that or did the designer?" the answer was "The designer."

I believe the fair thing to say at this time is that this software plays a role similar to that of personal computer software of a type called "productivity enhancers." Such PC software claims to help stimulate and organize ideas for writers and other creative people. One can brainstorm into the computer, diagram links between ideas, and so on. But such software cannot respond to the command "Write me an intelligent paragraph summarizing important research topics at this university."¹

Similarly, concept design research software at this time can be characterized as "inspired sketchpads," capable of generating and searching structured lists, constructing graphs that connect elements in various ways, or, occasionally, accessing rules or tables to help evaluate performance of elements. Good graphical user interfaces are being developed to support these activities. But little has been done so far to exploit the structures thus created in a systematic way, such as checking for correctness and completeness. Neither has anyone taken the obvious step of converting a correctly constructed graph into any existing systematic dynamic simulation modeling methods such as Bond Graphs, although most say they plan to do so.

In all of the above respects, the labs visited show remarkable similarity.

The bottom line is that except for focussed situations which essentially constitute redesign of an existing item using the same kinds of fairly simple elements, there is not very much progress on systems that really aid the designer other than bookkeeping. The reasons for this situation may be a lack of basic engineering knowledge, the lack of a mature concept of a product data model that can link form and function, and/or the lack of a mature concept of the "product design process."

The sections that follow reports on the following universities and laboratories:

- Technische Universität Aachen (two projects)

- Lancaster University (two projects)

- Technische Universität Berlin (summary of material in the ESNIB article "Design Research at the Fraunhofer Institute of Production Technology and the Technical University of Berlin, and an Industrial Application of Systematic Design Methodologies")

- University of Leeds (summary of material in the ESNIB article "From Geometric Modeling to Product Data Models: Collaboration Between Engineering, Computer Science, and Industry at Leeds University")

¹ Significantly, fewer products of this type are for sale now than were a year ago.

Technische Universität Aachen

The Werkzeugmaschinenlabor (WZL) (machine tool laboratory) at the T-U Aachen is one of the world's great mechanical engineering research centers. It is allied with the Fraunhofer Institut für Produktionstechnologie (IPT) next door that works with industry in similar areas. Both are devoted to a range of production technologies, the required research, and associated industrial consulting and technology transfer. The WZL and IPT occasionally step on each others' toes since the WZL wants to have contracts with industry while the IPT wants to work in some of WZL's research/applications areas, but the problems are small since the directors of the five main labs in both institutes are the same people. One feature of the division of labor between the two organizations is that IPT does not deal in issues relating to design of machine tools. That is done only at WZL.

Interestingly, the WZL's technology and research are far ahead of anything the German machine tool industry can absorb. "The companies are proud if they have one PC with AutoCAD," said one researcher. So while the name says "machine tools," the reality is that the best work is supported by and done for the automobile and aerospace industries.

Two design projects concerning conversion of requirements into a design are going on at WZL. One of these aims at permitting a designer to hook a set of machine elements together to form a system, analyze them, design them by CAD, and so on. The other is more conceptual and is intended to permit a designer to think up new configurations of functions and parts and look at them from the point of view of assembly.

Machine System Design

This project is being carried out by Mr. Repetzky, a Dr.-ing. candidate. He calls it CAE of the future. The goal is to integrate all of the tools a designer needs for designing a complex machine: CAD, dynamics, simulation, FEM, machine elements like gears and bearings, hydraulics, controls, and so on. He has been at work on it for a year.

A major goal is to combine the different kinds of data that a designer would need in order to attack such a problem, and store the data in a unified representation. This would be more like how designers think of things, he says, as opposed to the different kinds of data representations that present-day software uses. His approach is the object-oriented method.

The project is an outgrowth of the general problem of putting functional design capability into conventional CAD. His view of function in machine system design is that each machine element responds to inputs and delivers outputs. So his first efforts have gone into defining data objects for the elements and placing calculations (methods) into the objects that define the input-output relationships. A major problem for him is to decide when he has described an element in enough detail. For helical gears, for example, must he include the helix angle? Another issue is to design the software and the user interaction so that calculations will be launched when they are needed. It is not clear if he understands message-passing or events. I am not an expert in these either, but I believe that he would benefit from having more sophisticated computer science help in his research.

The software at present supports a mouse-menu interface that permits the designer to extract objects from a library, put them on the screen, and hook them together. The elements have the required hooks on them already, and a design is not complete until all the hooks have been connected to something. For example, a drive motor has a hook for fastening it to the ground and another for fastening it to a rotating shaft on another element. A bearing has a rotating element that can be hooked to a shaft and another that can be hooked to a housing. The hooks are each responsible for fixing one or more degrees of freedom (DOF), and the software will eventually be able to tell if all DOFs have been accounted for. On the hookup screen, the elements are represented by icons and do not have any particular geometry. They just have the mathematical properties described above.

If Repetzky hooks together a motor, a shaft, some bearings, a gear reducer, and a load, he can calculate the shaft torques on both sides of the gearbox. He could add tooth load calculations to the gear object but has not done so yet.

He has considered using Bond Graphs as a way to link these hookups to simulation. I believe Bond Graphs could help because they easily calculate the above mentioned torques; in addition, they are designed to model hybrid systems, such as electro-hydraulic, using the same symbols and math throughout. In addition, the Bond Graph method contains a number of internal consistency checks that prevent some kinds of basic modeling errors, such as failing to conserve energy or attempting to define both the force on, and the velocity of, a moving object. Bond Graphs are based on lumped parameter modeling and assume discrete elements like torque sources and inertias, so they are suitable for modeling what he wants to represent.

Combining the consistency checks of Bond Graphs with the DOF accounting he now plans would give his system some capability for evaluating the correctness and completeness of a model. I think this would be an important property for a modeling system to have. No researcher I have visited has given this kind of thing high priority. In some cases designer can draw what he wants and the computer will try to model it.

Mr. Repetzky also wants to connect his element modeler to a CAD system so he can draw the real shaft, for example, and tell the system that it is the same shaft as the element in the hookup. Also, he would like to be able to optionally begin part of the design in CAD mode and cut/paste pieces back into the hookup. He feels that a major problem will be to propagate design changes consistently from one representation to the other, and he worries that it will not be possible to accomplish this automatically. I feel sure that without the consistency checks mentioned above, reliable change propagation will be difficult to achieve.

Assembly-Oriented Design

Mr. Baumann, another Dr.-ing. candidate, described a system called DEMOS, on which he and others have been working for many years. It supports design of mechanical items and has several objectives. First, it seeks to bring assembly issues to the concept design phase. To do this, it permits the designer to describe the product in terms of graphs that link parts without the geometry of the parts being specified. Second, it allows the designer to study the design

and improve its assembleability. This, however, is done after geometry is defined. A commercial solid modeler (EUCLID from MATRA Datasystems) and a commercial data management system have been combined with WZL's own rule and databases and user interface to create this system. Future plans include converting to the ACIS modeler, a product of Spatial Technologies, Inc., and an object-oriented data representation. A paper contains additional information [Eversheim and Baumann].

Nominally, the design process supported by the system begins with a functional description that is admittedly not very abstract; it merely permits the designer to link functional needs to individual mechanical parts. The system's strongest features deal with development of two data structures: the function structure and the assembly structure. The designer inputs both of these and the software supports the process by keeping track of all the parts, logging the designer's choices for connection or assembly methods that link parts, later accepting the shape of each part, and finally performing Design for Assembly (DFA) analyses. By ordering the process this way, Baumann hopes to encourage assembly-oriented thinking by the designer before the geometry is completely described, even though the last steps require geometry. The sequence also adheres to the German design standard VDI 2221.

Both the function structure and the assembly structure are hierarchical graphs in which nodes are single parts or subassemblies, while links indicate some kind of relationship. In some cases, the relationship is that of assembly. In others, it represents some functional aspect of the design. So in the functional graph of a gearbox, (See Figure 1) the top node is the **gearbox**, and lower nodes represent **input**, **output**, and **housing**. **Output** has subnodes such as **shaft**, **gear**, **bearing1**, **bearing2**. In the corresponding assembly structure, the top node is again the **gearbox**, but the next level contains a two nodes: a **subassembly of housing top** and one **bearing cap**, and **all the rest** of the gearbox comprising, at the next lower level, a **subassembly for housing bottom** and the other **bearing cap**, plus **subassemblies** for the **input** and **output shaft-gear-bearings** sets.

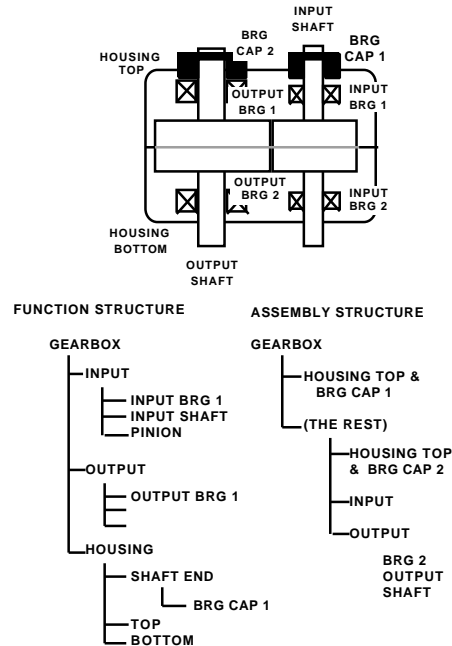


Figure 1. Illustration of Function and Assembly Structure of a Simple Gearbox.

Once the assembly structure has been drawn up, the system attempts to generate an assembly sequence. To do this without knowing much geometric data, the system uses several heuristics. For example, the part with the most connections to it is selected as the "base part" onto which others are added. (The upper housing or lower housing would be chosen in this example.) Parts with lots of connections to each other are candidate subassemblies. The system seeks to create as many independent subassemblies as possible using such rules. The designer can edit or alter the suggested assembly sequence.

Once the assembly structure and assembly sequence are determined, the designer can assign types of part mates to the assembly links. Library features like bearing seats can be called forth, including dimensions if the designer wishes. Last, the designer uses the CAD module to make up the actual shapes of the parts that contain these part mates. When the system has all the geometry and assembly structure information, it carries out a DFA analysis and suggests places where part count can be reduced. Tables from the Boothroyd method are used for this purpose.

Mr Baumann contrasted this work with that of Mr Repetzky by indicating that DEMOS seeks to support design of completely new things whereas Mr Repetzky was trying to model systems made up of standard components. The distinction is not clear to me, especially in view of the fact that a gearbox clearly can be built up of standard items. My comparison of the two projects is as follows:

Mr Repetzky is facing the problem of converting specifications and functional requirements into physical realizations, whereas Mr Baumann is facing the problem of helping the designer think of the product as something to be built rather than just something that works. The two methods, or descendants of them, will obviously have to be merged in order to create more complete design support systems.

Side Comment

The assembly sequence strategies used by the DEMOS software raise some interesting questions. For example, it is not obvious that maximizing the number of subassemblies is always a good idea. The reasons for doing or not doing this may depend on information from marketing or assembly machine experts, for example. Some of the considerations are discussed in the ESNIB article on Telemechanique,² where product modularity is the driving consideration. Other researchers (see ESNIB article on IEEE Robotics and Automation Conference: Special Symposium on Assembly Planning³) deal with these complexities by permitting the designer to impose other subassembly strategies or even generate all feasible assembly sequences and choose among them. However, these methods require the part geometry to be defined. The type of mates also influences these choices, but in DEMOS the assembly sequence is chosen before the part geometry and mate type.

The larger issue is whether one can make significant design decisions without having defined specific geometry. Assembly decisions illustrate the question here, but it comes up in many other contexts. DEMOS is a bold effort to create an environment where decisions can be made without geometry but at present it seems to me that many of those decisions are destined to be revised later.

The next question is whether the design is better anyway, even if the decisions were redone, just because the designer had to think about assembly much earlier than would normally be the case. No one is thinking about design processes this way: revisions are usually thought of as a sign of waste or inefficiency in the process.

Lancaster University

The two concept design projects described here are under the direction of Prof Michael French, who established the Mechanical Engineering Department at Lancaster 24 years ago. French is an experienced mechanical designer who returned to academia because he felt that design was not as well taught as it could be. His approach is to be as analytical as possible but to attempt at the same time to articulate "design principles" that go beyond analysis and describe what designers do.

Example principles are
In general:

provide clarity of function (know what each component does, ensure that it does it well and simply, and is not compromised by trying to do other things)

achieve matching (symmetry, equal sharing of functional responsibility, sharing out loads equally, etc.)

² Systematic Design of Modular Products at Telemechanique

³

take care in investigating alternative nesting orders for things (the nesting order of typehead and paper movement in old fashioned and ballhead typewriters is completely different)

In structural engineering:

- use short direct force paths

- use single rather than multiple force paths

French realizes that not all of design can be made analytical but he is skeptical of some "received wisdom" (the "expertise" that expert system builders try to capture) and suspects that trends seen in designs could be given an analytical explanation and foundation but simply have not. An example is his observation that in double track vehicles (cars) the steering function is nested between the wheels and the springs; in single track vehicles (motorcycles) the springs are between the wheels and the steering function. Asking designers why it is this way does not yield satisfying answers, but a serious explanation is probably possible, based on kinematics, dynamics and, possibly, manufacturing considerations.

The two projects described are funded by the UK government under the Engineering Design Centre program. French's project is aimed at mechatronic design, that is, design of mechanical systems that contain computation, measurement, and control as well as familiar kinematics, dynamics, fluids, strength of materials, and so on.

Schemebuilder

Schemebuilder is like Mr Repetzky's system in many respects, with similar aims but perhaps a more analytical underpinning. It is written entirely in the commercial expert system shell KEE. There are two windows: a "building site" and a "model library." The building site is like Repetzky's hookup window and functions in basically the same way. The example here is mechatronic, in keeping with the theme of the Centre: an autopilot for a yacht. A compass provides a control setpoint for a servomechanism that will drive the boat's tiller to correct a heading error.

Two kinds of elements can be called forth, those capable of transmitting power, and those dealing only with signals. These can be joined in ways similar to those provided by Bond Graphs, including keeping track of causality. But the full force of Bond Graphs has not been utilized in the sense that no generic components (inertia, compliance) have been defined. Instead, each element represents an individual physical type of thing (tiller with rotational inertia, boat with translational inertia...)

Design begins by calling forth specific elements, such as the yacht's tiller. These elements are described only qualitatively, such as noting that the water will exert a force on the tiller in one direction while the steering motor will exert a force the other way. The motor will act through an Acme screw, whose function is described qualitatively as converting rotational input into translational output. Quantitative descriptions are going to be added later in the project.

Each element has hooks that permit restricted kinds of connections to be made. A browser is available in the model library that helps the designer find suitable elements for a given hook type. For example, to provide translational power for the tiller, the list could include electric or hydraulic actuators but would not include things incapable of driving a load. However, the list presently also includes solenoids, which are too small to drive yacht tillers, an indication that qualitative factors dominate the system at the moment.

Several extensions of the current system are planned. One deals with design concepts like function and advice/warnings to the designer. The other deals with linking the concept to a CAD system so that space can be allocated for each of the components as it is placed in the building site.

Curiously, Prof French does not feel that the warnings and advice feature should be based on making the software understand engineering fundamentals. He feels that would be appropriate for design of really new things, whereas here he is dealing in some sense with hookups of already designed and understood things. Instead he would like to link the advice and warnings to a more sophisticated formulation of function as it relates to the elements selected from the library. He uses the term "function structure" to describe statements of functions perhaps at a semantic level, to which the system would respond with solution types built of library components that match the description. Once a pattern of solutions began to emerge for a given problem, the advice might draw on the existing design; for example, if some hydraulic components have been selected by the designer, the system might select more hydraulic components to mechanize other functions in order to exploit the hydraulic power supply that is already required. No approach to providing this semantic facility has been identified as yet.

System for Improving Mechanical Assembly Design

This piece of software is a model editor for assembling cylindrical things with a single rotational axis of symmetry. It is a deceptively simple context and a brilliant one because its simplicity forces certain basic issues into sharp focus while keeping side issues from clouding the discussion.

One type of assembly is involved, namely placing gears, turbines, bearings, spacers and their required fasteners onto stepped shafts. The designer must create the stepped shape and indicate the steps onto which the bearings will rest. Each set of parts starting from a step and proceeding along the shaft through bearings and spacers to a fastener is called a "stack." The system has been programmed to recognize stacks and to understand some of their inherent constraints. For example, a stack with no fastener is incomplete.

The system understands several geometric facts about such assemblies. For example, it can recognize stacked stacks: a step followed by a bearing, a spacer, another bearing, another spacer, and finally the fastener. It can also recognize the opportunity to create a stacked stack out of two serial stacks by responding to the command "Reduce number of fasteners." Finally, it knows when assembly is impossible because a step is too high to permit a bearing to pass

over it on the way to another step. This error can occur if the system discovers that a step is too short to support the bearing assigned to it, and attempts to make the step higher.

However, the system is unaware of some basic engineering facts that would seem to be natural for it, especially given the background of Prof French. For example, it does not really understand the concept of load path, that is, the idea that the fastener is going to push the spacer against the bearing which will in turn push against the step, trapping the bearing with a compressive force. At present, when the designer places a bearing near a step the system will not place the bearing against it in anticipation of the direction the force will ultimately point; instead the designer must move the bearing with the mouse until it coincides with the step visually on the screen.

An extension of this idea is to recognize when axial forces must be resisted. Helical gears generate such forces. Alternately, angular contact bearings require preloads. In such cases a load path to a fastener is needed, and the design is incomplete until this path is provided. All such paths comprise loops through the structure, with alternating loop segments in tension and compression, all adding up to zero net force around the loop. This is another case where "correctness and completeness" could be checked systematically. The student does not seem to want to add such understanding to the system but instead prefers that the designer realize it.

The potential benefit of adding it to the system is that this provides an opportunity to study in a simple but non-trivial context the question of how to link geometric design to real engineering (elements cause loads that have to be supported using steps and fasteners). It would be very satisfying to see a system of this type developed as a counter-example to other research where such physical facts are deduced as "rules" employed by designers.

Related Work at Other Universities

In other articles I have described in some detail work along these lines. Rather than repeat it here, I will summarize it briefly and refer the reader to the detailed article.

Technische Universität Berlin

The work of Prof Beitz and his students also consists of providing a computer interface to concept design in the form of linking elements on a computer screen. Each element comprises physical and logical properties that can be exploited in various ways. The user can make legal hooks between elements and can construct a "system" that the computer will ultimately be able to link to a simulation. As the design proceeds to more and more detail, the computer support becomes more substantial and the element descriptions become more physical. At the lowest level I saw (how to fasten hubs to shafts) there is an extensive rule base and some fundamental engineering calculations. However, constraint satisfaction is up to the user.

University of Leeds

Prof Neal Juster and his student Jim Baxter are trying to define functions provided by mechanical parts. Example functions are "provide support," "locate position," "stop leaks," and so on. A graphical representation is being tried, in which all the items that perform the same function are linked into a graph. Some syntactic checks are possible. For example, if the "support" graph is disconnected, it means that the design is in error since some of the parts are in fact unsupported. An engineering check could also be made: if the same part provides both fastening and location to another part, the design is a poor one, especially if both functions are to be provided by a screw. In good designs, location is typically provided by separate locating pins, spigots, steps, or other specific geometric features independent of the fasteners.

Comments

After reviewing the above projects, it is tempting to ask why all this seems to be so difficult. One researcher I visited spoke wistfully of gate synthesis in electronic logic. This was a done deal at least two decades ago. Algorithms exist that will convert a given Boolean algebra or truth table representation of a desired logic function into the minimum number of logic gates and their required hookup. Why is this possible?

There may be a good mathematical answer to this question, but I do not know it. This researcher and I surmised that logic gates have certain basic properties that mechanical elements just do not have, and these make the difference:

- each gate is discrete
- the gates do not back-load each other but instead behave as a one-way logical cascade
- a gate's behavior is dominated by logic; any physical behavior (heat, thermal expansion) is secondary to its behavior and does not affect it except catastrophically)
- each gate does exactly one thing, does it purely with no side effects, and does it so repeatably that tolerances are not an issue

Typical mechanical and mechatronic elements simply do not have these simple properties.

Conclusions

Each of the projects described above is attempting to tackle a genuinely difficult problem, one that often has been left to "creativity" and deemed too unstructured for systematic attack and computer aids. So far, it appears that the problem is living up to its reputation. There may be some underlying reasons for this.

The most likely one is that the problem is indeed too difficult because reduction of requirements to a concept requires too much knowledge; furthermore that knowledge is not well structured. In a recent paper [French 1990], Prof French lists the following knowledge areas that a designer must encompass to greater or lesser degree:

1. engineering science combined with physical insight
2. ability to model a problem analytically, including when to simplify
3. ability to organize work and proceed step by step
4. ability to recognize key decisions in a complex problem and link them together
5. invention (!)
6. aesthetic judgement enabling the designer to distinguish good solutions from bad ones
7. possession of a wide repertoire of methods and solved past designs
8. "received wisdom" about accepted practices in one area of expertise

Of these, he feels that research is feasible in areas 2, 3, 4, 7, and 8. In omitting 1, I believe he feels that the basic facts about statics, dynamics, and so on are already well understood. The task is to extend modeling to the repertoire and received wisdom areas and raise the latter two above the current level of "rules."

Thus I return to the theme of the ESNIB article called "Perspectives on Artificial Intelligence in Design:" the weakness in current approaches is at least in part due to insufficient understanding of the underlying engineering facts or insufficient effort to model them (the load paths in this article, or the machine tool spindle in the AI article - also involving load paths). The designers are the wrong people to ask since their approaches are too intuitive. In the near term it may not help to continue trying to build design aids using graphical interfaces, word searches, rule bases, or neural nets because they stand on a weak foundation in the engineering fundamentals. Instead, thought should be given to identifying areas in engineering science that could be recast in terms of analytical design principles. Design of machine elements is clearly a candidate where enough work may already exist to permit researchers to skip the expert systems and go straight to the theory.

References

[Brown] "1992 Conceptual Design Survey," a multiclient study done by D. H. Brown Associates, Inc., 222 Grace Church St, Portchester NY, 10573

[Eversheim and Baumann] W. Eversheim and M. Baumann, "Assembly-oriented Design Process," Computers in Industry, v 17 (1991), pp 287-300.

[French] Michael French, "Research in Engineering Design: "Some Proposals for Improving Research, Teaching, and Practice," J. Eng. and Technology Mgt, v 7 (1990) pp 145-51.

Points of Contact

Prof Dr.-Ing. Manfred Weck
Lehrstuhl für Werkzeugmaschinen
WZL - T H Aachen
Steinbachstr 53 B
D-5100 Aachen
Germany
phone 49 241 80 74 07, fax 49 241 87 34 42

Prof Michael French
Lancaster University
Engineering Design Centre
University House
Lancaster LA1 4YR
UK
phone 0524 65201, x3138
fax 0524 381707

Prof Dr.-Ing. Wolfgang Beitz
Technische Universität Berlin
Institut für Maschinenkonstruktion
Strasse des 17 Juni 135
D-1000 Berlin 12
Germany
phone 49 30 314 23341
fax 40 30 314 26481

Prof Alan de Pennington
Department of Mechanical Engineering
University of Leeds
Leeds LS2 9JT
UK
phone 0532 332112
fax 0532 424611