

Functional descriptions used in computer support for
qualitative scheme generation - “Schemebuilder”

R.H. BRACEWELL AND J.E.E. SHARPE

Engineering Design Centre
Lancaster University
Lancaster LA1 4YR
United Kingdom
Phone: (01524) 65201x3145
Fax: (01524) 593042
E-mail: edc@comp.lancs.ac.uk

FUNCTIONAL DESCRIPTIONS USED IN COMPUTER SUPPORT FOR QUALITATIVE SCHEME GENERATION - "SCHEMEBUILDER"

Abstract

With increased pressures coming from global competition and requirements for greater innovation in product development, designers are hard pressed to deliver designs of higher quality and variety using a repertoire of technological options from different disciplines. This interdisciplinary product development approach has not only removed many of the traditional constraints to design but has now given designers a much wider freedom of choice as to the best solution to a design problem. The focus of this paper is a knowledge-based design environment called Schemebuilder which is a comprehensive and integrated suite of software tools aimed at supporting the designer in the rapid development of product design models in the conceptual, through embodiment stages of design. Illustrated is the use of the software tools in the qualitative generation of alternative schemes, by application of stored working and decomposition principles in the development of a function-means tree-like information structure. With mechatronic product development as the main theme, this paper describes a closely integrated methodology that incorporates a bond graph approach to continuous-time energetic systems and high level Petri nets for the rigorous description of discrete-time information systems. Additionally, a technique is suggested for the decomposition of free format statements of need into the rigorously-defined design context and required functions which form the starting point of the function-means development process.

Keywords: Conceptual design, Interdisciplinary system design, First principles, Computer support, Mechatronics.

1. INTRODUCTION

The engineering design process typically entails the construction of a description of an artefact that satisfies a formal functional specification, within a defined context. It must meet certain performance requirements and resource and environmental constraints; be realisable in an appropriate target technology and perhaps satisfy one or more other criteria, for instance manufacturability, reliability, safety and testability. A common perception of engineering design is that of converting a need - expressed as an abstract concept in terms of general functionality - into a product fulfilling that need; the process involving the mapping of the specified functions onto realisable physical structures. To this end, most design is based on widely held working principles.

The engineering design process is, of necessity, a complex one that requires the designer to exercise initiative and structured thinking as well as deploy a wide range of skills and expertise in attaining a solution. In an interdisciplinary design environment, such as that involving a Mechatronics¹ approach to design, the designer is also often required to operate in a very general mode with a view to the possible use of a wide range of alternative technologies. The increasing occurrence of interdisciplinary product development has not only removed many of the traditional constraints to design but has given the designer a much wider freedom of choice as to the best solution to a particular design problem. Given the time constraints frequently imposed on present day product designers as a result of the constant demand to reduce the time-to-market of new products, designers are often not able to develop and assess sufficiently rapidly, the various alternatives that may be made available through the use of interdisciplinary design.

¹ Mechatronics may be considered as the integration, at all levels and throughout the design process, of mechanical engineering with electronics and computing technology to form both functional interaction and spatial integration in components, modules, products and systems. Common examples of products that are the result of a Mechatronics approach to design are camcorders and auto-focus cameras.

Much of engineering design is carried out by individuals or groups of individuals who have been trained in one engineering discipline with limited knowledge or experience of other fields of engineering. Consequently, these individuals often “lock” onto traditional and familiar technologies and design solutions that frequently result, are less than optimal. Logically, there are three obvious ways to enable interdisciplinary product design: (1) educate all designers with a generalists' knowledge of many technologies, (2) collocate technology specialists to encourage communication and co-operation in the design team and (3) to provide multidisciplinary computer support, which is the focus of this paper.

With the advances made in computer-aided design and the slow infusion of AI methods such as functional reasoning, constraint processing and planning into traditional computer-based design tools, it is not difficult to see how the third alternative of the above approaches might arise. A computer system might support and guide the designer through the range of technological options available, providing appropriate tools to perform processing of the design data at proper levels of abstraction, including concept generation, technology selection and matching, model execution, optimisation, and visualisation. However, no such integrated environment has yet been developed, hence the aim of Lancaster University Engineering Design Centre is to research the design methodology and underlying ontology required to create such a system.

This paper describes an integrated suite of software systems called Schemebuilder which are currently being researched and implemented. They provide highly integrated support for the rapid creation and evaluation of a wide range of outline schemes incorporating a range of technologies so that irrevocable decisions need not be taken on the basis of biased or inadequate information. Following an overview of Schemebuilder it is then shown how qualitative conceptual schemes and their alternatives can be created from first principles using

strictly defined and applied functional descriptions via the co-operative interaction of human and computer. Having generated alternative concepts, it is then necessary to size the components of the schemes developed by the process described in this paper. These may be visualised, dynamically simulated and animated as fully-rendered solids in 3d display windows. Although not described in detail here this is a major part of the overall Schemebuilder computer support - Bracewell et al. (1995).

2. THE SCHEMEBUILDER PROJECT

2.1. Aims of Schemebuilder

The main remit for Schemebuilder is to assist the designer in the conceptual and embodiment stages of design, including problem analysis, for interdisciplinary systems including machine design, process design and Mechatronics. The design of Mechatronic systems, which combine mechanical, electrical, electronic and software systems, presents a challenging array of possible implementations, often based on proprietary equipment but with opportunities for elegant and original design solutions that require the deployment of advanced electronics and integrated software to provide the required levels of functionality. The design of processes based on mass transfer presents another set of challenging design problems that are rarely integrated with machine or mechatronic systems. To take full advantage of these opportunities requires the ability to generate and quickly evaluate alternative schemes from first principles to practical embodiment.

Schemebuilder is seen as a highly integrated “design workbench”, where designers are guided through the range of technological options available, aided by the provision of tools for rapidly accessing relevant information. It acts both as a facilitator for the creation of a model of the system to be designed and an advisor on the appropriate means to achieve the design.

The definition of a scheme follows that of French (1985) where a scheme is an “*outline*

of a solution to a design problem, carried to a point where the means of performing each major function has been fixed, as have the spatial and structural relationships of the principal components. A scheme should be sufficiently worked out in detail for it to be possible to supply approximate costs, weights, and overall dimensions, and the feasibility should have been assured as far as circumstances allow. A scheme should be relatively explicit about special features or components but need not go into much detail....”.

It must be emphasised that the philosophy underlying our work is a move from automated design using the expert systems approach - Rychener (1988) - to one embracing a more co-operative relationship between man and machine - Oh (1993). We believe that the most effective approach to computer-based design tools is for the computer to provide decision support and allow the human designer to supply the judgement.

2.2. The Schemebuilder Environment

Schemebuilder provides an integrated assortment of design tools matched according to the specific needs of different design tasks building on a partly bond graph based ontology of rigorously defined physical and information functions. The provision of verification tools like those for simulation in the Schemebuilder environment (Figure 1) helps to tighten the link between design and analysis, and enables designers to predict the behaviour and performance of their digital mock-ups as early as possible. In short, Schemebuilder allows for the exploration of new ideas with minimal risks.

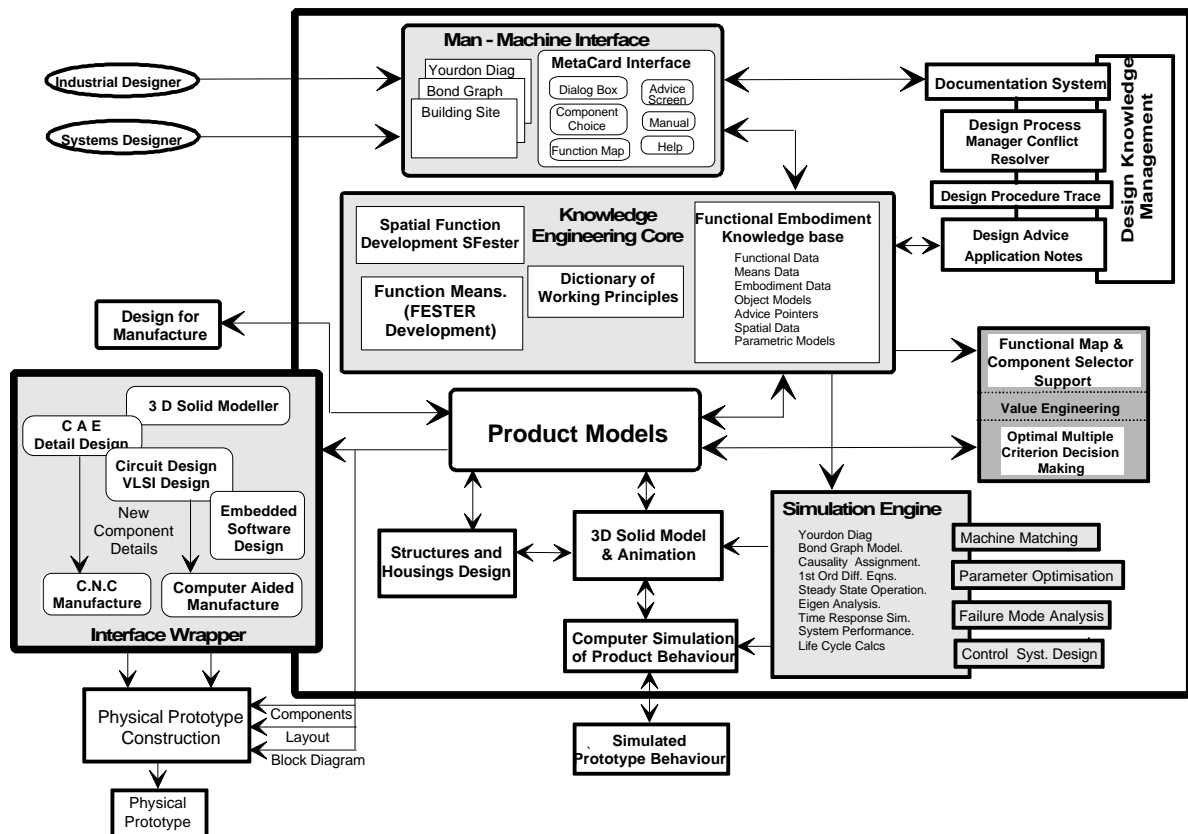


Figure 1. Overall architecture of the Schemebuilder environment.

The complete Schemebuilder environment will provide enabling tools to:

- assist in the production of specifications
- define functional context
- synthesise alternative functional schemes
- invoke necessary secondary support functions
- automatically generate context-based figures of merit for alternative function schemes
- provide advice and browsing facilities on available technologies
- allow definition of a working principles, as schemes of primary functions, classified under the higher level function that they embody
- allow classification of a component as a means of performing one or more alternative functions, with the aid of one or more supporting functions if necessary
- produce system models for dynamic simulation and parameter optimisation

- support embedded software design and development
- monitor system integrity, checking for continuity and matching
- produce fully rendered 3d solid layouts, which may be manipulated by the designer within known constraints and animated according to simulation results
- design casings and support structures that may be displayed and animated
- aid in the identification of “technological holes” in areas of useful functionality

2.3. Bond Graph Ontology

As products become more complex and highly integrated, design teams find it increasingly necessary to have a common language, covering all of the traditional engineering disciplines, in which to communicate. Bond graph methodology - Paynter (1961), Karnopp et al (1990) and Sharpe (1978), with its common structure and clear rules across engineering domains, provides a powerful basis for Schemebuilder's representation of the functional aspects of energetic systems.

Bond graphs are a formal representational language for analysing physical systems developed originally for system dynamics applications. They have been used successfully in the representation of formal models for mechanical, electrical and process design, for example - Finger and Rinderle (1989), Ulrich and Seering (1989), Cellier (1991). There has also been an increasing interest in the use of bond graphs for qualitative modelling in AI because of their simplicity and representational power -Fishwick (1989), Top and Akkerman (1991) and Soderman and Stromberg (1991).

The bond graph methodology is based upon the conservation of energy, with physical processes being linked in a labelled digraph through energy flows. Basic concepts like effort, flow, inertia, and capacitance are used in modelling, and their generality allows the method to

be applied across domains like thermodynamics, rotational and translational mechanics, fluid dynamics and electronics. Thus, bond graphs cover essentially all classical macrophysical domains in an integrated fashion. Once a bond graph has been constructed for a physical system, traditionally one can generate a block diagram or state-space representation of the full differential equations that describe the system and then proceed with formal analysis.

The essential value of the bond graph methodology is their provision of a very simple set of primary functions and the ability to develop useful qualitative inferences from the analysis of the causal ordering that occurs within a system. Similarly, it is possible to synthesise alternative functional schemes from known imposed causal requirements.

Whilst bond graph methodology forms the basis for the functional reasoning aspects of Schemebuilder within energetic domains, the actual component modelling and simulation utilises the object-oriented modelling facilities of Dymola. This commercial software package can be seen as a logical development of bond graph thinking and indeed was produced by leading bond graph researchers - Cellier (1991), Elmqvist and Brük (1995). Graphs and graph-theoretic algorithms still form the basis of Dymola, but they are now kept out of sight of the user. Models are instead defined as a collection of submodels - instances of classes, defined in an inheritance hierarchy - and/or equations, which are automatically rearranged by Dymola in order to assign the correct computational causality of the assembled model. A major advantage of Dymola is its ability to include 3d multibody dynamics, efficiently but rigorously, in its simulations - Otter et al. (1996). This is an area where traditional bond graphs are clumsy and often fudged, particularly in the modelling of mechanical references such as support structures.

3. COMPUTER-AIDED METHODOLOGY FOR QUALITATIVE DEVELOPMENT OF SCHEMES FROM FIRST PRINCIPLES

The early Schemebuilder design workbench described in Bracewell et al. (1993), Sharpe et al. (1993), has been enhanced so that schemes may be developed from first principles using a development of the function means tree approach as described by Buur (1990). The process is facilitated by application of functionally classified working principles and bond graph based decomposition principles. It is performed within the constraints of a design context and in conjunction with a state transition model. This approach allows for the progressive refinement and development of a design from required functions expressed abstractly to the eventual physical embodiment of those desired functions. Backed by an assumption-based truth maintenance system (ATMS) - de Kleer (1986), it is also possible to maintain the development of all distinct alternative schemes, even though the designer may only progress a limited number of them at any one time. Figure 2 illustrates conceptually the process of progressing from a statement of need a set of alternative schemes, followed by the assessment and embodiment of these schemes.

Although it is important to provide a mechanism whereby a design may be developed from first principles, it should be realised that conceptual design may begin at a number of different starting points. Any satisfactory computer aided procedure must allow for this. Thus it is also possible to use Schemebuilder in “bottom up” mode, connecting components together directly in order to try out an idea or conduct a “virtual experiment”. Similarly, existing designs can be reverse engineered, applying the decomposition principles backwards to extract the basic required function structure, before going “top down” again to generate alternatives.

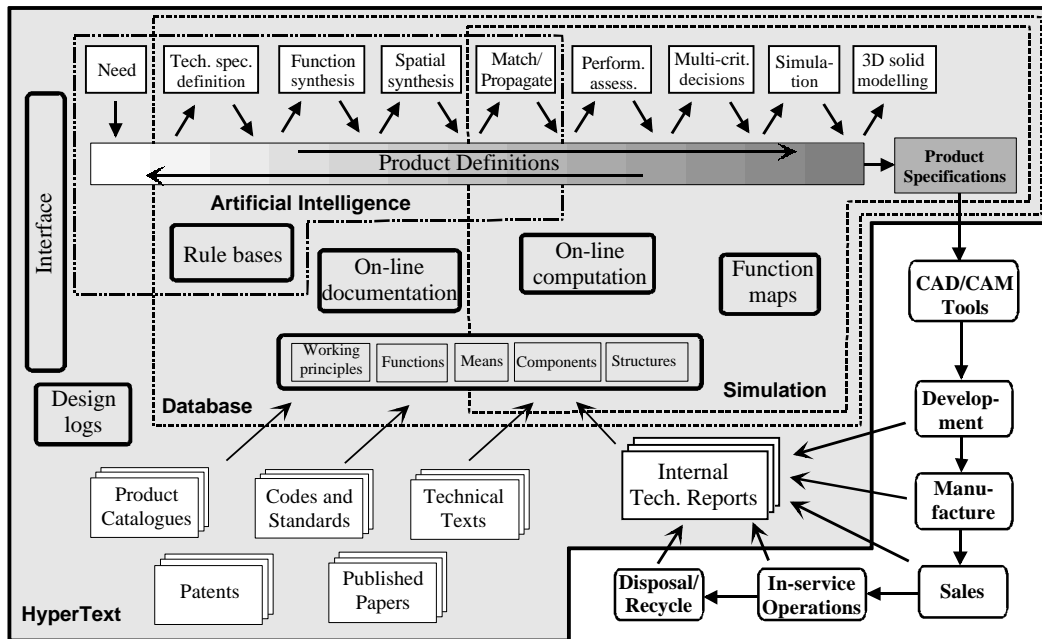


Figure 2. Schemebuilder conceptual design process.

3.1. Definitions

A *function* can be classified as either a *data function*, an *energy function* or a *mass transfer function*. All *functions* which are used must be pre-defined in the functional embodiment knowledge-base. *Functions* may have one or more defined *attributes*. *Functions* communicate with each other and with *components* via *links*.

A *link* may carry energy, data or mass. In a completed scheme, both ends of every *link* must be connected to a *component* within the scheme. The *link* may traverse up and down a number of levels within the *FEST-ER*, between its terminating *components*. Each *link* in the *FEST-ER* of a particular design project has a unique identifying number.

An *attribute* more precisely specifies the action of a *function*.

A *functional embodiment* can consist of either a *means* or a *principle*.

A *means* consists of at least one *component* and if necessary, one or more required *functions*, which may have certain required *attributes*. *Links* joining *components* and required *functions* within the *means* must be defined.

A *principle* consists of one or more required *functions*, which may have certain required *attributes*. *Links* joining required *functions* within the *principle* must be defined.

A *component* represents a family of known, available physical items, for which quantitative information is stored in a table of the component database. This data allows sizing, matching, layout and simulation to be performed later in the design process. Components possess *ports* which form the end-points of *links*, allowing input and output of energy, mass or data from the *component*.

FEST-ER is an acronym for **F**unctional **E**mbodiment **S**tructure - **E**xtended **R**ecursively. It is an acyclic directed graph-based information structure, inspired by and extending the capabilities of, the function-means trees of Andreasen (1990). The *FEST-ER* is developed by the designer's free but guided choice of alternative stored *functional embodiments*.

3.2. Decomposition of the Expression of Need

In Schemebuilder, the designer may develop a conceptual design from first principles, exploring the myriad of alternative functional schemes that are physically possible before choosing the best alternatives to pursue into detail. The process begins with some expression of need, combining normal language and related numerical information. For example, a typical expression of need might be defined as follows:

"The need is for a flow of chilled water at 5^oc for purpose of cooling power electronics. The flow is to be 1 kg per second within a closed circuit. The return water temperature is 15^oc. Power is to be taken from a variable frequency voltage source."

The first step is for this unstructured textual specification to be broken down into a design context, a set of required functions plus attributes, the required mass, energy and information flows and a set of required working points of the energetic flows. The design context consists

of ambient environmental factors and physical external entities with which the product must interact. These are illustrated in Figure 3. The required functions must be members of Schemebuilder's pre-defined, hierarchically classified, functional embodiment knowledge-base. This contrasts with traditional function-means tree approaches, which have tended to use a rather looser, more open ended definition of function, frequently but not exclusively using arbitrary verb-noun pairs. This is the approach described by Hansen (1995), who advocates instead the use of entity-relationship modelling in the breakdown of specifications.

Each instantiable function may be accessed via a number of associated keywords - the common usage terms by which they are generally known. The keywords in turn will appear in a thesaurus containing synonyms of those keywords which may also appear in statements of need. Using these facilities it is possible to help the designer, by means of prompts and iteration, to establish the context of the design, and the required functions. For example, considering the word "chilled" in the expression of need above, a search of the keywords identifying known functions leads to the required function HE.1, an ideal heat engine with unidentified energy domain for its work port.

There is considerable work going on world-wide into the creation and use of a software dictionary and thesaurus. Examples include the work of Mirbel (1995), which incorporates fuzzy reasoning to rank concepts in order of semantic nearness, the EDR project in Japan - Yokoi (1995) and the WordNet package from the United States - Miller (1995), which is freely available by ftp.

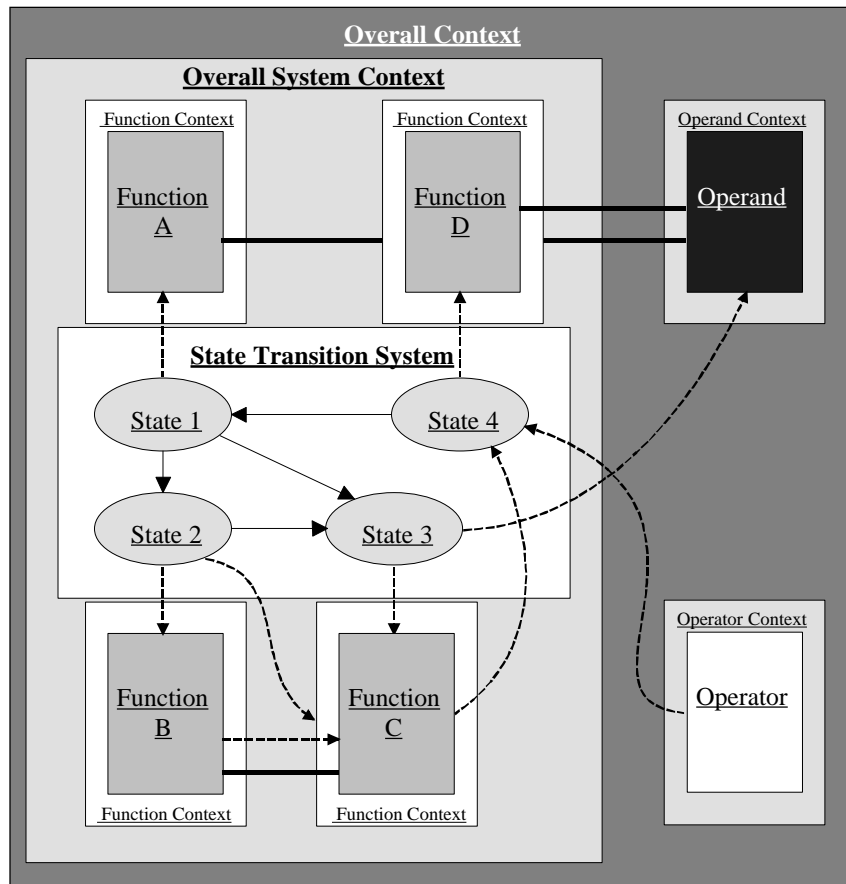


Figure 3. Functional and State Transition System Context

3.3. The Functional Embodiment Knowledge-base

The functional embodiment knowledge-base is a tree structure of functions, in which energy flow types, mass flow types and data carrier types are progressively defined as you approach the leaves of the tree. This allows principles to be stored at a high level of abstraction, with energy types undefined, then inherited into all appropriate energy regimes. A crucial aspect of the structure is the ability of single component types from the component database to appear in multiple means, embodying different functions at different points in the tree. This reflects the fact that components often are used in a number of alternative and sometimes unusual ways. A simple example of this is shown in Figure 4, where it can be seen that the component “chassis” appears in means of embodying both functions CFJ.ROT and CFJ.TRAN, which are rotational and translational common flow junctions, respectively.

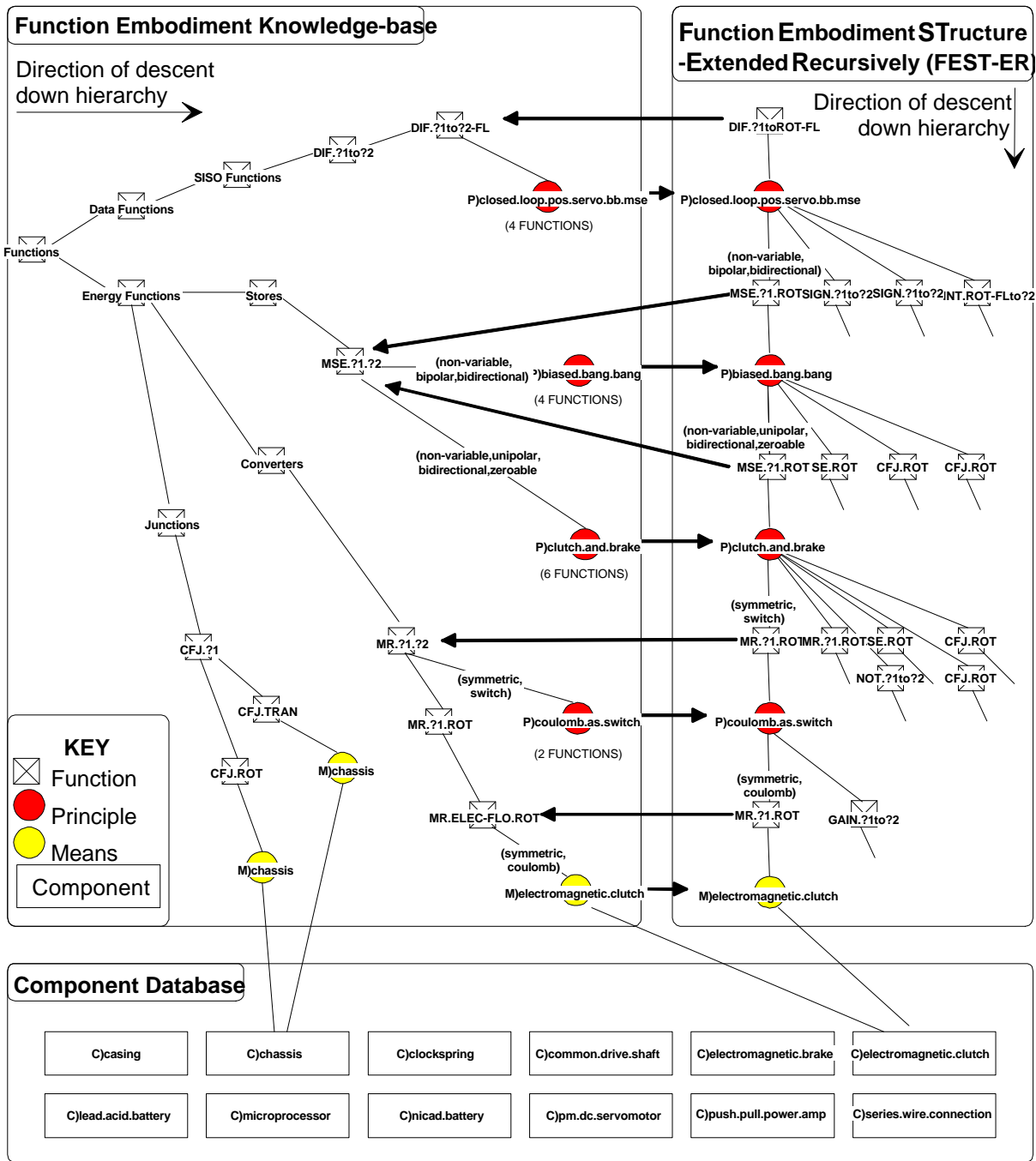


Figure 4. An example of the Functional Embodiment Knowledge-base

3.4. Working Principles and their Application in Functional Synthesis

Let us now consider how working principles can be applied in the embodiment of functional requirements. Recalling the chilled water example above, one of the required functions was HE.?1, an ideal heat engine with unidentified energy domain of its work port. Classified with

this function in the functional embodiment knowledge base are a group of working principles relating to refrigeration (refer to Figure 5) These include: air standard cycle; absorption cycle; vapour recompression; evaporation; Seebeck effect; Peltier effect and the Joule-Thompson Cooler. Each of these alternatives has a required set of primary functions that may each be embodied in a number of alternative ways. These functions, together with their interconnections, are held in the knowledge base. Thus for the chosen principle to be embodied, the functions are drawn on the schematic building site in the context of HE.?1.

For the refrigeration application, the choice of the best working principle is not clear and therefore requires recourse to background information on the application of the different principles. This is provided through on-line hypertext links to appropriate handbooks or application notes and functional performance maps held in MetaCard, a hypermedia development environment in which the Schemebuilder graphical user interface is created.

Having chosen one or more probable applicable working principles it is necessary to embody each of the required functions. This is performed by searching the knowledge base for available means to achieve the function. If nothing suitable is known, the required function may be further decomposed by either choosing another working principle, or applying a set of bond graph based principles of decomposition (see section 3.5), and searching for available means of performing the generated sub-functions.

In Figure 5, the choice of the vapour recompression principle leads to the requirement for a vapour compression function, a power transformation between the rotational and compressible fluid domains, TF.ROT.CFLU. Suppose the designer chooses fulfil this by applying the screw compressor working principle. This in turn instantiates a number of required functions, one of which is to synchronise the rotation of the screw elements - another transformation, TF.ROT.ROT. There are several means of achieving this using known components from the knowledge base, such as Helical Gear, Spur Gear, or Toothed Belt, or

alternatively another Working Principle might be chosen, such as that of Electronically Phase Locked Rotation. In this simple example, there are 3 or 4 levels of working principles, as shown in Figure 5. For a complex product there may be 7, 8 or more levels of aggregation.

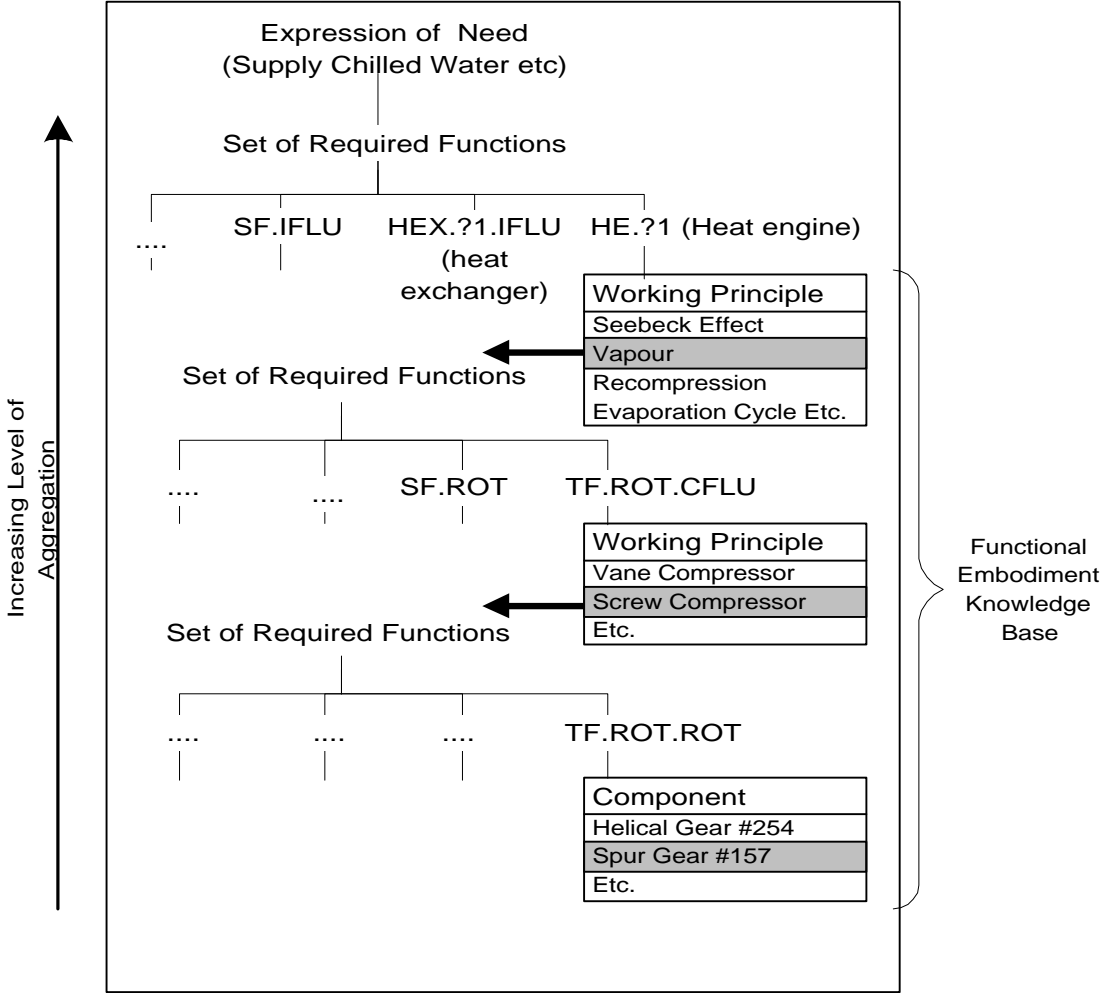


Figure 5. - Working principles in qualitative development of a scheme.

Spatial aspects of the functions, links to the environment and the need to provide physical structure may occur at any level in the hierarchy. Thus spatial principles of structures and kinematics may be invoked at any point. The starting part of the conceptual design decomposition is not restricted to any particular level of aggregation.

3.5. Bond Graph-based Principles of Functional Decomposition

Bond graphs - Karnopp et al. (1990) provide a unique set of principles for the embodiment of energetic systems. These principles are based on the facts that only compatible energy ports may be connected to each other, that the causality and the dependency of the energy covariables must be correctly propagated and that there are a limited set of primary functions that may be used. These include the transformer (TF), the gyrator (GY), the resistor (R), the compliant (C) and inertial (I) energy stores, the common effort (0) and common flow (1) junctions and sources of flow (SF) and effort (SE). Each has a clear set of principles governing its function and how it may be decomposed whilst still performing the same overall primary function - Sharpe (1978).

The following is a partial list of these decomposition principles:

- (i) Insert Transformer to provide intermediate energy domain
- (ii) Replace Source of Effort with a Compliant Store
- (iii) Replace Source of Flow with an Inertial Store
- (iv) Replace Source of Effort with Source of Flow + Conflict Resolver (Resistor or Gyrator)
- (v) Replace Source of Flow with Source of Effort + Conflict Resolver
- (vi) Replace Inertial Store with Compliant Store + Conflict Resolver
- (vii) Replace Compliant Store with Inertial Store + Conflict Resolver

The ability to split a function and introduce an intermediate energy domain allows the coupling of domains that are not normally possible. For example, it is not possible to transfer directly energy between electrical domain and the incompressible fluid domain, which will be illustrated in the subsequent metabolite infuser example where it is necessary to use either a translational or rotational stage as an intermediary.

3.6. The FEST-ER

The FEST-ER differs from the well-known function-means tree in that it supports two forms of multiple inheritance hierarchical links that are very important for the efficient development and representation of competitive designs.

The first of these is the embodied function reference, which is used when a function, which has already been embodied at some point in the FEST-ER, occurs again in a location which is non-mergeable with the original occurrence. Instead of embodying the function all over again, an embodiment function reference can be made, reusing the original work. There are many examples of this to be seen in Figure 8, where reference functions are surrounded by heavy boxes and references to them by light ones.

The second form of multiple inheritance link occurs where a single means is, when appropriate, allowed to embody more than one function, which can often be the key to a more competitive design. This type of link might occur where for example a linked microprocessor means and absolute angular grey-scale encoder means both have a required subfunction SE.ELEC, an electrical effort source. A single alkaline battery means is in this case able to embody both of these functions in parallel.

This technique is also the key to representing ASIC hardware code or embedded microprocessor software means, for implementing data processing functions. A number of different data functions from widespread parts of the FEST-ER, may be mapped onto either a single microprocessor means or a single ASIC means. Once this mapping has been made, the FEST-ER contains a complete Yourdon-style structured analysis of the information processing required of the microprocessor or ASIC.

If the individual data functions used, are stored in the functional embodiment knowledge-base with their representation as high-level Petri net fragments - Jensen and

Rozenberg (1991), then a complete net describing the desired functionality can be automatically assembled. The resulting net, if exported to the commercial software tool SystemSpecs, will then allow automatic generation of simulation or implementation code, in C++, Occam or VHDL - IvyTeam (1993).

The FEST-ER information structure will equally support alternative embodiments of the same functionality in say analogue electronics or even a special mechanical linkage, thus allowing trade-offs of moving solutions either way across the “mechatronic interface” to be easily examined.

The FEST-ER is implemented through the use of a Truth Maintenance System (TMS) in conjunction with an object-oriented information representation - Filman (1988). The first prototype FEST-ER system was implemented in KEE using the KEEworlds facility. It is now being rewritten along with the rest of Schemebuilder using the CLIPS package produced by NASA - Giarratano and Riley (1994). The CLIPS TMS is less sophisticated but more flexible than that of KEE, which has rather limited facilities for customization and would have made the more advanced multiple-inheritance features of the FEST-ER difficult if not impossible to implement.

4. APPLICATION OF FUNCTION-MEANS DEVELOPMENT

4.1. The FEST-ER for the Metabolite Infuser Example

Attention will be focused on the development of a FEST-ER for a portable metabolite infuser, which only involves one level of aggregation. The “expression of need” could be defined as follows:

“The need is for a light weight portable infuser capable of providing a continuous constant flow of drug into a patient's artery for up to 48 hours. It must provide an indication of correct functioning. The flow rate is to be 2.5 ml per day.”

This example may be decomposed into its alternative constituent components from a single functional requirement, a constant source of flow of an incompressible fluid, shown diagrammatically as SF.IFLU (Figure 6). Figure 7 shows the next stages of development of the FEST-ER, using the set of bond graph decomposition principles for energetic systems. Searching the function embodiment knowledge-base returns no known means. The insertion of a transformer [TF] to give an alternative intermediate energy domain shows that, for the intermediate rotation, there are several alternative transformers from rotary motion to incompressible fluid flow TF.IFLU.ROT but that there are no matching sources of rotation. Replacing the source of flow by an inertial energy store yields no known means. Replacing the source of flow by a source of effort or compliant store and a conflict resolver, which may be a resistor or a gyrator, yields a whole array of possible alternative energy domains and means. The alternative energy domains are electrical, rotation and translation, two stages of rotation, or two stages of translation.

The complete FEST-ER for the metabolite infuser is shown in Figure 8. This shows the total development of the alternative designs including the points at which the designer has chosen not to proceed. Any of these branches may be revisited at any time if required.

Each transformation or gyration between the domains has several different means, which may give rise to some 20 alternative schemes that have the potential to perform the desired primary function. Each represents a working principle and as such might be added to the dictionary of working principles. Each scheme represents the minimum functional structure and after this purely qualitative generation procedure, every component will be of the default size for its component class.

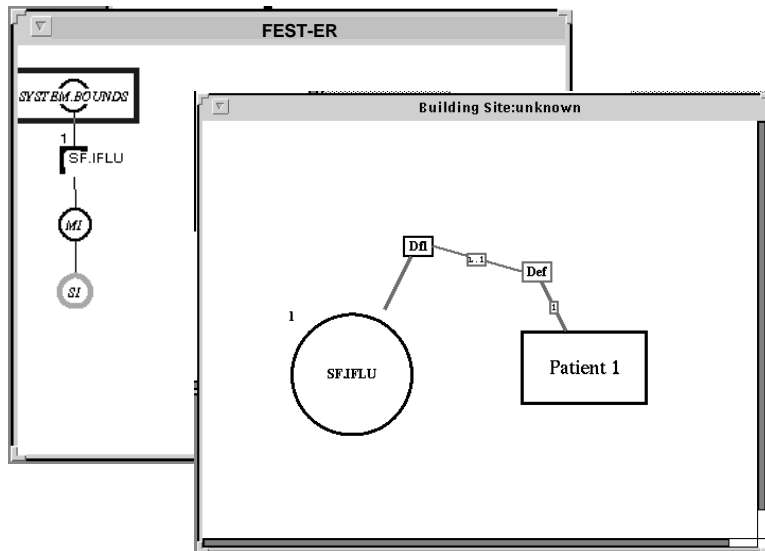


Figure 6. Initial development of the FEST-ER.

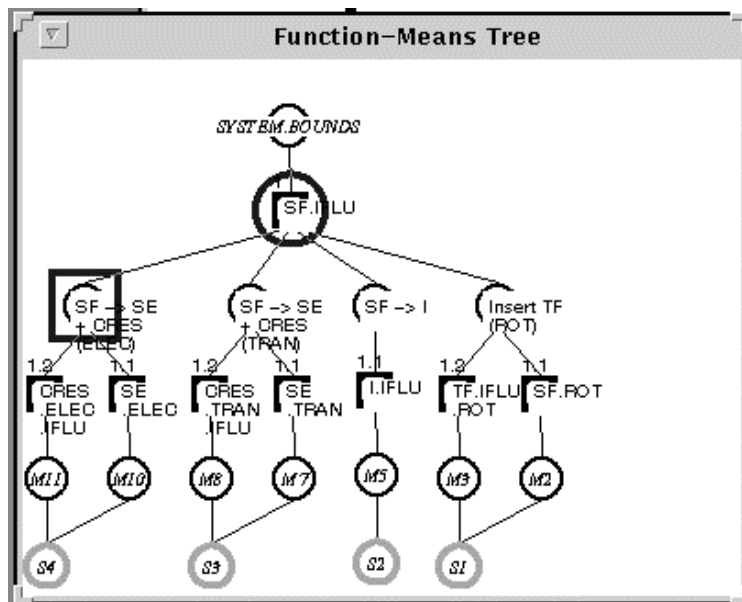


Figure 7. Further development of the FEST-ER.

4.3. Iterative Relationships of Function Means Development

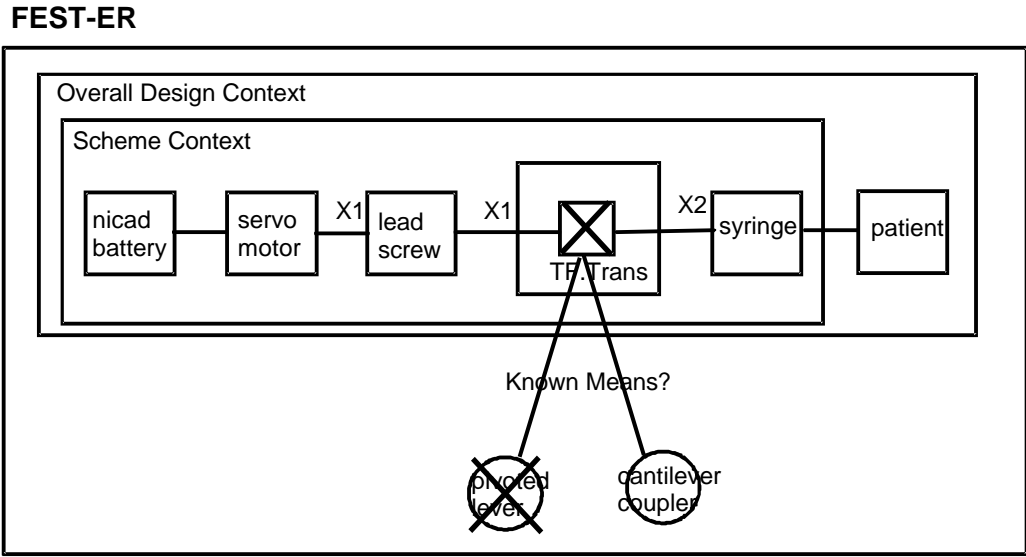


Figure 9. Function means interaction with functional axis.

The schemes generated by the propagation of information through the directed acyclic graph structure of the FESTA-ER, consist of assemblies of components each modelled in Dymola, including their spatial aspects and 3d dynamic behaviour. In the initial layout each fixed joint will reside in its home position. The layout may be viewed by the designer in fully lit, solid rendered 3d image windows, implemented by interfacing Criterion Software’s C language based “Renderware” library to the CLIPS object system. The remarkable efficiency of this library allows real time viewpoint changing, object manipulation and animation of models, even on low-end PC hardware. Thus, problems with the initial layout such as spatial interference of components or incorrect positioning or alignment of inputs or outputs are readily apparent. The designer may attempt to correct these problems by direct mouse manipulation of available degrees of freedom in the fixed joints of the scheme. However, often it will be necessary to introduce additional functions, principally spatial transformers, to avoid a spatial clash or give the correct alignment of functional axes. This is illustrated in Figure 9, where the requirement

is to transform from axis X1 to the parallel axis X2. The primary functional components and the initial context form the context for the function means development of the transformer.

This leads to the choice of a cantilever slide or a pivoted lever. With the choice of the cantilever, the new modified scheme may be laid out in the schematic building site and visualised in the 3d solid display window (Figure 10).

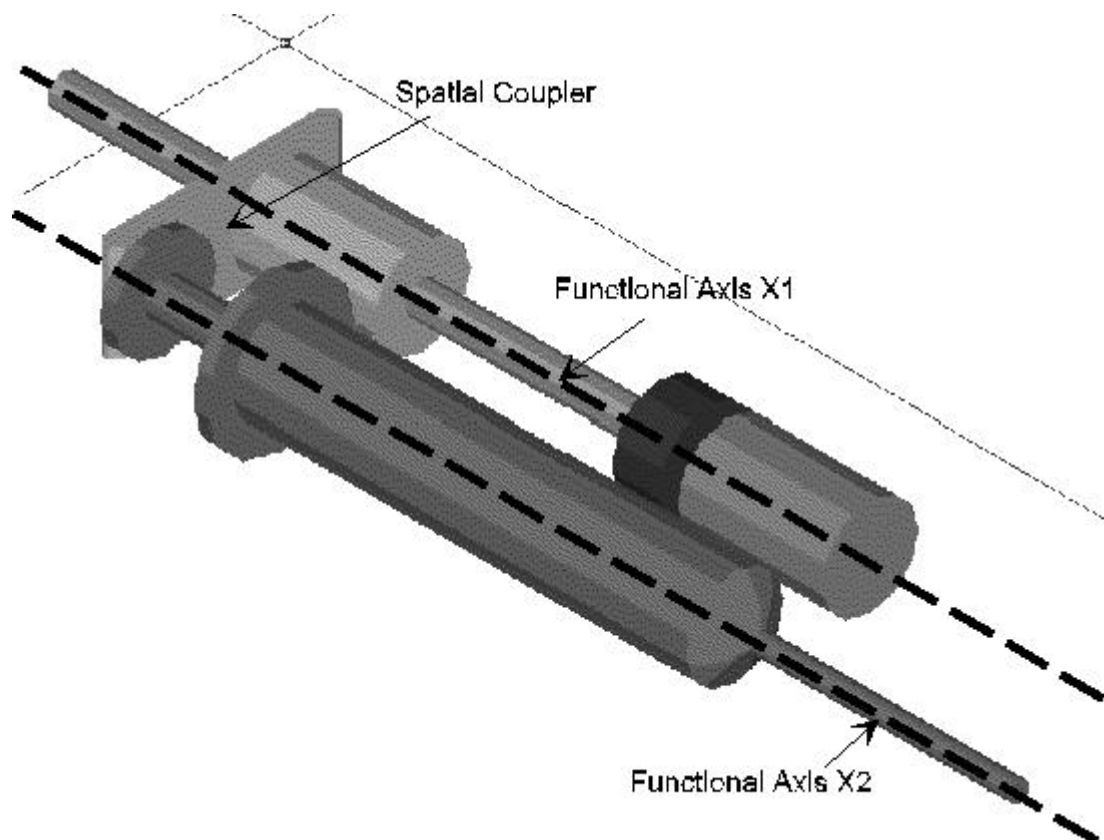


Figure 10. - 3D Solid Rendered Display showing Functional Axes and Spatial Coupler

During the component sizing process, propagation of the infuser's required fluid pressure/flow rate working point through the Dymola models of the syringe and leadscrew, leads to a torque/speed requirement of the motor which it is unable to supply. The required torque is too high and the low speed is in a highly inefficient area of its performance envelope. However, the motor is well able to supply the power required, so Schemebuilder will suggest that a rotational transformer be inserted to move the working point along a constant power line

into an area where the torque is manageable and the motor efficiency high. This is shown in Figure 10 with common axes (X1) for input and output. Reference to the functional embodiment knowledge-base reveals that the epicyclical and the multistage gearboxes are suitable means. The gear pair is not suitable because the input and output axes are offset. Thus there is a continued development of the FEST-ER as the detail matching and layout of the chosen schemes proceed.

The Dymola models generated for each scheme provide the basis for both the component matching described above and dynamic simulation, which together with the defined functional axes, allow the forces and torques within the functioning scheme to be superimposed on the 3d solid rendered, animated display of the design.

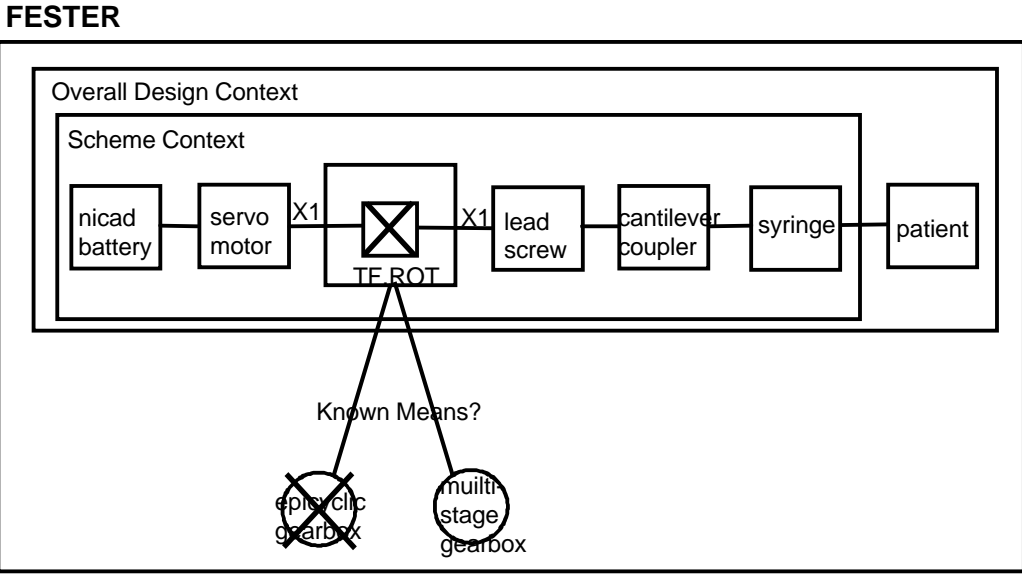


Figure 11. Function means interaction with matching component.

5. CONCLUSIONS

Design is an integrated and integrating process during which the designer must call upon vast amounts of knowledge and experience in a systematic manner to develop the design of a product from first principles to a satisfactory prototype. It is a combination of rigorous

intellectual discipline and creative thinking. In the development of the Schemebuilder environment, we have been very conscious of two important factors: (1) the need to allow the designer or designers if that be the case, to have total control over the design process at all times, allowing them to compare the alternatives in a clear and unambiguous way, and (2) to provide the underlying core structure of data and knowledge in such a way that the designer is not hindered in his actions and may freely develop new ideas and procedures whilst at the same time being equally free to revisit previous decisions whether in respect to the FEST-ER development or aspects of matching or simulation. The provision of a sophisticated audit trail of the designers action and the advice proffered or sought is important in this respect.

A further important aspect of the integrated set of AI tools is the manner in which the design process may be entered from a number of different points. This paper has largely followed French's model of conceptual design in describing how the different AI tools support the designer in the process. Practising designers often approach a problem not from first principles but using their experience or an existing design. This may be an effective starting point especially if supported by good simulation and the ability to build and test rapid prototypes as in Schemebuilder. Recent developments have demonstrated how the functional analysis may be used to determine the essential functionality of an existing design which may be used for the subsequent synthesis of alternative designs.

The greatest importance of the work that has been described must lie in the manner in which it has been integrated to allow the complete design process to be undertaken in one homogenous computer environment. The example given in this paper of the design from 'first principles' of a portable metabolite infuser for use with human subjects has demonstrated how the process may be used to facilitate the creation of a number of workable alternative designs represented within the computer as "virtual prototypes" for realistic comparative evaluation.

ACKNOWLEDGEMENTS

This paper represents a considerable effort over a period of years by a number of individuals.

The authors would like to thank the Staff of the EDC especially Marcus Duffy for his help and enthusiasm in preparing this paper and the Engineering Design Committee of the Engineering and Physical Sciences Research Council for their financial support of the Design Centre.

REFERENCES

- Andreasen, M. M. (1980). *Syntesemetoder på Systemgrundlag*. PhD Thesis, Lund Technical University, Lund, Sweden.
- Bracewell, R.H., Bradley, D.A., Chaplin, R.V., Langdon, P.M. and Sharpe, J.E.E. (1993). Schemebuilder: A design aid for the conceptual stages of product design, *Proc. 9th Int'l Conf on Engineering Design ICED'93, The Hague*, pp. 1311-1318. Heurista, Zurich.
- Bracewell, R.H., Chaplin, R.V., Langdon, P.M., Li, M., Oh, V.K., Sharpe, J.E.E. and Yan, X.T. (1995). Integrated Platform for AI Support of Complex Design - (Part II): Supporting the Embodiment Process. *Proc. First IFIP WG 5.2 Workshop (Knowledge Intensive CAD-1), Espoo, Finland*, Vol. 1, pp. 283-299. International Federation for Information Processing (IFIP).
- Buur, J. (1990). *A Theoretical Approach to Mechatronics Design.*, PhD Thesis, Institute for Engineering Design, Technical University of Denmark, Lyngby.
- Cellier, F. E. (1991). *Continuous System Modelling.*, Springer-Verlag, New York.
- Elmqvist, H. and Brük, D. (1995), Composite Constructs for Object-Oriented Modeling, *Proc. Eurosim Simulation Congress, Vienna, Austria*.
- Filman, R. E. (1988). Reasoning with Worlds and Truth Maintenance in a Knowledge-Based Programming Environment. *Communications of the ACM* 31(4), 382-401.
- Finger, S. and J. R. Rinderle (1989). A Transformational Approach to Mechanical Design Using a Bond Graph Grammar. In *Design Theory and Methodology - DTM '89*, (Elmaraghy, W.H.et al., Eds.), Vol. 17, pp. 107-115.
- Fishwick, P. A. (1989). Qualitative Methodology in Simulation Model Engineering., *Simulation* 52, 95-101.
- French, M. J. (1985). *Conceptual Design for Engineers*. Design Council, London.

Giarratano, J. and Riley, G. (1994). *Expert Systems: Principles and Programming*. PWS Publishing, Boston, MA.

Hansen, C.T. (1995). An Approach to Simultaneous Synthesis and Optimization of Composite Mechanical Systems. *Journal of Engineering Design* 6(3), 249-66.

IvyTeam (1993). *SystemSpecs 2.1 Reference Manual*. Zug, Switzerland,

Jensen, K. and Rozenberg, G. (1991). *High-Level Petri Nets, Theory and Application*. Springer-Verlag, ISBN 3-540-54125-X.

Karnopp, D. C., Margolis, D. L. and Rosenberg, R C (1990). *System Dynamics: A Unified Approach*., 2nd ed. Wiley, Chichester.

de Kleer, J. (1986). An Assumption-based TMS *Artificial Intelligence* 28, 127-62.

Mirbel, I. (1995). A Fuzzy Thesaurus for Semantic Integration of Design Schemes. *Proc. Lancaster International Workshop on Engineering Design, Ambleside*, pp. 319-335, Springer-Verlag, London.

Miller, G.A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM* 38(11), 39-41.

Oh, V. (1993). *Intelligent Design - Assistant Systems for Engineering Design*. Technical Report EDC2, Lancaster University Engineering Design Centre.

Otter, M., Elmqvist, H. and Cellier, F.E. (1996). Modeling of Multibody Systems with the Object-Oriented Modeling Language Dymola. *Journal of Nonlinear Dynamics* 9(1), 91-112.

Paynter, H. M. (1961). *Analysis and Design of Engineering Systems*. MIT Press, Cambridge, USA.

Rychener, M. D. (1988). *Expert Systems for Engineering Design*. Academic Press, Boston.

Sharpe, J. E. E. (1978). Bond Graph Synthesis of Robots & Telechairs. *Proc. 3rd CISM IFFTOMM Conference in Robotics & Manipulators, Udine, Italy*, 168-176.

Sharpe, J. E. E. and Bracewell, R. H. (1993). Application of Bond Graph Methodology to Concurrent Conceptual Design of Interdisciplinary Systems. *Proc. IEEE/SMC Conference: Systems, Man and Cybernetics '93, Le Touquet, France*, 7-13.

Soderman, U. and J. Stromberg (1991). Combining Qualitative and Quantitative Knowledge to Generate Models of Physical Systems. *Proc. 12th International Conference on Artificial Intelligence, Sydney*.

Top, J. and H. Akkermans (1991). Computational and physical causality. *Proc 12th Int'l Joint Conf on Artificial Intelligence, Sydney*, 1158-63.

Ulrich, K. T. and W. P. Seering (1989). Synthesis of Schematic Descriptions in Mechanical Design. *Research in Engineering Design* 1(1) 3-18.

Yokoi, T. (1995). The EDR Electronic Dictionary. *Communications of the ACM* 8(11), 42-44.

Rob Bracewell is Senior Researcher and Systems Manager of Lancaster EDC. An Engineering graduate of Cambridge University, England, he has since worked for 12 years at Lancaster, researching in the areas of Engineering Design, Mechatronics and the application of AI technology to both product and design process. His Ph.D. concerns the design of Ocean Wave Energy Converters. He has had prime responsibility for the design and implementation of the Schemebuilder software since the formation of the EDC in 1990.

John Sharpe has a PhD from Cambridge University on Optimal System Design and after a period as Senior Assistant in research at Cambridge, Engineer with the National Institute for Medical Research and 16 years with the University of London, directed the work of Lancaster EDC from 1992 to 1995. He has special interests in graph theory, particularly Bond Graphs, and first published a paper on Bond Graph Synthesis in 1978. He is a Fellow of the Institute of Mechanical Engineers and a Member of the Institute of Electrical Engineers. Currently, he has a serious hobby in the development of waste-to-energy schemes.