INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19–21, 2003

INDUCING CHANGE PROPAGATION MODELS USING PREVIOUS DESIGNS

Peter Matthews and David Lowe

Abstract

A large amount of design work can be classified as redesign. Effectively, this involves modifying a previous design by changing certain constraints and then redetermining the design parameter values. This paper describes a method for inducing a stochastic model of the design space. This model provides a means for guiding a designer through the parametric design modifications by searching for the most likely design outcome, based on Bayesian statistics. A graphical user interface is used to interact with the model. The guidance provides a designer with further constraints on the design space, thereby iteratively narrowing the search down to a small region of the design space. This is illustrated by two case studies: a flat screen display design problem and the design of a gas turbine combustor. The paper concludes with a review of the case studies and a summary of future work being undertaken.

Keywords: Probabilistic Design; Machine Learning; Information Analysis; Product Families; User Evaluation.

1 Introduction

Most design work can be classified as *redesign*. Redesign is where a product has already been fully developed, and there is a need for changing the design of this product by a small amount. For example, there are a large number of flat screen displays on the market. These can all be considered as a single product family, in that they all perform the same function, and the variants are a result of redesign work arising from slightly different specifications. However, it is not necessarily true that this product domain is well understood and therefore accurately modeled. Indeed, it is possible that there exist external influences that are impossible to model that can affect the design. This provides a challenge where inexperienced designers are tasked with redesign exercises: how are these designers to be guided through the design process without recourse to explicit models or experts?

The aim of this research is to investigate a stochastic approach for guidance through the design process. The focus is on determining the approximate values of design parameters between the end of the design concept stage and the early embodiment stage in the design process. This is achieved by allowing a designer to place an initial set of constraints on a design. The tool then supplies a series of further constraints to be placed on the design to focus the search of the design space. These additional constraints are ordered according to likelihood of generating a successful design. As these constraints are added, the effects are propagated through the rest of the stochastic model.

A key aspect of this approach is that it requires minimal expert intervention. Domain experts determine the set of design variables to be considered during this stage of the design process, but not to determine any relationships between the variables. By using previous examples, a stochastic design space model will be induced. This paper describes the mathematical basis and the implementation of this approach. Two case studies are provided to illustrate the implementation. Finally, conclusions are drawn from the implementation in its current state.

2 Background

There is an extensive body of work regarding the creation of design models. This paper concentrates on a stochastic approach of modeling the design space. Similar stochastic and stochastic-like approaches have been previously researched. One successful approach is based on Design Structure Matrices (DSM) which provides a means for aiding designers during this conceptual stage [1]. This requires the list of tasks to be completed and a precedence order for these tasks during the design process, which is supplied by domain experts. The DSM approach has proved to be a useful tool, and by adding probabilistic information has been expanded into an interactive tool (Signposting [2]) and a means to estimate the total time required by design process [3]. These provide assistance in determining the path a designer should take through the design process, but offer little help in either determining the values design parameters should take or provide early estimates of how well the design will meet its criteria.

However, it is necessary to represent the design space in a manner that is compatible with probability theory. In addition, the representation of the design space must also be compatible with the design process, so that such a tool can be easily implemented as part of that process. Such issues have been discussed in previous ICED conferences, e.g. [4, 5]. The previous representation work aimed to capture the design form at the earliest stage possible in the design process. This is a non-trivial task during the conceptual design stage, as good design practice demands that at this stage a wide variety of solutions must be considered [6].

Once a concept has been selected, the design embodiment begins and a more formal and fixed representation can be described. Typically, one such concept is used to form the basis of a product family. A product family is where a number of slightly modified products perform similar functions. Frequently, this occurs when a product is redesigned after it has been originally marketed, for example there are a large number of variants of flat screen displays on the market. These displays come in a large number of shapes and sizes, serving different purposes. Some are marketed for personal use, and these will be used in products ranging from personal digital assistants through to wide screen televisions and will experience at most mild conditions. Other will be used in public spaces, for example train timetable displays, and will need to be designed for greater reliability. This family of designs can be parametrically described most simply using the overall geometry of the product (width, height, depth), and the material used. This representation can then be augmented with various characteristics of the design, for example: weight, cost, and expected life. Some of these physical characteristics can be computed from the design parameters, however, it is also possible to include more subjective characteristics, e.g. some measure of desirability which is dependent on external factors for example how the product compares to its competitors. These subjective characteristics might only be able to be determined after the product has been marketed, and therefore these will not be a mathematical function of the design parameters.

Once a representation has been determined for a product family, mathematical relationships be-

tween the design parameters and characteristics can be determined. These most frequently are deterministic in nature, but there has also been work on achieving this stochastically, e.g. [7]. A major shortcoming of this approach has always been that these relationships can only be expressed by domain experts. This requires a large investment of time and effort by the domain experts, and there is the risk that it will be incomplete or inaccurate. For this reason, the aim of this work is to *induce* a model from previous members of the given product family. The approach adopted in this paper is a stochastic model of the design space. A designer will interact with this model by constraining the design space incrementally. Each time an additional constraint is placed on the design, a series of suggested further constraints will be supplied that guide the designer to the most likely subsequent design constraint, or move, that should be further placed on the design to ensure success. The designer is not required to follow any of these suggestions, as these are only guides to how most designs have been done in the past, but it they can be considered as the 'voice of experience'. This process is continued until the design has been fully constrained. This fully constrained design space represents the *most likely* region in which the desired design exists.

3 Mathematics

This work induces a probabilistic model of the design space based on a series of observations. This section provides a review of the mathematics required to create and verify such a model. First, the probabilistic notation is described and how it is applied to the design space. Next, the basic probability theory used is summarized.

3.1 Notation

The design space, which includes both design parameters and design characteristics, is described as a real-valued vector. For an n-dimensional space, this can be written down as: $\mathbf{X} = (X_1, X_2, \ldots, X_n)$, and the X_i's are referred to as *design variables*. The probability of a design with variable X_i taking values between a and b is expressed as $\mathrm{P}(a < X_i < b)$. Note that it is only possible to talk about design variables taking a range of values, as the probability of a design variable taking on an exact value is theoretically 0. In practice, the design space is transformed into a discrete set of intervals, and therefore the probability of X_i being in the jth interval is written as $\mathrm{P}(X_i = j)$. This is quite an intuitive manner, as during the early stages of design, designers are more interested in making estimates of design variables rather than their precise values. For example, when designing a flat screen display, a designer would be aiming to answer questions of the nature: 'should the display depth be: very thin, thin, medium thick, or thick'. These categories in turn are represented by value intervals, and the exact value will be determined later on in the design process. This approach also caters for categorical data, e.g. material selection, as there is no longer a requirement for the variable to be continuous.

Once the intervals and/or categories have been determined, computing the probability values is a trivial counting exercise. The probability of design variable X_i being in interval j is given by:

$$\mathrm{P}(X_i = j) = \frac{\text{number of examples with } X_i \text{ in range } j}{\text{total number of samples}} \tag{1}$$

This can be extended to the concept of *joint* probability, which is the probability of two events occurring together. The probability of $X_{i_1} = j_1$ and $X_{i_2} = j_2$ is written as: $\mathrm{P}(X_{i_1} = j_1,\ X_{i_2} = j_2)$[1]. An example of this would be asking the question 'what is the probability of the design being thin and having low cost?'.

Finally, there is the definition of conditional probability. Conditional probability provides a measure of the likelihood of an event occurring, given that some other event has already occurred. The conditional probability of event A occurring, given that event B has already occurred is given by:

$$\mathrm{P}(A|B) = \frac{\mathrm{P}(A, B)}{\mathrm{P}(B)} \tag{2}$$

For example, this could be used to ask the question: 'given that the design is going to be very thin, what is the likelihood of using PTFE as the material?'. The notation for this statement would be: $\mathrm{P}(\text{material} = \text{PTFE} \mid \text{depth} = \text{thin})$. If the value of this expression is high, then PTFE is commonly used for thin designs, and is likely to be a safe decision to use PTFE for this design. If the value is low then PTFE is not frequently used, it is likely to be a poor choice. It should be noted that this only provides a suggestion based on past designs. There is no reason, other than this experience, that a designer must follow this suggestion. Therefore the designer is at liberty to go against the suggestion.

3.2 Probability Theory

Probability theory is applied to guide the construction of the constraints on the design space. The aim is to incrementally constrain areas of the design space that are least likely to provide designs that meet the current set of constraints. This is achieved through the use of Bayesian learning [8]. Bayesian learning methods provide a simple and practical approach to hypothesis testing. In this case, the hypotheses that are to be tested are the incremental constraints to be placed on the design space.

The problem can be specified more formally as searching for the best hypothesis from a set of possible hypotheses, H. The search for this hypothesis will be based on the experience database given by D. Hence, the search can be expressed as the search for the hypothesis $h \in H$ that maximizes the expression $\mathrm{P}(h|D)$. This probability reflects the confidence of the hypothesis h given the experience database D, and so effectively the search is for the hypothesis that gives greatest confidence.

The problem with the expression $\mathrm{P}(h|D)$ is that it is not clear how to evaluate either the joint probability $\mathrm{P}(h, D)$ or the probability of the database occurring $\mathrm{P}(D)$ (see Equation 2). Bayes theorem offers assistance in computing this:

$$\mathrm{P}(h|D) = \frac{\mathrm{P}(D|h)\mathrm{P}(h)}{\mathrm{P}(D)} \tag{3}$$

This changes the nature of the computation. The term $\mathrm{P}(h)$ represents the prior probability of the hypothesis h and can reflect any prior domain knowledge. Typically, where no prior domain knowledge exists, this probability is set uniformly across all potential hypotheses. The term

[1] an alternative notation for this is: $\mathrm{P}(X_{i_1} = j_1 \cap X_{i_2} = j_2)$

$\mathrm{P}(D|h)$ represents the probability of observing the experience database given the hypothesis h. This can be computed, using Equation 2, by counting the number of examples in the experience database D that agree with the hypothesis h (i.e. the joint probability of D and h, $\mathrm{P}(D, h)$). The term $\mathrm{P}(D)$ is not relevant as this is a common denominator for all hypotheses to be tested. The maximally probable hypothesis, called the *maximum a posteriori* (MAP) hypothesis can then be computed as follows:

$$h_{\mathrm{ML}} = \underset{h \in H}{\operatorname{argmax}} \mathrm{P}(D|h) \tag{4}$$

where argmax returns the argument, h, that maximizes the probabilistic statement.

This is used as the basis for searching for design constraints to guide a designer to a specific area of the original design space. The full set of possible constraints is very large, and therefore constraints are added iteratively. At each iteration, h takes the form of constraining a single design variable to one particular value.

4 Computer Implementation

There are two main elements to the implementation of the theory. The first represents the inference 'engine', which is the implementation of the mathematics into computer executable code. The second is the interfacing between the engine and the user. Finally, it is also important to understand and be able to interpret the results supplied by the tool. These three aspects are discussed individually, followed by a note on the computational costs of the tool.

4.1 Inference engine

The implementation of the theory, and its application to the design domain, is achieved in a modular manner. The modules implement the data discretization; the constructor; the joint probability computer; and the MAP heuristic search. The data is assumed to be complete (i.e. no missing values or measurements), and readily available.

Data discretization Each design variable is discretized independently of the other variables. The user supplies the number of discrete categories that are to be generated, denoted k. The discretization algorithm aims for there to be roughly an equal number of elements in each category, as opposed to spreading the category boundaries at equal intervals. In addition to identifying the discretization boundaries, this module also returns the design data with the original values replaced by the categorical label. In the event that a design variable has fewer than k categories, these are identified and limited to the number of categories truly used. An example where one of the design variables is boolean (true/false), containing only two categories,

Constraint construction Constraints are constructed incrementally. The constraint construction module initially provides an 'empty' constraint structure that can be added to later. The constraint structure is an array of lists. Each design variable has a list of design categories that are permitted to be used. As constraints are added to the design space, no longer permitted categories are removed from the design variable list being constrained. It is also possible to

remove (slacken) constraints should the designer wish to re-open areas of the design space that were previously constrained.

Joint probability The joint probability is based on the experience database D, given by the discretized original dataset. The joint probability is computed by simply counting the number of examples from the database that pass all constraints divided by the total size of the database.

MAP heuristic The MAP heuristic searches for the next constraint that should be placed on the design space, as determined by Equation 4. The heuristic space that is searched is given by constraining each design variable to each possible setting, resulting in testing a maximum number of nk heuristics. Each candidate heuristic, which is a single constraint, is intersected with the current design constraints and the joint probability is computed. These are then sorted in descending order of likelihood and returned as the ordered list of suggested next design moves.

4.2 Graphical user interface

It is through the graphical user interface (GUI) that a designer both provides and receives information from the inference engine. It is important for this interface to be as intuitive and simple as possible for this to be quickly used by designers with minimal training.

The GUI is organized using two windows: the design state window and the move suggestion window (see Figure 1). The design state window consists of two lists: the left-hand list is used to select which design variable is to be constrained and the right hand list selects which ranges are permitted. Those that are not selected represent the design areas violating the constraints. Initially, the user sets the design state to reflect the design specification. It is assumed that this specification has left a number of the design variable unconstrained. Once these constraints have been selected, the user presses the 'Suggest Move' button on the design state window. This then updates the move suggestion window.

The move suggestion window contains the results of the heuristic search, as described above. These are ordered according to likelihood, and therefore, suggestions higher in the list are more likely to result in the design performing as expected than the suggestions lower in the list. The designer can select one of the suggested constraints from this list, and then press the 'Apply' button. This adds the selected constraint to the design state window, and recomputes the set of suggestions. Figure 2 illustrates this iterative process.

4.3 Interpretation

The suggestions can be interpreted as guidance towards designing the new product in the most frequent manner as has been done in the past that meet the current constraints. This is borne out from the Equation 4 which searches for the incremental constraint that satisfies as many of the previous designs as possible. This is good practice in product redesign, as the evidence from previous designs indicates that the new design will be feasible.

Conversely, in the event that there are no further suggestions, this indicates that the constraints represent a design that has never been attempted before. At this point there are two options: first is that the design has been overconstrained. The designer now slackens off the constraint

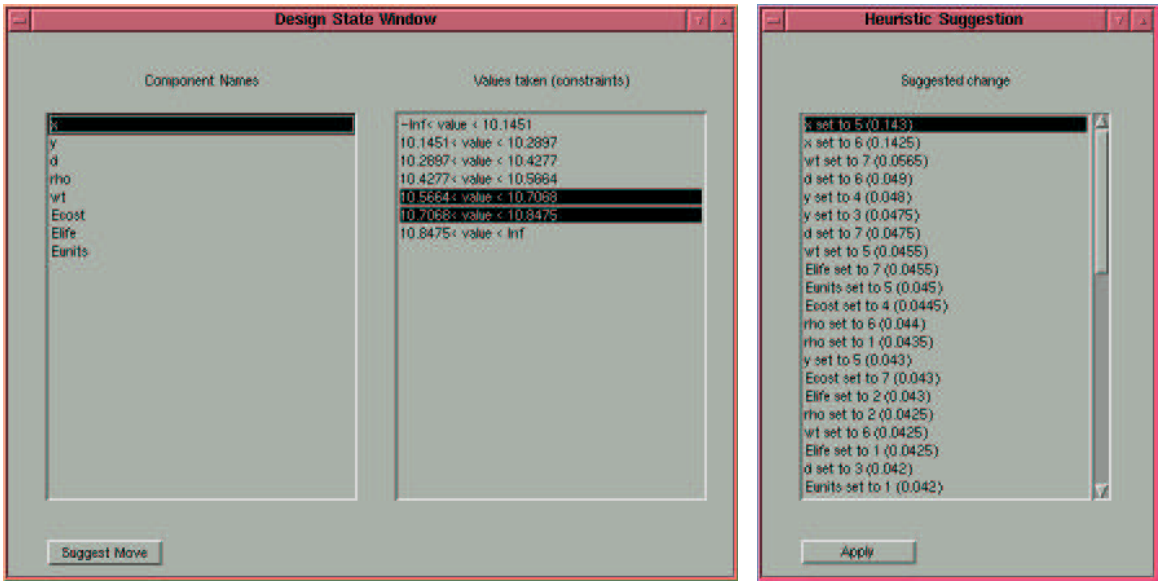

Figure 1. Left: The Design State window, constraining x to the penultimate two categories; Right: The Heuristic Suggestion window, listing suggested further constraints.

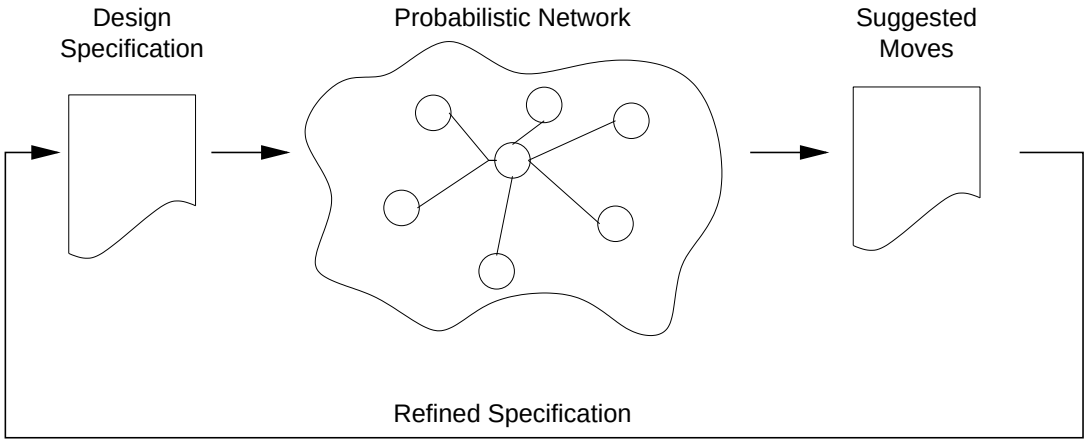

Figure 2. Overview of the iterative scheme: an initial design specification is tightened iteratively using the probabilistic network to suggest ever tighter design specifications.

set, and continues as before. The second is that the designer decides that this is due to the specifications that have been built up during the design refining process have led to a new product area. In this event, the designer must further investigate the new design using traditional methods to verify if this design is feasible. From this point on, the design is going beyond the experience database of this tool, and therefore the tool cannot be used anymore. If the design is successful, it can then be added to the experience database, and thereby extend the useful area the tool can operate in.

4.4 Computational cost

The overall computational cost of this approach is relatively cheap, and depends on the number of design variables (n), the number of categories to be used (k), and the size of the experience database (N). The individual heuristics constrain one design variable to one interval setting, and therefore there are $O(nk)$ heuristics to test. Each time a joint probability is computed, the whole experience database must be checked, which is of cost order $O(N)$. Hence, the total

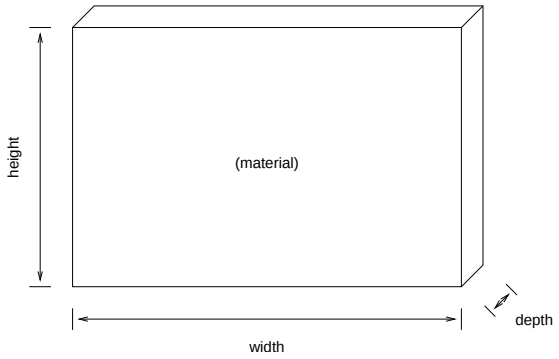

Figure 3. Design parameters of the flat screen display.

cost of computing the suggestion list is $O(nkN)$. Most designers will set $k < 10$. A typical design problem having in the order of ca. $n < 100$ design variables, with an associated design experience database of $N < 1000$. This represents an acceptable upper limit on the cost of computing the suggestion list.

5 Case Studies

Two case studies are presented to illustrate the design change propagation tool. The first case study is the design of a flat screen display. This is a controlled case, where the domain has been hand-crafted and is therefore fully understood. The second case study is the design of a gas turbine combustor. This uses industrial data to induce the design model, and is less well understood.

5.1 Flat screen display design

The flat screen design domain is described using eight design variables. Four of these are design parameters (i.e. parameters that can be directly changed by the designer, see Figure 3) and the other four are design characteristics. The design parameters are: width, height, depth, and material. The evaluation characteristics are: weight, cost, expected life, and expected sales volume. The relationships defining weight and expected life were defined strictly in terms of the design variables. Expected life and expected sales volumes were partly defined by 'external influences'. These two were related to each other and expected life. This was aimed to provide an open aspect to the design system, representing potential design parameters that were not, or could not, be explicitly expressed.

The aim of this case study is to verify that the method provides 'sound advice' as constraints are placed on the design domain. Using the prescribed model of the display domain, a database of 2000 designs was generated. This database was used as the experience database for the system. A number of experiments were performed, which were based on a set of different design specifications. These experiments mainly initially constrained either design parameters or design characteristics. The suggestions were followed until the design was fully constrained. Using the same design parameter constraints, an independent sample was drawn from the model, and the expected characteristics were compared to those from this new sample. It was found that the expected characteristics matched very closely the actual characteristics, demonstrating that for this case the method performs well.

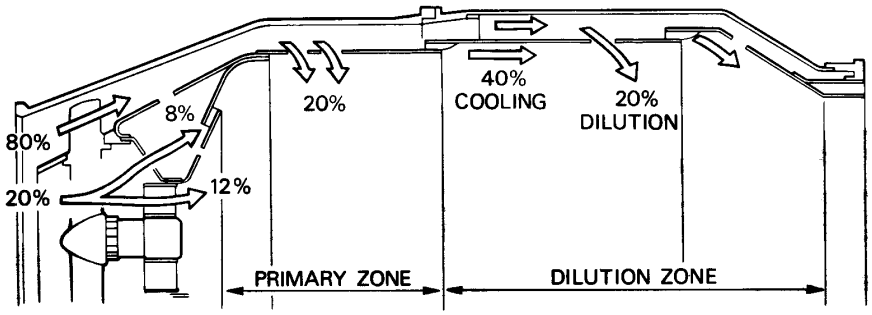

Figure 4. Design schema of a gas turbine combustor (courtesy Rolls-Royce, plc).

5.2 Gas turbine combustor design

The combustor of a gas turbine aero engine is a well defined module of the overall engine. The key function the combustor performs is the heating of the air flow through the engine by burning fuel in this air. The main challenge in achieving this is providing a good air-fuel mix throughout the combustor. The air flow into the combustor performs two tasks (see Figure 4): first is to allow the combustion process to take place and the second is to provide cooling films for the metal surfaces of the combustor which would otherwise melt.

The aim of this case study was to investigate the graphical user interface in an industrial setting. The system was provided with database of combustor mixing elements (e.g. holes, ducts, etc. that are placed on the combustor wall). The designers were given a previous combustor specification to meet as a test case. The overall impression reported back were supportive of the inference engine. It was noted that a preferred interface would guide a designer through making specified changes while keeping the remainder of the design as constant as possible. This will be investigated in future work.

6 Conclusions

Preliminary tests, both in academia and industry, indicate that this approach is a promising method for providing guidance in poorly documented domains. This guidance helps in locating the *region* in the design space that a designer should focus on. Theoretically, this region provides the greatest likelihood of the design being able to satisfy the given constraints. Nonetheless, there are a number of limitations to this approach. The main limitation of this method is that of the experience database. The method can only provide guidance based on the observed previous designs. Therefore, it can be difficult to explore novel regions in the design space as will occur when new technology appears. The second limitation is that this approach does not directly provide a designer with any *understanding* of the design domain. It is this understanding that will help a novice designer become expert in the domain. However, it should be possible to extract this understanding from the stochastic design model by computational analysis of the induced design model.

This paper provided a preliminary perspective of an induced stochastic design model. Discussions with industrial designers highlighted a set of key requirements. Given a specific design, a common requirement is to modify some aspect of the design. There will be a number of alternative means of achieving this modification, however typically each will be at the expense

of further effects on the design. It would therefore be preferable to be aware of all the trade-offs that involve the desired variable to be changed. There are two options to provide support for this issue: the first is to extract the trade-offs involving the desired variable and the second is to compute some estimate of total change to the full design for each possible action.

Ultimately, the aim of this research is to feed into the knowledge management aspects. This will be achieved through the analysis of induced design models. Prior to this analysis, the induction algorithms and models need to be validated. This work provides the first step towards that goal by introducing an environment where models can be induced and subsequently used by industrial designers. By selecting understood design domains, domain experts will be able to verify the induced models, and thereby validate the induction approach. Once the approach has been validated, the models will be analysed to extract domain knowledge. This knowledge then becomes explicit, and can be used to document the design domain.

Acknowledgements

This work is funded by the *University Technology Partnership for Design*, a collaborative research project between the universities of Cambridge, Sheffield and Southampton; and with industrial partners BAE SYSTEMS and Rolls-Royce, plc.

References

[1] D V Steward. The Design Structure System: A method for managing the design of complex systems. *IEEE Transactions on Engineering Management*, 28:71–74, 1981.

[2] J R Hamilton. *The capture and representation of knowledge to support adaptive aerospace design*. PhD thesis, Cambridge University Engineering Department, 1999.

[3] M Carrascosa, S D Eppinger, and D E Whitney. Using the Design Structure Matrix to estimate product development time. In *Proceedings of the DETC'98*, DETC98/DAC-6013, Atlanta, GA, 1998. ASME Design Engineering Technical Conferences.

[4] T N S Murdoch, N R Ball, and P C Matthews. Constraint based templates for design re-use. In A Riitahuhta, editor, *Proceedings of the 11th International Conference on Engineering Design*, volume 3, pages 267–270. Tampere University of Technology, 1997.

[5] P C Matthews, L T M Blessing, and K M Wallace. Conceptual evaluation using neural networks. In U Lindemann, H Birkhofer, H Meerkamm, and S Vanja, editors, *Proceedings of the 12th International Conference on Engineering Design*, volume 3, pages 1777–1780, August 1999.

[6] G Pahl and W Beitz. *Engineering Design: A Systematic Approach*. Springer-Verlag London, second edition, 1996.

[7] D R Wallace, M J Jakiela, and W C Flowers. Design search under probabilistic specification using genetic algorithms. *Computer Aided Design*, 28(5):405–421, June 1996.

[8] T M Mitchell. *Machine Learning*. McGraw-Hill Series in Computer Science. McGraw-Hill, New-York, NY, 1997.

Dr Peter C Matthews, MIEE

Engineering Design Centre, Cambridge University, Trumpington Street, Cambridge CB2 1PZ, England

Phone: +44 (01223) 332709, Fax: +44 (0870) 133 6961

`http://www-edc.eng.cam.ac.uk/` E-mail: `pm131@eng.cam.ac.uk`