INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 01 GLASGOW, AUGUST 21-23, 2001

A METHODOLOGY FOR COMPUTATIONAL DESIGN TOOL RESEARCH

R.H. Bracewell, K. Shea, P.M. Langdon, L.T.M. Blessing, P.J. Clarkson

Keywords: computer-aided design; systematic product development; knowledge-based engineering; empirical study; user evaluation

1 Introduction

A difficulty with research in computational design methods is in evaluating the capabilities and merit of the fundamental research output beyond initial benchmark tests that compare to theoretical examples. While this is a necessary step in method development, issues involving integration of computational methods within the design process are often overlooked. This need for better integration of tools throughout method development, was highlighted as an important issue for the future of design research at the UK DTI/EPSRC 1999 Workshop [1] as well as by Vergeest et al. [2] in the context of novel design tools. Fast feedback cycles from experimental use in education and practice could lead to the development of methods that are both theoretically capable and suited to achieving these capabilities within varying design processes. Thus, as researchers, we need the necessary tool set and support to rapidly prototype research systems without being unnecessarily hindered by implementation details and fast changes in computer technology and standards.

The development of computational design methods and tools requires knowledge of current advances in computer science, software engineering and hardware technology as well as familiarity with the application domain. One strategy for developing computational design methods is to split the tasks where the engineer, or domain expert, conducts the methodological research on paper and then hires a computer scientist to implement a proof-of-concept system. However, with today's programming job market, most research projects can not attract the necessary talent to implement the frequently complex software specifications and domain details. In addition, working out a method on paper can lead to difficulties in laying out diagrams of complex knowledge structures, making and propagating changes to data representations, and navigating the complex information spaces involved. These difficulties can make it very hard to address crucial issues of representation and logic in the early stages of method development. The result can be that a partial and computationally flawed specification is supplied to the programmer, leading to an implementation that most likely will not adequately reflect the method intentions and merits.

There is a common belief that software implementation of advanced design tools merely requires the application of mainstream software engineering techniques and thus is not a subject with which design researchers need be concerned. However, computational design research often entails a software design task that is highly specialised, involving the method complexity associated with numerical techniques combined with the human-computer interface issues of traditional CAD software. Additionally, it may be necessary to select and integrate advanced AI techniques with a range of continuously changing mainstream engineering software modules and packages. A welcome indication that the importance and

distinctiveness of this subject is starting to be recognised, is the introduction of courses teaching the fundamentals of Cae tool development at EPFL [3], CMU, Berkeley and elsewhere. While a thorough grasp of the fundamental principles taught in such courses is vital, constant awareness of the possibilities offered by the latest developments in information technology (IT) is also essential. However this demands a great breadth of experience and more investment of time than can generally be afforded by most academics, programmers and students involved in design research projects. In the related context of developing complex software product families, Meyer and Lehnerd [4] emphasise the critical importance of awareness of new generations of software tools, operating systems, and hardware as they often provide solutions to difficult problems and barriers confronting a software team.

The aim of this paper is to propose a practical methodology for Computer-aided engineering Design (CaeD) tool research, that is intended to enable the development of useable computational design tools early on in research projects. A distinctive feature is the incorporation of CaeDRe, a coherent, modular, up to date and extensible development environment providing a choice of high level languages, tools, reusable software components, integration mechanisms and code-hardening mechanisms. CaeDRe is described in more detail in Bracewell and Shea [5].

2 Defining a Methodology for Computational Design Tool Research

The methodology presented stems from the unification of three complementary approaches developed by the authors over years of practical experience in CaeD research. The first element is the design research methodology developed at the Cambridge EDC by Blessing and Chakrabarti [6], from which a four stage process has been adopted; see Figure 1. In the first stage, *Criteria*, measurable success criteria for the design tool are identified, such as "reduced time to market" together with chains of causal influences linking back to overall business objectives such as "increased profit". The second stage, *Description I*, analyses the existing design process to discover relations between the measurable criteria and the design process thus identifying where application of a design tool could lead to improvements. In the third stage, *Prescription*, insights gained in *Description I* are used to create a storyboard for an improved design process that could result from using the new design tool. For computer-based tools, this storyboard creates a starting point for specifying and implementing a prototype system. Finally in Description II the design tool is tested experimentally to determine whether it works as intended and whether it actually impacts the measurable success criteria. While this is one sequence that can be taken through the methodology, the case study presented in this paper will illustrate that it is not necessary to follow these stages in one specific order.

Software implementation issues were not a primary focus of the previous methodology as it was intended to be applicable to the development of both computer-based and manual design tools. Extending it to provide support on software design and the use of up-to-date computer science techniques as well as empirical methods for method evaluation throughout the research project has resulted in the CaeD tool research methodology shown in Figure 2. Software engineering issues are now addressed by incorporating a second approach, the Cae tool development methodology defined and taught at EPFL [3]. The development process is divided into five activities: (1) task definition, (2) choice of representations, (3) choice of methods, (4) definition of visualisation, interaction and distribution strategies and (5) theoretical and experimental validation. If the choices of knowledge level and implementation level representations are explicitly separated, and a distinction is made

between non-controlled and controlled experimental testing, the process fits neatly into the general methodology shown in Figure 1.

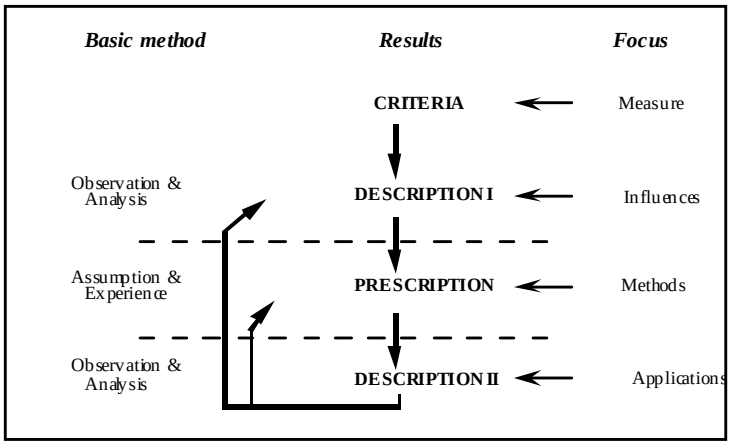

Figure 1. Design Research Methodology (Blessing, Chakrabarti and Wallace, 1998)

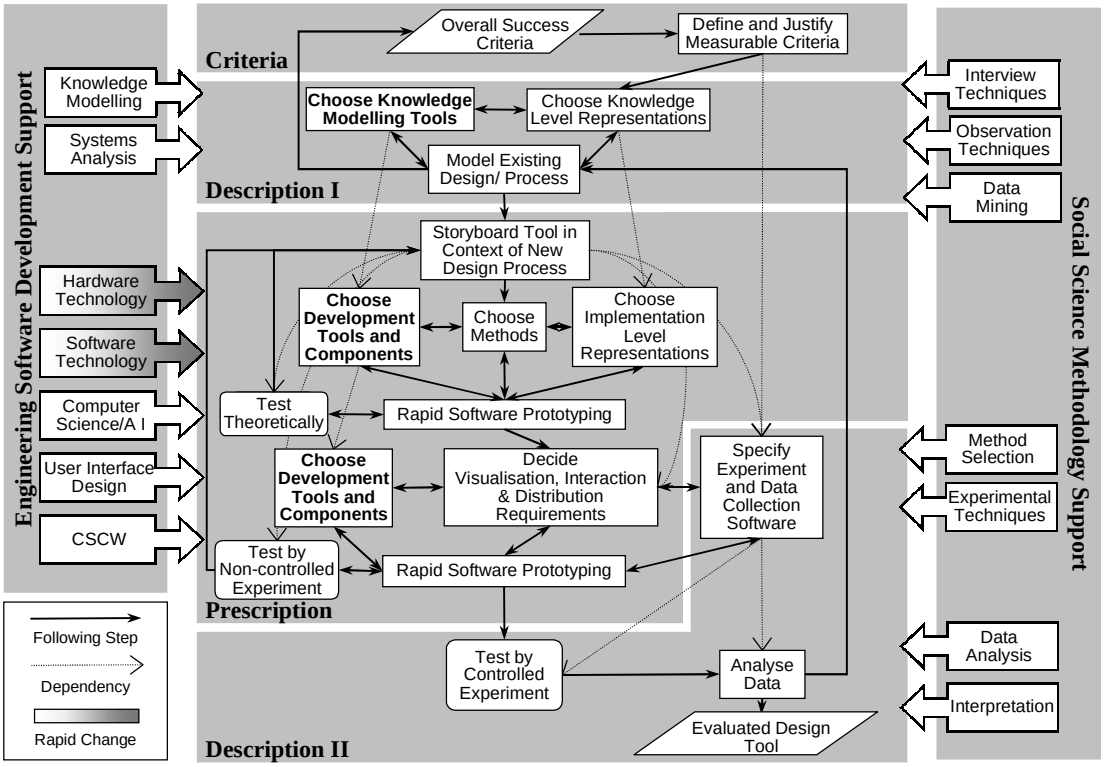

Figure 2. Computer-Aided Engineering Design (CaeD) Research Methodology

There are two main forms of method validation in CaeD design: (1) theoretical validation, i.e. comparisons to known benchmark problems and (2) experimental validation, i.e. testing with user groups. With some CaeD tools, such as production systems, method development often stops after comparisons to benchmark problems but does not continue to be validated empirically. The researcher usually performs the benchmark tests themselves thus, attention to visualisation, user interaction and distribution requirements as well as seamless integration mechanisms with other packages are not a priority. In some cases, for example automated routine design tools, there are explicit and commonly agreed criteria to assess the merit of a method, and thus theoretical validation may be sufficient. However, as we move to methods that are intended to enable design exploration and innovation, such claims can only be made

in a theoretical sense rather than a practical one. While theoretical validation is a necessary step in any method development, it is difficult to assess the effectiveness of a computational method in a practical sense without creating a reasonable prototype system.

The software design support necessary to create robust, functional prototypes, as well as integrate them into conventional and experimental design systems is provided by the third element of the methodology. This is the use of an integrated design research platform. The concept was initiated in the mid '90s at several design research centres all needing to integrate multiple computational design research projects into a unified whole. The off-the-shelf software development environments available at the time proved inadequate thus requiring awkward, ad hoc connections with other large, monolithic Cae and AI packages. In order to circumvent these problems, software platforms were designed to exploit the growing body of high quality, open source software becoming available and provide the capability of easily inserting individual research projects as extension modules. Examples of software platforms developed as part of large design research projects are SEED [7], n-dim [8], the Framework for Knowledge Intensive Engineering [9], DIICAD Entwurf [10] and the Schemebuilder Development System [11]. The last example, designed and assembled by the first author, was then developed further, exploiting recent advances in software technology, to create the Integrated Functional Modelling (IFM) Development System [12].

While use of the IFM Development System has proved highly effective in the functional modelling research domain, its monolithic nature restricted application to other projects within the research centre. This was due to the fact that the system philosophy was to provide a single preferred solution covering a wide range of high level functional requirements in a tightly co-ordinated whole. In strongly coupled research projects such as Schemebuilder and IFM, this worked well. However, in more loosely related but still potentially complementary projects often steered by different investigators with diverse software backgrounds, preferences and industrial collaborators this model is not appropriate.

A solution lies in the product platform concept. A product platform, as defined by Meyer and Lehnerd [4], is "a set of sub-systems and interfaces that form a common structure, from which a stream of derivative products can be efficiently developed and produced". Companies in diverse domains have successfully adopted product platforms, for example Hewlett-Packard for ink-jet printers, Black & Decker for power tools, and Visio Corp. for diagramming software. CaeDRe results from reworking the IFM Development System as such a product platform for computational design tool research where the "products" are modular, prototype tools that are sufficiently functional and robust for evaluation by industry. Using the product platform approach emphasises that it is not the choice of individual functional solutions that is vital but rather importance should be placed on definition of interface specifications to form a flexible but coherent architecture. Facilitating integration of research tools can remove the practical obstacles that frequently exist in evaluating separate but complementary design tools in combination. The design and structure of CaeDRe is described in detail in [5].

The use of CaeDRe in the methodology is shown in Figure 2 by the three bold "**Choose Tools/Components**" tasks, one for each of the three main iterative cycles in the research process. These are respectively the modelling of the existing design and/or design process, the definition and implementation of the method and finally the transformation of the fundamental method into a usable prototype tool with suitable user interface. A further insight gained from the work on the Schemebuilder Development System was the realisation that, from Human Computer Interface (HCI) considerations, the integrated software platform also provided ideal facilities for automated data collection and analysis for the empirical

evaluation of prototype design tools [13]. These considerations are incorporated in the methodology in the "Specify Experiment and Data Collection Software" task.

The side bars of Figure 2 show the incorporation of current software development and social science techniques that are necessary to the successful development of CaeD methods and tools. While in most cases provision of appropriate on-line guidance is sufficient, this should be emphasised with human expertise.

3 Case Study: Transforming Structural Shape Annealing into eifForm

The CaeDRe platform, a key element in the presented methodology, has recently been applied to transform a proof-of-concept implementation of the structural shape annealing method into a useable prototype system called eifForm. eifForm is an experimental computer-aided design system for structural synthesis aimed at providing architects and structural engineers with an enhanced means for lateral exploration of design possibilities. The generative method, called structural shape annealing, is composed of the following modules: a structural shape grammar, a structural analysis integration mechanism, simulated annealing optimisation and design performance evaluation algorithms. The method enables automated generation of innovative, performance driven free-form structures, providing both structural solutions and conceptual stimuli [14].

To illustrate the CaeD methodology presented in Section 2, the development history of structural shape annealing will now be placed within this context (Figure 3). Bold numbers relate to activities in the figure. The project began with a goal of creating a method that provided means for lateral exploration of planar truss structures using a grammatical formalism [15] as the knowledge level representation **1**. The conceptual design process for truss structures was modelled **2** considering a wide range of behavioural and performance issues based on design criteria discussed by Billington [16]. To allow for future extensions to three-dimensional trusses, the implementation level representation was chosen **3** as a modified winged-edge data structure inspired by work done by Heisserman and Woodbury [17]. Simulated annealing was chosen as the optimisation method **4** as a non-gradient based method was required for the representation and objective function desired [18]. Additional methods for rule selection and application were also developed. In the choice of development tools and components **5,** Unix-like systems were chosen as the OS platform, with ANSI C the development language, Emacs the coding environment, Geomview for 3D visualisation and FElt for finite element analysis. As theoretical testing was done by the researcher, visualisation, user interaction and distribution requirements were not explicitly addressed at this point. This implementation **6** only included utilities for translating final designs to selected analysis, CAD and visualisation file formats. Further details can be found in [5].

Until now, several papers have been published providing theoretical validation **7** of the method through comparisons to benchmark structural optimisation problems, for example single-layer space frames [14]. Further work successfully applied the method to the redesign of full-scale transmission tower for an energy company [19], with an overall success criterion **8** of reducing life-cycle costs of transmission lines. The company knew that costs were strongly related to structural mass, so reduction of mass was agreed as an easily quantified measurable criterion **9**. Tests have also been done **7** that compare theoretical capabilities for generating innovative solutions in comparison to human designers [20]. The next stage in method and system development is evaluating the capability for *achievable* innovation by designers using the system directly.

The proof-of-concept implementation was tested by non-controlled experiment **10** in collaboration with the Design Studio of the Future (DSoF) at MIT. In the studio, the system was used to explore generative design methods in the context of creating structural form and to assess the potential for enhancing design exploration and innovation. The expanded storyboard that resulted **11** is a digital design process+, i.e. non-digital mediums are always necessary, where the method is a key module within a process that incorporates many other current technologies.

The process starts with creating more elaborate and purposeful models of design scenarios through integration with CAD packages, digital topography maps, and material databases. Design generation using structural shape annealing is then invoked as either a fully automated or interactive process. The final designs are then transferred to visualisation and analysis tools, as the resulting designs often lie beyond the experience of the designer requiring attention to proper interpretation. This interpretation often leads to refinement of the input models for further design generation. Finally, selected designs are transferred to CAD/CAM tools for making physical models, using rapid prototyping.

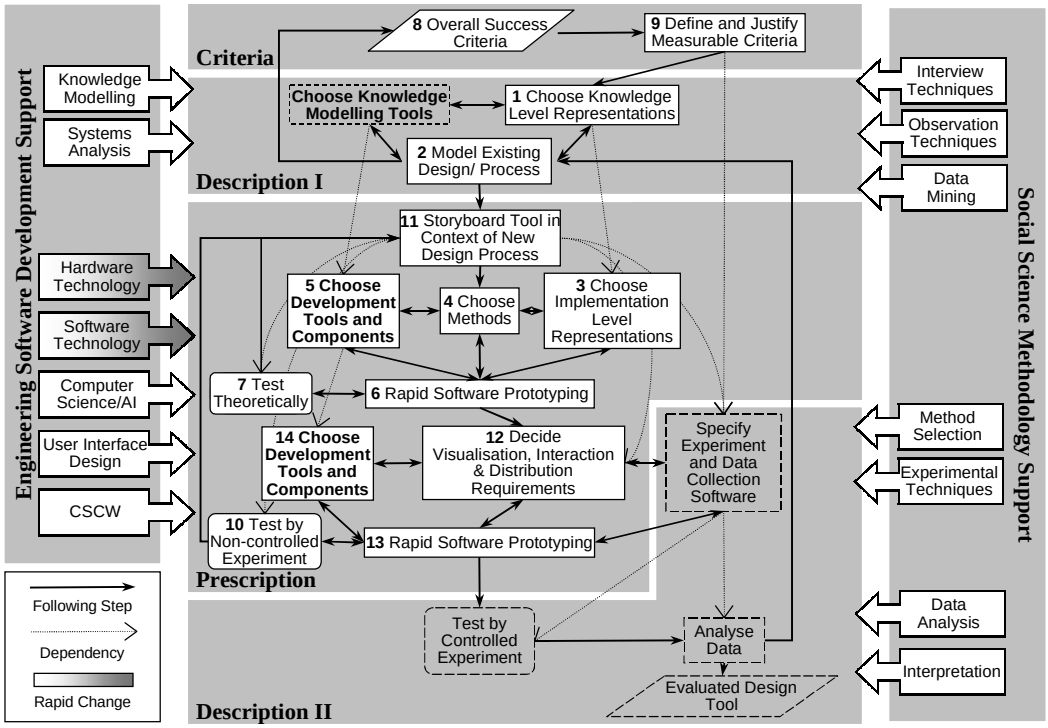

Figure 3. The CaeD Methodology as applied to eifForm

It was found, not surprisingly, that the level of implementation of the proof-of-concept system hindered the students' exploration of the theoretical method capabilities. Effective use of structural shape annealing involves expression of design intent through definition of geometry, structural behaviour and search models as well as proper interpretation and visualisation of generated designs. It is only through this process that designs that meet intent and are constructable will be generated. Important elements in creating an innovative design environment that will enable designers to access the theoretical capabilities of the method are user interaction and seamless integration with up to date digital technologies. As with most computational systems involving some level of automation, e.g. numerical analysis and rendering software, proper modelling will lead to more effective use. Equally important is transparency of the underlying method to foster user understanding of the algorithms. These

key elements for effective use all lie within the categories of visualisation, user interaction and distribution **12** that to now have not been formally addressed.

Further development of eifForm **13** will now be carried out using the methodology presented towards creation of a system for use in education and practice. The choices of development tools and components using the CaeDRe platform **14** are described in [5]. Its application in transforming structural shape annealing into eifForm is intended to create a modular, maintainable, extendable and customisable system for additional method testing in both non-controlled and controlled experiments. It is hoped that fast feedback cycles from experimental use will produce method and system enhancements that will lead to a computational design environment for exploration of innovative structural solutions.

4 Conclusions

This paper described a promising new methodology for practical, high level support of CaeD tool development within a research group setting. The methodology aims to provide a systematic process for producing evaluation ready prototype systems targeted at improving current and future designs and design processes. It is intended that using the methodology will lead to development of more appropriate methods suited to adoption in practice. The degree of merit of the methodology will only be truly established by comparing the capacity for integration and evaluation of tools produced using it with those derived from alternative approaches.

Acknowledgements

The authors would like to thank the UK EPSRC for providing funding for this work.

References

[1] Culley, S.J., *Future Issues In Design Research(FIDR) Workshop*, Final Report, EPSRC/DTI, 1999.

[2] Vergeest, J.S.M., Kleinhuis, S., Wiegers, T., Opiyo, E.Z., Lennings, A.F. and Horvath, I., "On Industrial Evaluation of Novel Design Tools", in *International Conference on Engineering Design, ICED 99*, Munich, v3 pp1857-1860, 1999.

[3] Raphael, B., Shea, K. and Smith, I.F.C., "A task and software independent CAE course", in AICIVIL-COMP99: *The fifth international conference on the applications of AI to Civil and Structural Engineering*, Oxford, England, 1999.

[4] Meyer, M.H. and Lehnerd, A.P., *The Power of Product Platforms*, The Free Press, New York, 1997.

[5] Bracewell, R.H. and Shea, K. CaeDRe: A Product Platform To Support Creation And Evaluation Of Advanced Computer Aided Engineering Tools, Ibid., 2001.

[6] Blessing, L.T.M., Chakrabarti, A. and Wallace, K.M., "An Overview of Descriptive Studies in Relation to a General Design Research Methodology", in *Designers - The Key to Successful Product Development*, ed. Frankenberger, E., Badke-Schaub, P. and Birkhofer, H., pp42-56, Springer, 1998.

[7] Flemming, U. and Woodbury, R., "Software Environment to Support Early Phases in Building Design (SEED): Overview", *Journal of Architectural Engineering*, v1, n4, Dec 1995, pp147-152, 1995.

[8] Subrahmanian, E., Reich, Y., Konda, S L, Dutoit, A, Cunningham, D, Patrick, R, Thomas, M, Westerberg, A W, "The n-dim Approach to Creating Design Support Systems", in *ASME DETC*, Sacramento, California, 1997.

[9] Tomiyama, T., Kiriyama, T. and Umeda, Y., "Toward Knowledge Intensive Engineering", in *Int. Workshop on Engineering Design*, Lancaster, pp319-337, 1994.

[10] Grabowski, H. and Lossack, R.S., "Knowledge Based Design of Complex Products by the Concept of Design Working Spaces", in *IFIP TC5 WG 5.2 International Conference on Knowledge Intensive CAD*, Pittsburgh, PA, USA, pp79-98, 1996.

[11] Bracewell, R.H., Bradley, D.A., Chaplin, R.V., Langdon, P.M. and Sharpe, J.E.E., "Schemebuilder: A design aid for the conceptual stages of product design", in *9th Int'l Conf on Engineering Design ICED'93*, The Hague, Heurista, pp1311-1318, 1993.

[12] Bracewell, R.H. and Johnson, A.L., "From Embodiment Generation to Virtual Prototyping", in *International Conference on Engineering Design (ICED)*, Munich, Germany, WDK, v2, pp685-690, 1999.

[13] Langdon, P.M. and Cheung, H.F., "A Hypertext Documentation System for Schemebuilder: The 'Mechatronics' Book", in *Joint Hungarian-British Mechatronics Conference*, Budapest, 1994.

[14] Shea K., and J. Cagan, "Innovative Dome Design: Applying Geodesic Patterns with Shape Annealing," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **11**:379-394, 1997.

[15] Stiny, G., "Introduction to Shape and Shape Grammars," *Environment and Planning B*, **7**:343-351, 1980.

[16] Billington, D.P., *The Tower and the Bridge: The New Art of Structural Engineering*, Basic Books, New York, 1983.

[17] Heisserman, J., and Woodbury R, "Geometric Design With Boundary Solid Grammars," in *Formal Design Methods for CAD (B-18)*, J.S. Gero and E. Tyugu, eds., Elsevier Science B. V., North-Holland, pp. 85-105, 1994.

[18] Swartz W., and C. Sechen, "New Algorithms for the Placement and Routing of Macro Cells", *IEEE proceedings: Cat No. 90CH2924-9, IEEE Conference on Computer-Aided Design*, Santa Clara, CA, November 11-15, pp. 336-339, 1990.

[19] Shea, K., and I. Smith, "Applying Shape Annealing to Full-Scale Transmission Tower Re-Design", *Proceedings of DETC99: 1999 ASME Design Engineering Technical Conferences*, September 1999, Las Vegas, NV, DETC99/DAC-8681, pp. 1-9, 1999.

[20] Shea K., and J. Cagan, "The Design of Novel Roof Trusses with Shape Annealing: Assessing the Ability of a Computational Method in Aiding Structural Designers with Varying Design Intent", *Design Studies*, **20**(1):3-23, 1999.

Dr. Rob Bracewell
Engineering Design Centre, Department of Engineering, Cambridge University, Trumpington Street, Cambridge CB2 1PZ, UK, Phone: +44 1223 766962, Fax: +44 870 1337199, Email: rhb24@eng.cam.ac.uk, URL: http://www-edc.eng.cam.ac.uk