A TOOL FOR CAPTURING DESIGN RATIONALE

R H Bracewell, K M Wallace

Abstract

This paper describes the research, implementation and industrial evaluation, of DRed (Design Rationale editor), a software tool that allows designers to record their design rationale (DR) at the time it is deliberated. DRed provides an easy to use graphical structure to present the issues addressed, options considered, and associated arguments for and against each one. The statuses of individual elements, and the overall state of completion of the design are clearly visible. The resulting DR graphs are clear, comprehensive, printable, and highly suitable for presentation in review meetings and inclusion in design reports. Freehand sketches, and screen captures from any other PC software used in the design, can easily be incorporated in the DR graphs. DRed is a derivative of the venerable IBIS concept, but contrary to most of the published research in this area, designers in industry have found it natural and helpful to use on complex problem diagnosis and design tasks. It is shown how the rapid research, robust implementation and successful introduction into industry of DRed, was facilitated and guided by the use of a new methodology for researching computer-aided engineering design tools, that has been proposed and is being adopted at the Cambridge Engineering Design Centre (EDC).

Keywords: Design rationale, aerospace design, IBIS, research methodology, CaeDRe

1. Introduction

This research was undertaken as a part of the collaborative Knowledge Capture, Sharing and Reuse (KCSR) project. This involved researchers in engineering design, work psychology and computer science from three universities, working with two major companies in the aerospace sector, Rolls-Royce (RR) and BAE SYSTEMS (BAES). The objective of the KCSR project was to research and implement a software toolset, to support designers in their day to day activities, embedded within appropriate social systems to maximise the effectiveness of the design process. Through an extensive programme of interviews, presentations and workshops, requirements for these systems were elicited from designers, evaluated and prioritised [1]. A prime identified requirement, was to provide a tool capable of capturing DR without undue imposition on designers.

Research into capturing and mapping the rationale for complex decisions can be traced back over 30 years to Kunz and Rittel's pioneering work on Issue-Based Information Systems (IBIS) [2]. The basic concept of IBIS is simple. It is a directed graph, where some nodes represent issues to be solved, and are linked by arcs to other nodes representing alternative solutions, that are each in turn linked to nodes representing arguments for or against them. As can be judged from its many derivatives, (e.g. gIBIS [3], PHI [4], QOC [5], ÉGIDE [6], PROSUS [7], Compendium [8]) the simplicity and expressive power of IBIS holds strong intellectual appeal. However it is now acknowledged that rarely have such techniques been successfully applied in industry, except perhaps in the context of facilitated meetings [8]. Of commercial software tools of this type, only Questmap [8] survives, and development of that seems to have ceased. As a result, in the Design Research field, this topic has become something of a backwater. A search of the last four ASME DETC CDs and the ICED01 CD reveals a total of just 3 citations of Kunz and Rittel's IBIS work. The received wisdom appears to be that IBIS once seemed like a good idea, but many people have shown that it does not work well in practice.

This paper will explain why this seemingly well-explored and unpromising approach was nevertheless selected, and how the application of a newly proposed research methodology has led to the creation of DRed, a novel software tool, which by contrast with earlier results, seems to have considerable potential.

2. Methodology

This research employs a methodology for researching computer-aided engineering design (CaeD) tools, that was recently proposed and is being adopted at the Cambridge EDC. It is summarised in Figure 1.

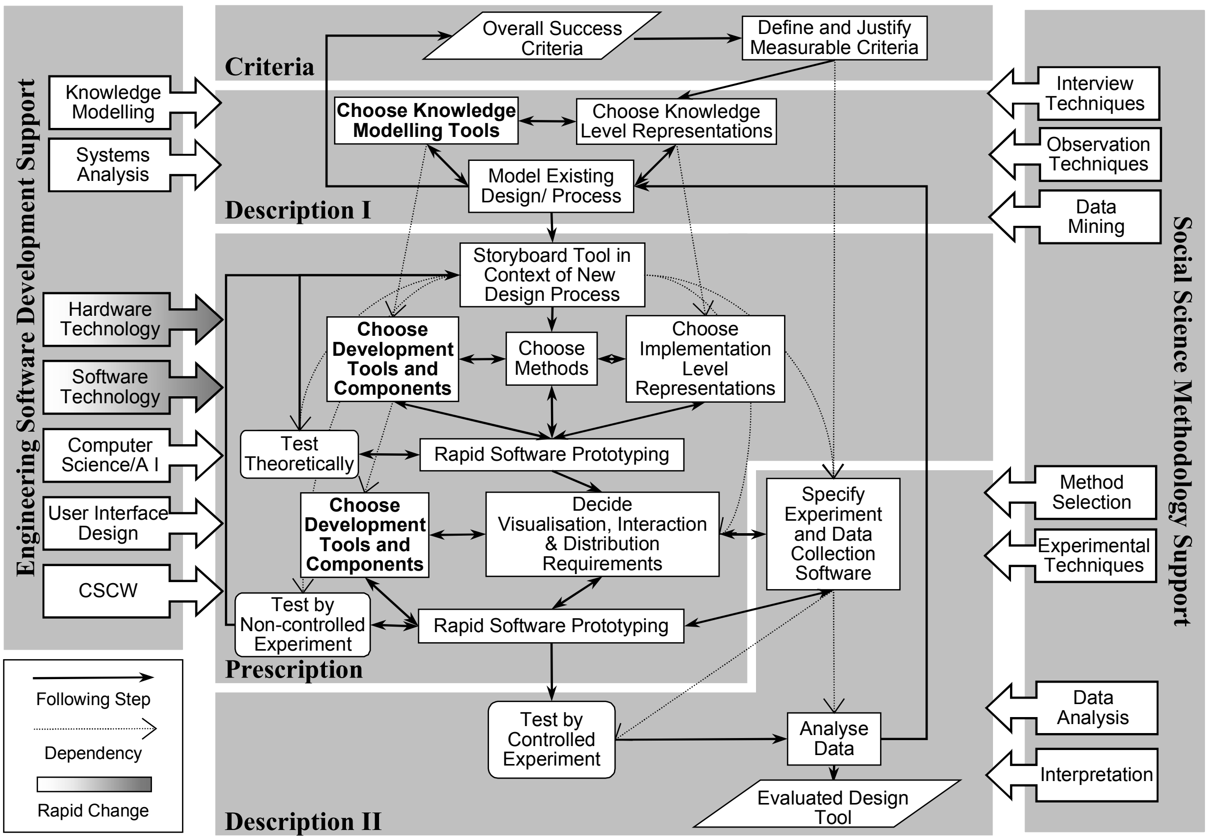

Figure 1 Methodology for Researching Computer Aided Engineering Design (CaeD) Tools

It fits within the well-known general methodological framework for design research proposed by Blessing, Chakrabarti and Wallace [9]. The CaeD research methodology is a practical approach, intended to enable the development of usable software prototypes as early as possible in the research. It is described in twin papers published at the ICED01 conference [10] [11]. A distinctive feature is the use of CaeDRe, a modular, "product platform" style of development, supporting research across the whole EDC. CaeDRe provides a choice of high level languages, tools, reusable software components, integration and code-hardening mechanisms. The research described in this paper, conducted in the space of just 12 months, covers the Description I, Prescription and Description II stages of the methodology.

2.1. Criteria

Figure 2 shows how the CaeD methodology was applied to the research and implementation of DRed, and the tasks are numbered in the order that they were performed. Task "**1** Overall Success Criteria" stated the basis for judgement of success or failure of the whole KCSR project. The criteria were:

- Did the project correctly identify the crucial problems faced by aerospace industry designers, in the capture, sharing and reuse of knowledge?
- Did the project suggest how one or more of these problems could be solved?

In task "**2** Define and Justify Measurable Criteria", for the identified problem of the difficulty of recording DR, it was considered that the merits of a technical solution might be established experimentally, by finding if it allows a larger quantity of DR to be captured per designer hour than standard practice, and whether the DR so presented is easier to understand.

2.2. Description I

The Description I stage aims to model the existing design process in question, using well defined knowledge structures, to inform a proposal as to how that process could be measurably improved. The existing RR standard practice for DR capture is the preparation of a textual Design Definition Report (DDR), and the compilation of a loosely-structured paper Design Scheme Folder (DSF) at the end of a project. Despite the perceived poor usability of IBIS-like approaches, a preliminary examination of the contents of DDRs suggested that IBIS was at least worth a try. Hence for task "**3** Choose Knowledge Level Representations" a provisional choice was made of IBIS-like graphs. The next step was "**4** Choose Knowledge Modelling Tools", with the aim of finding a convenient and flexible tool to allow the representation to be tested and iteratively refined by instantiating it with DR from real cases. As argued in [12], the choice is far wider than the small range of specialist knowledge modelling tools exemplified by Protégé 2000 [13]. Depending on circumstances, the best choice can range from a pencil and Post-It notes, through diagramming tools such as Visio, to systems engineering or product data management tools based on industrial strength databases like Oracle or Versant. Here, the first option evaluated was to use an existing commercial IBIS tool such as Questmap [8] or DRAMA (a derivative of ÉGIDE [6]). However neither of these were suitable. The main problem with Questmap is the way nodes are represented by an icon and a short, single-line text label. The full text of the node is viewed or edited via a dialog invoked by double-clicking. This means that for every issue, answer or argument captured, the user needs to summarise it into no more than 5 or 6 words. As the basis for DR capture, this was thought to be an intolerable burden on the designer. A further serious problem is that due to the text hidden in the nodes, no complete and easy to follow printed hardcopy of the DR is available for inclusion in reports. DRAMA was found to share the same problems, with the additional disadvantage that the DR is forced into to a strictly hierarchical structure. It was clear that while large parts of the argumentation in DDRs is, indeed, tree structured, there are also non-hierarchical relationships that must be captured.

The knowledge modelling tool finally chosen was Graphlet, a general purpose interactive graph editor, free for non-commercial use, from the University of Passau [14]. In common with the CaeDRe platform, it employs the "two language" philosophy in which a robust and efficient compiled core underlies a configurable user interface written in a very high-level interpreted language. Graphlet was thus used in task "**5** Model Existing Design/ Process" to explore the use of IBIS-like representations to structure graphically the DR contained in existing DDRs. A typical result is shown in Figure 3, side by side with the original DDR.

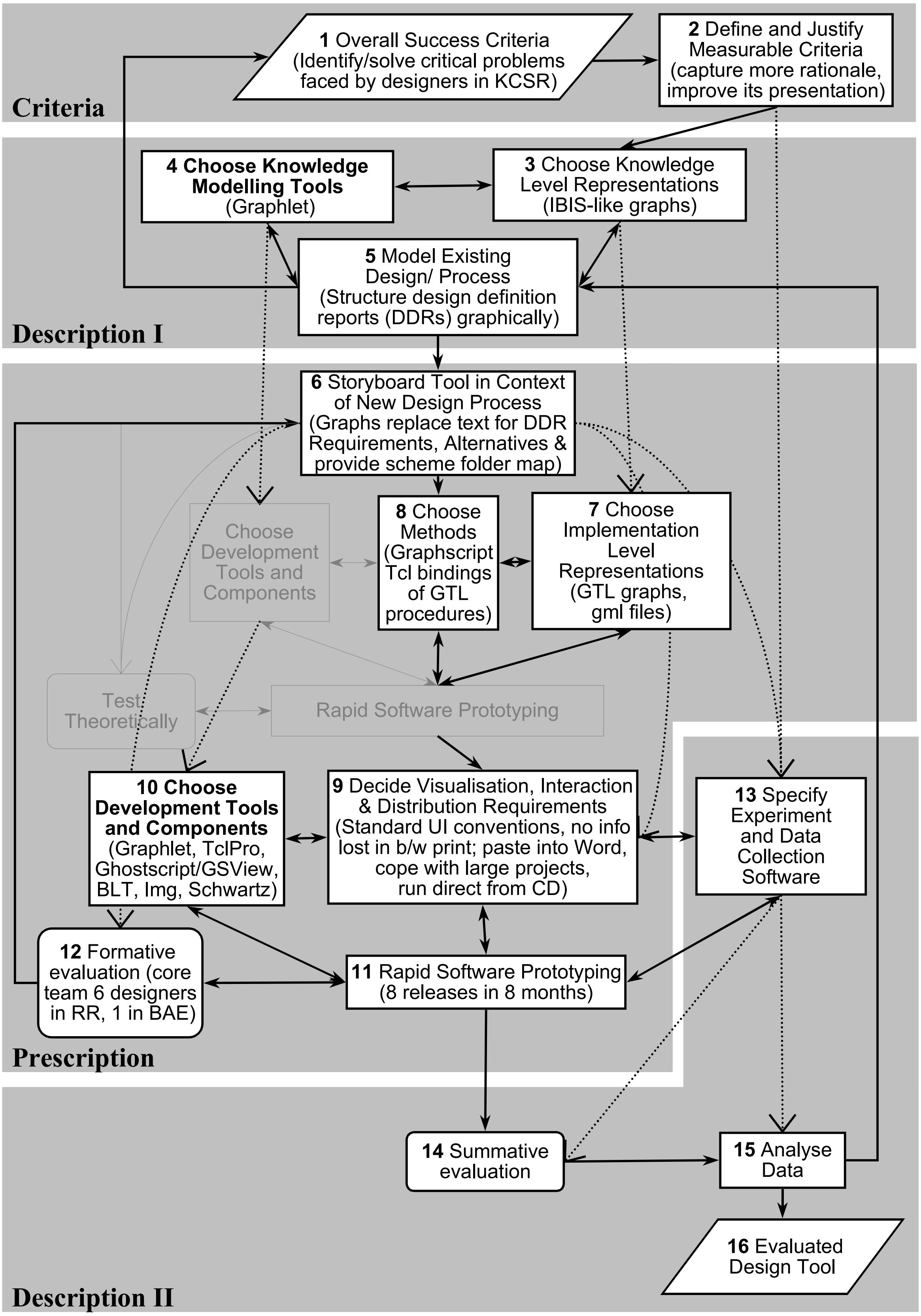

Figure 2 CaeD methodology applied to DRed Research and Implementation

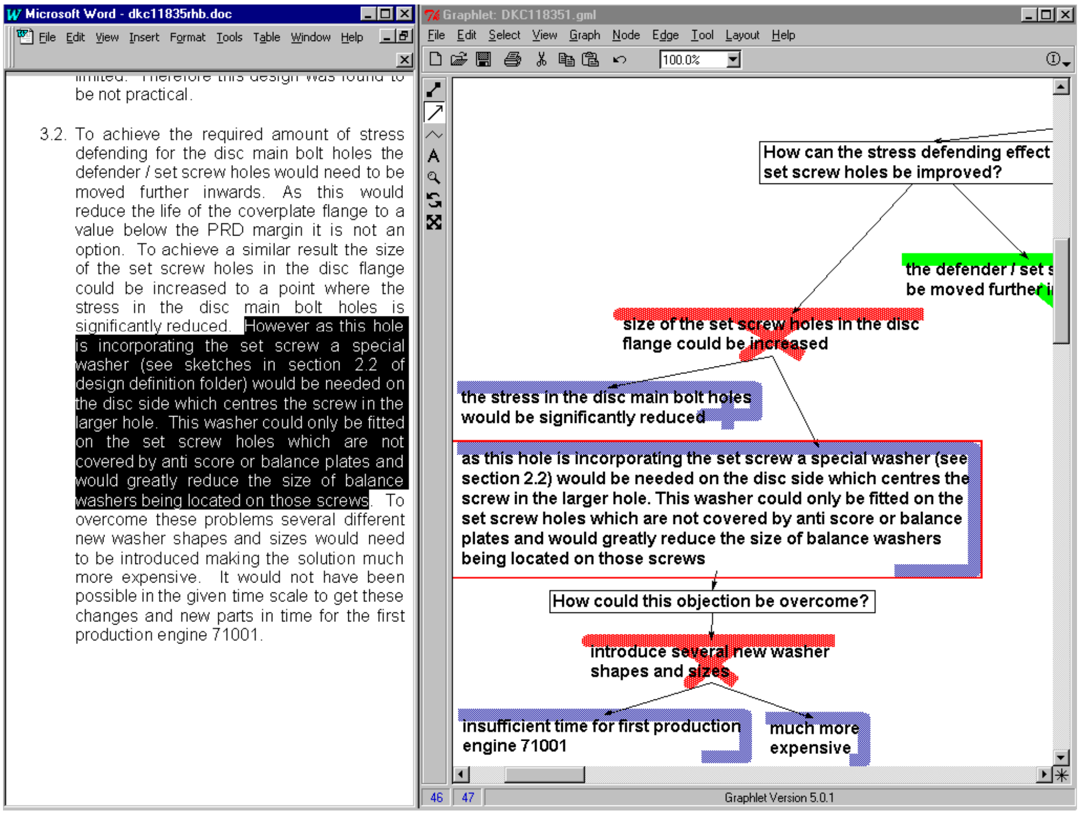

Figure 3 DR from an existing design modelled using Graphlet (showing structure only)

These reverse engineered DR graphs were felt by both academic and industrial members of the KCSR team to be easier and quicker to understand than the textual reports from which they had been derived. Unlike the labelled icon node representations of Questmap, the issues, answers, pro and con nodes resized to contain multiple lines of text, behind which coloured graphics clearly signified the node type. The advantages, were that eliminating icons saved screen area, there was no need to summarise into labels, and nothing was hidden so a printed hard copy was readily available. The conviction grew that this approach was promising, and that the frequently reported problems with IBIS were surmountable if careful attention was paid to usability. The use of DR from real projects in this study convincingly demonstrated the feasibility and possible benefits of such a design support tool, attracting great interest in the companies. Designers were beginning to want to try out the approach for themselves.

2.3. Prescription

Graphlet now really proved its worth. The knowledge modelling exercise had suggested what sort of tool might answer the companies' DR capture needs, but Graphlet does not fit the bill directly. However, it had been chosen with forethought about the possibility of its use in the Prescription stage as well. This was prompted by consideration of the dotted dependency arrows in Figure 2, from "**4** Choose Knowledge Modelling Tools" to "**10** Choose Development Tools and Components" and from "**3** Choose Knowledge Level Representations" to "**7** Choose Implementation Level Representations". These show that there may be an opportunity to jump-start tool implementation, if the chosen knowledge modelling tool can also support rapid software prototyping, and precisely defined, robust, implementation level representations as well as the less formal knowledge level.

If a research prototype software tool is to be tested by designers in industry on live projects, it is imperative to consider the practical details of its introduction. This is the first task of the

Prescription stage; "**6** Storyboard Tool in Context of New Design Process". In the case, the initial storyboard, was for designers simply to use the tool to create DR graphs, that could either be directly printed and stored in the DSF, or imported into DDR Word documents.

In some CaeD tool research projects, the underlying methods and representations are implemented first and tested by the researcher(s) before adding a proper user interface to allow third party use. The CaeD methodology shows this by two levels of rapid software prototyping in the Prescription stage. In this case, by taking the approach of progressively modifying Graphlet using Tcl/Tk scripting, it was possible to combine these levels and have a proper user interface from the outset. Hence the upper level in Figure 2 is greyed out.

Graphlet is built upon an underlying C++ Graph Template Library (GTL), that allows graphs to be represented, navigated, modified, loaded from and saved to, graph mark-up language (gml) files. Experience manipulating large graphs in the Description I stage, had shown these facilities to be both efficient and robust. Moreover, the GTL representation is extensible, in that attributes for any node or arc can be defined on the fly, and are saved to file. Hence Graphlet's GTL was a chosen for "**7** Choose Implementation Level Representations", and Graphscript, its Tcl binding of the GTL procedures, was suitable for "**8** Choose Methods".

The next task was "**9** Decide Visualisation, Interaction & Distribution Requirements". In terms of visualisation and interaction, it was considered vital that the tool should be as intuitive as possible to anyone familiar with Windows diagramming software such as MS Draw or Visio. Fortunately, Graphlet's user interface already follows this principle reasonably well. It was further decided that though colour might be used to make the DR clearer on screen, the meaning should be unambiguous if the graphs are printed in monochrome. Referring to the storyboard, it was decided that a way would have to be found to import Graphlet's Postscript output into MS Word. The biggest difficulty in distribution of the software was the fact that RR and BAES, as has become customary in large companies, have outsourced their IT provision. This means that for an experimental software tool to be installed on networked PCs, a prohibitively expensive and time-consuming process would need to be followed with the external IT service company. Initially, this problem was addressed by finding a small number of dedicated, standalone PCs to be used alongside the designers' normal machines – a workable but not ideal solution.

Next was "**10** Choose Development Tools and Components". Graphlet was run as a Tcl script and DLL from within the TclPro development environment, chosen mainly for its excellent debugger. The Source Navigator IDE was chosen to aid the navigation, understanding and editing of Graphlet's large body of Tcl source code. Ghostscript and GSView were also chosen, as together they provide a command line utility called pstoedit, to convert Graphlet's Postscript output into Windows metafiles that can be imported into Word documents.

A common bottleneck in CaeD tool research lies in task "**11** Rapid Software Prototyping". This is the time consuming and difficult problem for the researcher to convert partially worked out research ideas into a sufficiently precise specification for satisfactory implementation by a programmer. The only way to avoid this problem completely, is for the researcher to code the software personally, which can be very productive indeed if the right choices of languages, components and tools have been made. This was the approach taken here, with all the software being written by the first author. By writing Tcl scripts to modify incrementally the Graphlet user interface, it was possible to release in less than a month, an initial prototype DR capture tool, that was named DRed v0.1. While limited, this was sufficiently functional to support real design work. Moreover, due to its solid foundations in the unmodified GTL library, and the run-time error checking of the modified and newly written Tcl scripts, it was immediately robust enough for designers to use with confidence.

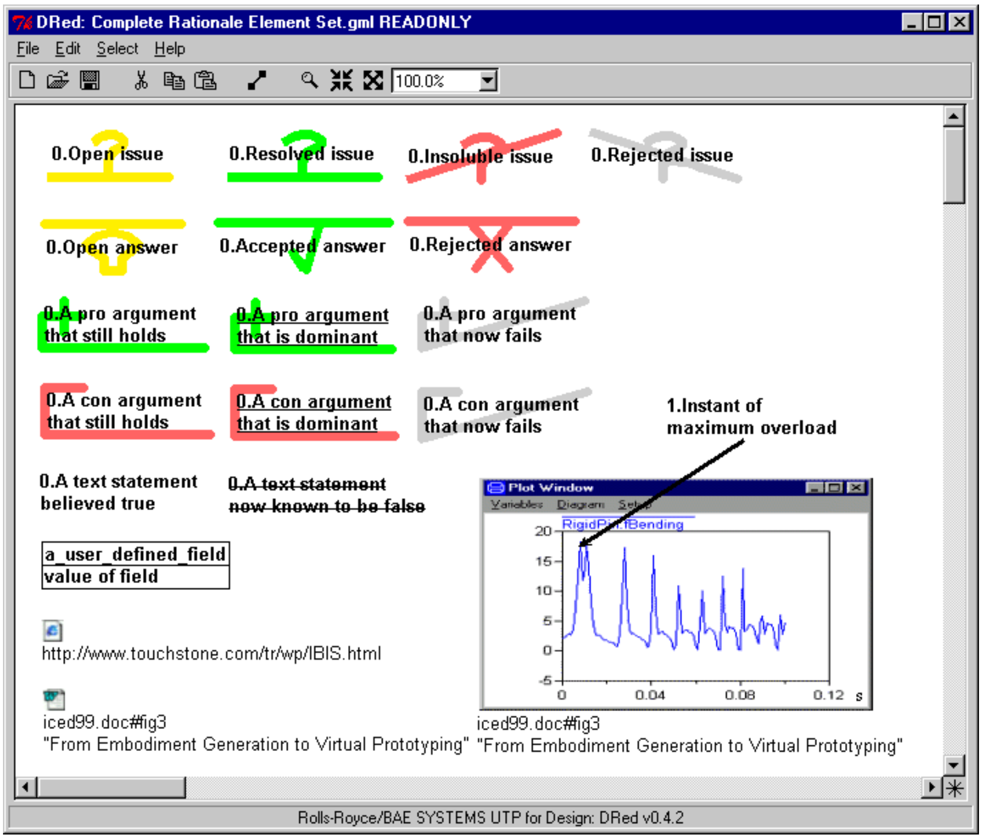

Figure 4 The DR element types and their statuses

As the basis for task "**12** Formative evaluation", a core team of six designers in RR and one in BAES was recruited to use DRed in their daily work. Immediate reactions were very positive, and accompanied by many helpful ideas about functionality and usability. There were also suggestions that its use in recording the design process, far from being a hindrance, was actually proving beneficial in lending structure to designers' thoughts. An efficient and productive research cycle of idea generation, implementation and testing ensued. Seven further releases of steadily increasing capability, and refinement of the underlying concepts, followed in the next eight months.

Various changes were made to the representations used (**7**). New element types were added, along with the idea that *all* elements, not just answers, should have a set of clearly indicated statuses. The current complete set is shown in Figure 4. The initial arbitrary choice of link arrow direction was reversed, such that when the status of any node is changed by the designer, the arrows point from that node to all others for which the status should consequently be reviewed. This can be seen in Figure 5, showing an example DR graph, simple enough to be self-explanatory, that explores the best way of getting to the railway station when the car will not start. A desire to use DRed to capture the conceptual design of thermodynamic cycles for a new range of small jet engines, led to the concept of "tunnelling links" being devised. These allow such large problems to be handled, by allowing DR graphs to be distributed, visualised and navigated across multiple files, as shown in Figure 5. Links appear to tunnel below the surface and reappear elsewhere in the same or a different file, continuing on to their destination element. In browsing, the tunnel is traversed by double-clicking the small circle representing the tunnel mouth, and the pointer is taken to the far end.

In the course of the iterative research and implementation process, there was also a highly beneficial change of distribution strategy (**9**). It was realised it that DRed need not be installed on individual PCs, since, by virtue of it's use of Tcl, it can easily be made to run directly from a CD-ROM or networked home directory. This enabled DRed to be run on designers' normal, networked PCs, instead of the unsatisfactory initial practice of using dedicated PCs solely for DRed. With an additional element type, the file reference (the bottom left element shown in Figure 4), this enabled a major expansion of the storyboard for use of the tool (**6**). The DRed graphs could now be used to provide a map of the computer files generated during the design, and stored in an electronic DSF. The choice of three new software components (**10**), the BLT, Img and Schwartz libraries for Tcl, solved the problem of allowing file reference nodes to be represented, if desired, by a bitmap graphic rather than the default icon (see the bottom right element in Figure 4). This provided another major functionality enhancement requested by designers, as it meant that the output of any PC software, such as CAD or a spreadsheet, could easily be captured from screen and included in the DR graphs. Similarly design sketches, whether drawn directly onto a Tablet PC or scanned from paper, could easily be intermingled with textual rationale, combining the flexibility of a design notebook with a rigorous structure for arguments and dependencies.

2.4. Description II

Once the main functionality of DRed had stabilised, the research moved on to the more formal, summative evaluation of the Description II stage. This determines whether or not the use of the tool has had a positive impact on the measurable success criteria identified in task **2**. In task "**13** Specify Experiment and Data Collection Software", there is a trade-off between formality, and reality, of the experiment chosen. It is easy to set up a tightly controlled experiment for a simple problem in a laboratory setting, but the result may well be less reliable than for one less controlled, but with more extensive use on real problems in an industrial setting. Since there was a strong drive in the companies for rapid expansion in use of the tool, the latter approach was chosen. The proposed experiment, involves a wide range of people using DRed, and comparing their rate of DR capture with similar projects that they have recently conducted, using the old approach of textual DDRs and paper DSFs. The mean number of issues, answers, pros and cons captured per designer hour using DRed will be counted, and compared with figures obtained by reverse engineering DDR and DSF contents from earlier projects into DRed. Additionally, a questionnaire given to the users after 6 months will record their personal assessment of the tool. Controlled laboratory experiments will be separately performed, comparing designers' ease of understanding of DR that has been structured graphically using DRed, with the same arguments presented conventionally as narrative text, to see whether the presentation really is improved.

Task "**14** Summative evaluation" was initiated by giving a two hour hands-on training course in small groups to around twenty new users. It is planned to train a similar number of new users in the next few weeks. The evaluation has, at the time of writing, been ongoing for two months and informal feedback continues to be highly positive.

3. Conclusions

The use of the recently proposed CaeD research methodology has been highly successful, and led to the creation of a DR capture tool that aerospace designers are enthusiastic to use. These conclusions are still tentative because evaluation results so far have been essentially formative, involving quick, pragmatic non-controlled testing in a tight iterative research and implementation loop. However a more formal and extensive summative evaluation process is now well underway, with strong support from the funding and collaborating companies.

Acknowledgements

The authors acknowledge the support of EPSRC, Rolls-Royce and BAE SYSTEMS through the UTP for Design, and would particularly like to thank Michael Moss, Dave Knott and all the designers who bravely evaluated DRed, for providing invaluable ideas, feedback and encouragement.

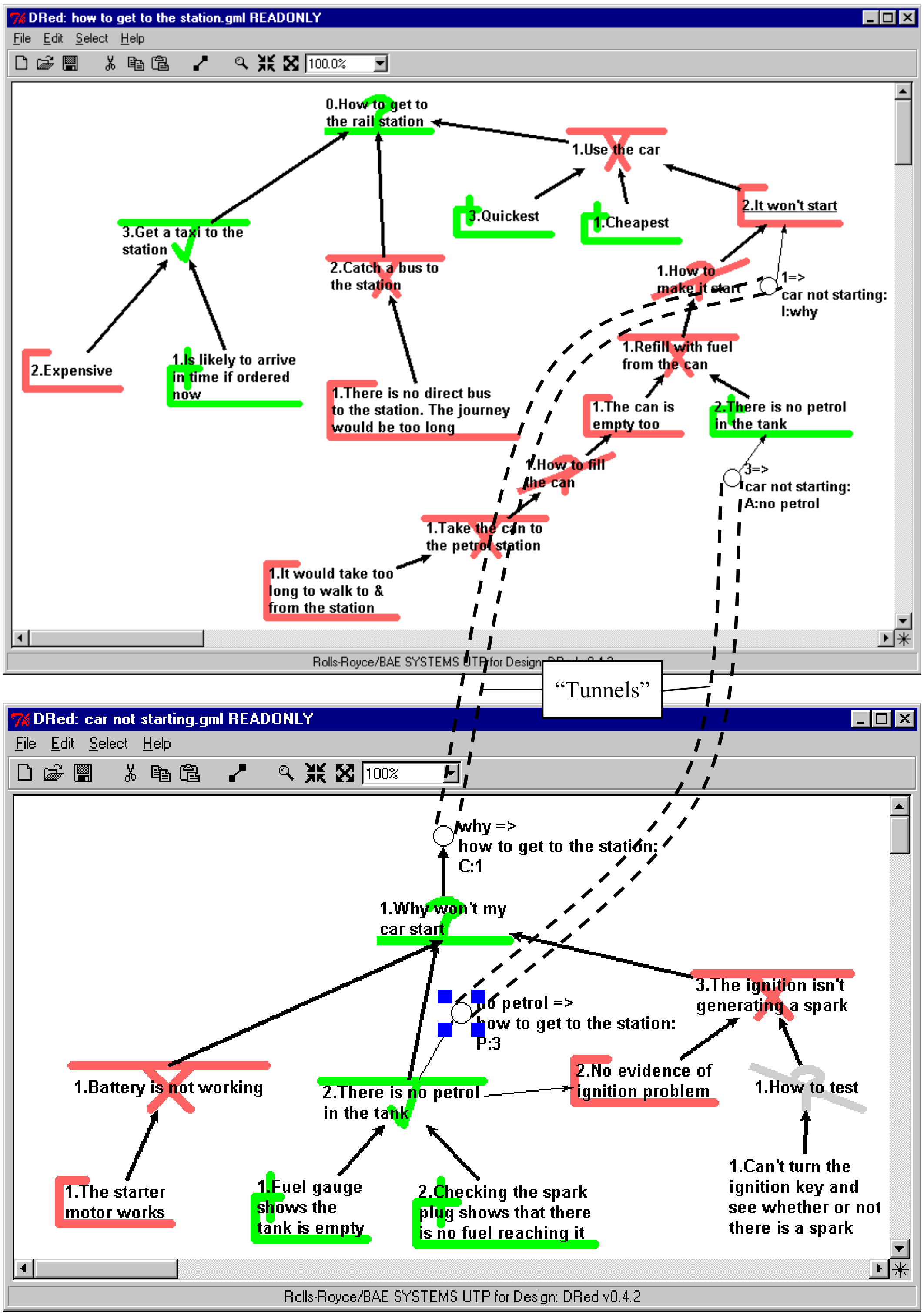

Figure 5 Example DR graph distributed between files using tunnelling links

References

[1] Kerr, M.P., Waterson, P.E. and Clegg, C.W., "A socio-technical approach to knowledge capture sharing and reuse in aerospace design", Proceedings of DETC-CIE21254, 2001.

[2] Kunz, W. and Rittel, H.W.J., "Issues as Elements of Information Systems. Working Paper 131", Center for Planning and Development Research, Berkeley, USA, 1970.

[3] Conklin, J., Begeman, M.L., "gIBIS: A hypertext tool for exploratory policy discussion", ACM Transactions Office Information Systems, Vol. 4, 1988, pp.303-331.

[4] McCall, R. "PHI: A Conceptual Foundation for Design Hypermedia", Design Studies, Vol. 12, No. 1, 1991, pp.30-41.

[5] MacLean, A., Young, R.M., Bellotti, V. and Moran, T.P. "Questions, Options, and Criteria: Elements of Design Space Analysis", Human-Computer Interaction, Vol. 6, 1991, pp.201-250.

[6] Bañares-Alcántara, R., King, J.M.P. and Ballinger, G.H., "ÉGIDE: A Design Support System for Conceptual Chemical Process Design", AI System Support for Conceptual Design, ed. Sharpe, J.E.E., Springer-Verlag, 1995, pp.138-152.

[7] Blessing, L.T.M., "A Process-Based Approach to Computer-Supported Engineering Design", Ph.D. Thesis, University of Twente, 1995.

[8] Conklin, J., Selvin, A., Buckingham Shum, S. and Sierhuis, M., "Facilitated Hypertext for Collective Sensemaking: 15 Years on from gIBIS", Proceedings 12th ACM Conference on Hypertext and Hypermedia, Århus, Denmark, 2001.

[9] 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, Springer, 1998, pp42-56.

[10] Bracewell, R.H., Shea, K., Langdon, P.M., Blessing, L.T.M. and Clarkson, P.J., "A Methodology For Computational Design Tool Research", Proceedings of ICED01, Glasgow, Scotland, Vol. "Design Research…", 2001, pp.181-188.

[11] Bracewell, R.H. and Shea, K., "CaeDRe: A Product Platform to Support Creation and Evaluation of Advanced Computer Aided Engineering Tools", Proceedings of ICED01, Glasgow, Scotland, Vol. "Design Methods…", 2001, pp.539-546.

[12] Bracewell, R.H. and Wallace, K.M., "Designing a Representation to Support Function-Means based Synthesis of Mechanical Design Solutions", Proceedings of ICED01, Glasgow, Scotland, Vol. "Design Methods…", 2001, pp.275-282.

[13] Grosso, W.E., Eriksson, H., Fergerson, R.W., Gennari, J.H. and Tu, S.W., "Knowledge Modelling at the Millennium (The Design and Evolution of Protégé-2000)", Proceedings of KAW'99, Banff, Alberta, 1999.

[14] Himsolt, M., "Graphlet: Design and Implementation of a Graph Editor", Software - Practice and Experience, Vol. 30, No. 11, 2000, pp.1303 – 1324.

For more information please contact:

Rob Bracewell, Cambridge University EDC & Rolls-Royce/BAE SYSTEMS UTP for Design,
Department of Engineering, Trumpington Street, Cambridge, CB2 1PZ, UK.
Tel: +44-1223-766961 eFax: +44-870-1337199 Email: rhb24@eng.cam.ac.uk
URL: http://www.eng.cam.ac.uk/~rhb24