AN EXPERIENCED-BASED APPROACH TO DECISION SUPPORT SYSTEMS

Steve Russ, Suwanna Rasmequan, Meurig Beynon

Department of Computer Science, University of Warwick, Coventry, CV4 7AL, UK

Tel: +44 (0)24 7652 3193 Fax: +44 (0)24 7657 3024

{ sbr, suwanna, wmb }@dcs.warwick.ac.uk

ABSTRACT

We identify and motivate a fundamental challenge for the effective use of decision support systems (DSS) in the future: how can we substantially improve the quality of interaction, and the degree of flexible engagement and integration, between humans and computers? We propose a new approach to using computers that is experience-based and human-centred, and that offers the potential to address this challenge through the construction of computer artefacts that are based upon the concepts of observables, dependency and agency.

Keywords: Decision Support Systems, Expert Systems, Modelling, Observable, Dependency, Agency

1. THE CHALLENGE FOR DECISION SUPPORT SYSTEMS

Over the last three decades the development of Decision Support Systems (DSS) has seen the opportunistic exploitation of many kinds of technology. Yet in spite of the huge technical advances over these years the improvement to be seen in the effectiveness and extent of DSS usage has been relatively modest. There seems to be a significant pattern operating here. On the one hand, applications where useful interactions between human and computer are simple and they can largely be preconceived - such as word processing, accounting software, e-mail and web-browsing - have escalated in use and sophistication very rapidly. On the other hand, applications demanding a high quality of interaction because of human factors, the role of the physical context, or the unpredictability of the environment (e.g. engineering design, and business modelling) have made much slower progress. There is a *prima facie* case here that a fundamental challenge facing computer applications, and DSS in particular, is that the quality of interaction, and the degree of integration possible between human and computer activities, is affected very little by technology.

We approach this challenge from a computing perspective that is unconventional: we are not so much advocating new technological solutions but rather proposing the adoption of a different paradigm for the use of computing technology. Despite a number of signs of change, contemporary views on computers and programming still seem dominated by a reliance on logic and formal methods in constructing and interpreting applications. Such a logicist outlook affects thinking about data and models, and tacitly discourages human interaction. The paradigm that we propose here is experience-based, human-centred and founded on the concepts of observable, dependency and agency. It is thus very different from a logicist outlook but more primitive conceptually and capable of embracing formal concepts rather than challenging them.

We first explain why we regard the challenge of improving the quality of interaction and the degree of integration between humans and computers to be such a fundamental and important challenge for the future direction of DSS. By way of illustration, consider the task of manually constructing a timetable for a class of students each of whom is to give an oral presentation to a

specified group of staff. Though it is possible to give an abstract characterisation of a feasible timetable, and indeed to write a computer program to generate a timetable that meets every prosaic constraint concerning availability of staff and students, this is not what the human expert understands by solving the time-tabling problem. The expert has many tacit expectations about what qualities the timetable can have, and these expectations will be tempered by a process of exploration that attempts to construct a solution with these qualities. If the expert chances upon many possible solutions, or is frustrated by a lack of solutions, her evaluation criteria may consequently be revised. Though the coverage of the solution space may be far less comprehensive than that offered by an automated procedure, there will be no counterpart for the expert's personal understanding of those parts of the solution space that have been visited. In constructing and contemplating possible solutions, the expert will have explored the relationship between the abstract assignment of symbols to slots and its real-world referent in ways that are highly situation-specific, and can neither be preconceived nor automated. Her evaluation may indeed reflect the subjective judgements of all the staff and students involved. In some circumstances, the expert's knowledge of the timetable will show itself in an ability to make adjustments in response to changes in requirements following unexpected events.

Thus, even an expert will typically not know initially how to identify, or formulate, a problem in a circumscribed manner suitable for a programmed solution. Generally the expert will also have no systematic method of finding solutions. What is judged to be a satisfactory solution will depend highly upon the specific problem-solving situation, and upon how skilful, fortunate and conscientious the expert has been in their exploration of the situation. They will know what to do next only when they find themselves in a situation. The problem-solving activity is typically guided by what is encountered, when it is encountered. Furthermore, the expert may not have any explicit heuristic for evaluating and selecting solutions. The criteria applied in evaluation may be qualitative or impossible to preconceive.

These issues about the expert's perspective relate to an abstract model of computer use in decision support. This is not to deny the value of automation in problem solving activity. Even in manual solution of a task such as time-tabling, an expert can exploit automation in several ways.

Automation is useful in imposing known constraints. For example, a computer interface can be constructed to prevent or inhibit constraint violation. The computer can perform routine calculation and inference that needs no reference to the real-world referent, as in the updating of a spreadsheet, or automatic execution of slot assignment steps that are obligatory consequences of a particular scheduling action. Automated search procedures can also address the routine aspects of exploring the solution space. Computer generation of potential solutions cannot provide the in-depth understanding of a region of the solution space such as the human interpreter acquires, but it typically leads to an examination of the search space that is more comprehensive, more systematic, and deals in a more uniform way with all potential solutions. Where quality criteria can be suitably framed, evaluation can also be automated. By recording, classifying and monitoring the results of searches and evaluation, the computer can also supply global knowledge about the solution space that is otherwise inaccessible.

The above discussion highlights the need for both human and automatic processing activity in decision support and similar examples could easily be multiplied. A key issue is how these activities can be effectively integrated. Only the human observer can interpret the abstract solution in relation to its referent. Only the computer can deliver the processing power to support more efficient and comprehensive exploration and analysis. Introducing a wide range of preconceived ways of evaluating solutions does not attack the essence of this problem. The judgements made by the human expert concern a relationship between the abstract solution and its real-world referent; they typically involve observations that cannot be comprehensively pre-conceived and pre-programmed.

The limitations of current paradigms for integrating human and computer activity in problem solving are most prominent in turbulent environments. In a time-tabling context, for instance, it is usually necessary to adapt to changes in requirement that arise unexpectedly, either during the process of compiling a timetable or even whilst it is in operation. For such purposes, it is essential to adapt the existing solution rather than develop a new solution - a process for which there may be no automatic support if the existing solution has been generated automatically. Such adaptation may be carried out much more effectively where human insight has played a significant role in the development of a solution, especially if the change in requirements pertain to features of the solution space that have been thoroughly explored.

There is an essential distinction between the treatment of observables by humans and the treatment of data by computers. The evidence for this is clear in the sharp separation that differentiates the conception and construction of a program from its execution. In program development, human imagination and knowledge of the world is paramount. In program execution, human interaction and interpretation invokes knowledge of the world in preordained and preconceived ways. There is a fundamental mismatch between abstract data that is interpreted by the human in direct association with its counterpart in the real-world referent and situation, and abstract data that is manipulated according to computational rules that can only take account of pre-specified and pre-programmed features of this association. The next section examines an approach to computer-based modelling which aims to address this issue.

2. THE PRINCIPLES OF EMPIRICAL MODELLING

Our approach can be viewed as a radical generalisation of spreadsheets – in terms of the interface, the underlying algebra and allowing many users. The key significant idea of spreadsheets - state change through dependency and agency – has not really been taken up seriously in conventional software development. The approach adopted here takes the concepts of state, and of state change, as fundamental. We are using 'state' in a broad sense to encompass states of mind in subjective experience as well as the state of a computer artefact and the state of its real world referent. The concept of programming for us includes not only algorithms and data but also the configuring of a whole collection of components, including human agents, into a provisional system. This is such a broad concept that we have usually referred to our artefact construction as modelling rather than programming. And because of the emphasis on experience (particularly on observation and experiment) we describe our research as Empirical Modelling (EM). EM is a novel approach to modelling developed by Dr Meurig Beynon and his collaborators at the University of Warwick, UK since 1983 (see http://www.dcs.warwick.ac.uk/modelling).

The way in which we construct an EM model depends on construing a phenomenon through three basic perceptions: those of observables, dependencies and agents. These perceptions are based upon the modeller's viewpoint and interpretation of the domain or, in other words, the modeller's personal experience of the domain. The initial statement of perceptions is organised into an account written in a special purpose notation. The contrast between such an account of a domain and a conventional system specification may be likened to that between an artist's preliminary sketch and an architect's final drawing. The former is spontaneous and personal, a soft boundary to be continuously developed into a piece of artwork, the latter is the culmination of earlier drafts, a hard boundary not intended for revision.

The EM approach offers a distinctive kind of modelling. With its focus on state, and lack of control structures, it is more primitive than conventional programming. State change is only achieved through automatically updated dependencies or the explicit action of an agent (who may be the modeller). With a conventional approach, the modeller must preconceive the inputs and outputs before starting the construction of the model in order to prescribe the processes involved. If there is a need for new inputs or outputs, e.g. an additional system feature is required, then the

whole process may have to undergo costly revision and redesign. With the use of EM, revisions can in principle be made at any point during the modelling process with immediate effect. This is because the state of a model is captured in a script of definitions that resemble spreadsheet formulae.

During the initial stages of script construction the effects of certain interactions will show the need for new observables and new, or refined, dependencies. The script, built up in this experimental and incremental fashion, represents the modeller's knowledge of the domain in a similar way to that in which a spreadsheet embodies some of the financial and operational knowledge of its user. At the same time the script corresponds to the state of an evolving computer model of the domain. Because of the origin of this model in personal experience it has a situated quality. That is, it is organically connected to its referent or domain through being continuously open to revision by comparisons between experience with the domain and experiences of interaction with the model. In recognition of this style of evolution we often call our models 'interactive situation models' (ISMs, see (Beynon et al., 1999)).

There is now also a distributed version of our tools that allows a variety of modes of communication between many users. This offers the potential to support collaborative modelling where many people with different viewpoints and knowledge may interact with a common model and work to achieve consensus on properties or objectives that are initially in conflict.

3. HUMAN-COMPUTER INTEGRATION

Classical expert systems were primarily directed at displacing human decision making (Sauter, 1997). In keeping with our aspirations for decision support systems, the use of ISMs potentially offers broader scope for flexible human-computer engagement. Some of the possibilities can be illustrated by the timetable ISM described in (Beynon et al., 2000a). Human and computer activities can be combined in the environment supplied by the timetable ISM in several ways, each exploiting a slightly different configuration of dependencies between key observables, or reflecting a different mode of interaction. These include:

Timetable comprehension and maintenance: It is typically difficult for a human time-tabler to evaluate an automatically generated timetable. The timetable ISM can be regarded as essentially composed of a collection of observables relating to the available resources and their present assignment, together with a family of agents that monitor the state of the timetable and can be authorised to change the state. The source of the resource assignment data is immaterial - it is possible to extract data from any given partial or complete timetable and insert it in the ISM. In this way, it becomes possible for the human time-tabler to explore the neighbourhood of given solutions to the time-tabling problem in much the same way as if they had constructed it themselves.

The implementation of procedures or algorithms for time-tabling: It is not usually possible to develop or to study an algorithm by interleaving automatic computation with manual step-by-step execution and revision. This can be done using an ISM. Though it is not possible for the human time-tabler to intervene in the execution of an optimised algorithm, it is possible for the results of the execution to be integrated into the manual time-tabling activity. For instance, the product of such an algorithm might be a suitable observable (such as whether the completion of a partial timetable is provably impossible) or a timetable that can become a new subject for comprehension and exploration.

Opportunistic extension of the ISM to encompass a richer family of observables: One of the weaknesses of automatic activity is that it is typically specified in such a way that it is hard to adapt to a new requirement, and that it is typically executed in a manner that is oblivious to its environment. Providing support for timetabling is a task that raises many issues concerning unexpected interactions between human and automatic activities.

Each of these scenarios demands a form of close integration of human and automatic activity. In principle, the EM paradigm is well-suited to the task, but the technical challenges call for the development of our current tools. Introducing new scripts is an exceptionally versatile technique for the dynamic extension and revision of the internal computer model. The development of interfaces to reflect such an open functionality is a more perplexing issue, since these cannot be preconceived, and are subject to dynamic extension and reconfiguration.

4. CONCLUSION

This paper, which is a much-shortened and modified version of (Beynon et al., 2000b), has described how the introduction of a new paradigm for computer-based modelling can lead to a closer integration of manual and automated activities. Though the illustrations that have been discussed are simple, the potential implications for decision support are broad and wide-ranging. More evidence for this claim can be found in the distributed ISMs that have been constructed to simulate a historic railway accident (Beynon,1999 and Sun et al., 1998), to implement a virtual electrical laboratory and to provide support for restaurant management (Rasmequan et al., 2000). Current goals in this research are to exploit the potential of our tools for group DSS and strategic DSS, and to develop improved tool support that can make the use of Empirical Modelling principles and methods more accessible to non-technical decision makers.

5. REFERENCES

Beynon, W.M. (1999), "Empirical Modelling and the Foundation of Artificial Intelligence", Proceedings of CMAA'98, Springer Lecture Notes in AI, 1562, Springer-Verlag, pp. 322-364.

Beynon, W.M., Cartwright, R.I., Sun, P-H. & Ward, A. (1999), Interactive Situation Models for Information Systems Development, Proceedings of SCI'99 and ISAS'99, Orlando, USA, 1999

Beynon, W.M., Ward, A., Maad, S., Wong, A., Rasmequan, S. & Russ, S. (2000a), The Temposcope: a Computer Instrument for the Idealist Timetabler, Research Report 372, Department of Computer Science, University of Warwick, UK

Beynon, W.M., Rasmequan, S. & Russ, S. (2000b), "A New Paradigm for Computer-based Decision Support" submitted to Journal for Decision Support Systems.

Rasmequan, S., Roe, C. & Russ, S. (2000), Strategic Decision Support Systems: An Experience-based Approach, Proceedings of the 18th AISTED Conference on Applied Informatics, Innsbruck

Sauter, V.L. (1997), Decision Support Systems: An Applied Managerial Approach, John Wiley & Sons, New York

Sun, P-H. & Beynon, W.M. (1998), Empirical Modelling: a New Approach to Understanding Requirements, Proc. 11th International Conference on Software Engineering and its Applications, Vol.3, Paris.