Choosing appropriate tools by means of intelligent Decision Support.

John Darzentas, Jenny Darzentas, Thomas Spyrou

Research Laboratory of Samos, Department of Mathematics,
University of the Aegean, GR 83200 Karlovassi, Samos, Greece
email: idarz@aegean.gr

ABSTRACT: *This paper examines an increasingly common problem: that of understanding which tool or tools out of an array is the most appropriate for a problem owner to apply to solve his problem. Several attempts at providing decision support for such cases have been attempted. This paper briefly surveys this work and presents the use of an implemented system which is for use by designers of computer systems who seek to use tools and techniques for improving the usability of the systems they design. Besides aiding the user to choose tools appropriate for helping with his problem, the system has also shown to be useful as a learning system, as well as a transfer mechanism for conveying the results of laboratory research into the real world.*

RESUMÉ: *Cet article traite d'une problèmatique qui devient de plus en plus commun, celle de comprendre quel outil conviendrait le plus à assister à résoudre un problème. Plusieurs essais de faire des SIEDS pour cet problèmatique ont été fait. Cet article résume brièvement ce travail et présente l'utilisation d'un système actuel qui est disponible aux réalisateurs-concepteurs qui cherchent à employer des outils et techniques qui pourrait améliorer l'usabilite des systèmes qu'ils construisent. En plus de son fonction d'aide à choisir outils, ce système s'est montré un moyen efficace pour aider l'utilisateur à apprendre plus sur son problème, et aussi à aider le transfert des résultats de recherches aux monde réel et son marché.*

KEYWORDS: *intelligent decision support systems, technology transfer, human computer interaction*

MOTS CLÉS: *systèmes interactifs aide a la decision intelligents, transfert des résultats de recherches technologiques, dialogue l'homme-machine*

1. Introduction.

In today's predominantly technological world, tools of ever increasing sophistication are constantly being developed to aid users to overcome various problems that beset them. However, very often these tools are not used. Researchers report that users are often confused and overwhelmed by the wealth of information about tools and methods (Mili and Cioch 1990; Basru and Blanning 1994; Darzentas et al., 1995; Favier,. 1995), and find themselves unable to choose what is best for them or to take advantage of them. So widespread is the problem that it is claimed (Fischer et al.,1985; Desmarais *et al.*1993) that even within a software application, such as a spreadsheet or a word processing package, perhaps only 50% of a system's functionality is used. It appears that the speed with which new technology is being introduced; the vastness of the arena; and the requirement to spend time to become a specialist in order to adequately survey the tools on offer, combine to form insurmountable obstacles to actually getting the tools used.

Several attempts have been made to remedy the problem of accessibility of tools to users in various domains. For instance, by defining and cataloguing the use of mathematical models and methods (Mili *et al*, 1990), by representing to the user the structure and components of DSS and functionality and capabilities of those components (Basru and Blanning, 1994), and by software sited within an application, whose aim is to make users aware of more efficient use of a system's functionalities (Desmarais *et al.*).

In this paper, the use of a system called DDAS (Designer's Decision Aiding System) is presented to illustrate another means of overcoming this type of problem of users' accessibility to tools. It is intended for use by designers of computer systems who seek to use tools and techniques for improving the usability of the systems they design.

In a previous paper (Darzentas *et al* 1995) the work that went into the designing of the DDAS architecture based upon the philosophy of soft systems methodology (SSM) and incorporating a reasoning mechanism which uses fuzzy logic has been extensively described.

In the sections that follow, section 2 gives a brief overview of several attempts to provide assistance or decision support for the problems of improving user's access to tools, section 3 discusses the specific problem area of computer systems usability, and the requirements that it raised, while section 4 describes the implementation and the operation of the DDAS. In the concluding section 5, a discussion of the system and its features, in particular those which support generalisability, and the directions of future work are presented. In the appendix which follows, an example session with the DDAS is presented for illustrative purposes.

2. Overview of some related work.

Several pieces of research work can be related to the work presented here in terms of the general problem they tackle. Mili and Cioch (1990) have attempted to construct a common framework within which to place various mathematical methods and models, in terms of what they can do and offer users. The researchers suggest a systematic approach to analysing the relationship that exists between methods and the problems they solve, which involves assessing whether a method is applicable, the scope of its applicability, the extent of its applicability (e.g. what approximations are made and at what cost) the robustness of the applicability (which changes to the problem impact positively or negatively the applicability) etc. Such information, the researchers claim, allows users to select the "right" (sic) method, and more importantly, it allows them to understand the strengths and weaknesses of each method in context. In order to make this information available to the user, the researchers have developed an assessment framework consisting of a documentation template and a procedure used to fill in the template for each association of method to problem.

The work of Basu and Blanning (1994) also takes as starting point the recognition of the inability of users to make efficient use of tools, in their case, models in Decision Support Systems (DSS): "Many DSS have incorporated so many models that the average end user is bewildered and gravitates towards the a few familiar models and remains unaware of other resources" They define the problem of the user as to be inability to ascertain both the range of facilities within DSS that they use, and also, whether or not they have the correct data to "feed" to the models that the DSS makes use of. They suggest a graph theoretic construct, termed metagraphics, to help users make more effective use of the DSS tool by offering graphical visualisation of the structure of a system, and hence offer more insight into the way a system is structured and behaves.

In consultant advisory systems, which borrow from ITS (Intelligent Tutoring Systems), Desmarais et al. (1993) start from the same premise, that there exist tools which users do not have time or capability to learn about, facilities which exist within the very computer programmes and packages they are using, but which users do not avail themselves of. "users of powerful, but complex software packages do not take full advantage of the functionality of their tools." They claim that evidence from scientific literature and from experience suggest that most users would gain considerable efficiency by learning some of the unknown functionality of their software applications. They implemented an intelligent help facility which intervenes when a user performs a task using a roundabout way, and explains to the user how the task can be performed more efficiently.

What all these research reports have in common with the work described here is the same basic problem: accessibility to tools. However, the means of tackling the problem takes very different paths. For Mili and Cioch, the establishing of the framework is seen as the objective of the work; for Basu and Blanning, a means of

visualising the attributes of a system in order to get to grips with it. while for Desmarais et al. working to produce an implementable system, the task of offering advice is complicated by considerations of when to offer advice and also how generalisable their methodology can be.

In the first two examples mentioned, the researchers' vehicle for providing aid is not based upon an automated computer system, whereas as in case of the consultant advisory system, software is added on to an existing system which instructs users in the finer arts of of the software application. Of course the shape and form of the problem space as well as its content influence the shape of the solutions and aids proposed and each come up with a different means to support the user in his task of distinguishing what is appropriate for him, e.g. a framework; a visualisation technique; an intelligent "help" system. However, between cases just cited there are commonalities.

All deal with technological innovations, which are likely to have "overtaken" the user, who, struggling to keep up with latest developments, is aware these tools are useful but needs guidance. Secondly, all are rather unstructured and hard to define areas. It is not a matter of making a taxonomy of tools and leaving it at that because, to begin with, the tools are not easily comparable, but mostly because the user needs to be made aware of how these tools can deal with the problems they face. Thus, thirdly, each approach tries to set out the merits of the tools and then relate them in some way to problems they are meant to deal with so that the user can make a more informed choice. In addition, there is also an added dimension to the overall problem to be taken into account. i.e. new tools often change the very nature of the work they have been designed to help with (Dix *et al.*, 1994) Nowhere is this more apparent than in collaborative technologies used and/or being developed for Computer Supported Co-operative Work (CSCW). Favier (1995) attempted to tackle the problem of helping businesses decide which collaborative technologies are the most appropriate in various asynchronous and distributed work settings, by establishing a list of criteria by which companies could judge which technologies were most appropriate given variables such as groupwares, tasks, people and teams. In longitudinal studies that were conducted in real work contexts, it was found the new technologies imposed new ways of achieving tasks that had not previously been evisaged, and which seriously impact the established notions of tasks and task contexts. To summarise, the wealth of new technologies, and their usage, is a major problem to be confronted, and one which is being tackled it appears in varying manners.

In the next section, the problem domain tackled by the DDAS, and the specific requirements which it imposed is presented, alongwith an attempt to categorise the characteristic features of the system.

3. Meeting the requirements of the specific problem area - using intelligent decision support.

The specific problem domain dealt with by the DDAS system was the problem of getting a specific class of users, designers of software systems, to be aware of the potential power of a range of usability tools and techniques. The HCI (Human Computer Interaction) tools and techniques, which were being developed by a research project (AMODEUS 1989-95) in order to help designers design more usable systems, were felt to be not easily accessible to a wider audience for mainly two interrelated reasons.

Firstly, due to the *nature of the project research results*. The techniques, originating from wide variety of disciplines, (computer science, cognitive science, ergonomics, etc.) consequently differed considerably in approach, in scope, and in degree of formality, and although all touched on the problems of usability in the design of computer systems, some covered certain areas more than others, and were in turn less concerned with some areas than others, some are more predictive than prescriptive, etc. Considerable effort was expended to try to integrate the approaches but the diversity of the approaches and the multidisciplinarity of the research did not lend itself to the establishment of an overarching theory. Thus there was no clear and internally coherent "package" of results to transfer.

The second reason, related to the first, concerned the *nature of the design process*[]. In empirical designer studies that were carried out to try to establish how designers design, it was repeatedly found that strict notions of design processes were not adhered to in practice. The formal software engineering view of design suggested a linear process expressed as:

$$\text{requirements} \Rightarrow \text{design} \Rightarrow \text{specifications} \Rightarrow \text{implementation}$$

had, in reality, relatively little in common with any product design procedures. In real life, the process was one of "muddling through" (Terrins-Rudge and Jorgensen, 1993 and was "complex, variable and disorderly" (Hannigan and Herring,1986).

Thus there was not even a conception common to both designers and the HCI tool and technique developers, of what constitutes design activity, nor for that matter between designers themselves and consequently no well defined place within design activity for HCI usability techniques.

In order to deal with these problems and to transfer the fruits of very valuable research to the design community, several means were used, including workshops and tutorials, and training of graduate and undergraduate students in the use of the techniques and methods. However, it was also felt that the use of some sort of automated aid for helping designers to take advantage of the fruits of this research would be a useful approach, and this was the motivation for the genesis of the DDAS.

Thus the long term objective of DDAS is to help users to make use of sophisticated tools which originate from research laboratories and which are therefore not packaged for use by the lay user, that is, as a transfer activity.

The strength of the DDAS as a means of transfer lies in allowing the user to explore the tools on offer having in mind his own concerns. This is best understood in relation to other means of transfer, for example, instruction or seminar. In these cases, a general introduction and explanation of the strengths and weaknesses of the

tool or technique is provided but this is not (and cannot be expected to be) tailored to the user's needs. It is up to the user to make the link between what is presented and what he is looking for in the way of a tool to deal with his situation of concern. By contrast, the DDAS takes as a starting point the user's problems and from there directs the user to the most appropriate tools. In this way, the user is able to go bypass generalisms and go straight to the subjects which are important to him. In the DDAS, there is no attempt to subsequently explain the use of the tool; the assumption being that the decision support needed was at the higher level of helping the user choose tool or tools from an array which are appropriate for dealing with his situation of concern. As an analogy, the system is a kind of intelligent directory, which first helps the user to define his problem and then recommends the tool or tools which give the best coverage of that problem, rather than an encyclopaedia. The emphasis is on mapping the user problem to the potential of the tools, which is by no means a trivial task. To explain the use of the tool, what skills are needed to use it and so on, was the task of other mechanisms (Buckingham Shum et al,1995) not included in the system.

Although the long term motivation of the DDAS was as a means of transferring research results, the expectation that this effort would focus upon the needs of the designer in relation to these tools was powerful incentive study the vehicle of decision support to deal with the problem space.. The targeted user group of the DDAS, -computer systems designers- asked for assistance in helping them decide which tool or technique was most useful to them. They wanted an advisory system which would be flexible enough to allow them to interact with it quickly, ie without having to give vast amounts of detailed information about their problem and without receiving from the system in-depth complicated reports that they would need time to analyse and discuss. (This effort they wanted to expend on using the tool, not on preliminary steps of deciding which tool to use). Furthermore, these were users who were familiar with technology and wary of any "black-box" which might "compute" them a recommendation based upon hard quantitative data about constraints and criteria. Their needs were for something that was transparent and which allowed "soft" qualitative knowledge and beliefs to be taken into account in the decision-making process.

The DDAS meets these requirements by being based upon a "co-operation" paradigm in which both the system and user are equals. The user is not treated as a non-expert and the emphasis is upon aid and support rather than trying to make the decisions. This has the effect of making the computer an *active* and *intelligent*, as well as a useful, partner in the decision making process. It becomes an interactive tool without the system dominating and the user worrying about loss of control, or about making definitive choices too early on in the interaction. The two adjectives "intelligent" and "active" that have been applied to the new generations of DSSs are used to denote notions encountered in the co-operation paradigm, as well as more specific definitions. For instance, Intelligent DSS have been described as systems which use expert system techniques and/or knowledge bases. The DDAS possesses a knowledge base which is made up of rules describing the potential of the tools to

deal with different problems. The set of problems is itself made up from those problems which the tool developers claim that the tools solve, or at least go some way to solving.

In addition, the reasoning and justification components which use test score semantics technique based upon fuzzy logic (Zadeh 1989, Darzentas 1995) contribute to the intelligence of the system. It has been claimed (Gottinger and Weimann, 1992) that most AI systems are not well equipped to handle small differences in outcomes on a variety of attributes which may affect decision making. For example, most planning systems are based upon developing a plan which can be proven to achieve a specific goal. In the DDAS, the user is presented with the problem descriptions mentioned above arranged in hierarchical groupings which are consistent with the commonly held beliefs about the way they are viewed in the domain[rich picture paper]. The user is able to browse the descriptions, marking with varying degrees of importance those which interest him This enables him to reflect the small preferences/ beliefs which distinguish him as an individual with his own view of the problem situation. He can also review his choices and change them at will. The fuzzy reasoning component of the system is sensitive to such changes and computes the outcome of the variables in such a way as to reflect them and the strength of belief of the problem owner in the importance of the chosen descriptions.

Active DSS are best understood in contrast to passive DSS, systems which tended to produce mostly hard data based results and leave the user to interpret them and thus take decisions based upon them, whereas an active DSS takes more initiative and intervenes more in the actual process of decision making with recommendations and advice. In the DDAS, the user makes selections and choices, but the system intervenes with a comments on these, if they appear unusual to it. However, the user has the last word, in such a way that his beliefs are those which the system eventually computes.

The next section describes in more detail the way a user interacts with the system

4. Implementation and the operation of the DDAS

Implementation was carried out using CLIPS, an expert system environment developed by NASA and HARDY, a hypertext based diagram editor for X-windows and windows 3.1. developed by AIAI of the University of Edinburgh.[] Using CLIPS enabled the use of a logic based programming environment needed for the task of manipulating qualitative knowledge, combined with required expert systems features. The use of HARDY enabled the utilisation of visualisation techniques to interact with the user, making use of graphics and labelled nodes within networks.

The interaction with the user is based upon two types of presentation elements: the graphic display of the *problem descriptions*, and the *commands* that manipulate the interaction.

The *problem descriptions* are displayed in the form of labelled shapes and are laid out in a series of screens browsable by the user. Two types of shapes are used,

one to represent the fact that there exist more specific problem descriptions in the knowledge base, while the other shape represents the most specific expression of a subproblem contained in the knowledge base. Shapes may be linked by arcs which denote different types of relationships existing between problem descriptions, for example, green arcs represent "high possible concurrency" and blue arcs, "low possible concurrency".

Commands are displayed as buttons on a toolbar which is permanently on screen. These commands aid the user to choose amongst the available facilities of the system, for example the facility of moving to diagrams/screens that correspond to different levels of analysis is performed by double arrow buttons. (Fig 1)

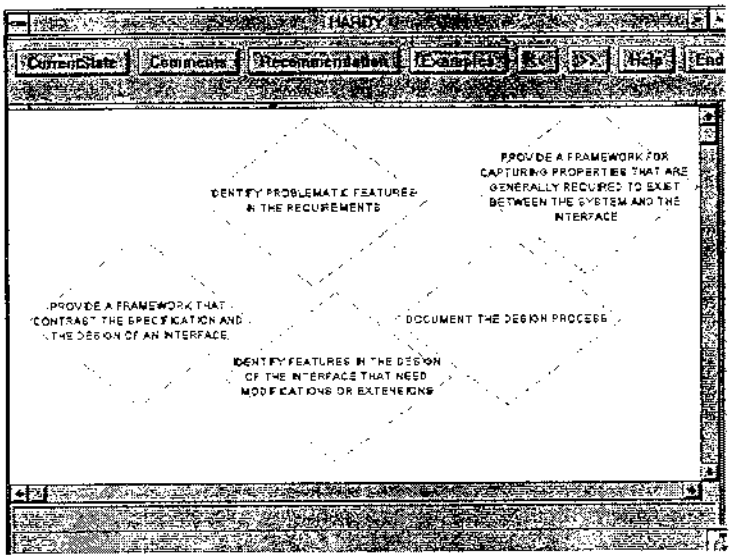

Figure 1. *A snapshot showing some problem descriptions (general level) and command buttons*

During the interaction the problem descriptions are displayed to the user, so that he can search for and identify those that he considers as most relevant to his problem. The objective is for him to make a selection of these relevant descriptions as a way of expressing his situation of concern. Whilst selecting, the user can also specify the degree of relevance of the descriptions to his problem, and should he change his mind, he can unselect any thing he has already chosen. Each time he clicks on a problem description, its colour changes. Each colour shows the degree of importance of the specific problem descriptions to the user. In order to guide the user through the network of problem descriptions, these are presented to him at

various levels of detail. It is possible for the user to go backwards and forwards between screens.

The user can ask for comments from the system about the set of the problem descriptions he has chosen so far. The comments are based on the relationships of the problem descriptions that exist in the knowledge base. For instance, the user who has chosen both of the problem descriptions that are parts of a "low-possible-concurrency" relationship, is warned that these problems are not usually concurrent. The user can ignore the warning messages, but if he wants to follow the advice given, he may decide how he wants to solve the implications, by either selecting and unselecting accordingly.

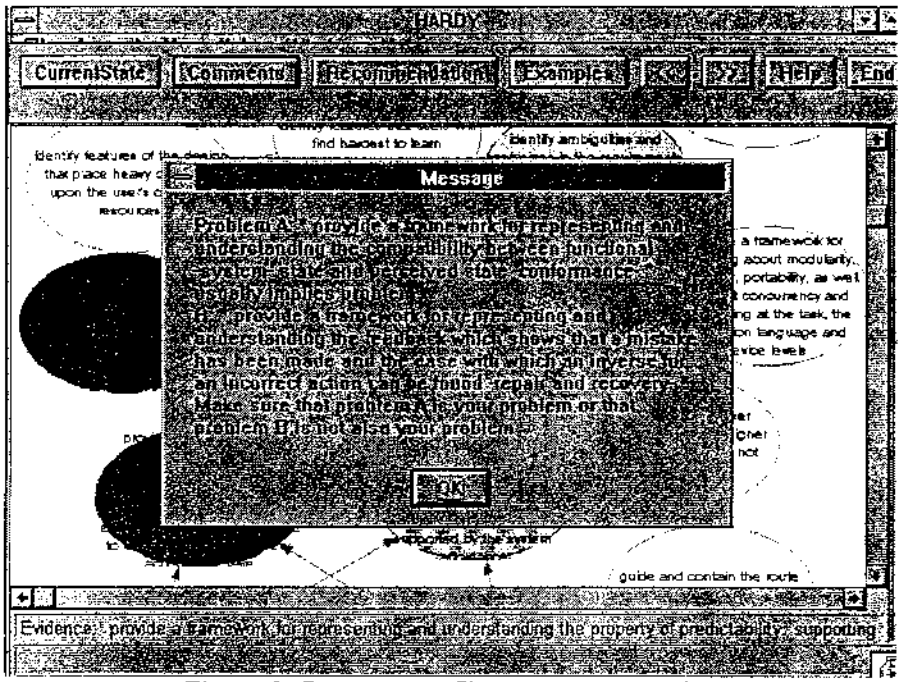

Figure 2. *Comments on Choices message window*

A further facility available at any time is that of providing a formatted text description of the set of problem descriptions chosen. The relationships that exist in the knowledge base form the basis for the text description of the chosen problems. This helps the user to recapitulate what selections he has made. (Current state)

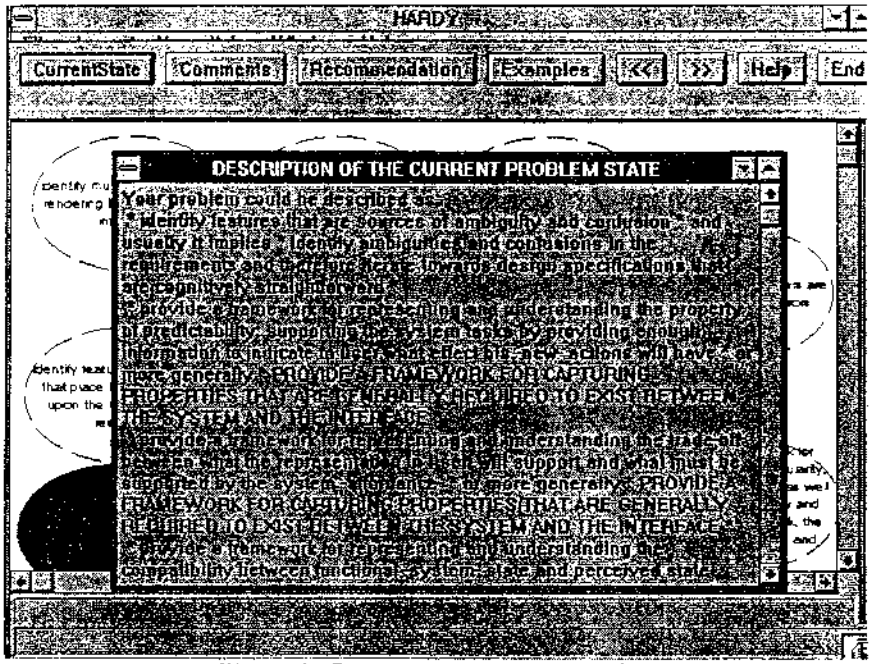

Figure 3. *Current state message window*

Should the user want an illustration of a particular problem description, he can obtain examples of use. This feature can be useful in helping the user decide about how close (if at all) the specific subproblem description is to his own particular problem.

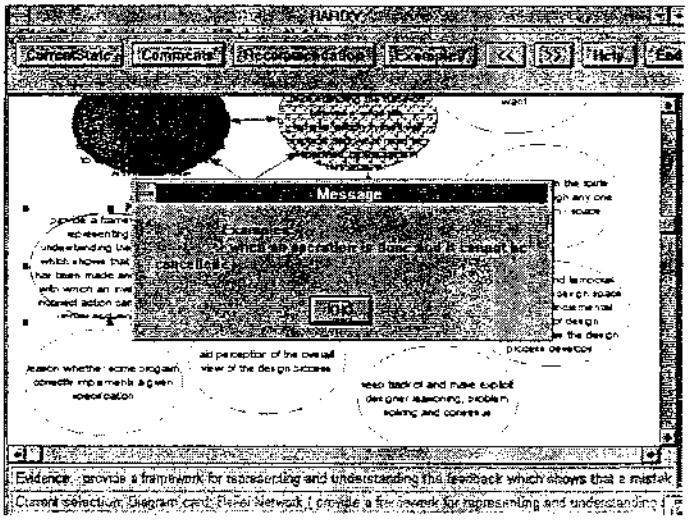

Figure 4. *Message window with the available example(s) for the chosen specific subproblem*

When the designer-user feels that the problem descriptions he has chosen describe his situation adequately, he can request a recommendation from the system. The recommendation is given as formatted text which recommends to the user the most appropriate technique(s). The reasoning behind this recommendation, based upon fuzzy logic, is also given in the formatted text, in order to give the user the justification of the rationale behind the recommendation. The compensation oriented score operator from test score semantics (Zadeh, 1989 Darzentas 1996b) is used to compute the recommendation. For its computation the quantifier value (linguistic or otherwise) that specifies how important each chosen problem description is to the user and the quantifier values of how well the modelling techniques satisfy each chosen problem description are used.

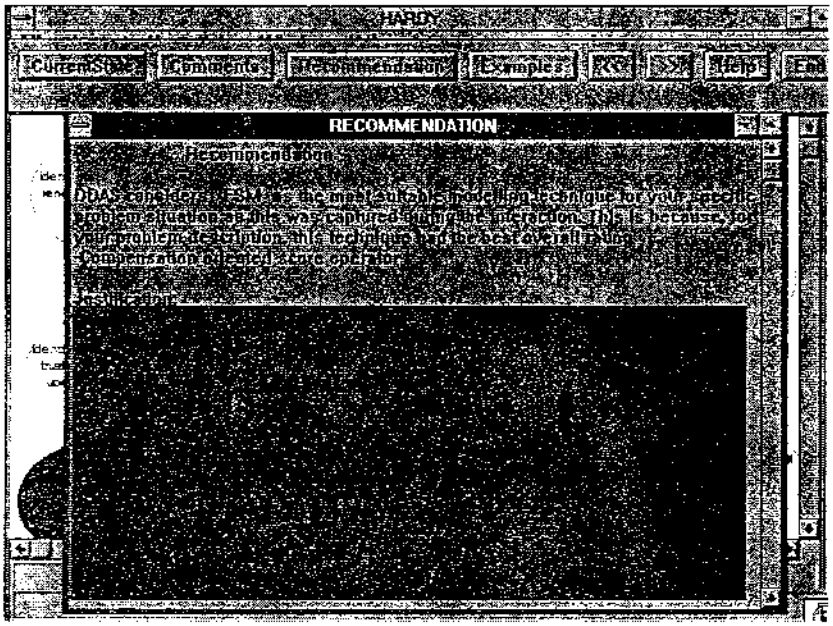

Figure 5. *Recommendation message window*

The user is then able to accept the recommendation, or to begin again the interaction, varying colours and choices to see if these substantially affect the outcome. This type of exploratory interaction, is further commented on in the next section.

An example detailing a designer's specific problem and how he can handle it using DDAS illustrates some of the system's capabilities and is given in the appendix.

5. Conclusions: Discussion and work in progress

The system described in this paper has as primary objective to aid users to discover which tools are available and suitable for dealing with a situation of concern. As has been discussed above, this is by no means a trivial problem, as the plethora of tools available simply overwhelms, in most cases, the problem owner, not to mention the struggle to keep up the ever quickening pace of innovation, as last year's tools are outmoded by this year's.

The system presented here uses a methodology which, it is believed, is generalisable to many other situations of modern day life, where problems exist, and tools to tackle them too, but they co-exist in separate worlds. This paper presented a system as an illustration of what could be done to help bring research results out of the laboratory and into the market. It is intended for use by designers of computer systems and the tools on offer have in common the notion of usability, from the differing, but often overlapping, viewpoints of system, task and user.

It was noted in trials that, by and large, the system met this first objective of assisting users to understand which tool(s) is appropriate for them to use to deal with their problem.

In addition, the use of the system assists the users, in this case, mostly human factors experts and software engineers, to deepen their understanding of the domain, by helping them to see the linkages and connections amongst the elements of the problem situation and between these and the tools available. This role of the system as a tool for *learning* is considered an important by-product especially when considered against the backdrop of the paradigm of life long learning, where the ability of organisations to become active learning organisations is increasingly seen as one of the keys to a firm's success (Isaacs and Senge 1992; March 1993).

Thirdly, as has already been noted elsewhere (Darzentas et al. 1996a), systems like these, i.e. tool based approaches to conveying knowledge from one centre of activity to another, perform a very important function in helping in the *transfer* of research results from the laboratory to the real world.

This paper discussed a system which has the means of associating problems to potential of tools, a means of evaluation users' problems, an interface which promotes visualisation and interactive co-operation. It is strongly believed that this system could be generalised to other domains where the problem of accessibility to tools is paramount.

Currently, the system is being applied to three other ranges of tools, namely: computer languages, OR techniques and homeopathic medicine. Each of these domains represents an area where the task of mapping tools, problem solving methods and remedies onto ill-defined problems represents a task where the power of the features of the system described above: namely, the manner of organisation of the knowledge base and the fuzzy logic based inference and evaluation mechanism could offer substantial aid.

To validate the use of these systems, the recommendations given by the systems are being compared to those given by the experts who are not involved at all in

system development or knowledge acquisition. Further, comparisons between the reasoning on available facts by experts and system are being made. Lastly, comparisons between the domains will be made to test for generalisability of the methodology leading to the decision aid.

Acknowledgements: *This work was funded by the ESPRIT Basic Research Action 7040 AMODEUS (Assaying Means of Design Expressions for Users and Systems)*

6. References:

AMODEUS, 1994 : ESPRIT Basic Research Action 3066, 1989-1992. AMODEUS (Assimilating Models of Designers Users and Systems) and ESPRIT Basic Research Action 7040 AMODEUS II 1992-1995 (Assaying Means of Design Expressions for Users and Systems) Documentation available by anonymous ftp (ftp.mrc-apu.cam.ac.uk).

BASU, A. BLANNING, R.W. 1994 Metagraphs: A tool for modelling decision support systems *Management Science* 40, pp.1579-1600,

BUCKINGHAM SHUM S., JORGENSEN A.H., HAMMOND N. AND ABOULAFIA A.(Eds), 1994 : «Amodeus-2 HCI Modelling and Design approaches: Executive Summaries and Worked Examples , *Amodeus Project Document: TA/WP16.*

DARZENTAS, J., DARZENTAS, J.S., SPYROU, T. 1995 DDAS: A Designer's Decision Aiding System, *Journal of Decision Systems*, Vol 4, no.1, pp 9-22

DARZENTAS, J., DARZENTAS, J.S., SPYROU, T. 1996a. « Helping the user to make use of the tools- Transferring Research Results from the Laboratory to the Market». *Proceedings of the Human Comfort and Security Workshop*, Brussels 26th October. Springer Verlag, in press.

DARZENTAS, J. SPYROU, T. TSAGARIS, CH.: 1996b «Evaluating Option-related Text Descriptions for Decision Aiding» in *Perspectives on DSS* eds. Darzentas, Darzentas and Spyrou, University of the Aegean Press (In press)

DESMARAIS, M.C., GIROUX, L., LAROCHELLE, S. 1993 An advice-giving interface based on plan-recognition and user-knowledge assessment. *Int. J. Man-Machine Studies* 39, pp 901-924

DIX, A. FINLAY, J. ABOWD, G. BEALE, R. 1993 Human-Computer Interaction Prentice Hall

FISCHER, G. LEMKE, A.C. AND SCHWAB, T 1985 «Knowledge based Help Systems». Human Factors in Computing Systems *CHI85 Conference Proceedings*, ACM pp 161-167

FAVIER, M. 1995 Asynchronous and distributed work: which collaborative technology to select? *Journal of Decision Systems*, Vol 4, no.1 pp 63-78.

FISCHER, G MASTAGLIO, T. 1993 «Computer-Based Critics» in *Current Research In Decision Support Technology* eds R.W. Blanning and R.King IEEE Press pp147-156

GOTTINGER, H.W. WEIMANN, P. Intelligent Decision Support Systems, *in Decisions Support Systems* 8 1992 pp317-332

HANNIGAN H, HERRING V., 1986 : *The role of Human Factors Inputs to Design Cycles*, Deliverable A1.2b, HUFIT CODE, HUFIT/8-HUS-11/86.

ISAACS, W. SENGE, P. 1992 Overcoming limits to learning in computer based learining environments *Journal of Operational Research*.Vol pp.183-196 .

MARCH, J.G. 1991. Exploration and Exploitation in Organisational Learning. *Organisation Science*, Vol 2, no.1, pp.71-87.

MILI, F. CIOCH, F.A. 1990 «Documenting Decision Models for Informed and Confident Decisions». *Proceedings of 23rd Hawaii International Conference on Systems Science IEEE*, pp 494-503

TERRINS-RUDGE D., JORGENSEN A.H., 1993 : «Supporting the Designers, Reaching the Users» in: *Computers, Communication and Usability: Design Issues, research and methods for integrated services* Byerly P.F., Barnard P.J., May J. Eds. Elsevier, pp. 87-98.

ZADEH .L.A., 1989 : « Knowledge Representation in Fuzzy Logic », *IEEE Transactions on Knowledge and Data Engng ? n° 1*, pp. 89-100.

7. Appendix : an example session

To illustrate some of the system's capabilities, an example detailing a designer's specific problem and how he can handle it using DDAS follows. In this example, a designer's concern is that interface users are often confused by the outcome of clicking a button X. e.g. there can be two different results of clicking the same button X in two different contexts respectively.

The designer wants to resolve this problem. For the sake of the example, it is assumed that he wants the solution to enable the users of the interface to distinguish clearly what the corresponding effects on the system are when a button is pressed. In addition he also wants to check that this problem of the design of the interface does not start from a confusion in the requirements.

The designer is firstly presented with a diagram which uses rhombii to represent the most general subproblem descriptions, such as that given in Fig 1. The designer searches through the diagram for labels which come closest to expressing his problem. In this case, he chooses the rhombii with the following labels and assigns to them a degree of relevance:

- *IDENTIFY FEATURES IN THE DESIGN OF THE INTERFACE THAT NEED MODIFICATIONS OR EXTENSIONS* (red)
- *IDENTIFY PROBLEMATIC FEATURES IN THE REQUIREMENTS* (yellow)
- *PROVIDE A FRAMEWORK FOR CAPTURING PROPERTIES THAT ARE GENERALLY REQUIRED TO EXIST BETWEEN THE SYSTEM AND THE INTERFACE* (magenta)

The designer presses the button with the label >> in order to move to the next diagram with the more specific subproblem descriptions. He is then presented with a diagram which uses circles and arcs to represent the possible subproblem descriptions and the relationships between them, such as that given in Fig 6.

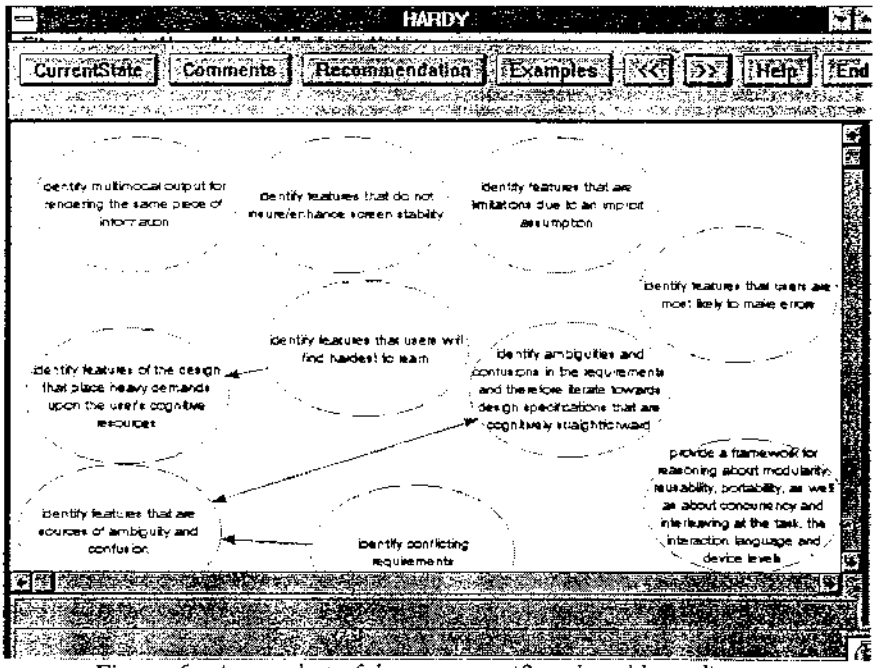

Figure 6 . *A snapshot of the most specific subproblems diagram*

The designer searches further through the network diagrams for labels which come closest to expressing his problem. In this case, he chooses the circles with the following labels and assigns to them a degree of relevance (colour) as below:

- *identify features that are sources of ambiguity and confusion* (red)
- *identify ambiguities and confusions in the requirements and therefore iterate towards design specifications that are cognitively straightforward* (yellow)
- *provide a framework for representing and understanding the compatibility between functional (system) state and perceived state (conformance)* (magenta)
- *provide a framework for representing and understanding the trade-off between what the representation in itself will support and what must be supported by the system (affordance)* (turquoise)
- *provide a framework for representing and understanding the property of predictability: supporting the system tasks by providing enough information to indicate to user what effect his new actions will have* (magenta)
-

Before going on to choose some more subproblem descriptions from the DDAS diagram, the designer would like to have a commentary from the system about his choices. He clicks on the «COMMENTS ON CHOICES» grey button. This advice is given in a message window .

In this particular case the displayed message comments that according to the system, the subproblem description «*provide a framework for representing and understanding the compatibility between functional (system) state and perceived*

state (conformance)» usually implies the one with the label *«provide a framework for representing and understanding the feedback which shows that a mistake has been made and the ease with which an inverse for an incorrect action can be found (repair and recovery)»* and therefore the second could also be chosen.

The designer can shift-left-click on a subproblem in order to see the available examples (if any) of the specific subproblem. The examples help him understand some characteristic situations that the subproblem should be chosen. The examples are given in a message window.

Each time the designer wants to see a text description of the chosen subproblem he clicks on the «CURRENT STATE» grey button. A window appears with formatted text which consists of sentences that contain either one selected subproblems description or two selected subproblems that are related with a type of relationship expressed in words.

In this way, the system, utilising its knowledge of the design space, and subproblems associated with it, prompts the user and aids him to consider subproblem descriptions which may be relevant to his problem of concern and which he has not chosen. The user considers the system's advice and is free to reject it should he not think it relevant.

Otherwise, the system highlights the subproblems mentioned with a black outline (Figure 7) to help the user find the subproblems that the message refers to.

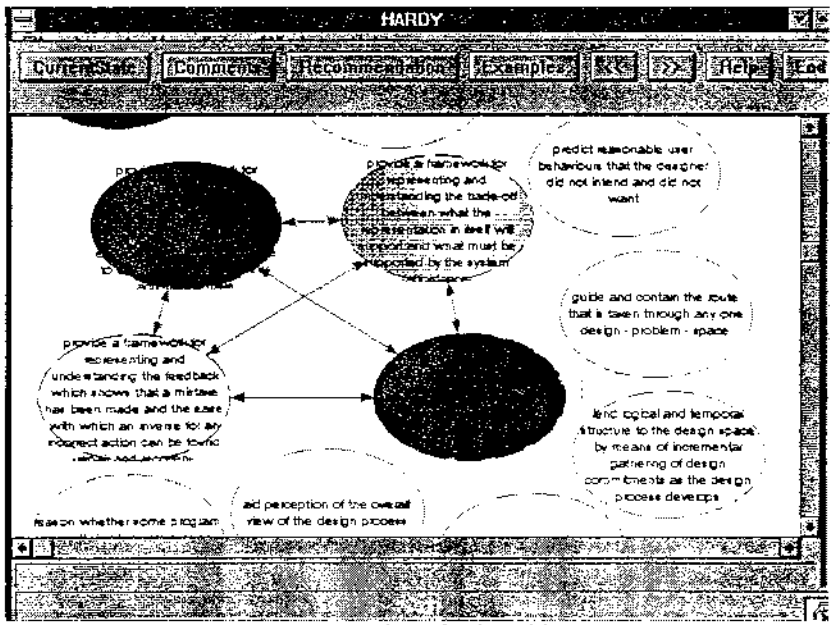

Figure 7. *Selected subproblems*

The user continues in this way, making selections, reading the comments on current choices and reselecting until he is satisfies with what the current selection

represents. During this cycle he can get at any time a text description of the current state.

When the designer is satisfied that he has a final set of chosen subproblem descriptions (i.e. he doesn't want to choose any more subproblem descriptions by clicking on them and that he doesn't want to change his belief about the importance he gave to the selected subproblem descriptions, by changing their colour), he then clicks on the «Recommendation» button to get a recommendation about the most appropriate modelling technique(s) for his problem. A window appears with the recommendation.

The user can accept this recommendation and search to find out about the method recommended, or he can choose to go through the cycle again, making differing choices and assigning different degrees of belief.