PROTEUS: AN APPROACH TO INTERFACE EVALUATION

Jonathan Crellin.

People and Computer Interaction Systems Research Group (PACIS), The Computing Department, Faculty of Mathematics, The Open University, Walton Hall, MILTON KEYNES, MK7 6AA. United Kingdom.

PROTEUS is a number of software tools which allow the implementation of an iterative, user centred approach to software (particularly interface) design, using rapid prototyping. The tools allow automated collection of questionnaire data, logging of system usage, and the central technique which is the collection of a qualitative representation of users perception of an interface design space, using the Construct Elicitation System. This data is fed back to the designer, and increases understanding user needs in relation to an interface. The paper describes the development of PROTEUS as an integrated evaluation tool, and reports on some of the empirical work underlying the approach embodied by PROTEUS, including its integration into the design of a small but real system.

1: INTRODUCTION

1.1. The failure of analytic approaches

It is possible to distinguish between interface evaluation methods, and interface evaluation approaches. Approaches is used here to describe the underlying philosophy of the evaluation. Methods are the techniques of data collection employed. Different approaches may use the same methods of data collection, but analyse the data in different ways. Formal approaches focus on the structure of the interface, within the context of human cognition models. They evaluate features of interface such as complexity and consistency, Grudin (1989). Such methods frequently fail to predict the future success of interfaces.

1.2. Problems with Empirical Approaches

Empirical approaches use an experimental or controlled observational paradigm. They involve evaluating performance of users with the interface on bench mark tasks aiming to predict how effective the interface will prove to be in real use. External variables (such as interruption) are controlled, and clear behavioural performance measures for performance on the interface are stated before evaluation takes place. However such evaluation approaches still fail to predict performance in real situations.

1.3. Contextual Approach

Whiteside et al (1988) suggests a contextual approach to interface evaluation. Such an approach collects experience as it happens, in the context where it usually occurs. This approach seeks to understand the experience of using an interface, as it is used in a real situation.

1.4. The problems of contextual methodology

The methods employed by contextual approaches are usually video observation, verbal protocols, and software logging. Unfortunately the first two methods for collecting experience are usually highly intrusive, and likely to alter the very phenomena which one wishes to observe.

1.5. Knowledge elicitation techniques as a basis for evaluation.

Techniques used for knowledge elicitation for expert systems may offer some useful methods for use within software evaluation, Briggs (1987). If such techniques are used it remains necessary to distinguishing between an objective description of an interface, and the subjective experience of using the interface. The knowledge elicitation approach can be used to identify users mistaken assumptions about an interface. However what does mistaken actually mean in this context? It could be argued that user mistakes are actually due to designers poor conception of a users viewpoint. Such users mistakes highlight a difference in designer and user perspectives.

1.6. Designer and Users

Designers and users have distinct differences. Designers see systems as a whole, they see the source code which makes up the system, and by the nature of their profession they spend a lot of time considering the link between the underlying system and the interface. Users are only directly aware of a systems interface, and knowledge of the underlying system is inferred from their experience with the interface. There are also differences which stem from a different knowledge of what is possible within the technology. The lack of knowledge of what is possible can make users ask for things that are difficult to achieve (for example a natural language interface), but at the same time, that lack of knowledge can mean users do not think of possible alternative ways of doing things (for example considering using pull-down menus rather than buttons).

Involvement of users in the design process is usually considered a 'good' thing, as it helps avoid the most obvious design trap, of ending up with a design that is considered at best uncomfortable or unpleasant to use, or at worst simply unusable. The term 'user-centred design' has been used to describe a range of design methodologies. At one end of the range of user centredness this can be a fairly small amount of evaluation of the software as it is used by a number of typical users. The evaluation can often involve looking at a feature of the interaction which the designer has decided (in the light of psychological/ergonomic guidelines) to be relevant to the performance of the system. At the other end of the spectrum, the user can design the majority of the system himself, by using some type of application-builder. The dimension here is one of emphasis on a designers

design skills. At one extreme the responsibility for identifying the important features of the design lie with the designer, at the other end the responsibility is entirely placed on the user. One can see advantages with both approaches. Designers are expected to have knowledge which transcends a particular domain, and thus can see things about a design which may not be apparent to users-as-designers. If designers have any value this is what their value must be. Marcus(1983) refers to three perspectives on design.First the outerface, which are the final products of computation, text, tables, graphics. All of these can be printed, projected, or can appear on a VDU. People who use this information need have very little real understanding of how computers work. Second the interface which is the frames for command/control and documentation, that the computer system user encounters. This human computer connection allows the human to manipulate and control the machine, without which the computer is a useless tool. Thirdly the innerface which is the frames of command/control and documentation that are avaialble to the designer and only he sees. They depict programming languages, software tools and operating systems. This special perspective that only a designer has can lead to enhanced control over the design space but also misunderstanding of user perspectives.

System designers are used to dealing with computer system concepts, and use language to express those concepts. Users language is usually specialised to cope with their own task domain. These specialised languages form a signifigant barrier to communication between users and designers.

1.7. An ideal method for evaluation

An ideal method for interface evaluation will have the following characteristics:-

* Scope for automation
* User centred
* Not intrusive into the experience of using an interface
* Supports communication between users and designers.
* Quantitative and qualitative data.
* Ecological validity supported

The repertory grid, which is used as a knowledge elicitation tool, is an obvious candidate for software evaluation. The method has been used for the collection of evaluative data in a number of different domains, architecture, Honikman (1976), marketing Stewart et al (1981). The Construct Elicitation System (CES) has been developed at the Open University for collection of repertory grid information in evaluation tasks, Crellin (1988). The system has also been used in an investigation of the meaning of a researchers abstraction, Robson and Crellin (1988).

2. DESCRIPTION OF PROTEUS

PROTEUS is not designed to replace existing methods of evaluating usability and performance, but to provide a parallel analysis of user centred issues, and an input into the design decision making. It is a tool designed to aid the social process of designing a piece of software. The social process is one of extending the mutual understanding between the designer and the user. As such the tool is best suited to circumstances where a designer is not very experienced, and lacks the ability to gain a users perspective, or where the perspectives of designer and user might be expected to be radically different.

The components of PROTEUS are a shell, the different software prototypes, a help system, and the Construct Elicitation System. The methodology of PROTEUS is to allow users to use the prototype systems in as near as possible an ecologically valid environment. The system is therefore robust, and portable, and can be used in a normal workplace. PROTEUS can be used on a minimal Macintosh system (Mac Plus with single internal drive), and is consistent with the Macintosh interface guidelines. It therefore requires no special equipement, and can be used in a normal working environment by any Macintosh user. Usage data is recorded unobtrusively by the system during use, additional system logging (for example keystroke level recording) can be recorded by code embedded in the prototype systems. Data elicited by the Construct Elicitation System is available for user editing, during the session. The aim of the approach is to collect the maximum of information about user behaviour and experience, without intruding on the experience of using the prototype systems. The user should feel comfortable in the use of the system, and not feel themselves to be the object of unwanted scrutiny.

The PROTEUS shell presents an on-line questionnaire to collect demographic data. Access to the Construct Elicitation System is controlled via the shell. A user can only start using CES when she has used all the prototypes.

The Construct Elicitation System is an open ended description and rating system, based on the Personal Construct Psychology repertory grid Kelly (1955). Users are asked to provide textual labels for the ways in which they discriminate between the different prototypes. These verbal labels form the basis of semantic differential grids, which provide information both about the similarities between different ways of rating, and between the different prototypes.

3. PROTEUS AND THE RAS HELP SYSTEM

3.1. Introduction: The RAS System

RAS is a demonstration relational database program to be used as a teaching resource in the Open University's undergraduate program for 1990. Its primary role is to demonstrate the structure of a relational database, and to introduce the interrogation of relational databases using Structured Query Language (SQL).

Although most of the system was specified by teaching requirements, the HELP system had not been finalised. The role of a HELP system in a teaching package is to assist the student in completing their exercises, but not to provide a substitute to the course material, through which primary teaching takes place, or to distract learners from the course material.

3.2. Subjects

Subjects were of varying backgrounds. Some were post-graduate students working in the Faculty of Mathematics (n=4), others were Open University technical and research staff (n=2), and tutors from the Database course (n=4). All subjects had prior experience of micro computers. Only the tutors on the database course had more than a slight knowledge of SQL and relational databases. Additionally the system's designer completed the task.

3.3. Materials

The experimental materials consisted of two floppy discs. A 3.5 inch Macintosh boot-up disc, which contained the PROTEUS shell, including the CES program, and a number of prototype help systems for the RAS teaching material. Each of the prototype help systems were implemented on the Mac, running in a command language environment which was very similar to the MSDos RAS environment. Only the help commands were implemented on the Mac versions of RAS, other functional aspects of the RAS system were not implemented, and returned a short message to that effect if the user attempted to type them on the Mac version of the RAS system. Three distinct HELP systems were devised, and one more system was added by combing elements of the others. The HELP systems provided were:-

Declarative Textual Help, describing the purpose and effects of the commands in detail, but not providing information about how to implement the command.

Exemplar Help, which consisted of a valid example of the command (executable on the RAS system), and an example of the reply RAS would deliver if such a command was executed.

Syntax Help, which provided a BNF syntax diagram of the command.

Finally, *Declarative and Syntax*, which combined the contents of Declarative and Syntax above.

Each of the Help systems is syntactically identical, having a simple structure of the HELP command and a single argument. The screen display of each system is also very similar, resembling a message output from an IBM PC.

Subjects were also given a 5.25 inch MSDos floppy disc which contained the RAS program, and MSDos system files. In addition to the floppy discs subjects were given copies of the relevant OU teaching material. This consisted of an audio tape containing a guided tour of the RAS program, and a draft copy of the written OU course material . The course material introduces the RAS system and teaches interrogation of the database using SQL.

Subjects who completed the study in the computing laboratory had available a twin drive Amstrad 1512 PC, a Mac Plus computer, and a tape cassette player equipped with headphones. This equipment was located in a section of a computing laboratory partitioned off from the 'public' area, and was available for use anytime during the test period.

3.3. Procedure

Two procedures were adopted. The first was closer to a laboratory procedure, with subjects working in a variable number of sessions on the laboratory Macintosh and Amstrad PC away from their normal workplace. The audio material was available to these subjects and was listened to through headphones. A draft copy of the database teaching material was also available. The sessions took place in a screened off section of a computing laboratory.

The second procedure was more ecologically valid, and involved giving the subjects discs of the materials, and allowing them to work on them in their normal place of work, using familiar equipped. The return rate from this procedure, was however much lower. Data from this group is still being analysed and is not reported in this paper.

Both groups were given the experimental materials and were asked to listen to the audio tape (introducing the RAS system). They then had to read the extract of OU teaching material, and try the exercises in using SQL. When they needed help on the RAS system they were instructed to use the help systems running on the Macintosh, although the exercises themselves were completed on the Amstrad PC version of the RAS system.

On booting the Macintosh, demographic data was collected from subjects by a computer delivered questionnaire. After this each of the prototype HELP systems were available to subjects, accessed by clicking the appropriate button on the Mac screen. A graphic indication of the time spent on each prototype HELP system was given by the gradual filling in a pie chart next to the appropriate button. Additional help on the experimental procedure was given to subjects on screen, accessed from a pull down menu. Finally one more button was available on the screen, allowing access to the Construct Elicitation System. This button could not be selected until all the prototype HELP systems had been tried.

After completing the RAS exercises (which took up to one hour to complete) subjects went on to the Construct Elicitation task. Before they started generating constructs they were asked to rate the prototype HELP systems on a ten point semantic differential scale, with poles Unpleasant to Use, and Pleasant to Use. The Construct Elicitation System was loaded and the process of triadic elicitation started. The CES uses both an open ended way of asking questions about what characteristics of individual interfaces make them distinct from the other interfaces (triadic elicitation) and an ethnographic method (implication laddering) for gathering correlates of the elicited constructs. The elicited constructs form the poles of bi-polar semantic rating scales, on which all the interfaces are evaluated. Feedback of the data is given to subjects as an aid to further construing. Finally the subject is asked to give an extended text description of each construct.

When subjects have finished generating constructs they can leave CES and then quit from the PROTEUS shell. If at any stage a subject wants to refresh her mind about the prototype HELP systems, she can leave CES and view the particular prototype HELP system, then return to CES. When subjects leave CES for the last time they are again asked to rate the prototype HELP systems on the ten point semantic differential scale. These ratings are currently used to estimate if the process of construing alters how the interfaces are perceived, but also provides a mean subjective usability rating for each interface by each subject.

The users actions inside PROTEUS are monitored. When a stimulus interface is called logging is suspended unless supported by code inside the prototype. It is possible for

subjects to leave the RAS shell, and continue the evaluation task at another time. Such breaks are also logged

3.5. Data

Subjects spent between two hours seven minutes and four hours forty seven minutes on the task, and completed the task in between one and three sessions. Between five and ten constructs were elicited from each subject.

3.6. Analysis of Data

A review of the data is presented in table 1. This shows the quantitative data from the experiment.

Table 1: Review of Quantitative Data collected by PROTEUS.

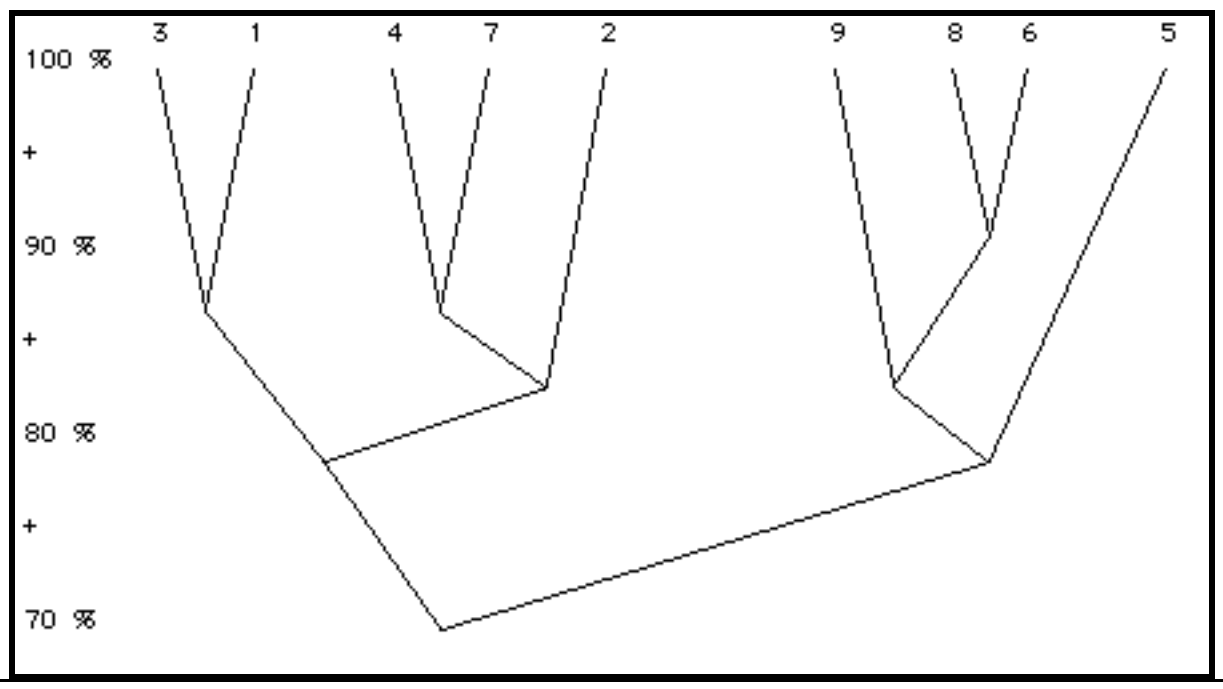
	INTERFACES			
	Syntax Help	Textual Help	Exemplar Help	Syntax and Text Help
Ratings Data (1=unpleasant, 10=pleasant) (ranking).	1=7(2) 2=4(2.5) 3=4.5(2.5) 4=1(4) 5=7.5(3) 6=3.5(4) 7=2.5(3) average=4.3(3)	1=8(1) 2=4(2.5) 3=3(4) 4=2(3) 5=6(4) 6=5(2) 7=1(4) average=4.2(2.9)	1=3.5(3.5) 2=4(2.5) 3=7(1) 4=8(2) 5=9(1.5) 6=4.5(3) 7=6.5(1) average=6(2)	1=3.5(3.5) 2=4(2.5) 3=4.5(2.5) 4=9(1) 5=9(1.5) 6=7(1) 7=6(2) average=6.2(2)
Number of Uses	1=1 2=2 3=3 4=3 5=6 6=6 7=3 average=3.4	1=3 2=5 3=3 4=4 5=6 6=5 7=3 average=4.14	1=2 2=3 3=2 4=3 5=9 6=2 7=2 average=3.28	1=2 2=4 3=2 4=3 5=2 6=6 7=2 average=3
Total Usage (seconds)	1=2651 2=1261 3=2014 4=321 5=327 6=404 7=305 average=1041	1=294 2=183 3=309 4=173 5=393 6=532 7=176 average=294	1=819 2=816 3=1532 4=142 5=852 6=653 7=244 average=723	1=315 2=223 3=109 4=154 5=386 6=4626 7=191 average=858

The quantitative data shows that subjects used the different help systems between three and four times during the study. Total usage times are probably less informative in this study than number of uses, since users spent time doing the RAS learning task whilst a particular help was running. Number of uses does not vary significantly between the different interfaces. The ratings data does give a clear indication of user preferences regarding the different systems. The Syntax and Text help and the Exemplar Help systems are rated higher than the Syntax on its own or the Text on its own. This observation is reflected in the CES data of most subjects, with the exemplar and text and syntax help systems matching closely on six of the seven element trees.

The data from the Construct Elicitation System has been analysed using the FOCUS algorithm, Shaw (1980), Jankowicz and Thomas (1982). Results from the analysis are displayed as two binary trees per subject. The binary trees show the similarity matchings for the constructs elicited from subjects, and for the interfaces as the subject saw them. This method of displaying the construct data makes certain features more apparent. It is possible to see which interfaces are seen as very similar, and which interfaces are seen as quite different. It is also possible to compare individuals and see if those similarities are common to most people or unique to individuals.

In this study the element trees of most subjects showed a high level of correlation between the Exemplar and Text&Syntax help. This supports the view that the CES data is reflecting perceived usability, as intuitively the Text and Syntax help system would be correlated more closely with the either of the other two systems, which are identical in part. The CES data appears to reflect perception of the system as a whole, rather than simply the physical characteristics of the systems.

Construct Labels and Construct Tree for Subject 2



CONSTRUCTS:-

1: EXPLANATION STRATEGY
A: DESCRIPTIVE
B: EXEMPLARY
"Descriptive attempts to explain the relevant process, whereas exemplary shows it."

2: FORM
A: STRUCTURAL
B: NON-STRUCTURAL
"Structural uses a formal notation, the other a natural language description"

3: CONTENT
A: FORMAL
B: SUBSTANTIVE
"This distinction is between showing the form or grammar of a command, and giving the substantive application of it"

4: STYLE
A: EXPLANATORY
B: DEMONSTRATIVE

5: APPROACHES TO EXPLANATION
A: SINGLE
B: MULTIPLE

6: SCOPE
A: MINIMAL
B: FULL

7: UTILITY
A: USEFUL
B: LESSUSEFUL
"I avoided the phrase useless, as I found all of the systems had some utility, depending on the nature of my problem eg. wanting to know the correct grammar, or reminding myself of the meaning of a particular term"

8: ELEGANCE
A: ELEGANT
B: INELEGANT
"An aesthetic response."

9: EASE
A: INTUITIVE
B: OBSCURE
"Whilst the syntactic form has elegance, it is not immediately intuitive to me"

The construct tree for subject 2 is fairly typical of the structure of data generated by CES. The construct trees make more apparent the number of distinct ways an individual is using to distinguish between the different interfaces. For example it may be that constructs with very different verbal labels attached to them are used in approximately the same way by an individual. This is true in the Subject 2's construct tree, where 'Style' is seen as similar to 'Utility'. Looking at the raw rating data from subject 2 it is possible to see that this construct is used in a similar way to describe all the help systems, although the poles are reversed, so that 'Useful' systems are aligned with 'Demonstrative' systems, and 'Less Useful' systems with 'Explanatory' ones. This helps illuminate one aspect of the task presented in this study, where subjects were asked to complete simple operations in SQL. In this task understanding of syntax is much more useful than understanding why a particular command should be use. From the point of view of a learning package, this type of help can distract attention from the underlying principles that are being taught.

3.6. Giving the Data back to the Designer...

PROTEUS allows the identification of user significant issues, within an overall context of designer significant issues. It does this by identifying both a verbal description of the issue, and its relationships to other issues, and to particular designs, and their features. It also allows a designer to identify the extent to which he is cognizant of particular issues, even though they may be expressed in different linguistic terms. Constructs are not represented only by verbal labels, which are open to misinterpretation and misunderstanding especially where designers meet users of a fairly different background (for example system designers and OU students). The ratings data provides a framework in which the verbal labels take on an extra dimension.

Currently the system designer is evaluating the data from this study, and will consider what form of improvements should be made to the RAS help system.

4. SUMMARY

Although the study was not taken in an optimally ecologically valid setting, in practice the laboratory environment used was probably not distinctly different from an ideal students learning setting. Equally the subjects used were not directly concerned with studying an Open University course. Fortunately all the subjects were interested in finding out about the learning material, and after the study all the subjects stated that they had learned a little more about relational databases.

Although the systems evaluated were very similar to each other, the PROTEUS methodology has produced distinguishing data between the prototype systems. The process of producing recommendations for the system designer is still to be completed, and further work is being undertaken to develop methods for giving the data back to the designer. These include improved interactive graphical representations of the data.

5. FURTHER WORK

The issue of giving back of data to the designer has not been fully addressed in this paper. Current work involves exploring ways of making the significant aspects of the CES data more explicit to designers. Data from the CES is currently essentially individual data, however because the data is generated by experience of the same set of interfaces, it becomes possible to look for similarities in construing between individuals. Further work is currently being undertaken in the representation of group data generated from clustering algorithms.

ACKNOWLEDGEMENTS

The work described in this paper was supported by an Open University research grant.

REFERENCES

Briggs, P., (1987) Usability Assessment for the Office: Methodological Choices and their Implications, in, Psychological Issues of Human-Computer Interaction in the Workplace, (Frese, M., Ulich, E., and Dzida, W), North-Holland, Amsterdam .

Crellin, J. M., (1988) Personal Construct Psychology and the Development of a Tool for Formative Evaluation of Software Prototypes, in: Proceedings of the Fourth European Conference on Cognitive Ergonomics, (Green, T. R. G. et al. ed.), Cambridge.

Grudin, J., (1989) The Case Against User Interface Consistency, Communications of the ACM, 32, 10.

Honikman, B., (1976) Construct Theory as an Approach to Architectural and Environmental Design, in The Measurement of Interpersonal Space Vol 1, (P. Slater ed.) Wiley, London.

Jankowicz, D. and Thomas, L., (1982) An algorithm for the cluster analysis of repertory grids in human resource development, Personnel Review, 11, 4, pp.15-22.

Kelly, G., (1955) The Psychology of Personal Constructs, Norton, New York.

Marcus, A., (1983) Graphic Design for Computer Graphics, in: Readings in HCI:A multidisciplinary approach, (Buxton, W., and Baecker, R.), Morgan Kaufmann, Los Altos, California.

Robson, J. I., and Crellin, J. M., (1989) The Control Implications Program, in: Contemporary Ergonomics 1989: Proceedings of the Ergonomic Society's 1989 Annual Conference, (Megaw, E. D. ed.), Taylor and Francis, London, pp.172-177.

Stewart, V., Stewart , A., and Fonda, N. (1981) Business Applications of Repertory Grid, Mcgraw Hill (UK) Ltd.

Whiteside J., Bennett J., Holtzblatt K., (1988) Usability Engineering: Our experience and evolution, in: Handbook of Human Computer Interaction, (M.Helander ed.), Elsevier Sciences Publishers, Amsterdam.