INTERACTIVE ELICITATION AND EXCHANGE OF KNOWLEDGE

MILDRED L. G. SHAW

Department of Computer Science, University of Calgary,
Calgary, Alberta, Canada T2N 1N4

This paper gives a state-of-the-art report on the use of techniques based on personal construct psychology to automate interactive elicitation of cognitive structures for group learning and decision making. It outlines the structure and key components of the RepGrid implementation on a network of Macintosh computers, which includes tools for interactive repertory grid elicitation and analysis, and a system for interactively exchanging and comparing repertory grids from multiple simultaneous users on the network. The users of RepGrid are attempting to extend and understand their own thinking and problem-solving capabilities by interacting with other people. They are able to:

- *see the relationship of their points of view to those of others*
- *explore differing terminology for the same constructs*
- *become aware of differing constructs having the same terminology*
- *extend their own construct systems with those of others*
- *provide others with constructs they have found valuable*
- *explore a problem-solving domain using the full group resources*

This paper describes a distributed system for participative elicitation of construct structures based on a network of computers. It aids group problem solving by enabling the participants to share in a construct elicitation process based on repertory grid techniques. Since their development in the context of clinical psychology, these techniques have found a wide range of applications in education, management, and expert systems development. A similar rationale applies: in education the learner is attempting to acquire a specific construct system for the subject matter, and in management people with different construct systems are attempting to work together toward common objectives. The RepGrid system described in this paper enables a number of individuals to interact through networked personal computers to develop mutual understanding of a problem domain through the use of repertory grid techniques.

Kelly's (1955) personal construct psychology (PCP) provides a

Copyright © 1989 by Hemisphere Publishing Corporation

systemic model of the cognitive and behavioral processes of the individual, including interpersonal and social interaction. In PCP, learning is seen as the development of the construct system, social processes are seen as the development of shared constructs, and problem solving is seen as the active anticipation of the future. The theory provides complete foundations for all human psychological and social processes. Hence it is possible to build technologies based on it that are very general in concept and application.

ELICITATION OF CONSTRUCT SYSTEMS

The manual elicitation of construct systems using structured interview techniques is tedious, time-consuming, and requires skill, practice, and an appropriate personality. In the last decade, as part of the advance in psychological testing technology, computer-based techniques have been developed for on-line elicitation and analysis of construct systems. These have proved very successful and are now in widespread use (Shaw, 1980, 1981). Many of these are interactive programs that allow a greater degree of autonomy and more control to be given to the user.

PEGASUS is a system for eliciting repertory grids, and is part of the PLANET suite of construct elicitation and analysis programs (Shaw, 1982; Shaw & Gaines, 1988).

Based on past experience with the PLANET system, the PEGASUS individual grid elicitation procedure was taken as an initial model for RepGrid. Interaction with PEGASUS, with feedback of construct and element matches, is known to be a stimulating and enlightening experience. Then, to ensure that the group interaction through RepGrid is seen as important in its own right, the features have been included to:

- select additional elements from those entered by other participants
- compare one's own constructs with those of other participants
- exchange complete grids with other participants and evaluate mutual agreement and understanding

It is often desirable to compare constructs across groups. Shaw (1980) discusses *exchange grids* for the measurement of understanding and agreement between either two people or one person on two occasions. For example, two people may each elicit a grid in an area of common knowledge. Each may choose his own elements indepen-

dently of the other, and elicit and rate his or her constructs quite separately. Each then makes two copies of the grid, leaving out the rating values. Both these copies are filled in by the other person, one as he uses those constructs on those elements and the other as he thinks the original was completed. By comparing pairs of these grids it is possible to map the extent of overlap of the agreement and understanding between the two people. Such exchange grid procedures are known to aid individuals in coming to understand their agreements and disagreements with the way others perceive the world, and to help them to understand and use the constructs of others (Shaw, 1980). They give a basis for group problem solving, and hence RepGrid may be presented to interested users as a system around which group problem-solving procedures may be built.

DESIGN CONSIDERATIONS

In choosing a technology for the implementation of RepGrid, the importance of a simple and attractive user interface has been paramount. PEGASUS was designed as a structured interview for users with little or no knowledge of problem-solving techniques. The program largely controls the interaction and limits the user's choices to a few significant decisions. Greater flexibility was both desirable in RepGrid and also intrinsically required by the overlay of group interaction on the basic construct elicitation procedures.

A poor person-computer interface can completely negate the value of the most advanced system. This has been a major consideration in the RepGrid design, and the Apple Macintosh was chosen as a user terminal to RepGrid because:

- The windowing facilities make it possible to present multiple channels of information in a modular and comprehensible fashion.
- The graphic data presentation makes it possible to present grid data structures in an attractive and comprehensible format, replacing the display of numeric ratings with one based on spatial relations.
- Much of the interaction with RepGrid can consist of spatial manipulation using the mouse to click and drag objects on the screen, rather than typing text or commands.
- The proper use of the toolbox routines leads to a uniform interface that is consistent with other Macintosh programs that users already know.

- The Macintosh has a reputation for being simple to understand and easy to use without training or reading manuals, reducing user anxiety and encouraging expectations of being able to operate the system.

The Macintosh interface encourages an object-oriented programming paradigm in which the user has access to a number of discrete, manipulable objects, most of which are potentially accessible at any time. That is, there is very little imposed flow of control. This gives the system tremendous flexibility, allowing much freedom in use, and enabling the users to impose their own mode of working upon the system rather than having to conform to some preset sequence. However, flexibility and freedom are not in themselves necessarily advantageous in human-computer interaction. Users may find the decision-making necessary to drive the system too great a burden, or they may never discover how to use the system effectively because it gives little indication of reasonable modes of use. Thus the Macintosh interface in its own right is no panacea for effective human-computer interaction. It needs careful system design based on an understanding of its underlying concepts, functionality, and the psychology of interaction.

These considerations led to the following design rules for the RepGrid user interface on the Macintosh computer:

- Follow the Macintosh standards (defined by the toolbox and its use by programs such as the Finder, MacWrite and MacDraw) as closely as possible so that Macintosh users find RepGrid natural in use.
- Make each object defined by a window a natural concept unit of the grid elicitation and analysis process so that its purpose and functionality are obvious.
- Design each object to be self-describing so that minimal explanation and use of the manual are necessary.
- Use a graphic representation of data structures and results of analysis.
- Base user data entry on object manipulation using the mouse wherever possible.
- Make the system tolerant of user actions giving ease of system exploration through the capability to reverse actions and recover from errors.
- Within these constraints offer the user the maximum freedom of action possible.

These rules are largely ones of Macintosh programming style that any program should follow. There are, however, many commercial programs that deviate from this style, often because they have been transferred from other computers. Worse, there are a number of programs also that apparently conform to the style but fail to follow it in some important features. These latter programs undermine the user's confidence in the system and his capability to understand it. It has been important to avoid such problems in RepGrid.

RepGrid goes far beyond PLANET and any derivative systems. RepGrid is fully object oriented, in that no numbers are used to hold place values, but the elements can be directly manipulated using the click and drag technique of the Macintosh. The mouse can be used for all manipulative interactions, and the keyboard need only be used for entering the names of things. Also, many mouse actions have an alternative keystroke for those who want this choice.

In addition to the natural spatial elicitation, RepGrid is very fast, giving analyses in a few seconds, complete with all the graphics. For example, FOCUS takes 1 or 2 seconds, and the resulting graph has the cluster links drawn on the trees; the user may choose to place the element tree above the grid as is usually done, or to the right below the construct tree. PrinCom produces all the principal components, and plots the two specified. The lines and crosses representing the constructs and elements in the output are fixed, but the labels can be moved by the user to make them easier to read. These and other facilities make RepGrid significantly different from its forerunners, in terms of the interaction with the user, and the potential usefulness to the experimenter/therapist with a client.

SINGLE GRID ELICITATION

When starting up the RepGrid system, a double click on the icon shown in the top left corner of Fig. 1 will yield the initial screen. This asks the user for a *Name and description,* which is used to monitor the user's interaction and stored in the *Log* file.

As an example, we use *Donna,* who is going to look at *systemic family therapy.* She chooses to start an elicitation, so she first has to enter her preliminary data shown on the *Status* screen, such as the grid title and the rating scale, which ranges from −99 to +99. Donna types in her name as the *Title,* then clicks on *Purpose.* A new user will have a topic with a specific purpose, which should be kept in mind as the interaction proceeds. In this example Donna is construing *ther-*

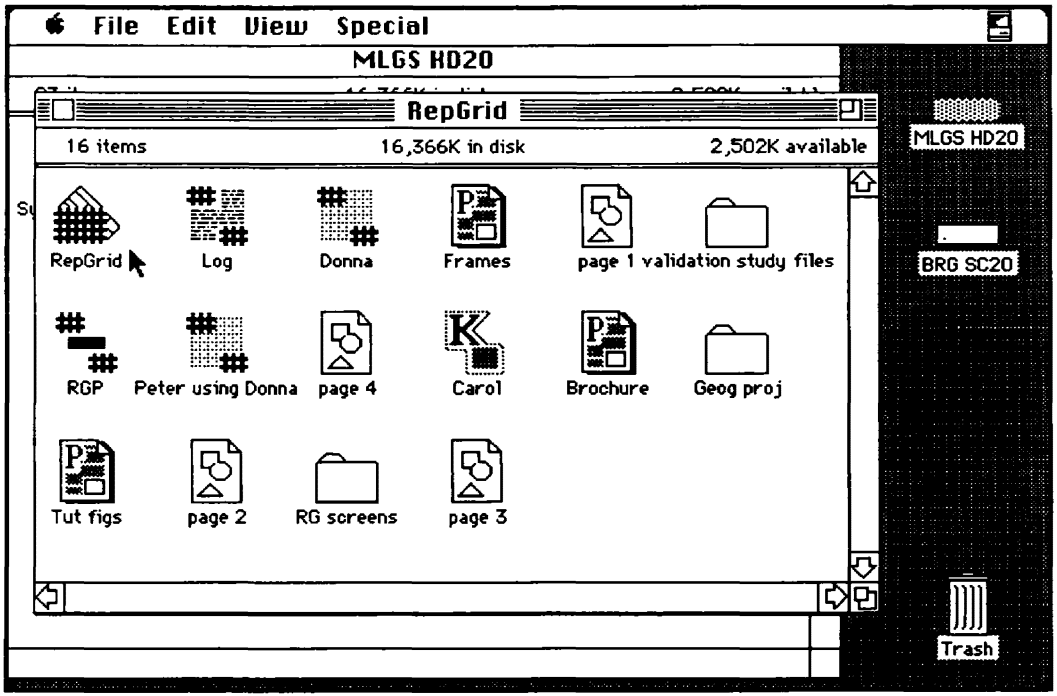

FIGURE 1 RepGrid desktop showing the icons.

apists in order to *distinguish therapies.* She chooses a 1–5 scale to make these distinctions. She also renames the default labels *Element* to therapist and *Construct* to attribute. These labels are then used in subsequent feedback and instruction. When she has finished she can press *Return* or click on *OK* to start the elicitation.

The next thing for Donna to do is to think of some therapists and add these as elements. She knows that at least six is the best number to start with, and that less than three will make triadic elicitation impossible. She adds 11 at this time, shown in Fig. 2. If a typing error is made, or she wants to change an element, she can just go back and correct it at any time just by highlighting it and clicking on the *Edit* button.

As Donna is just starting her grid, she decides to elicit a construct from a triad, so she clicks on the *Triad* button. She sees the three elements in the triad, and decides that Alan Parry is the one that is different from Jay Haley/Cloe Madanes and Karl Tomm. The triad is shown in Fig. 3.

Now Donna has to decide why she thinks Jay Haley/Cloe Madanes and Karl Tomm are alike and different from Alan Parry. She types in the description of the pair, *hierarchical,* on one pole of the construct, then she names the other end of the construct: the way in which Alan Parry is different is that he is *collaborative.* Donna now has to rate all the other elements on this scale (Fig. 4).

Now she has got her first construct, and continues adding several more constructs from triads before she decides to look at high matches. After some time, she chooses to edit the first construct and can do this just by calling it up and clicking and dragging the elements she wishes to move to their new locations.

After this, Donna notices that the bar in the top right is showing a high match between two of the therapists. She clicks in the bar to find out what it is, then chooses to *Show* it by clicking on that button, and can see in Fig. 5 why the therapists are matched. The line pointing up indicates the placing of Minuchin, and the line pointing down that of Milan.

She then decides to break the match by adding a new construct that distinguishes between them. This takes her to a new screen, placing the matched elements at each end of the construct bar. The new construct can then be named and rated in the usual way. The first thing she does after rating the elements is to edit one of the pole names, as shown in Fig. 6.

Donna then decides to add several construct names before rating all the elements on them. Constructs with unrated elements are shown by a bullet mark (•) between the pole names. Donna then

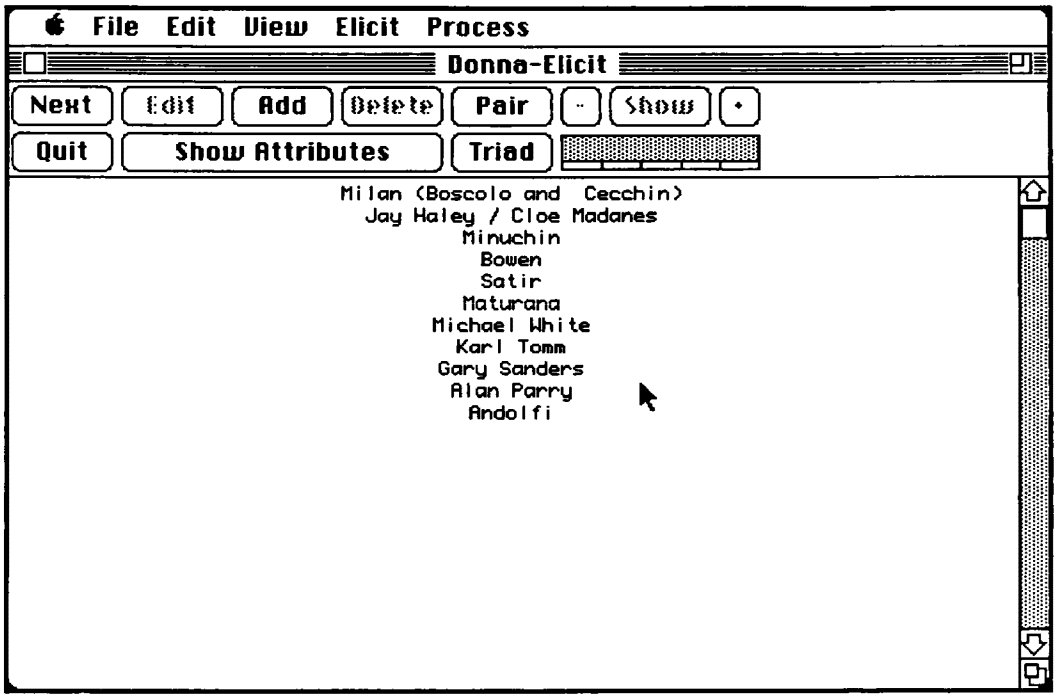

FIGURE 2 RepGrid elements window.

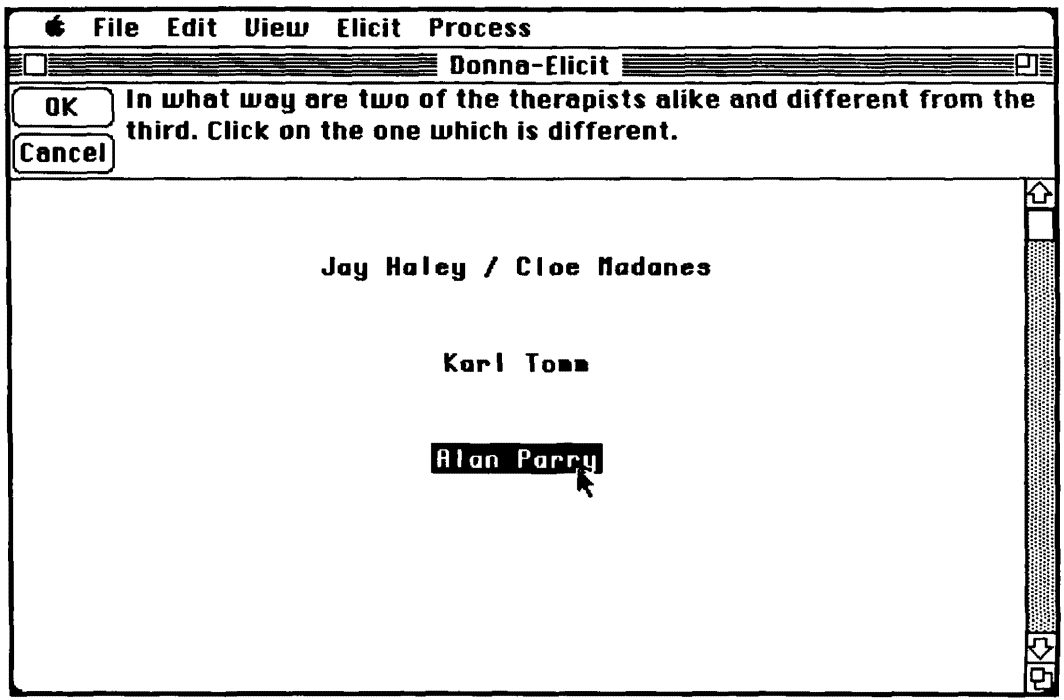

FIGURE 3 RepGrid triadic elicitation.

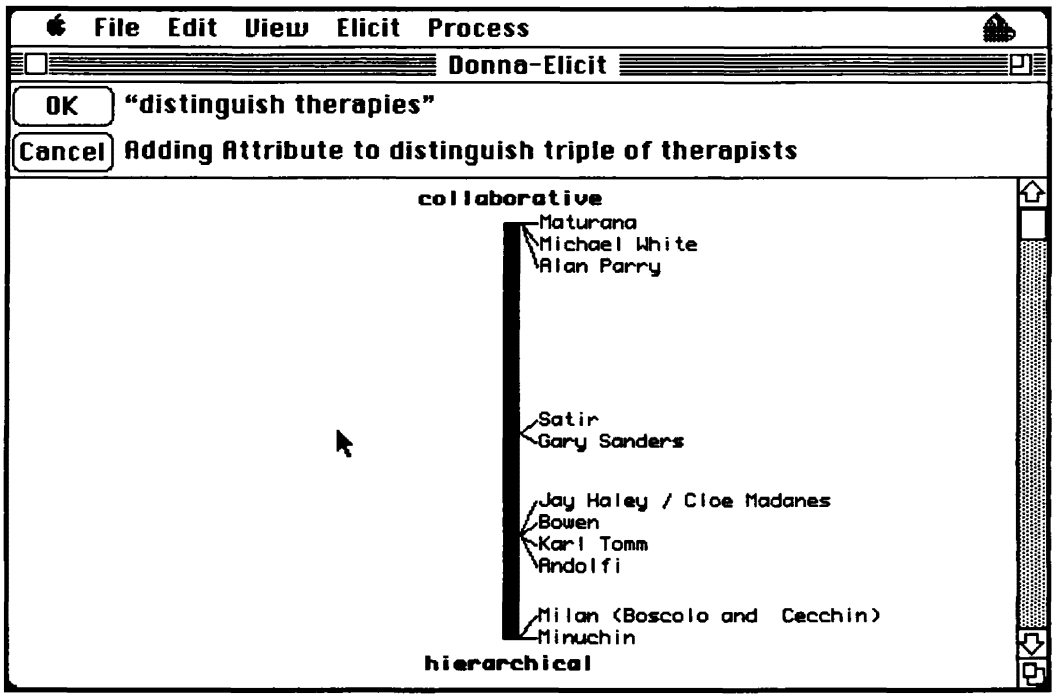

FIGURE 4 RepGrid first construct.

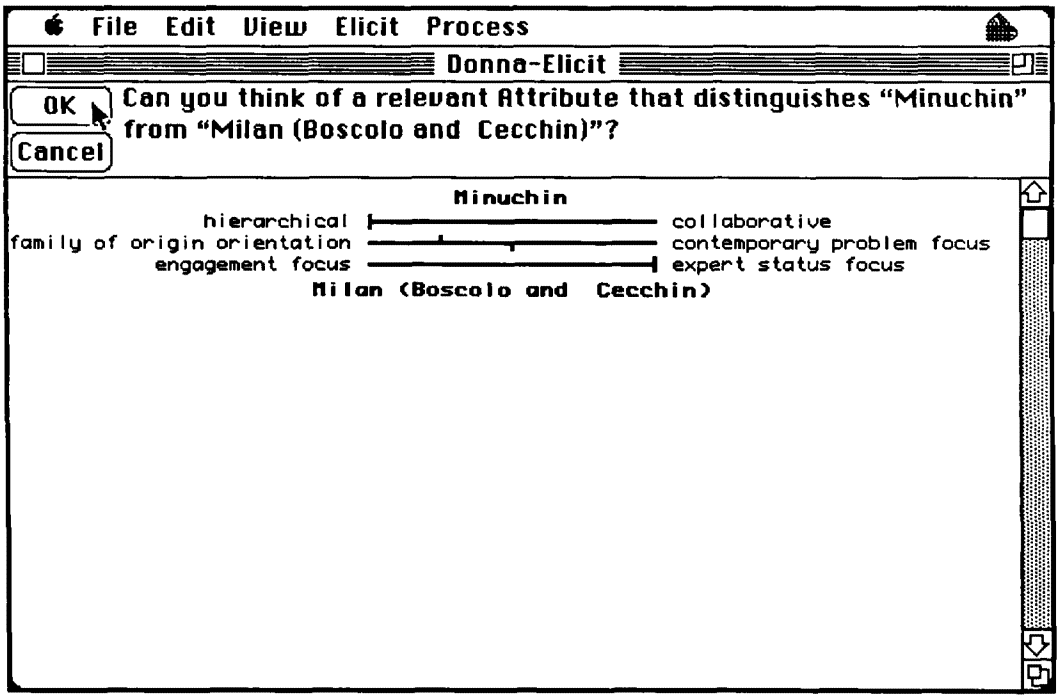

FIGURE 5 RepGrid showing the element match.

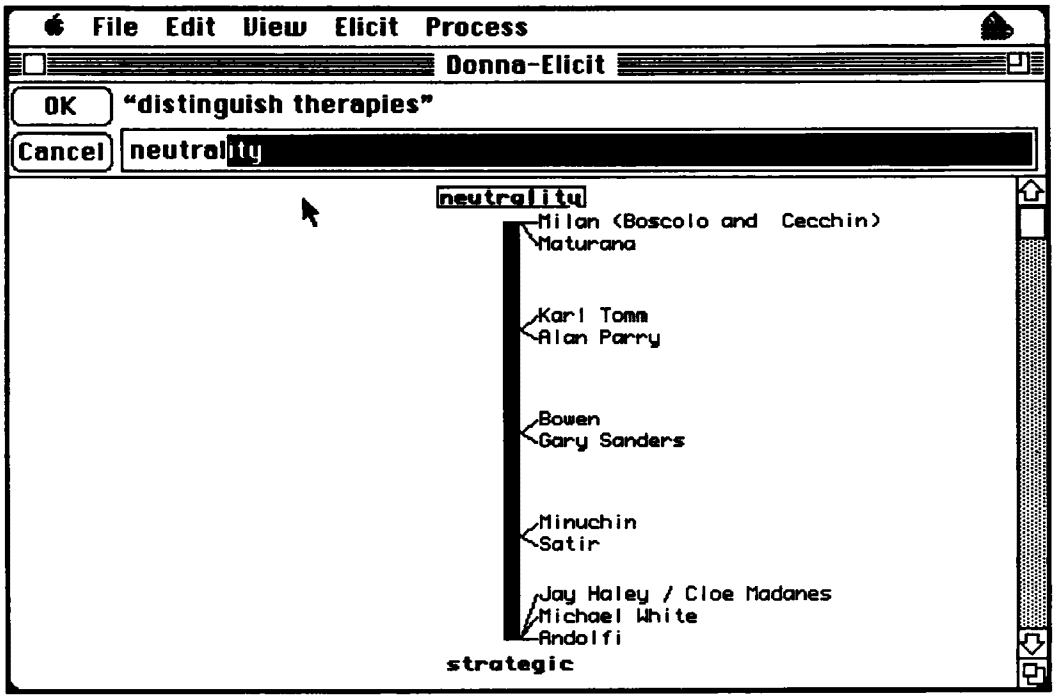

FIGURE 6 RepGrid editing a construct pole name.

continues to add constructs either from triads, or from considering element matches. She also chooses to look at a number of construct matches, but not to split any by adding another therapist at this stage in the elicitation.

Donna then looks at the high matching constructs by clicking on the match bar, then on the minus (−) to step down one level at a time, or on the plus (+) to step back up. She decides to *Show* the highest one, and this takes her to a new screen showing how the constructs are matched, shown in Fig. 7.

Donna decides that she can add another element, and types in a new therapist *Watzlawick*. Figure 8 shows how she rates *Watzlawick* on each of her constructs so far elicited.

Donna can continue to elicit her grid with the options available as described. At any time she can add or delete elements or constructs, use triads or pairs, or just type in names and ratings. She can see the *Status* screen at any time and change it if she wishes—for example, adjust the purpose to be more relevant to what she is doing. When she chooses she can *Display* her grid, or analyze it using FOCUS or PrinCom and then continue with the elicitation. Figure 9 shows how a FOCUS analysis can be requested with clusters exhibited up to certain levels of match, the element tree above or to the right of the grid and shading and/or numbers indicating the ratings. If *Text* is selected, element and construct matching score values will be produced, and put into the *Log* file or some other file of the user's choice. In this case, Donna chose to have the element tree on the right with cut-off values of 65 for elements and 75 for constructs, and both numbers and shading displayed.

Looking at the FOCUS output in Fig. 10, we can see therapists with a similar orientation by looking at the clusters at the bottom right. The one who is different from all the others appears at the top of the main cluster. There are three main construct clusters shown at the top right, the single one at the top and two larger clusters below. This, or any other, picture can be saved as a *RepGrid* or *MacDraw* file in the usual way. A screen of this is shown in Fig. 11.

The first two principal components are shown in PrinCom in Fig. 12. The clusters shown here are almost identical to those shown by the FOCUS analysis. The largest construct cluster is shown on the left and right, the smaller one top and bottom, and the one unrelated one is a short line between the two larger clusters. The therapists also are on the same sides of the picture as in FOCUS, but also, as in FOCUS, not highly clustered. However, the highest cluster of Minuchin, Jay Haley/Cloe Madanes and Bowen occurs in both analyses. Any two components may be chosen to be displayed in PrinCom, for

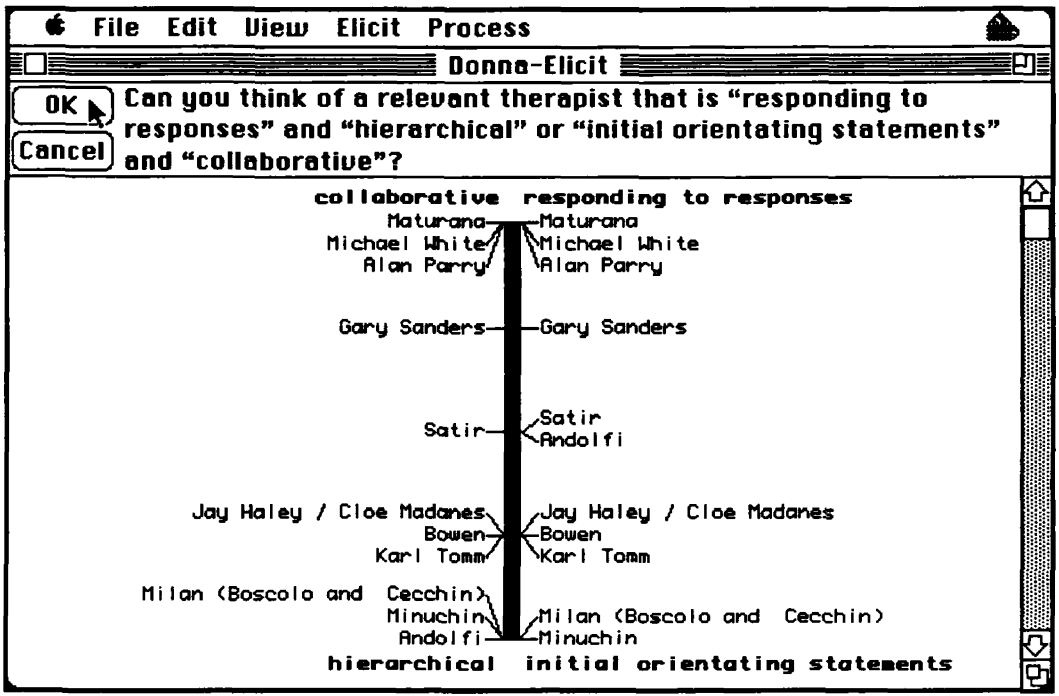

FIGURE 7 RepGrid showing the construct match.

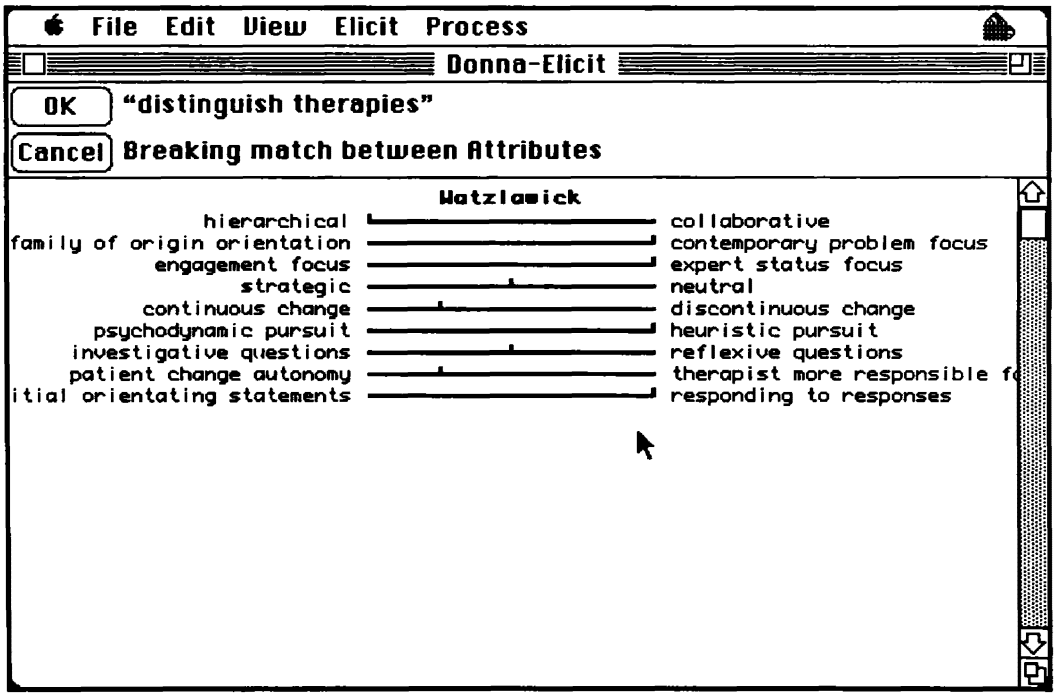

FIGURE 8 RepGrid ratings for the new element.

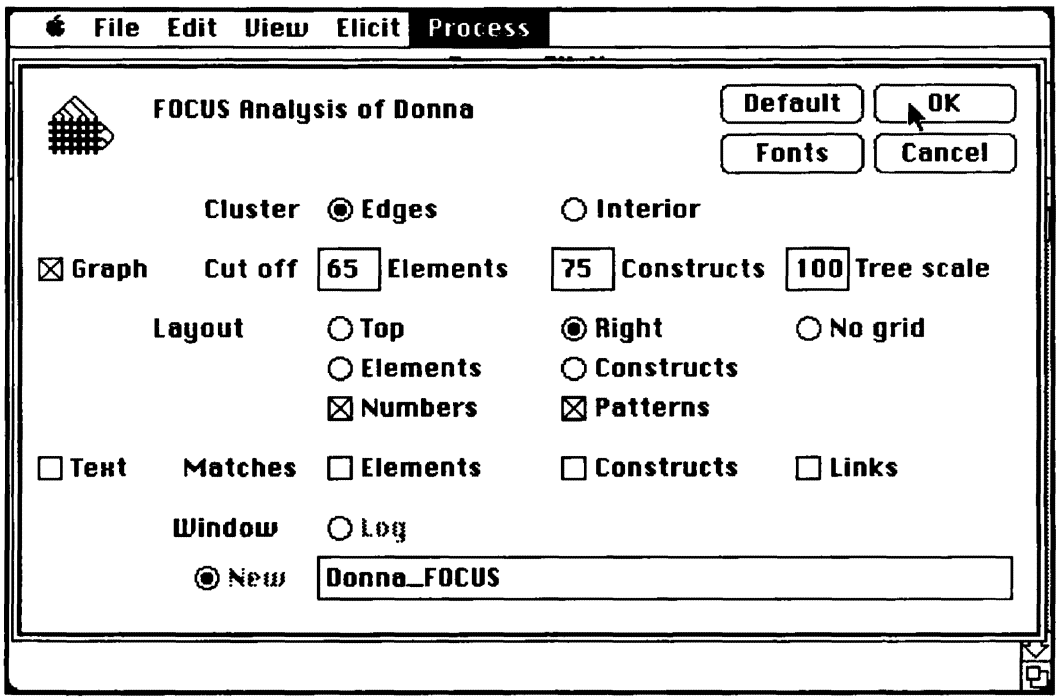

FIGURE 9 RepGrid FOCUS analysis dialog box.

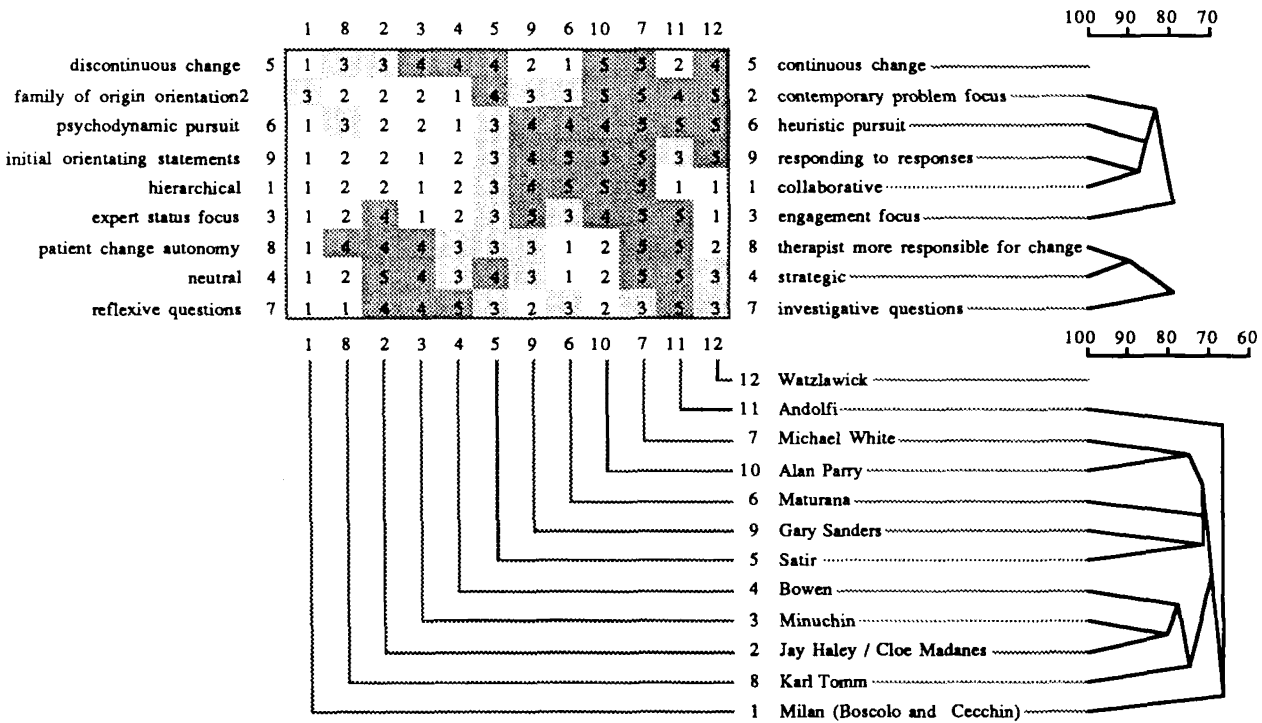

FIGURE 10 RepGrid FOCUS display of Donna's grid.

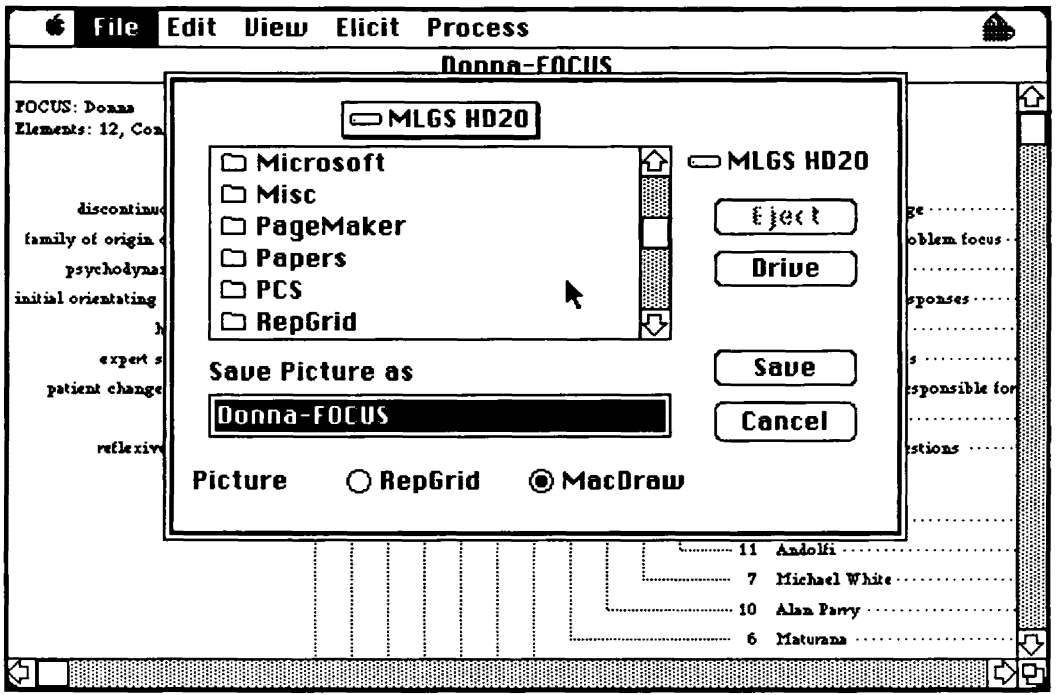

FIGURE 11 RepGrid saving a picture RepGrid.

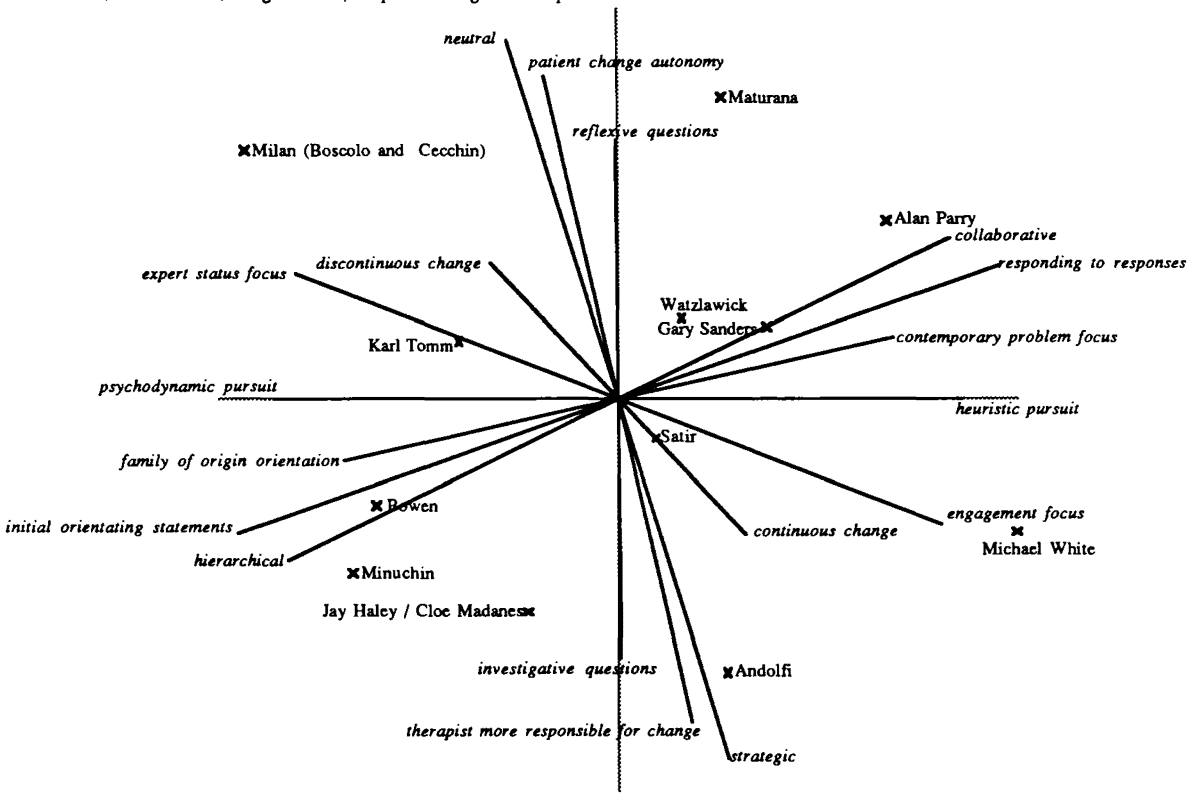

FIGURE 12 RepGrid PrinCom display of Donna's grid.

example, components 1 and 3. The lines in the output are fixed, but the labels can be moved by the user to make them easier to read.

OTHER FACILITIES

The interaction is totally under user control. That is, the user can choose any option at any stage in the elicitation. However, the user can also choose automatic item selection at any time, which can be invoked by clicking on the *Next* button when the decision has to be made. For example, if a match has been found it will be presented to the user, who is asked to split it; if no other action is suggested, the user will be presented with a new triad. *Both* and *Neither* options can be set up, allowing elements to be placed on both poles of the construct, or neither pole. This covers the situation reported by Landfield in his work with suicidal subjects (Landfield, 1976).

All the facilities in the elicitation system are also present in the *Exchange* system described in the next section. If the user wishes to use offered elements and/or offered constructs then the procedure for setting up the *Exchange* system should be followed, as described in the next section, but only the *Elements* and/or *Constructs* boxes should be checked respectively.

EXCHANGE OF KNOWLEDGE

As well as a system for the elicitation of single grids, RepGrid also has a system for interactively exchanging and comparing repertory grids from multiple simultaneous users on the network. The users of RepGrid are attempting to extend and understand their own thinking and problem-solving capabilities by interacting with other people. They are able to:

- see the relationship of their points of view to those of others
- explore differing terminology for the same construct
- become aware of differing constructs having the same terminology
- extend their own construct systems with those of others
- provide others with constructs they have found valuable
- explore a problem-solving domain using the full group resources

There are two forms of construct *Exchange*—rating the chosen partici-

pant's elements on his or her constructs as (1) the user would have done or (2) the user thinks he or she did. These two procedures enable the user to compare (1) the agreement with the other participant and (2) the understanding of the other participant. The user can select either of these forms, or do both, one after the other. Whichever is selected, the same procedure will be followed. In either case, the user selects the file icon of the grid with which he wishes to exchange and compare his knowledge or ideas. The user must change the title to one of his own, and select only the elements and constructs as shown by the X's in the boxes in Fig. 13. (For offered elements and/or constructs the respective boxes should be checked.

The purpose may be changed, but it is inappropriate to change the rating scale. Now the user can see the list of elements and constructs used in the original grid, and choose to *Delete* some if necessary. When the elicitation starts, all the constructs will have unrated elements as shown by the bullet marks, and the user will be able to choose each construct in turn, in any order, and rate all the elements that are listed on the left of the screen. When all the elements have been rated on a construct, that construct is marked in the construct list so that the user can see at a glance which ones he still has to do. For the Agreement Exchange, the user must rate his or her elements on each construct *as she/he would have done* if they were her/his elements and constructs. To repeat this process for *Understanding* the other participant, the user must rate the elements on the constructs *as she/he thought they had been originally rated.* The screens look the same for each construct, just the rating of the elements will be different. Figure 14 shows the constructs that have been completed with a hyphen separating the pole names, and those that still have to have the elements rated on them shown by a bullet between the pole names.

By comparing pairs of these *Exchange* grids it is possible to map the extent of overlap of the agreement and understanding between the two people. Such exchange grid procedures are known to aid individuals in coming to understand their agreements and disagreements with the way others perceive the world, and to help them to understand and use the constructs of others. This then gives a basis for group problem solving, and explicitly seeing the content of any disagreement or misunderstanding of the topic in question.

CONCLUSION

Personal construct psychology can be seen as a theory of cognitive science that has deep systemic foundations, models the person as an

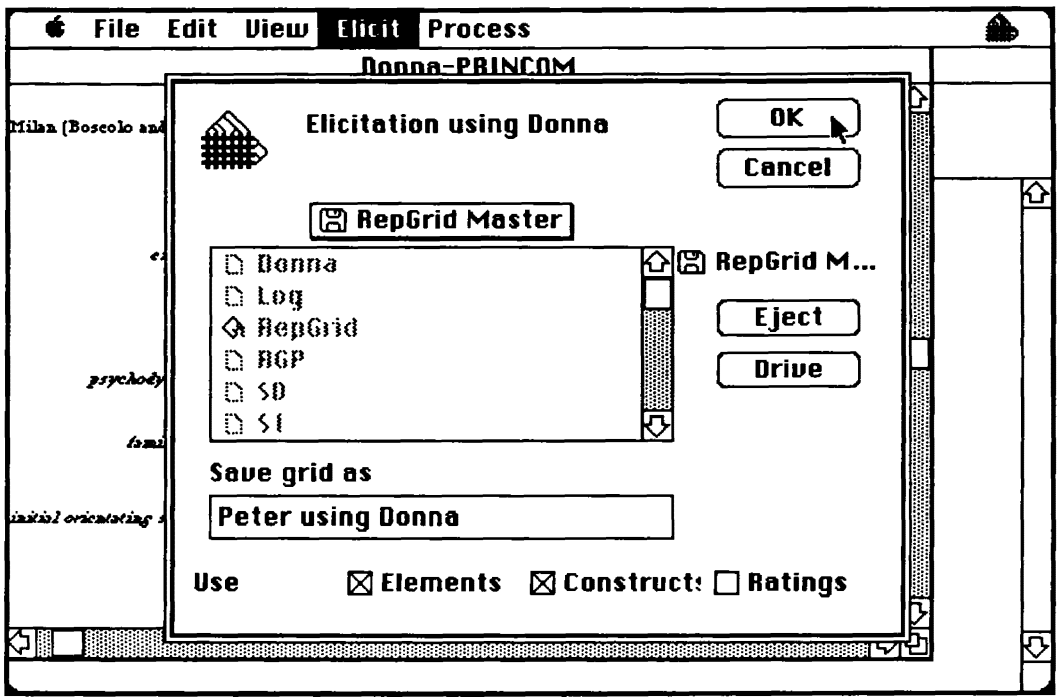

FIGURE 13 RepGrid exchange dialog box.

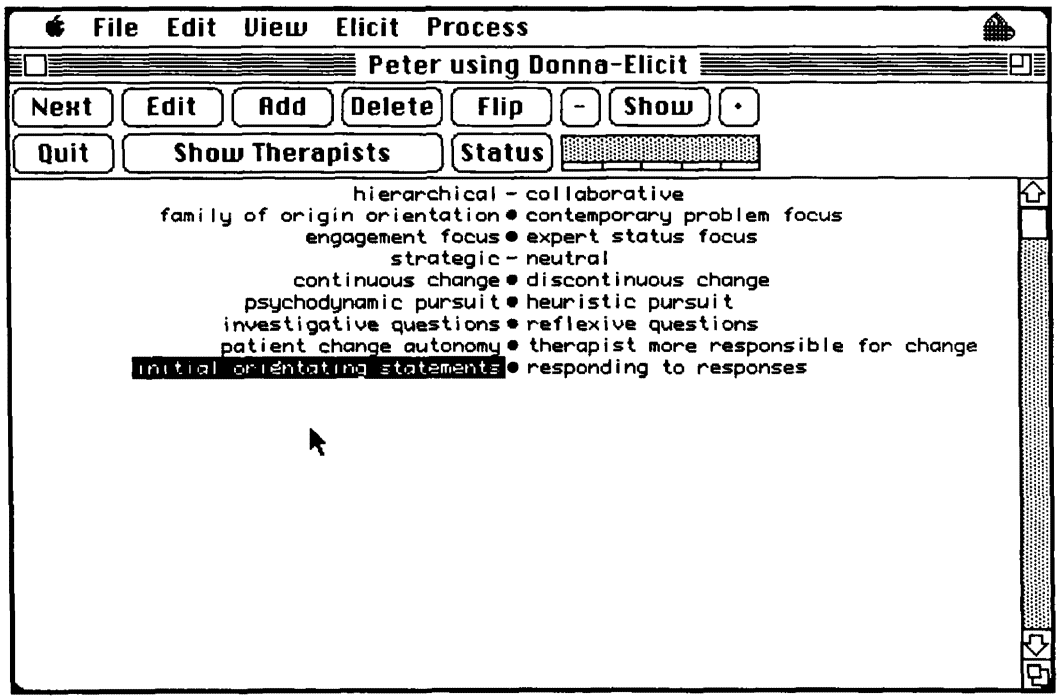

FIGURE 14 RepGrid finished and unfinished constructs.

anticipatory system, and has been operationalized through computer programs that have proved very powerful in expert system development and group problem solving (Shaw, 1985). The new technology of interactive, networked personal computers makes it feasible to develop systems for group problem solving. This paper has described such a system that interactively elicits, exchanges, and processes repertory grids from multiple users on Macintosh computers. The users of the system can extend and understand their own thinking and problem-solving capabilities by interacting with other people to see the relationship of their points of view to those of others. They can also explore differing terminology for the same constructs; become aware of differing constructs having the same terminology; extend their own construct systems with those of others; and provide others with constructs they have found valuable. Thus RepGrid is a new development that enables a group of people to explore a problem-solving domain using the full group resources.

REFERENCES

Kelly, G. A. (1955). *The psychology of personal constructs.* New York: Norton.

Landfield, A. (1976). A personal construct approach to suicidal behavior. In P. Slater (Ed.), *Dimensions of intrapersonal space,* Vol. 1 (pp. 93–107). London: Wiley.

Shaw, M. L. G. (1980). *On becoming a personal scientist.* London: Academic Press.

Shaw, M. L. G. (Ed.), (1981). *Recent advances in personal construct technology.* London: Academic Press.

Shaw, M. L. G. (1982). PLANET: Some experience in creating an integrated system for repertory grid applications on a microcomputer. *International Journal of Man-Machine Studies, 17*(3), 345–360.

Shaw, M. L. G. (1985). Communities of knowledge. In F. Epting, & A. W. Landfield (Eds.), *Anticipating personal construct psychology* (pp. 25–35). Lincoln: University of Nebraska Press.

Shaw, M. L. G., & Gaines, B. R. (1988). PLANET: A computer-based system for personal learning, analysis, negotiation, and elicitation techniques. In J. C. Mancuso & M. L. G. Shaw (Eds.), *Cognition and personal structure* (pp. 27–65). New York: Praeger.

Requests for reprints should be sent to Mildred L. G. Shaw, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4.