Knowledge Modeling on the Web

Mildred L.G. Shaw and Brian R. Gaines
Knowledge Science Institute
University of Calgary
Alberta, Canada T2N 1N4
mildred@cpsc.ucalgary.ca, gaines@cpsc.ucalgary.ca

Abstract: WebGrid-II is a knowledge acquisition and inference server on the World Wide Web (the *web*) that uses an extended repertory grid system for knowledge acquisition, inductive inference for knowledge modeling, and an integrated expert system shell for inference. This article describes WebGrid-II, and illustrates its use in eliciting and modeling knowledge

1 Introduction

Developing knowledge-based systems involves knowledge acquisition from a diversity of sources often geographically distributed. The sources include books, papers, manuals, videos of expert performance, transcripts of protocols and interviews, and human and computer interaction with experts. Expert time is usually a scarce resource and experts are often only accessible at different sites, particularly in international projects. The process of building and testing knowledge models across a distributed community would be greatly expedited if wide-area networks could be used to coordinate the activities at different sites. The objective of the research reported in this paper has been to use the web to support distributed knowledge acquisition by porting existing knowledge acquisition tools to operate through the web (Gaines, 1995a). A further objective has been to use the web on a continuing basis to support distributed knowledge management in which the acquisition, representation and application of knowledge become an integral part of the activities of an organization.

Repertory grids based on personal construct psychology (Kelly, 1955) have been used for knowledge acquisition since the early years of knowledge-based system development (Shaw and Gaines, 1983; Boose, 1984) and have been refined over the years to support increasingly complex knowledge structures (Boose, Bradshaw, Koszarek and Shema, 1993; Gaines and Shaw, 1993). Since grid elicitation tools are used directly by experts it would be very useful to have them accessible through any personal computer or workstation with access to the web. WebGrid (Shaw and Gaines, 1995) provides such access in a way that fully integrates with the web, supporting the use of hypermedia systems to manage a large volume of heterogeneous data, interactive graphic interfaces to present knowledge models in a form understandable to experts, rapid prototyping systems to test the models in operation, and model comparison systems to draw attention to anomalous variations between experts. It can also be used as a service integrated with other web-based knowledge modeling systems (Gaines and Shaw, 1998; Tennison and Shadbolt, 1998).

WebGrid extends the forms of constraint that can be represented in a grid in order to increase its representational power, while retaining a simple user interface to the tool in order to maintain its usability (Gaines, 1995a). It provides categorical and numerical data types in addition to rating scales, and also allows symbolic values such as "any", "none", and "not applicable". We have developed a complete semantics for CLASSIC and similar KL-ONE derivatives in terms of grids in which the basic constraints are derived from set-theoretic constraints upon a filler set for a given role (Gaines, 1993; Gaines, 1994). An entity-attribute-value grid with the 'values' extended to include more general constraints can emulate general concept-role-constraint knowledge representation systems, and the analysis of the grid can model the underlying hierarchies in these structures (Shaw and Gaines, 1998).

This article provides an overview of WebGrid from differing perspectives

2 Eliciting and Modeling Knowledge with WebGrid

Figure 1 shows the initial screen of WebGrid-II. It requests the usual data required to initiate grid elicitation: user name; domain and context; terms for elements and constructs; default rating scale; data types allowed; and a list of initial elements. It also allows the subsequent screens to be customized with an HTML specification of a header and trailer—this capability to include links to multimedia web data is also used to allow annotation, text and pictures, to be attached to elements. The small "?" icons give access to context-sensitive on-line help.

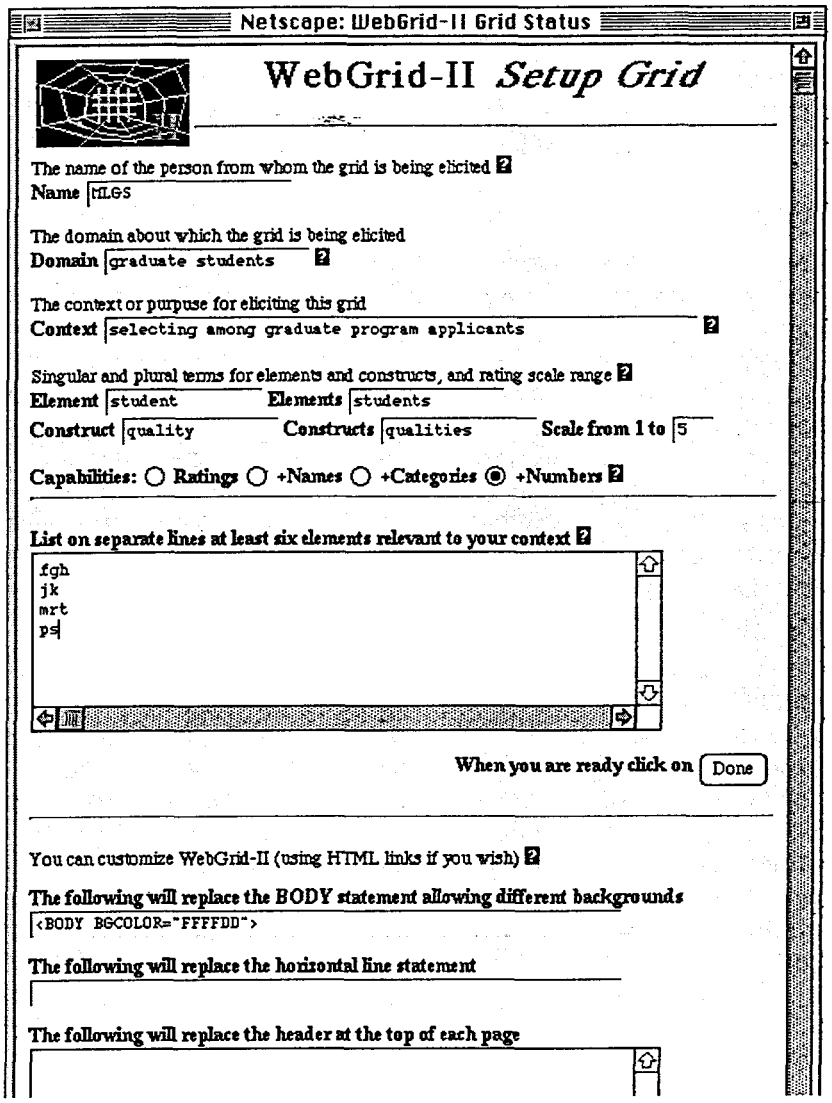


Figure 1 WebGrid-II initial screen

The problem domain is that of "selecting among graduate program applicants" and the intention of the user is to develop a set of rules defining the criteria for selection. The system will incorporate the text typed in by the user into the later dialog, and the user has stated that the elements in the grid are to be called "students" and the constructs "qualities." The user has set the types of constructs to be "Ratings+Names+Categories+Numbers"—the "Ratings" option elicits a standard repertory grid, the "+Names" option indicates that the user wants to use more advanced facilities for knowledge modeling, the "+Categories" allows categorical constructs to be defined, and the "+Numbers" option allows numeric constructs to be defined. She has left the default rating scale to be 1-5 and has entered the initials of 4 students typifying the range of selection decisions normally made.

When the user clicks on the "Done" button at the bottom, the browser transmits the data entered to the remote server which passes it through its common gateway interface to a specialist knowledge acquisition auxiliary server. The server processes the data and generates an HTML document that it returns to the browser resulting in the screen shown in Figure 2 eliciting a construct from a triad of elements.

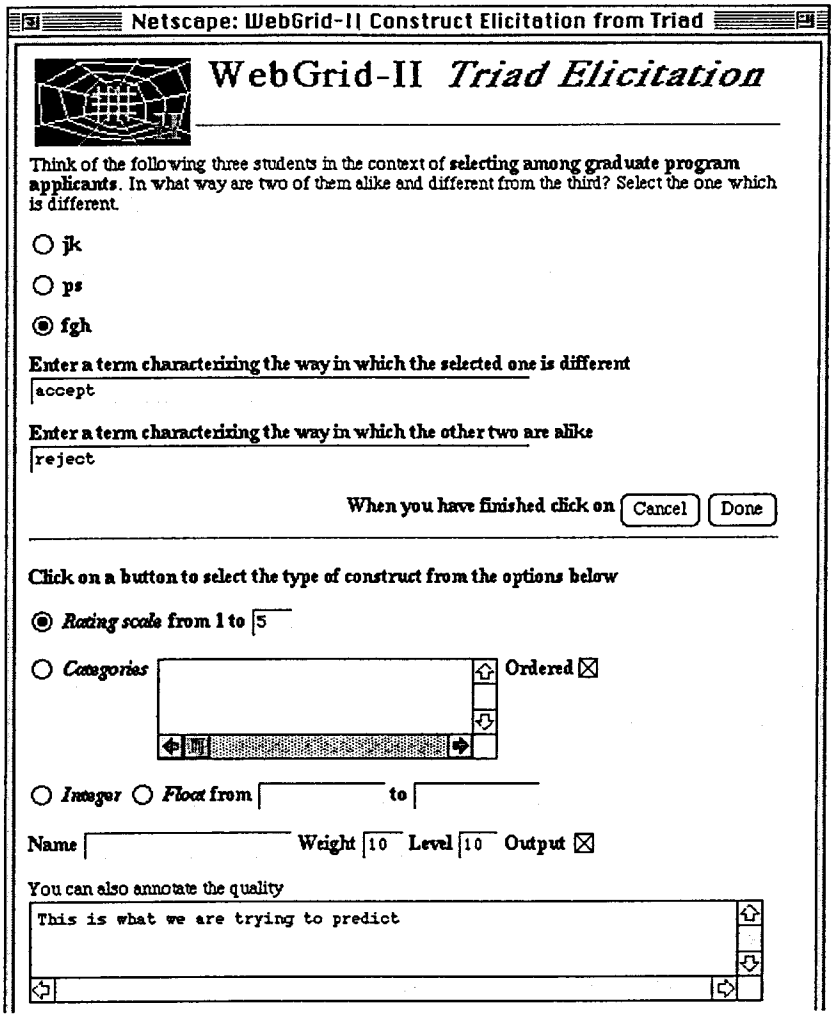


Figure 2 Construct elicitation from a triad

The upper part of the screen assumes that a rating scale will be used. The lower part of the screen is generated because advanced features were selected through the "+Numbers" option. It allows the user to vary the rating scale for an individual construct, to select a categorical, integer or float construct, to name the construct, to give it a weight in clustering, a priority level in asking questions in an expert system application, and to specify that the construct is an input for decision making purposes or an output to be anticipated. Since the construct entered is the main decision to be made, to accept or reject a student, and the user wants the system to anticipate the correct decision from the way the student is construed she has made it an output.

The user clicks on a radio button to select an element which she construes as different from the other two and enters terms characterizing the construct. When the user clicks on "Done" the server generates the screen shown on the left of Figure 3 which places a popup menu rating scale alongside each element enabling the user to rate each one along the new construct as shown on the right. She is also able to change the terms used if they seem inappropriate in the context of all the elements, and to change the ratings of the already entered elements if appropriate.

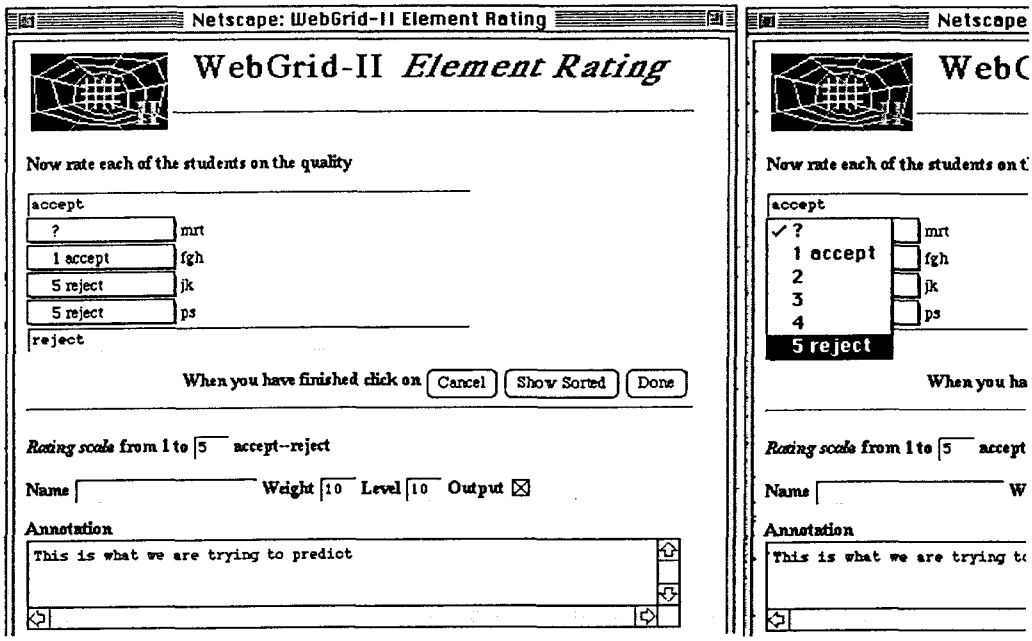


Figure 3 Rating elements on constructs

Clicking on the "Done" button in Figure 3 sends the ratings back to the server which generates the status screen shown in Figure 4.

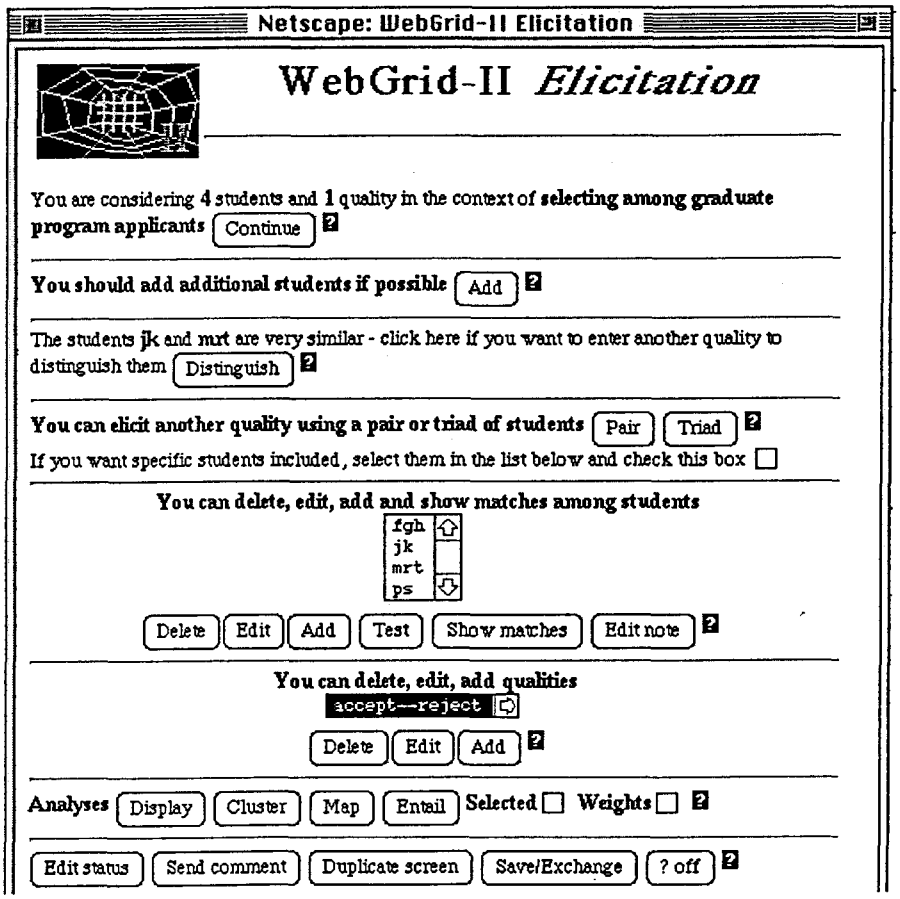


Figure 4 Status screen

This shows the elements and constructs entered, allowing them to be selected for deletion, editing and so on. It also offers various suggestions as how to continue the elicitation based on the data entered so far, facilities for analysis, saving the grid, and so on. The user decides to develop another two constructs from triads of elements and enters "high gpa—inadequate gpa" and "good references—inadequate references" since she knows that these are critical to acceptance.

The user notes that the students' grade point average (gpa) is important in admission decisions and click on "Add" under the list of constructs to add it. She fills in the screen shown in Figure 5, entering a construct called "gpa" which is a floating point number between 0 and 4. She uses the "Categories" box to define some categories based on gpa values, that 0 through 2.99 is inadequate, 3 through 4 is adequate and 3.5 through 4 is high.

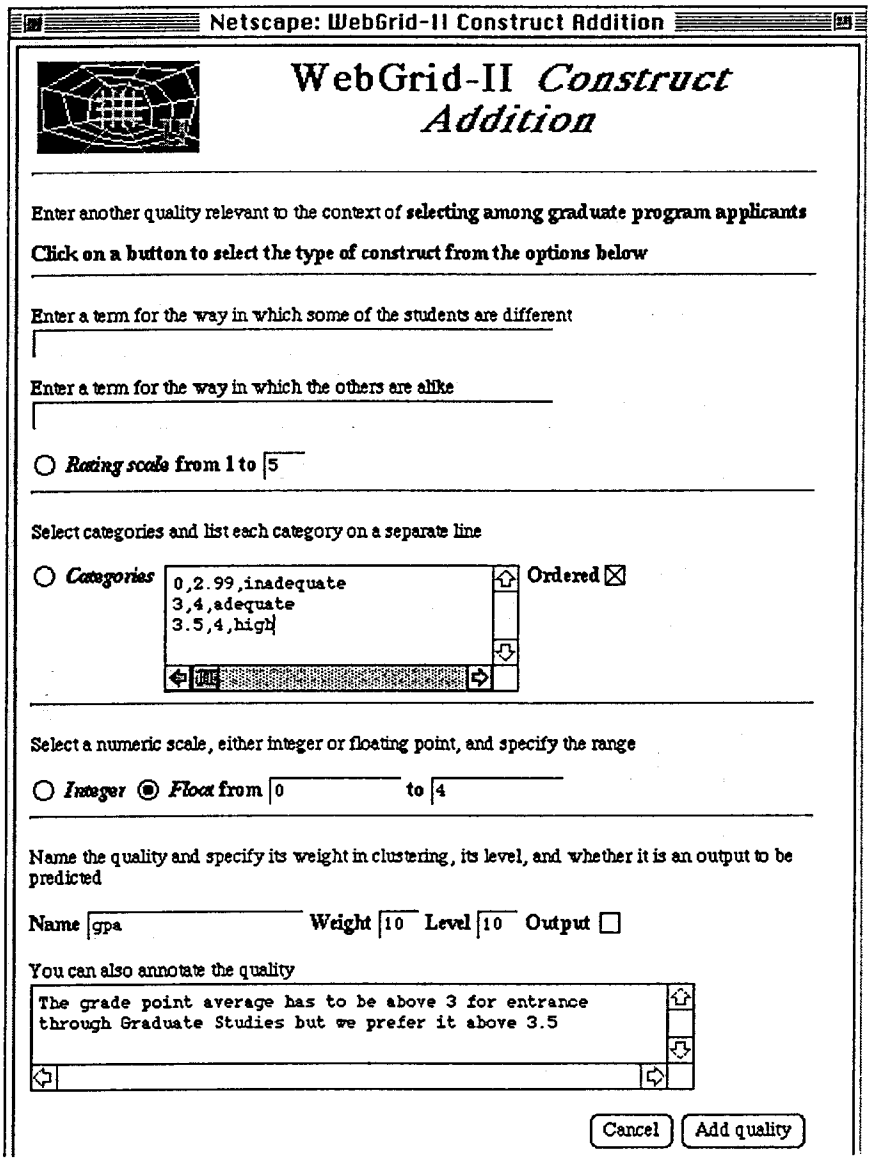


Figure 5 Defining a numeric construct

When she clicks on "Add quality" WebGrid-II returns the screen shown in Figure 6 asking her to enter numeric values for the gpa of each student.

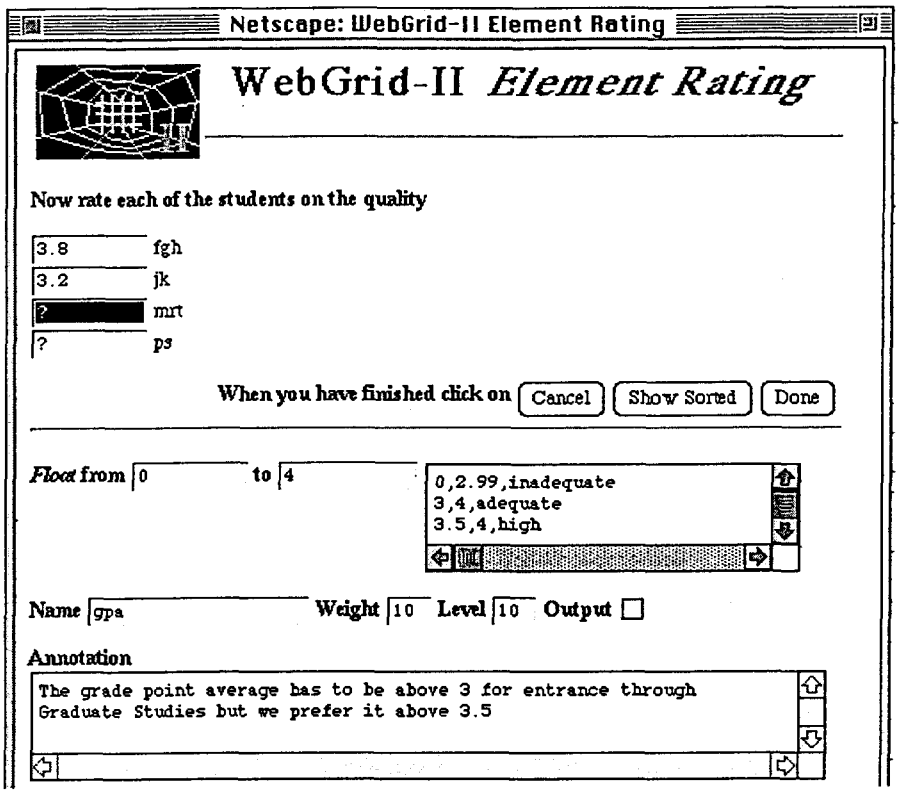


Figure 6 Entering numeric values

She enters another construct, "good references—inadequate references", and decides to run rule entailment to see if the constructs enter account for the admission decisions. This generates the screen of Figure 7 which shows that "ps" might have been expected to be admitted but was not. She realizes this is because he did not having anyone willing to supervise him, enters the construct "willing supervisor—no supervisor", runs rule entailment again and this time gets rules that correctly predict all 4 cases.

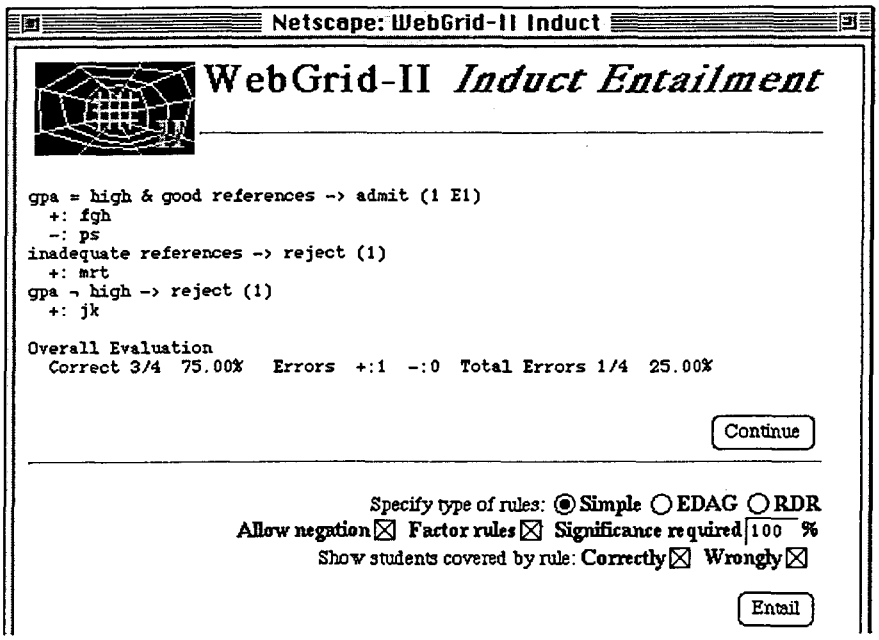


Figure 7 Rules entailed after 3 constructs

She then decides to check the rule against some more test cases and clicks on the "Test" button under the elements list in Figure 4. WebGrid-II generates the screen shown in Figure 8 which enables her to enter data on a test case and infer a conclusion from the rules.

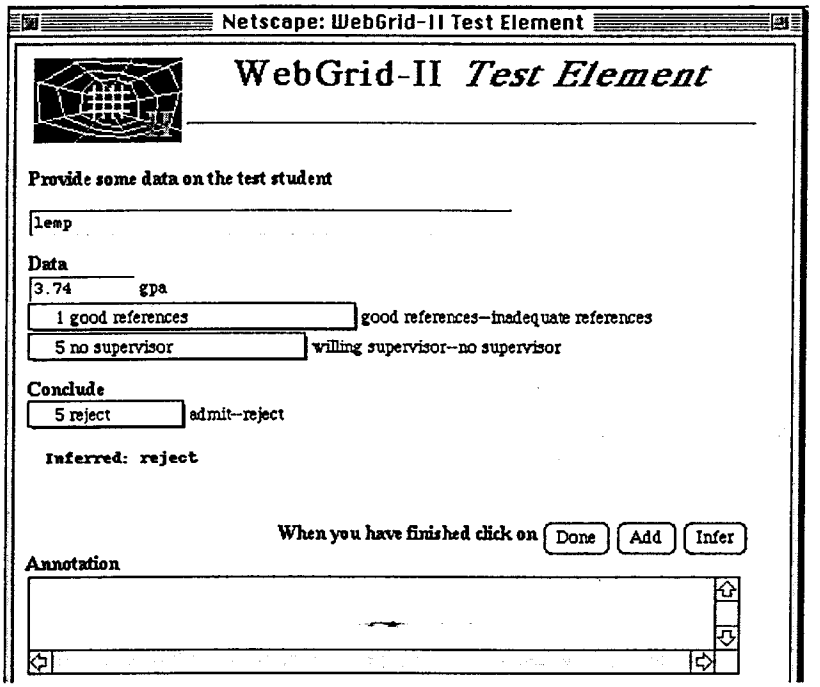


Figure 8 Inference with a test case

When she enters a student "lemp" WebGrid-II infers that he should be rejected but she remembers he was accepted because he had a scholarship and it was expected he would be easily able to find a supervisor. She clicks on "Add" to add this case to the grid and then adds the construct "has scholarship—no scholarship". She goes on adding cases and constructs using the test facility, the suggestions about matches on the main screen, and so on, until she has developed a grid of 12 elements and 7 constructs which results in the rules shown in Figure 9.

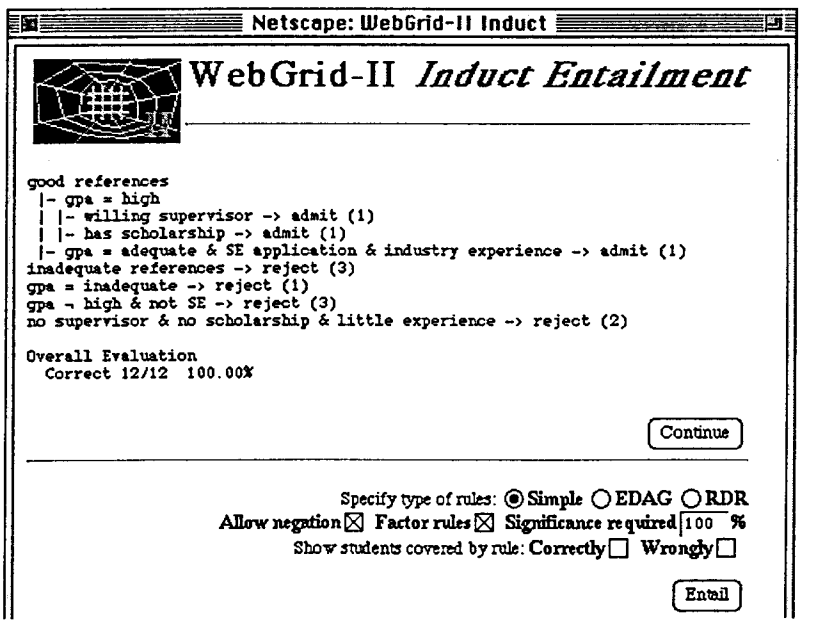


Figure 9 Final rules

The rule for admission is that the candidate must have good references and either a high gpa with a willing supervisor or a scholarship, or must have an adequate gpa and be applying for the software engineering specialization which also requires significant industry experience. She decides also to look at rules with exceptions (Gaines, 1991a; Gaines and Compton, 1995), selects the "EDAG" button in Figure 9 and clicks on "Infer" to generate the screen of Figure 10. This says that a candidate should be rejected unless they have good references and in addition their gpa is high but they do not have no supervisor or no scholarship, or their gpa is adequate, they are applying for the SE specialization and have industrial experience.

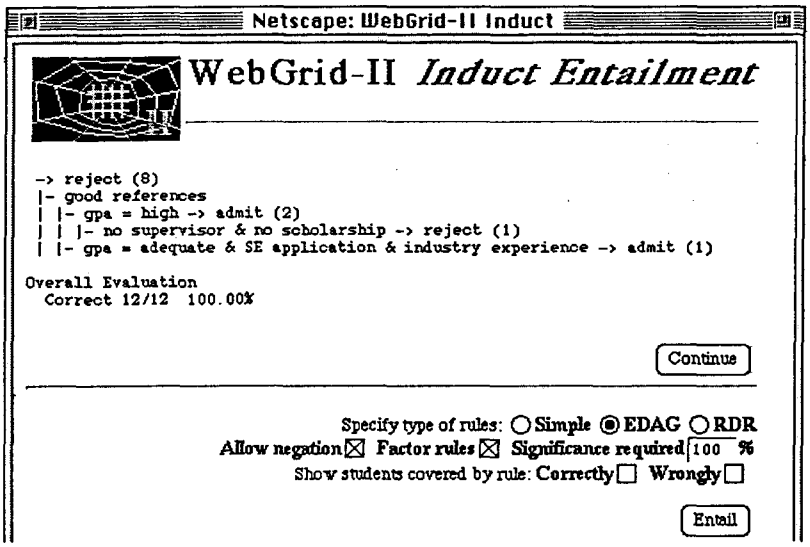


Figure 10 Final rules with exceptions

It can be seen in both Figure 9 and 10 that the induction of rules has itself generated a hierarchical conceptual structure. This is evident when the rule is represented as an EDAG (Gaines, 1995b; Gaines, 1996) in KDraw (Gaines, 1991b) as shown in Figure 11. The rules generated from the grid data result in an elaborate hierarchical conceptual structure.

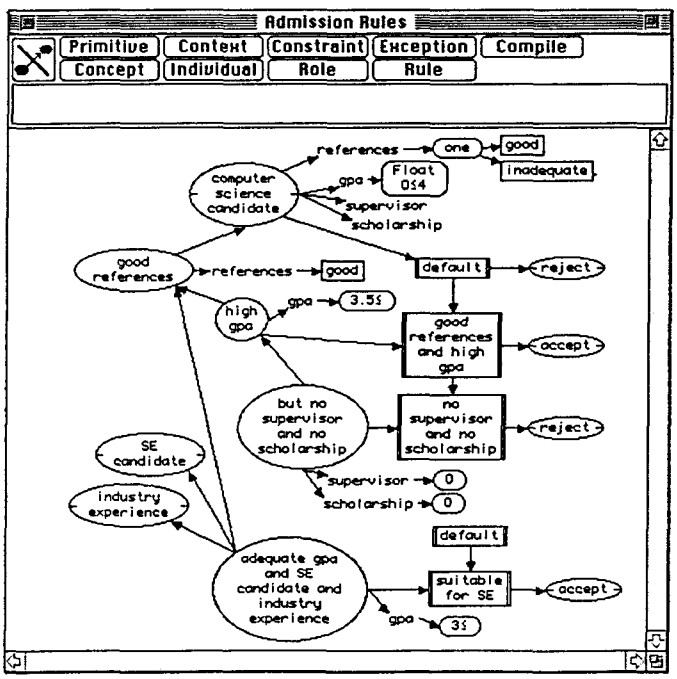


Figure 11 Final rules with exceptions in KDraw

3 Hierarchical Knowledge Structures from Backward Chaining Rules

Gaines and Shaw (1994) present a study of using knowledge acquisition and representation tools to support scientific communities in which they use a laddering technique to analyze the definition of an "IMS ideal artifact" in a paper by Tomiyama (1992) that defines the objectives of the international Intelligent Manufacturing Systems project. The analysis is presented as a knowledge structure in KDraw in Figure 12. The knowledge structure shows an IMS ideal artifact as being defined by its inheritance of 3 essential qualities, each of which is itself defined by its inheritance of 3 essential qualities. The qualities are themselves defined in terms of bipolar constructs consisting of pairs of disjoint primitives.

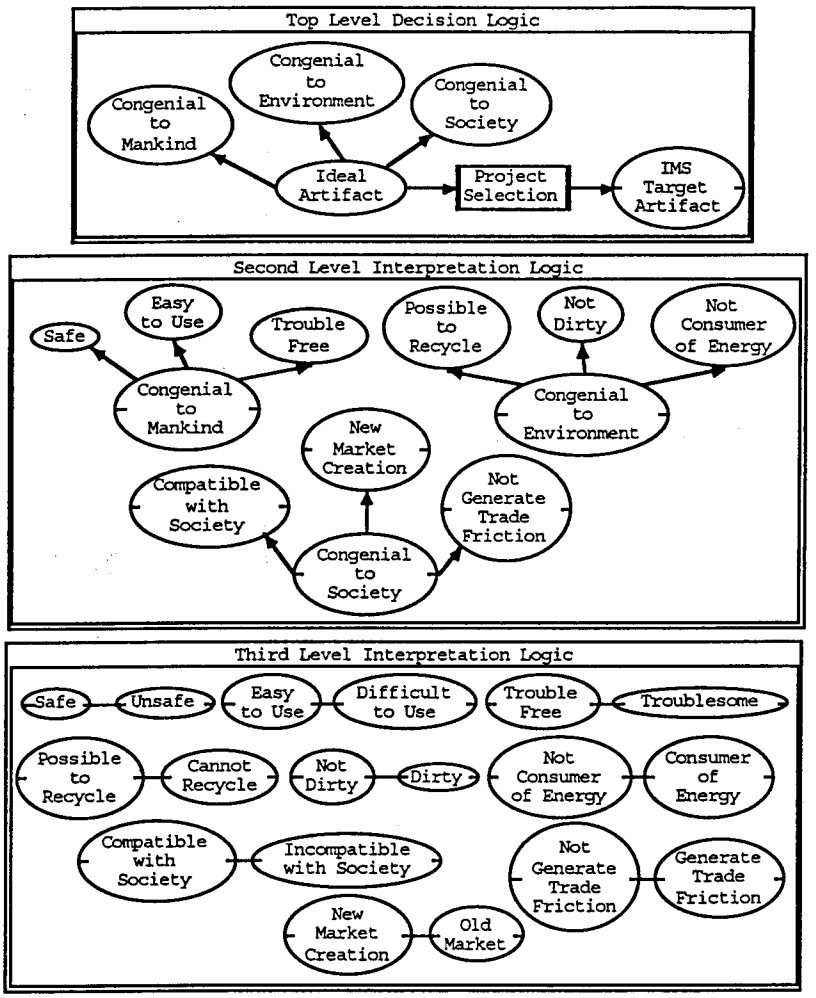


Figure 12 Tomiyama's definition of an "IMS Ideal artifact" represented in KDraw

It is possible to reconstruct this knowledge structure in WebGrid-II by entering a grid of 10 elements and 13 constructs where the elements are an instance of an ideal artifact together with the 9 near-misses that fail to be an ideal artifact by having one inappropriate value. The 4 derived constructs in the first level of Figure 12 are specified as outputs and running rule entailment in WebGrid-II derives their relation to one another and to the operational constructs as shown in Figure 13. The rules derived are a backward-chaining set reproducing the knowledge structure of Figure 12. When they are run on a test case, as shown in Figure 14, all four of the output constructs are derived allowing the reasons to be seen as to why a specific case is, or is not, an IMS ideal artifact to be seen—that is, the "why" or "why not" question is automatically answered in terms of the sub-goals.

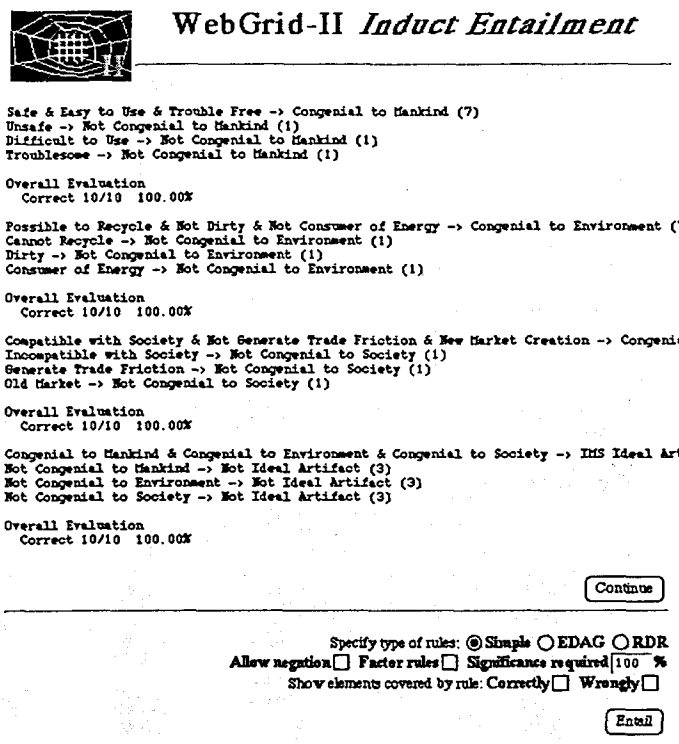


Figure 13 Rule entailment for a grid representing an ideal case and near misses for Fig.16

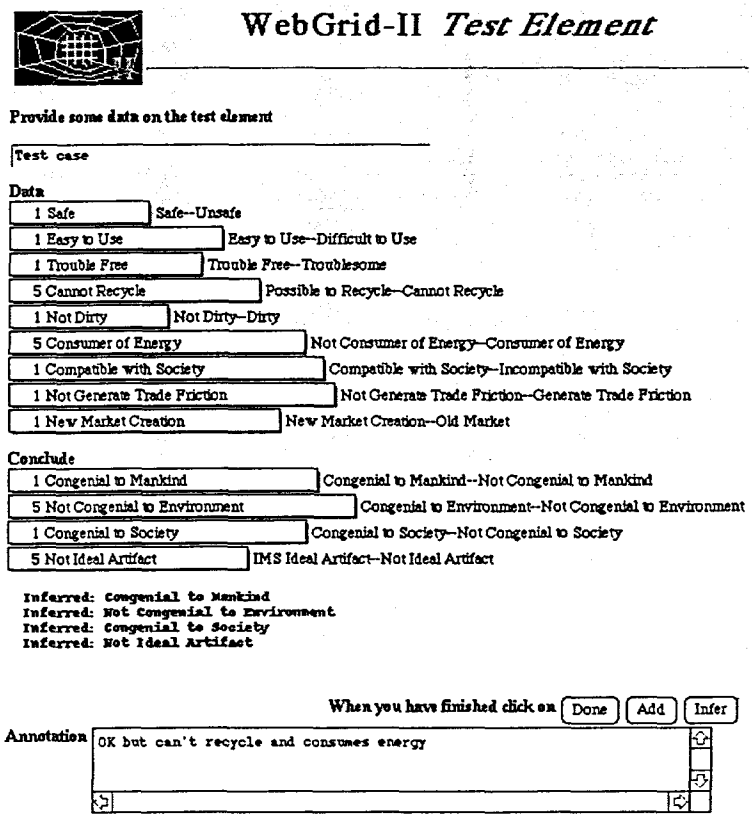


Figure 14 Testing a case as a potential IMS ideal artifact

4 Comparing Expert Models

One area of application of personal construct psychology that has proved useful in many disciplines is the modeling of social knowledge processes through comparing the constructs and terminology of different individuals in the same domain. The comparative analysis of repertory grids to make such similarities and differences apparent was one of the earliest applications of computers to repertory grid analysis (Shaw, 1979, 1980), and the techniques involved have developed over the years to provide powerful tools for comparing knowledge models that have been widely applied (Shaw and Gaines, 1991a,b, 1993) and incorporated in WebGrid (Shaw and Gaines, 1995, 1999). The theoretical foundations for the application of repertory grid elicitation and analysis to modeling social knowledge processes have been described elsewhere (Shaw, 1985). Figure 15 shows the type of situation to be modeled.

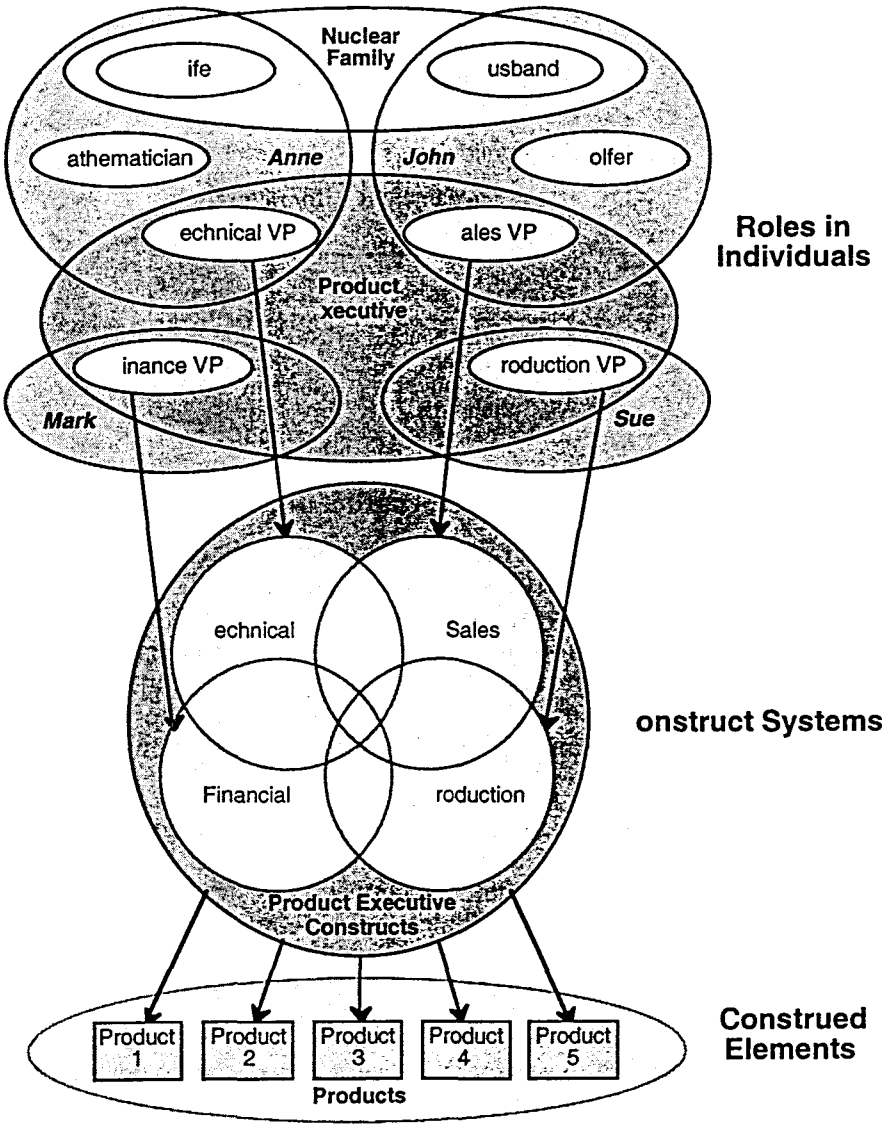


Figure 15 Construct system supporting social processes

Anne, John, Mark and Sue are individuals who play a number of different roles in a number of social situations. Anne is shown as having three roles, wife, mathematician and technical vice president. John is also shown as having three roles, husband, golfer and sales vice president. The wife role in Anne together with the husband role in John are involved in the social processes of a *nuclear family*. Similarly, the technical VP role in Anne, the sales VP role in John, the finance

VP role in Mark and the production VP role in Sue involved in the social processes of a *product executive*. The center part of Figure 15 shows the construct systems supporting the interactions between the roles within people constituting the product executive, and applied by them in the lower part of Figure 15 to construe a domain of products.

When those in the product executive are interacting they expect to have some common constructs reflecting their joint corporate culture and to have some understanding of one another's constructs reflecting their capabilities to engage in discourse. However, a major problem in sharing knowledge in such social organizations is the use of the same terminology for different constructs and the use of the different terminology for the same constructs. Figure 16 shows the possibilities that arise.

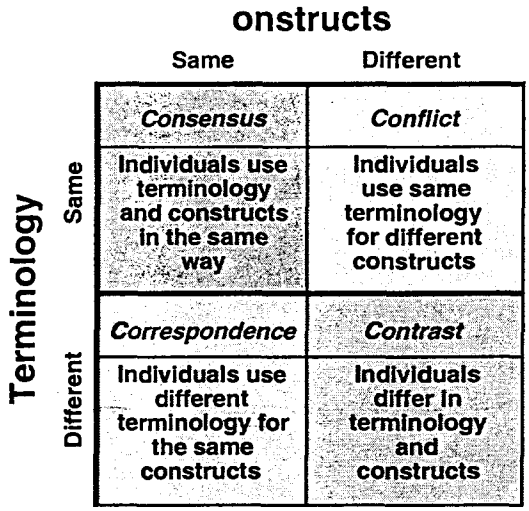


Figure 16 Consensus, correspondence, conflict and contrast in construct systems

The recognition of *consensual* constructs is important because it establishes a basis for communication using shared constructs and terminologies. The recognition of *conflicting* constructs establishes a basis for avoiding confusion over the labeling of differing constructs with the same terms. The recognition of *corresponding* concepts establishes a basis for mutual understanding of differing terms through the availability of common constructs. The recognition of *contrasting* constructs establishes that there are aspects of the differing knowledge about which communication and understanding may be very difficult, even though this should not lead to confusion.

Analysis of repertory grids provides a basis for recognizing consensus, correspondence, conflict and contrast in construct systems. If an individual in one role attempts to fill in a grid from another individual in another role then matches between constructs indicate consensus and major mis-matches indicate conflict. If an individual in one role attempts to construe elements in a grid from another individual in another role then a match between a construct in one grid and that with different pole names in the other indicates correspondence, while lack of such a match indicates contrast.

To set up a distributed elicitation experiment in WebGrid-II, the experimenter first elicits a grid in the normal way resulting in a screen such as that shown in Figure 17. The grid shown was developed to enable students in a class on advanced information system to compare their construct systems with that of their instructor. To make the grid available to the class the instructor clicks on the "Save/Exchange" button at the bottom of the screen which leads to the screen shown in Figure 18. Clicking on the "Cache" button will cache the grid on the server to make it available to others. If the experimenter also wants to collect the exchange grids then clicking on the check box below will store the grid in a new directory and automatically store any grids derived from it in the same directory.

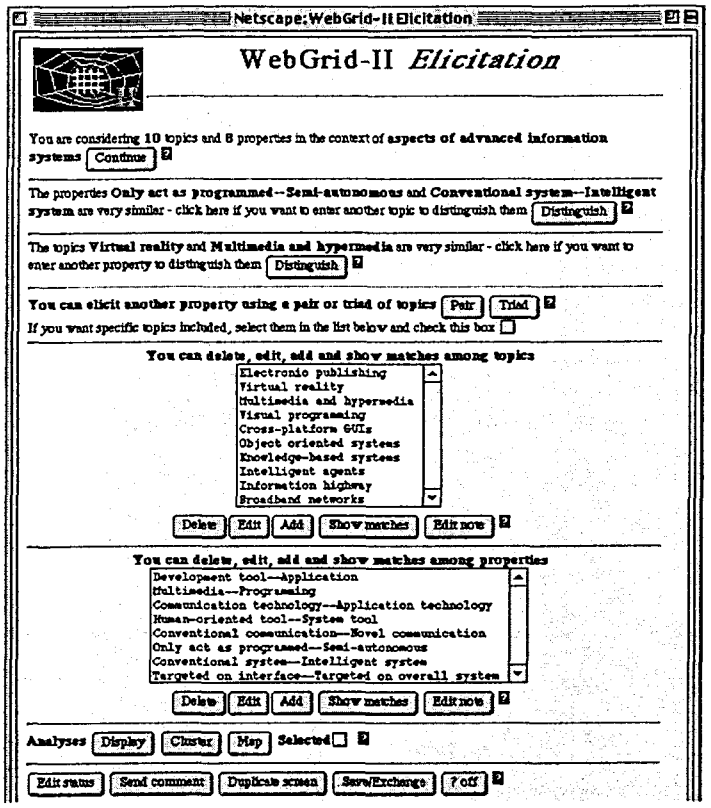


Figure 17 Initial grid elicited using WebGrid-II

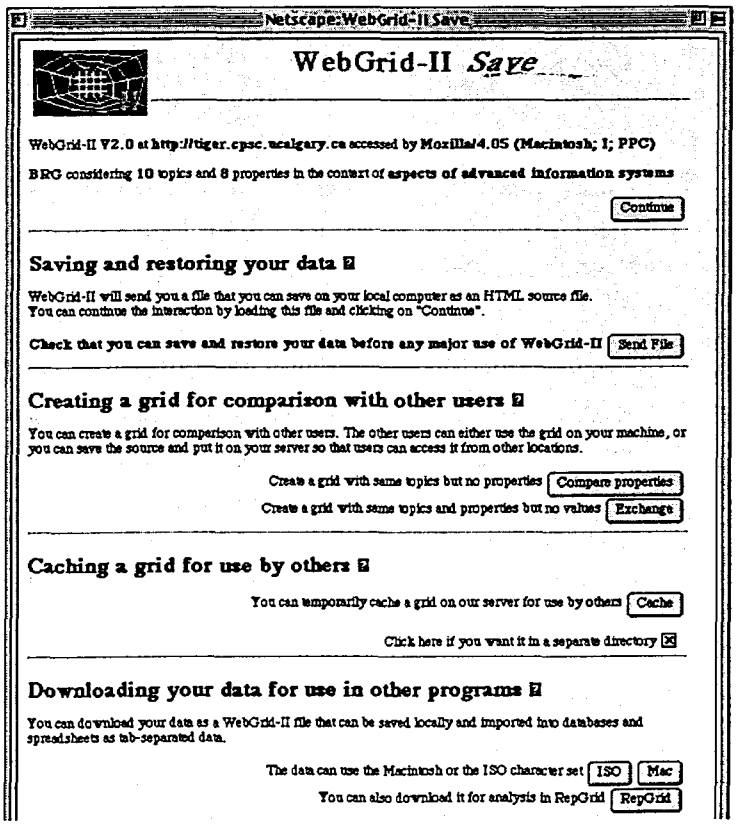


Figure 18 Caching the initial grid for access by others

Figure 19 shows the screen returned when the "Cache" button is clicked in Figure 18. The experimenter is provided with 4 URL's, uniform resource locators that give access to material on the World Wide Web. The first gives access to the original grid, the second to a grid with only the elements and the third to a grid with the elements and constructs but no values. The fourth URL is only provided when the "separate directory" option is selected in Figure 18 and it gives the experimenter access to the set of grids collected.

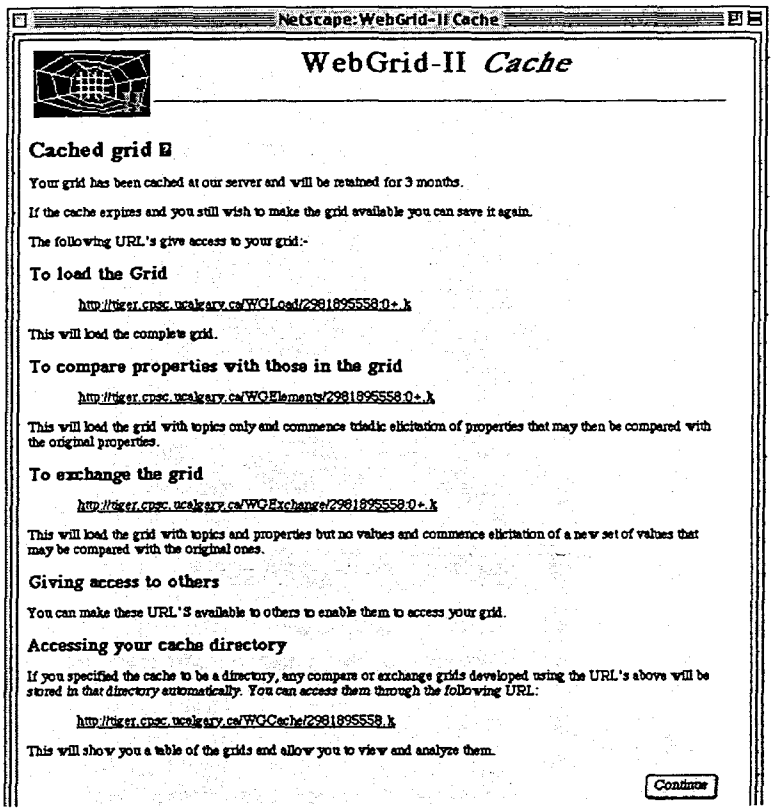


Figure 19 Caching the initial grid for access by others

In this example, the experimenter asked a colleague who was co-instructor for the course to use the URL http://tiger.cpsc.ucalgary.ca/WGExchange/2981895558:0+.k to develop an exchange grid to see how similar the two instructors were in their use of the constructs in it. The students on the course were given the URL http://tiger.cpsc.ucalgary.ca/WGElements/2981895558:0+.k and asked to develop their own constructs and compare them with those of the instructor.

When the co-instructor accessed the URL for exchanging grids shown at the end of the previous section WebGrid-II loaded the original grid with elements and constructs but no ratings and took the co-instructor through a series of screens for rating the elements on the constructs. When this was complete it generated a screen similar to that of Figure 17 but with a "Compare" button in the analyses row after the "Map" button. Clicking on this returns the difference grid shown in Figure 20. It can be seen that the instructors have a clear consensus on 6 out of the 8 constructs and a potential conflict in the use of the other two, mainly arising from the way they construe four of the elements.

When a student accessed the URL for developing constructs with the given elements shown at the end of the previous section WebGrid-II loaded the original grid with the elements only and took the student through a series of construct elicitation processes. When this was complete it generated a screen similar to that of Figure 17 but again with a "Compare" button in the analysis row after the "Map" button. Clicking on this returns a comparison grid, an example of which is shown in Figure 21.

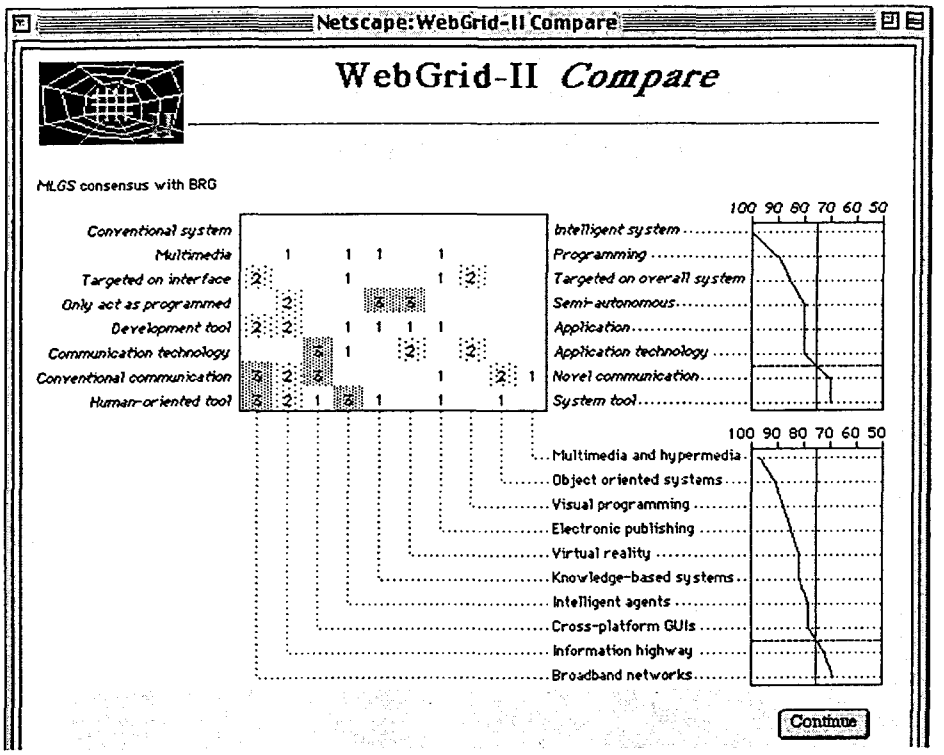


Figure 20 Difference grid comparing the two instructors

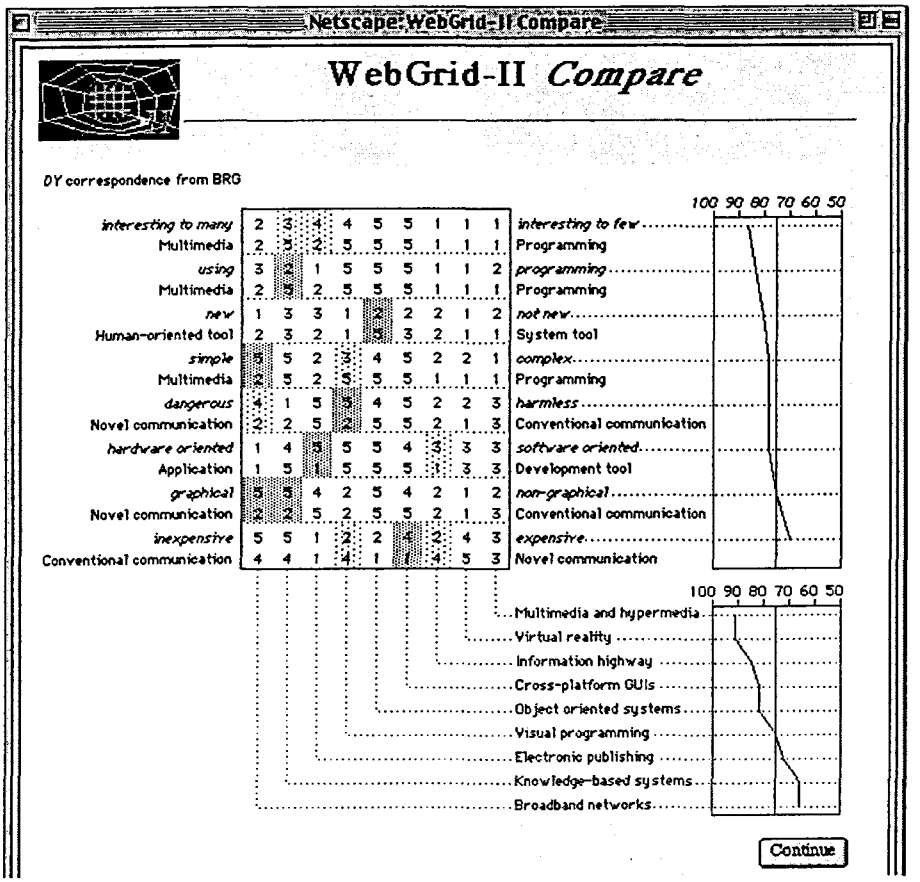
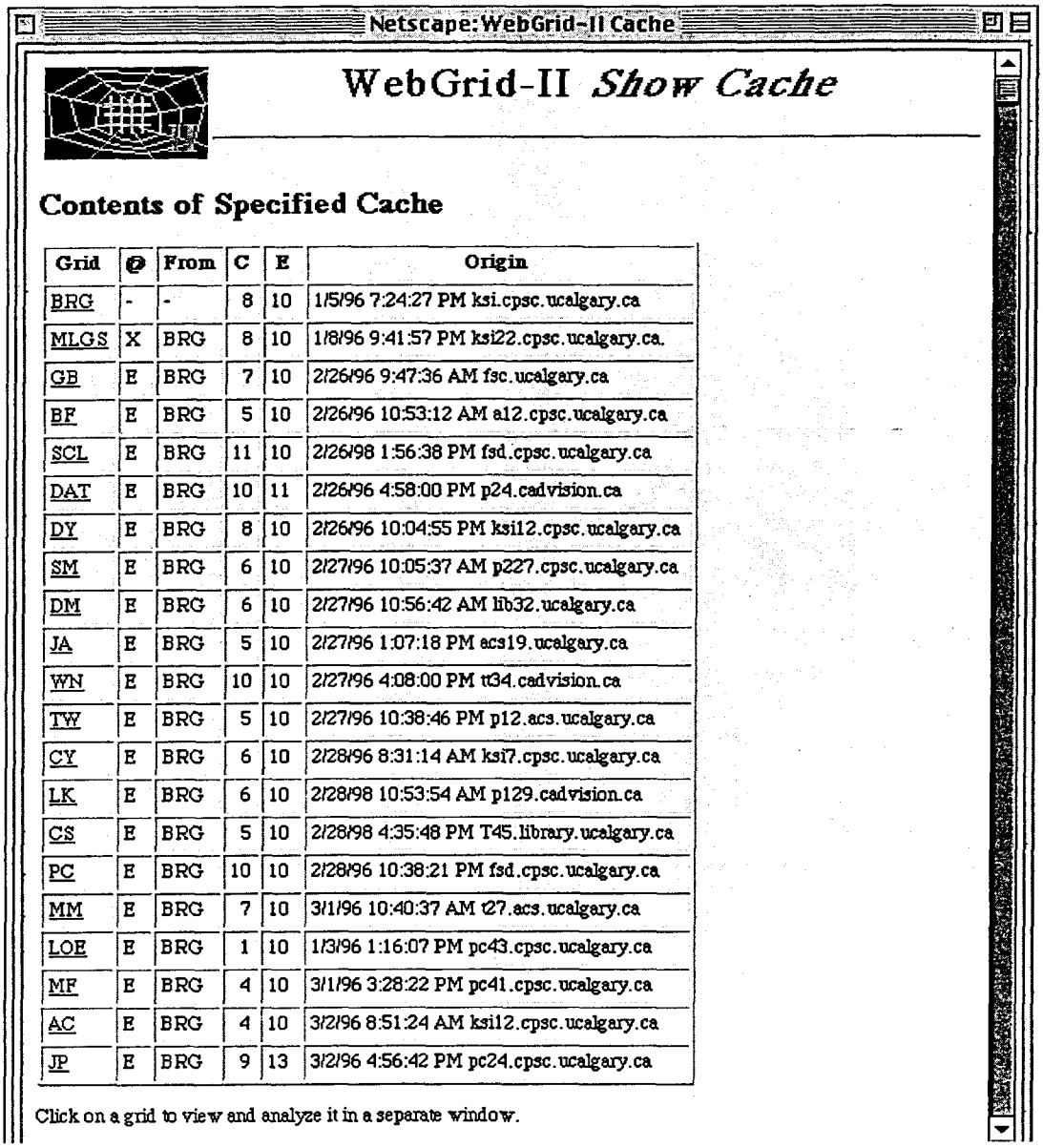


Figure 21 Comparison grid comparing student and instructor constructs

From Figure 21 it can be seen that the student's construct "interesting to many—interesting to few" matches with the instructor's construct "Multimedia—Programming", an example of correspondence. From the element matches at the bottom right it can be seen that the instructor and student may have difficulties in discussing "Broadband networks" since they have few corresponding constructs making similar distinctions for these elements.

The examples above show how WebGrid-II provides the clients with comparative analyses. It also provides the knowledge engineer with full access to all the grids collected through the final URL at the bottom of Figure 19. This generates the screen shown in Figure 22 listing the original grid and all those generated from it.



Grid	@	From	C	E	Origin
BRG	-	-	8	10	1/5/96 7:24:27 PM ksi.cpsc.ucalgary.ca
MLGS	X	BRG	8	10	1/8/96 9:41:57 PM ksi22.cpsc.ucalgary.ca.
GB	E	BRG	7	10	2/26/96 9:47:36 AM fsc.ucalgary.ca
BF	E	BRG	5	10	2/26/96 10:53:12 AM a12.cpsc.ucalgary.ca
SCL	E	BRG	11	10	2/26/98 1:56:38 PM fsd.cpsc.ucalgary.ca
DAT	E	BRG	10	11	2/26/96 4:58:00 PM p24.cadvision.ca
DY	E	BRG	8	10	2/26/96 10:04:55 PM ksi12.cpsc.ucalgary.ca
SM	E	BRG	6	10	2/27/96 10:05:37 AM p227.cpsc.ucalgary.ca
DM	E	BRG	6	10	2/27/96 10:56:42 AM lib32.ucalgary.ca
JA	E	BRG	5	10	2/27/96 1:07:18 PM acs19.ucalgary.ca
WN	E	BRG	10	10	2/27/96 4:08:00 PM tt34.cadvision.ca
TW	E	BRG	5	10	2/27/96 10:38:46 PM p12.acs.ucalgary.ca
CY	E	BRG	6	10	2/28/96 8:31:14 AM ksi7.cpsc.ucalgary.ca
LK	E	BRG	6	10	2/28/98 10:53:54 AM p129.cadvision.ca
CS	E	BRG	5	10	2/28/98 4:35:48 PM T45.library.ucalgary.ca
PC	E	BRG	10	10	2/28/96 10:38:21 PM fsd.cpsc.ucalgary.ca
MM	E	BRG	7	10	3/1/96 10:40:37 AM t27.acs.ucalgary.ca
LOE	E	BRG	1	10	1/3/96 1:16:07 PM pc43.cpsc.ucalgary.ca
MF	E	BRG	4	10	3/1/96 3:28:22 PM pc41.cpsc.ucalgary.ca
AC	E	BRG	4	10	3/2/96 8:51:24 AM ksi12.cpsc.ucalgary.ca
JP	E	BRG	9	13	3/2/96 4:56:42 PM pc24.cpsc.ucalgary.ca

Figure 22 Directory of cached grids for access by experimenter

The listing in Figure 22 shows the identifier entered by the user, "E" for elements only or "X" for exchange, the numbers of constructs and elements, the date and time of storage and the Internet address of the machine used. The experimenter can click on any identifier and view the grid, compare it with the original as shown in the previous section, or otherwise analyze it.

5 WebGrid-II Architecture

Figure 23 shows the overall WebGrid-II architecture. At the right, when the server receives a document from a user client it extracts the grid from the hidden fields and extracts the command and associated data entered by the user. It then executes the command drawing upon the underlying repertory grid tools as necessary, updates the grid if appropriate and establishes the next state of the interaction. It then generates a document to return to the user using the script for that state, passing to it appropriate parameters from the grid. Under certain circumstances, such as the support of collaboration, it may draw upon other grid data cached at the server by the user.

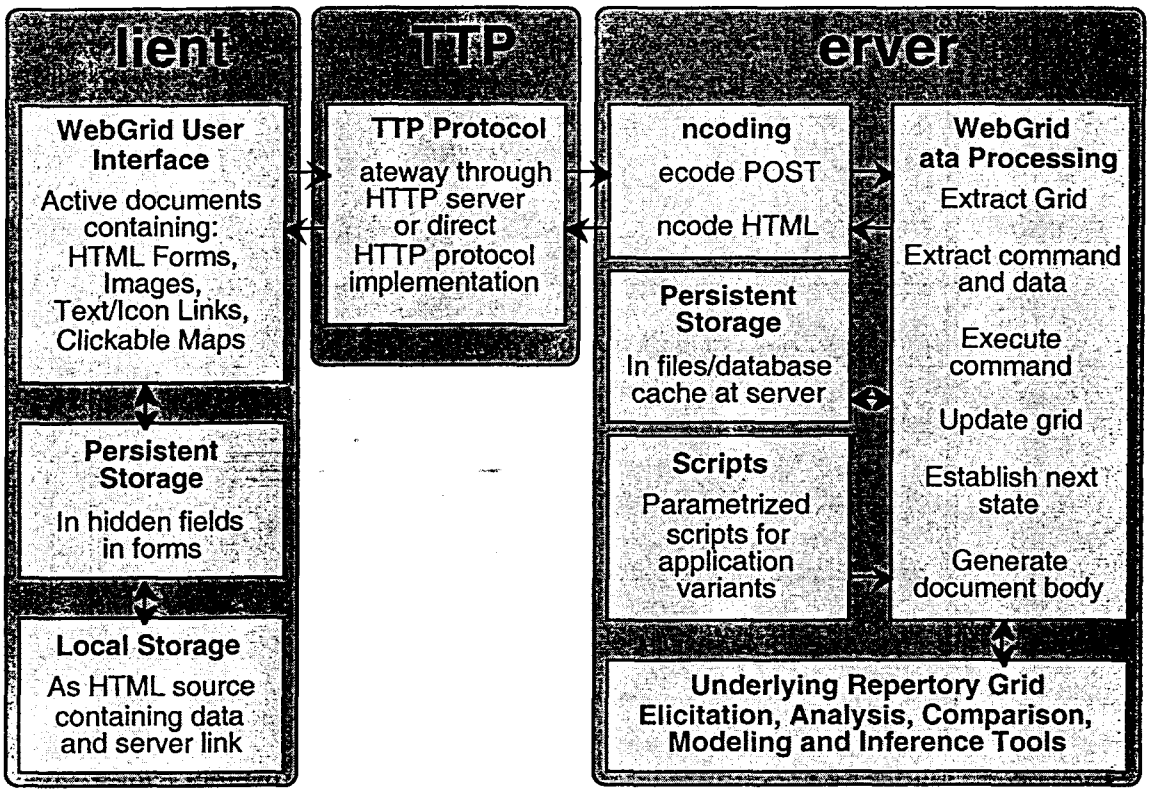


Figure 23 WebGrid-II architecture

One objective of the WebGrid-II design was to support users who wished to collect data on elements and constructs through their own methodology, pass them to WebGrid-II for further elicitation and analysis, and then return data and control to their own system.

Passing control to WebGrid-II is simple since all one has to do is send it a document with a data structure of the form shown in Figure 24. This is normally generated by WebGrid-II itself but is simple to output from any system that has gathered relevant data. Passing control back is also simple since WebGrid-II supports user-specified headers and trailers and these can contain JavaScript that returns the WebGrid-II data and control of the interaction to another server. WebGrid-II also supports additional hidden fields containing data from the original server which are passed back to it when control returns, thus enabling it to maintain state information.

The calling system may wish to restrict the WebGrid-II features available to the user and to exert some control over the interaction. WebGrid-II is designed such that the scripts may be simply edited to remove features and the system defaults to the expected behavior. It also supports a command language embedded in hidden fields that enable interaction to be controlled, for example, to take the user through a sequence of construct elicitations or ratings on prescribed constructs. The command language can also be used to program WebGrid-II to return control to another server when the prescribed interactions are complete.

These capabilities have been used to support a colleague in a management department in Florida who is conducting research on 360-degree survey methodologies (Jones and Bearley, 1966). He has developed his own form for element collection, and the demonstration form designed to show him how to do this is shown in Figure 24. It collects some demographic data which will not be used by WebGrid-II but just sent in with the grid data. It then asks for the names of 6 managers, characterized intensionally.

Figure 24 Customized data collection front-end to WebGrid-II

It is also possible for the other programs to ignore the user interface portions of a WebGrid-II document and use it as a server anyway. Tennison (1997a,b; Tennison and Shadbolt, 1998) has used this approach to make WebGrid-II a service integrated with her APECKS (Adaptive Presentation Environment for Collaborative Knowledge Structuring) system on the web. APECKS provides a virtual environment where people who are not in the same place in the real world can meet and talk, and supports a cyclical process of construction, comparison and discussion about knowledge representations through which an organizational memory is gradually constructed.

6 Conclusions

Repertory grid knowledge acquisition and modeling tools have been widely used in the development of knowledge-based systems, and their availability on the web supports distributed knowledge modeling. This article has demonstrated how WebGrid operates to elicit and model knowledge, how grid analysis can generate hierarchical knowledge structures from extended grids, how various sources of knowledge may be compared, and how other systems may be integrated with WebGrid. Access to WebGrid is freely available through the URL: **http://repgrid.com/WebGrid**.

Acknowledgments

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. We are grateful to Paul Compton for the opportunity to present this paper at PKAW, and for many stimulating discussions at KAW meetings world-wide.

References

Boose, J.H. (1984). Personal construct theory and the transfer of human expertise. **Proceedings AAAI-84**. pp.27-33. California, American Association for Artificial Intelligence.

Boose, J. H., Bradshaw, J. M., Koszarek, J.L. and Shema, D. B. (1993). Knowledge acquisition techniques for group decisions support. **Knowledge Acquisition 5**(4) 405-447.

Gaines, B.R. (1991a). Integrating rules in term subsumption knowledge representation servers. **AAAI'91: Proceedings of the Ninth National Conference on Artificial Intelligence**. pp.458-463. Menlo Park, California, AAAI Press/MIT Press.

Gaines, B.R. (1991b). An interactive visual language for term subsumption visual languages. **IJCAI'91: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence**. pp.817-823. San Mateo, California, Morgan Kaufmann.

Gaines, B.R. (1993). A class library implementation of a principled open architecture knowledge representation server with plug-in data types. **IJCAI'93: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence**. pp.504-509. San Mateo, California, Morgan Kaufmann.

Gaines, B.R. (1994). Class library implementation of an open architecture knowledge support system. **International Journal Human-Computer Studies 41**(1/2) 59-107.

Gaines, B.R. (1995a). Porting interactive applications to the web. **4th International World Wide Web Conference Tutorial Notes**. pp.199-217.

Gaines, B.R. (1995b). Structured and unstructured induction with EDAGs. Fayyad, U.M. and Uthurusamy, R., Ed. **Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)**. pp.124-129. Cambridge, MA, MIT Press.

Gaines, B.R. (1996). Transforming rules and trees into comprehensible knowledge structures. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R., Ed. **Knowledge Discovery in Databases II**. pp.205-226. Cambridge, Massachusetts, AAAI/MIT Press.

Gaines, B.R. and Compton, P. (1995). Induction of ripple-down rules applied to modeling large databases. **Journal for Intelligent Information Systems 5**(3) 211-228.

Gaines, B.R. and Shaw, M.L.G. (1993). Basing knowledge acquisition tools in personal construct psychology. **Knowledge Engineering Review 8**(1) 49-85.

Gaines, B.R. and Shaw, M.L.G. (1994). Using knowledge acquisition and representation tools to support scientific communities. **AAAI'94: Proceedings of the Twelfth National Conference on Artificial Intelligence**. pp.707-714. Menlo Park, California, AAAI Press/MIT Press.

Gaines, B.R. and Shaw, M.L.G. (1998). Developing for web integration in Sisyphus-IV: WebGrid-II experience. Gaines, B.R. and Musen, M.A., Ed. **Proceedings of Eleventh Knowledge Acquisition Workshop**. pp.http://ksi.cpsc.ucalgary.ca/KAW/KAW98/gaines/.

Jones, J.E. and Bearley, W.L. (1966). **360° Feedback: Strategies, Tactics, and Techniques for Developing Leaders**. Amherst, MA, HRD Press.

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

Shaw, M.L.G. (1979). Conversational heuristics for eliciting shared understanding. **International Journal of Man-Machine Studies 11** 621-634.

Shaw, M.L.G. (1980). **On Becoming A Personal Scientist: Interactive Computer Elicitation of Personal Models Of The World**. London, Academic Press.

Shaw, M.L.G. (1985). Communities of knowledge. Epting, F. and Landfield, A.W., Ed. **Anticipating Personal Construct Psychology**. pp.25-35. Lincoln, Nebraska, University of Nebraska Press.

Shaw, M.L.G. and Gaines, B.R. (1983). A computer aid to knowledge engineering. **Proceedings of British Computer Society Conference on Expert Systems**. pp.263-271. Cambridge, British Computer Society.

Shaw, M.L.G. and Gaines, B.R. (1991a). Extending electronic mail with conceptual modeling to provide group decision support. **COCS'91: Proceedings of Conference on Organizational Computing Systems**. pp.153-158. New York, ACM Press.

Shaw, M.L.G. and Gaines, B.R. (1991b). Supporting personal networking through computer networking. **Proceedings of CHI'91: Human Factors in Computing Systems**. pp.437-438. New York, ACM Publications.

Shaw, M.L.G. and Gaines, B.R. (1993). Group knowledge elicitation over networks. Bramer, M.A. and Macintosh, A.L., Ed. **Research and Development in Expert Systems X. Proceedings of British Computer Society Expert Systems Conference**. pp.43-62. Cambridge, UK, Cambridge University Press.

Shaw, M.L.G. and Gaines, B.R. (1995). Comparing constructions through the web. Schnase, J.L. and Cunnius, E.L., Ed. **Proceedings of CSCL95: Computer Support for Collaborative Learning**. pp.300-307. Mahwah, New Jersey, Lawrence Erlbaum.

Shaw, M.L.G. and Gaines, B.R. (1998). WebGrid II: Developing Hierarchical Knowledge Structures from Flat Grids. Gaines, B.R. and Musen, M.A., Ed. **Proceedings of Eleventh Knowledge Acquisition Workshop**. pp.http://ksi.cpsc.ucalgary.ca/KAW/KAW98/shaw/.

Shaw, M.L.G. and Gaines, B.R. (1999). Modeling the social practices of users in Internet communities. Kay, J., Ed. **UM99: User Modeling: Proceedings of the Seventh International Conference**. pp.77-86. New York, Springer.

Tennison, J. (1997a). APECKS: Adaptive Presentation Environment for Collaborative Knowledge Structuring. Department of Psychology, University of Nottingham. http://www.psychology.nottingham.ac.uk/staff/Jenifer.Tennison/APECKS/.

Tennison, J. (1997b). Linking APECKS to WebGrid-II. Department of Psychology, University of Nottingham. http://www.psychology.nottingham.ac.uk/staff/Jenifer.Tennison/APECKS/WebGrid.html.

Tennison, J. and Shadbolt, N.R. (1998). APECKS: a Tool to Support Living Ontologies. Gaines, B.R. and Musen, M.A., Ed. **Proceedings of Eleventh Knowledge Acquisition Workshop**. pp.http://ksi.cpsc.ucalgary.ca/KAW/KAW98/tennison/.

Tomiyama, T. (1992). The technical concept of IMS. RACE Discussion Paper, No. RA-DP2, Research into Artifacts, Center for Engineering, The University of Tokyo.