Knowledge Acquisition, Modeling and Inference through the World Wide Web

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

Abstract: The development of knowledge-based systems involves the management of a diversity of knowledge sources, computing resources and system users, often geographically distributed. The knowledge acquisition, modeling and representation communities have developed a wide range of tools relevant to the development and management of large-scale knowledge-based systems, but the majority of these tools run on individual workstations and use specialist data formats making system integration and knowledge interchange very problematic. However, widespread access to the Internet has led to a new era of distributed client-server computing. In particular, the introduction of support for *forms* on World Wide Web in late 1993 has provided an easily programmable, cross-platform graphic user interface that has become widely used in innovative interactive systems. This article reports on the development of open architecture knowledge management tools operating through the web to support knowledge acquisition, representation and inference through semantic networks and repertory grids.

1 Introduction

Advances in computing technology that generate new opportunities for interactive computing generally also introduce new human factors considerations. In the 1980s the major new technology was the personal computer with graphic interaction leading to the *direct manipulation* paradigm (Shneiderman, 1983). In the 1990s it is the interconnection of such personal computers world-wide through the Internet that is leading to innovation through client-server distributed interaction. New paradigms for *groupware* and *agent-based computing* are some well-known outcomes. However, there are other impacts of the partitioning of interactive functionality between client and server that are less obvious, equally momentous, and highly relevant to the support of the groupware and agent paradigms.

The explosive growth of the World Wide Web (the *web*, Berners-Lee, Cailliau, Luotonen, Nielsen and Secret, 1994a) in the 1990s is a significant example of new trends in human-computer interaction evolving through grass-roots development world-wide. In particular, web *forms* support a graphic user interface (GUI) that can be simply programmed as an interactive document. Anyone who can use a word processor can become a user interface designer. The embedding of user interface elements within a document in itself promotes a new approach to interaction in which the interface emerges from the content and context. The flexibility of a dynamically programmed *emergent interface* may itself constitute a new paradigm for human-computer interaction in the 1990s.

This article describes some experience in porting knowledge acquisition tools from personal computers to operate on the web through emergent interfaces embedded in active documents. The development of 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. Knowledge acquisition methodologies and tools have developed to take account of these issues by using 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

1

the models in operation, and model comparison systems to draw attention to anomalous variations between experts.

However, existing knowledge acquisition tools are largely based on personal computers and graphic workstations, and their use in a distributed community involves moving software and often computers from site to site. 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 initial objective of the work reported in this paper has been to use the World Wide Web to support distributed knowledge acquisition by porting existing knowledge acquisition tools to operate through the web. 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.

The next section gives an overview of the web architecture and protocols, and following sections illustrate how knowledge acquisition systems have been ported to the web to operate in a distributed client-server environment.

2 The Architecture of the World Wide Web

The World Wide Web was conceived by Berners-Lee in March 1989 as a "hypertext project" to organize documents at CERN (Berners-Lee and Cailliau, 1990). The design involved a simple hypertext markup language that authors could enter through a word processor, distributed servers running on machines anywhere on the network, and access through any terminal. In March 1993 the web was still being presented as primarily a hypermedia retrieval system (Berners-Lee, 1993), but in November that year a development took place that radically changed the nature of the web. Andreessen (1993) issued NCSA Mosaic version 2 using tags to encode definitions of Motif widgets embedded within a hypermedia document, and allowed the state of those widgets within the client to be transmitted to the server. With the addition of such *forms* providing basic graphic user interface capabilities, the web protocols became the basis of general interactive, distributed, client-server information systems.

The Hypertext Transfer Protocol (HTTP) used between web clients and servers is a generic, stateless, object-oriented protocol that uses Multipurpose Internet Mail Extensions content encoding (MIME, Borenstein and Freed, 1993) to transmit arbitrary data. The content encoding used for web documents is the Hypertext Markup Language (HTML, Berners-Lee, Connolly and Muldrow, 1994b) which is a document type definition in the ISO 8879 Standard Generalized Markup Language (SGML, Goldfarb, 1990). The adoption of an SGML tagged encoding schema allows web documents to include not only text, typographic and multimedia material but also to carry arbitrary additional data. HTML has been standardized at four levels:

- level 0 - text with embedded links;
- level 1 - adds typographic text with embedded images;
- level 2 - adds embedded graphic user interfaces (forms);
- level 3 - adds tables and other capabilities.

Figure 1 shows the client-server architecture of the web. A client accesses servers on the Internet using various protocols. It communicates with various helper applications that extend its functionality. When it accesses a web server using the HTTP protocol that server can also access various helper applications through server gateways.

hypermedia
document
archives

client-server transactions
responding by generating
dynamic documents

file
archives

**FTP
Server**

**Web
Server**

**Auxiliary
Servers**

common
gateway
interface

news
archives

**News
Server**

FTP

HTTP

databases

NNTP

computational
services

text
archives

**Gopher
Server**

Gopher

Internet
protocols

*Servers at local
or remote sites*

*Local
workstation
with TCP/IP
connection to
Internet*

**Client
helpers**

**Web
Client**

sounds,
movies,
spread-sheets
arbitrary applications

hypermedia
browser
GUI

User

Figure 1 Client-server architecture of World Wide Web

Knowledge acquisition systems can be implemented on the web as: *client helpers* running locally on the same machine as the web browser and communicating with it through its application program interface; or *auxiliary servers* running remotely on the same machine or network as the web server and communicating with it through its common gateway interface. The advantages of client helper implementation are that existing knowledge acquisition tools can be used, and the full power of the local machine graphic user interface is available. The disadvantages are that the tool needs to be implemented for all platforms, minimally UNIX, PC and Mac, and the tool has to be distributed to all users. The advantages of auxiliary server implementation are that a single implementation for one server platform only needs to be developed and maintained, and the program does not have to be issued to other sites. The disadvantages are the limited capabilities of the user interfaces encoded in HTML forms and the transmission delays of operating through the network.

Figure 2 gives an overview of what is involved in factoring an application to operate in client-server mode through the web using existing web browsers as clients. At the left the client component consists of a web browser providing a graphic-user interface through:
• HTML documents supporting typographic text
• Images supporting graphics, possibly embedded in the text
• Hyperlinks attached to text or images such as icons
• Clickable maps supporting returning to the server the location of a mouse click in an image
• HTML forms providing a programmable GUI returning data entered to the server.

At the left center, the client communicates through the net with a web server that is either an implementation of the HTTP protocol in the application server component, or a standard HTTP server providing a gateway to the application server component.

At the right the server component splits into 4 parts:
• Encoding the application output as HTML and decoding the incoming POST parameters
• Representation of the application output as a document and extracting incoming data
• Data processing as the non-interface parts of the application
• Parameters as separate data structures or scripts providing application variants and dialog in different languages.

3
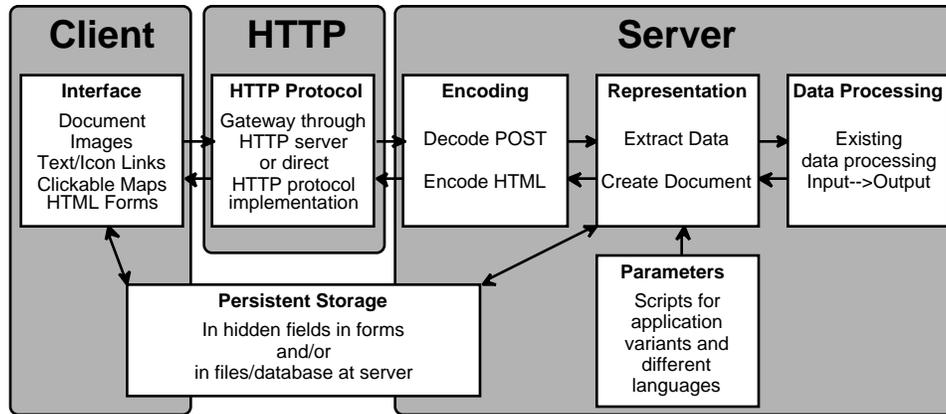
Figure 2 Factoring an application for client-server operation through the web

In addition, at the lower left the persistent storage for the application has to be factored to be represented either in hidden fields which can be stored as an HTML document at the client or in files or a database stored at the server, or some combination of both.

The following sections exemplify the implementation of major knowledge acquisition tools as client helpers and auxiliary servers operating through the web.

3 Semantic Networks on the Web

The graphic representation of formal knowledge structures in semantic networks is common in most knowledge acquisition tools because experts find it easier to understand and critique the knowledge model when presented in diagrammatic form. The graphic user interface available through HTML is currently inadequate to support diagrammatic editing, but existing semantic net tools can be interfaced to the web as client helpers. They can then use the web as a multimedia data management system for informal knowledge that is linked to the formal structures developed from it.

3.1 Semantic Network Editing through a Client Helper

Figure 3 shows a semantic network editor in the front window communicating with a web browser in the rear window. The web document shown is the specification of the Sisyphus I room allocation problem (Linster, 1994). The decision procedures in the document are being translated into formal knowledge structures in a semantic network (Gaines, 1994) using KMap, a concept map editor for the Macintosh platform (Gaines and Shaw, 1995a; Gaines and Shaw, 1995b). The knowledge engineer has annotated the document with HTML anchor tags making the significant definitions available through hypertext links. When she clicks on a link in the web browser to access a definition the browser scrolls to that definition and also transmits its URL (uniform resource locator) to the semantic net editor.

The knowledge engineer copies text from the document, pastes it to the editor to name a knowledge element and clicks on one of the buttons at the top of the editor to create a graphic representation of that element. The editor automatically stores the URL sent by the browser as data attached to that element, so that the formal structure is linked to the original text and can be used to access it on the web. For example, in Figure 3 the cursor is over the concept "group head without office" and has changed to a button shape to indicate that linked data is available. Clicking on the concept sends a request through the browser to the server to show the relevant text which appears at the top of the browser as shown.

Figure 3 Generation of a semantic net linked to a web document

3.2 Semantic Networks Uploaded as Clickable Maps

The use of a semantic network editor as a client helper enables formal knowledge structures to be developed but does not in itself make them available through the web for inspection by others at remote sites. This is done by providing a facility in the editor for uploading the knowledge structure from the helper to the web server using the capability of the browser to post data to the server under control of the editor—the knowledge engineer simply selects the "Upload" option in a popup menu that appears when she clicks outside the knowledge structure. The server is then able to transmit the knowledge structure as an image in the Compuserve GIF format which acts as a clickable map in any web browser without requiring access to the editor. Figure 4 shows the semantic network of Figure 3 being used as a clickable map. The cursor is again over the concept "group head without office" and has changed to a hand to indicate that linked data is available. Clicking on the concept sends a request to the browser to show the relevant text which will appear at the top of the browser as before.
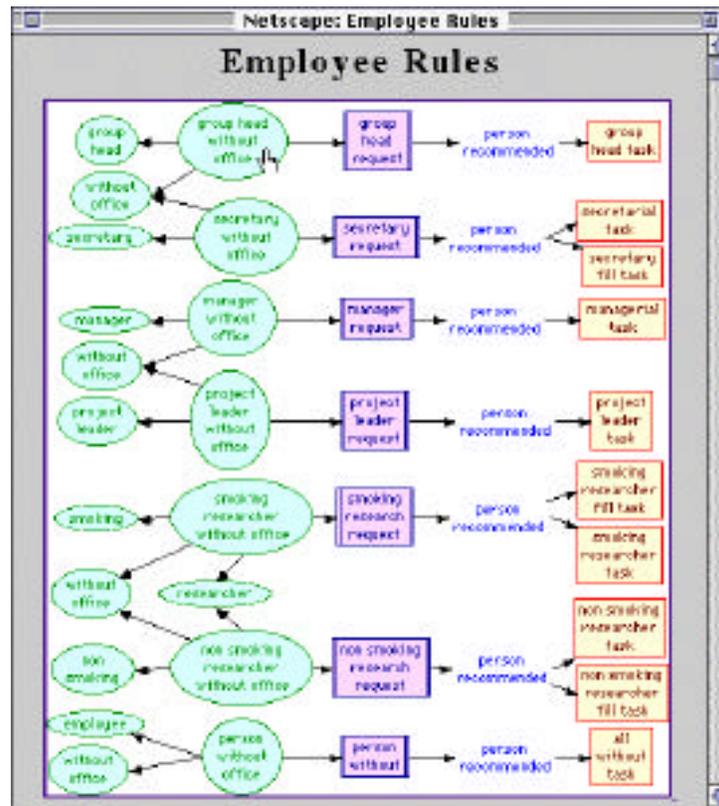
Figure 4 Semantic net uploaded to web as a clickable map

Figure 5 shows the HTML generating the document in Figure 4. The line beginning "<IMG SRC" gives the URL of the image and defines it through "ISMAP" as a clickable map. When the browser receives this line it requests that the image be sent, and the server loads the semantic network, converts it to a GIF and returns it to the browser for display. The line beginning "<A HREF" gives the URL to which the coordinates are to be sent when the mouse is clicked with the cursor within the map. The two URLs are identical because the server can distinguish a request for an image from a notification that a map has been clicked. When the server receives the coordinates of the click it again loads the semantic network, computes what node has been clicked, retrieves the hidden data and sends the browser a redirection command indicating the URL that it should retrieve.

6

```
<HTML><HEAD><TITLE>Employee Rules</TITLE></HEAD>
<BODY><H1 ALIGN="CENTER">Employee Rules</H1>
<P ALIGN="CENTER">
<A HREF="http://www.noodle.org/WebMap/users/ferio/Employee%20Rules.k">
<IMG SRC="http://www.noodle.org/WebMap/users/ferio/Employee%20Rules.k" ISMAP>
</A></BODY></HTML>
```

Figure 5 HTML generating document in Figure 4

The HTML specifications of Figure 5 may be embedded within any document, making active semantic networks available as was done with the KWrite knowledge-based word processor (Gaines and Shaw, 1992; Gaines and Shaw, 1993c). Thus, a knowledge structure developed locally can be made available globally together with its linked derivation so that knowledge acquisition and validation can proceed on a distributed basis. The network uploaded is the full data structure so that it also becomes available for download to those having the semantic network editor, and it can be compiled at the server to run as part of a knowledge base operating through a server gateway and accessible through the web. Hence, a web document containing links to semantic networks as shown in Figure 5 not only serves to display them as active diagrams, but also serves as a knowledge base allowing the networks to be loaded and used for problem solving. Moreover, since the document is available over the Internet, the inference system can be a server agent at a different site from the document that is sent the document URL, uses this to determine the URLs of the knowledge structures, fetches these using the HTTP protocol, and then uses them to manage an interactive problem-solving interaction across the web.

3.3 Semantic Network Editing through Java Applets

The preceding sub-sections have illustrated the use of a platform-specific client helper to create semantic networks linked to material on the web, coupled with the capability to upload the networks to a server and deliver them as clickable images that maintain the links. However, the incorporation of Java (Arnold and Gosling, 1996) support in web browsers now enables one to combine the two functions and offer platform-independent semantic network editors through the web that can be programmed through JavaScript to support interaction with other web capabilities.

Figure 6 shows Flores-Mendez's (1997) implementation of JMap, a Java-based version of Kremer's (1996) CMap concept mapping system, being used to create the semantic networks previously shown. Cmap was itself a reimplementation of KMap to operate as a Netscape plugin and support the editing of concept map and semantic network editing on the web. The Java implementation essentially downloads the graphic software dynamically avoiding the need for platform specific plugins. It makes the application program interface of the software available through JavaScript embedded in the web document so that HTML forms may be used to provide a programmable user interface embedded in the same document as the semantic network.

The Java applet supports direct manipulation of the graphic user interface so that nodes can be selected and dragged to new positions, and JMap provides multi-level undo/redo capabilities, so that the overall user interface is as supportive as in a stand-alone application. Embedding semantic networks in active documents on the web enables the documents to act as complete knowledge bases and to be used used to drive inference engines as was done with KWrite (Gaines and Shaw, 1992). Since the Java applet is also able to communicate with the server it is possible to use the concept map as an extension to the user interface functionality provided in HTML forms, and hence to program client-server systems customized for specific applications. In particular, CMap and JMap, in both their web and stand-alone versions, are able to interoperate through a reflector at the server to provide groupware development of concept maps and semantic networks whereby changes made at one location are propagated to other sites.
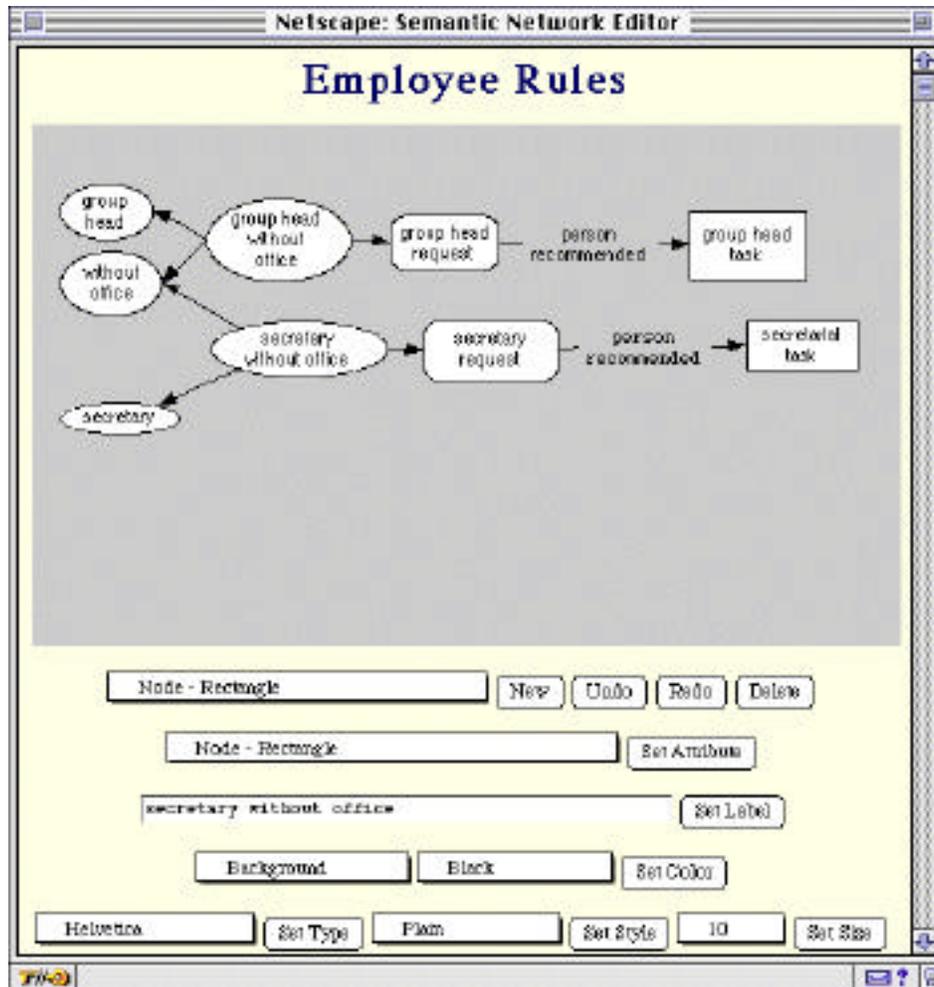
Figure 6 Editing a semantic net on the web through an interactive Java applet

4 Repertory Grid Client-Server Implementation

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; Ford, Bradshaw, Adams-Webber and Agnew, 1993; Gaines and Shaw, 1993a). 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. This is feasible because the primary data input format is through rating scales, and this can be done effectively by using popup menus which are available through the HTML graphic user interface.

The following sub-sections describe experience in porting repertory grid knowledge elicitation, modeling and inference tools to operate through the World Wide Web. Applications to an igneous rocks classification test case, group comparison of manufacturing system models, and inference with a standard inductive modeling dataset are described. In addition, the techniques are described which were used to support various aspects of the interface such as returning graphic analyses, storing data in documents, providing multi-level undo through the web browser "Back" command, and supporting multiple languages and customization.

8

4.1 Knowledge Elicitation through WebGrid

Figure 7 shows the initial screen of WebGrid, a repertory grid elicitation tool operating through the web using a standard web browser and requiring no client helpers—an example of the client-server architecture shown in Figure 2. The HTML form 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.

Figure 7 Repertory grid elicitation initial screen

9

The problem domain is that of a proposed Sisyphus III problem on the classification of igneous rocks (Shadbolt, 1996), and the interaction reproduces one of the example datasets. 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 8 eliciting a construct from a triad of elements.

Figure 8 Construct elicitation from a triad

10

The user clicks on a radio button to select an element which he construes as different from the other two and enters terms characterizing the construct. The pictures of the rocks were entered as HTML annotation of the elements by the knowledge engineer. The lower part of the screen allows the user to specify the data type with the default being a standard grid rating scale. He retains the default option and clicks on "Done". The server generates the screen shown on the left of Figure 9 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. He 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.
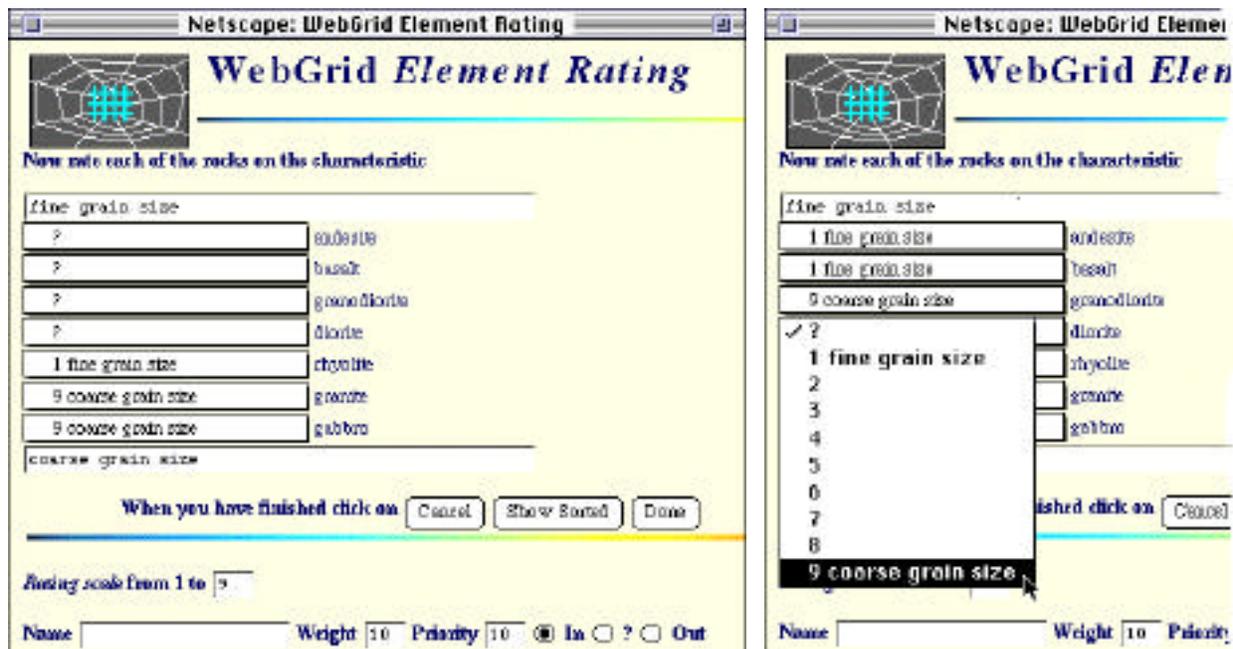
Figure 9 Rating elements on constructs
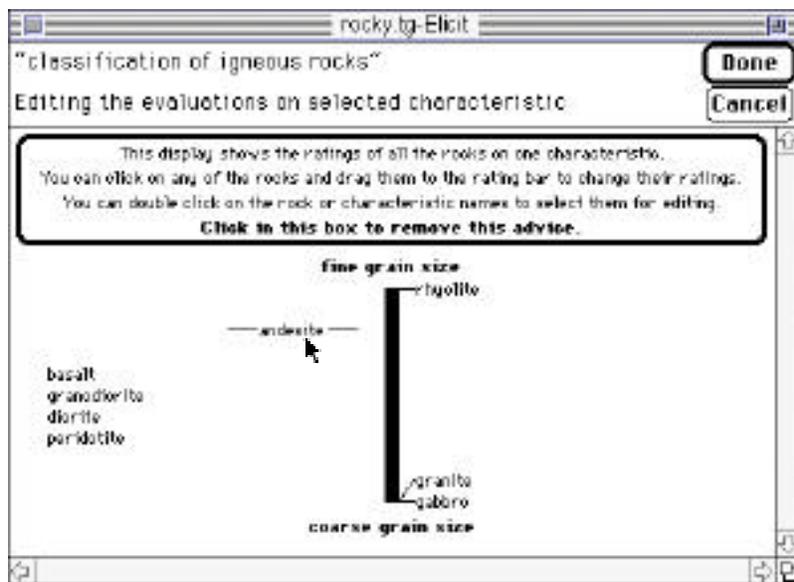
Figure 10 Comparison with previous stand-alone application

11

It is interesting to compare the entry of ratings through popup menus on the web as shown on the right of Figure 9 with their entry through the specially programmed graphic user interface of the stand-alone application RepGrid (CPCS, 1993) as shown in Figure 10. In RepGrid the user rates elements by dragging them to the appropriate position on the rating scale. This has the advantage that the relative positions of all elements are immediately visible whereas in WebGrid the relative positions do not change as the ratings are entered. The "Show Sorted" button visible near the bottom right of the left hand screen shown in Figure 9 allows the user to request that the server redisplay the elements sorted by rating so that this feature of RepGrid may be emulated.

We have undertaken informal comparisons of WebGrid and RepGrid in classes of students using both tools and have found no overwhelming preference for one form of interface over the other. The popup menu and the customized widget seem equally effective in this application which is not what we had expected.

Clicking on the "Done" button in Figure 9 sends the ratings back to the server which generates the status screen shown in Figure 11 (shown after 8 elements and 5 constructs have been added). This draws the user's attention to matching constructs, suggesting a new element be added to break the match, to matching elements suggesting a new construct be added to break the match, and also offers the opportunity to elicit another construct from a pair or triad of elements. The elements and constructs are listed in selection boxes allowing selected items to be deleted and edited, new items to be added, matches to be shown, and HTML annotation to be added to elements. Further buttons enable the data elicited to be saved or to be displayed or analyzed in various ways.
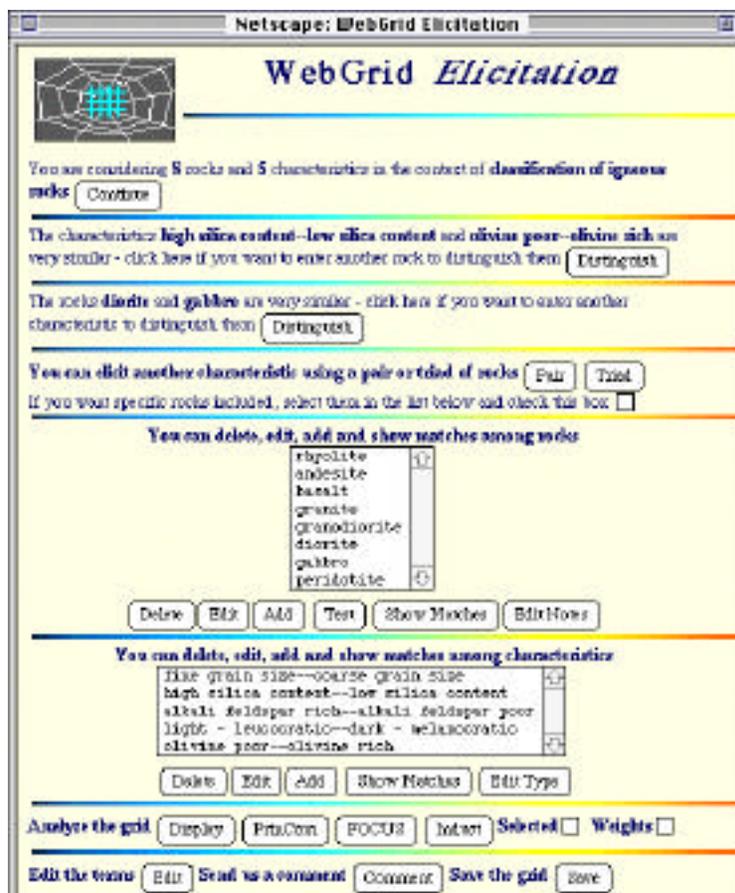
Figure 11 Status screen showing matches

12

4.2 Returning Graphic Analyses through the Web

Figure 12 shows the output returned when the "FOCUS" button is used to develop a hierarchical knowledge model by sorting the elements and constructs so that similar ones are close together (Shaw, 1980). The results of analysis are usually presented as graphic output. This is generated at the server, converted to GIF format and returned to the client where it can be examined and saved if required. It can be seen that the four elements, *gabbro*, *diorite*, *granodiorite* and *granite* are very similar indicating that the constructs entered may be inadequate to distinguish them and that further discriminating constructs are required. It can also be seen that the two constructs, *low silica content—high silica content* and *light - leucratic—dark - melanocratic* are very similar indicating that they do not provide independent discrimination of the elements entered and that further elements might be elicited.

Figure 13 shows the output returned when the "PrinCom" button is used to provide a principal components analysis of the grid by rotating it in vector space to give maximum separation of elements in two dimensions (Slater, 1976; Gaines and Shaw, 1980). The results of analysis are again graphed, converted to GIF format and returned to the client where they can be examined and saved if required. It can be seen that there are two major dimensions, one being grain size and the other being four highly correlated constructs, again indicating that it would be appropriate to enter more elements.
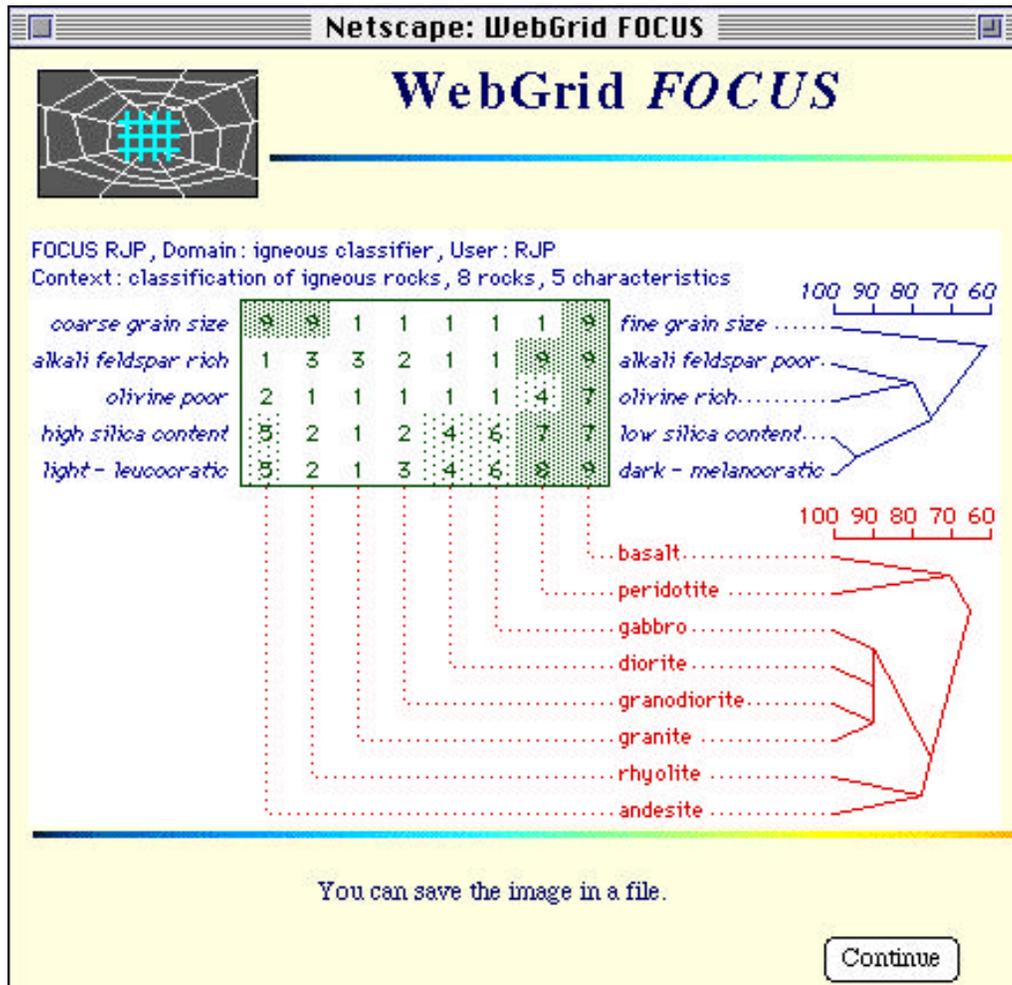
Figure 12 FOCUS hierarchical cluster analysis of grid

13
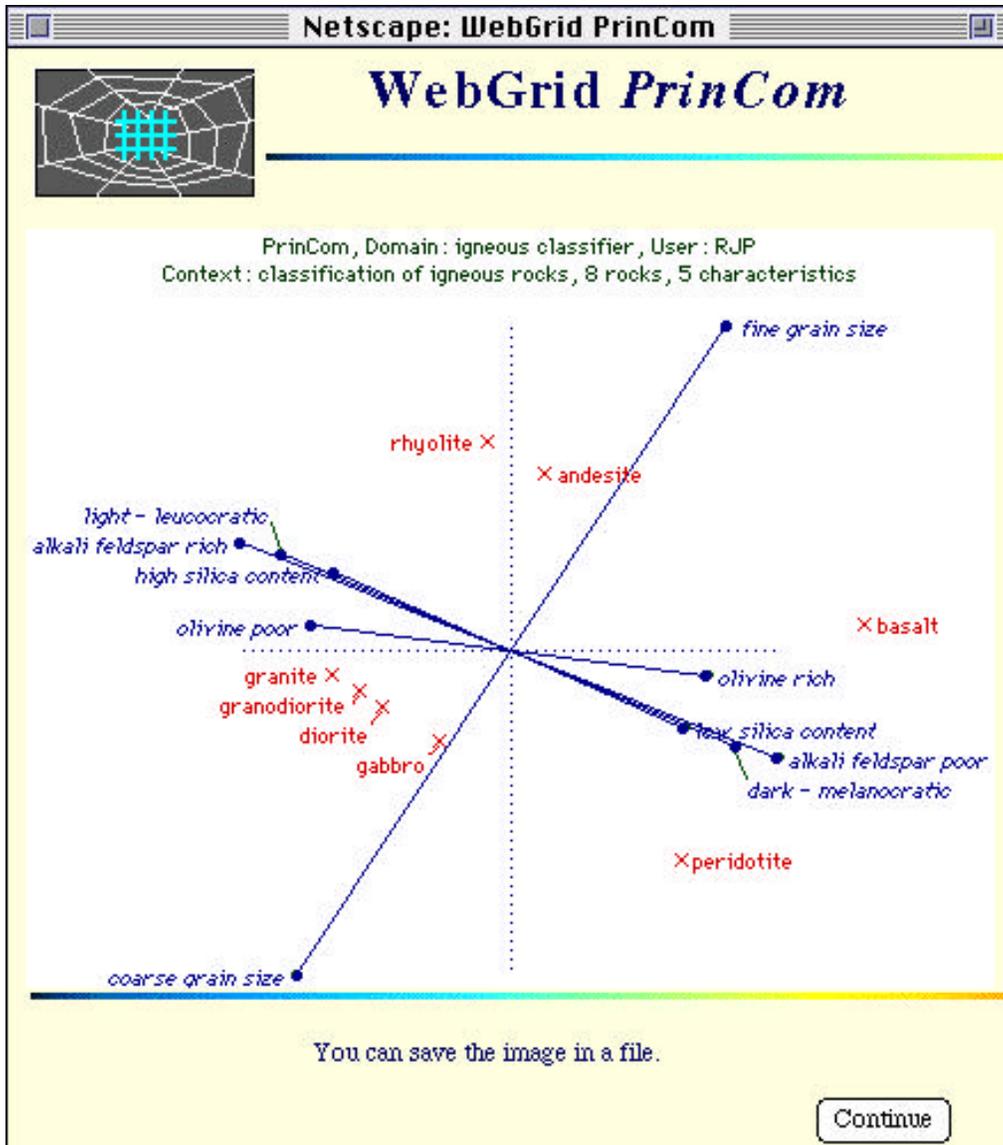
Figure 13 PrinCom principal components analysis of grid

4.3 Storing Data in Web Documents

The knowledge acquisition server is designed to operate on a stateless basis in which data is not normally stored at the server unless it is specifically uploaded. This is achieved through the server storing all the grid data in hidden fields in the HTML document which do not display but are returned to the server along with the entered data. Figure 14 shows the first part of the HTML generating the document of Figure 11 where the hidden fields encoding the grid are apparent. The user can store the data at the client machine by saving the source of the document in a file, and can continue the elicitation and analysis at any time by loading this file. This is possible because the server has returned a full URL to itself in the form ACTION. When the document is loaded locally and user action is recommenced it will reconnect to the server using this URL. One advantage of this approach is that no data is stored at the server and no management of persistent data is necessary, and another is that the user can use the "Back" command in the browser freely as a multiple-level "Undo" without any special support at the server.

14

```
<HTML><HEAD><TITLE>WebGrid Elicitation</TITLE></HEAD>
<BODY  BGCOLOR="FFFFDD"  TEXT="000066"><IMG  SRC="http://tiger.cpsc.ucalgary.ca/WebGrid/WebGrid.gif"
    ALIGN="left" ALT="WebGrid Icon">
<H1 ALIGN="center">WebGrid <I>Elicitation</I></H1><IMG SRC="/IMG/RAINBOWSEP.GIF"><BR>
<FORM ACTION="http://tiger.cpsc.ucalgary.ca/WebGrid/Main.k" METHOD=POST>
<INPUT TYPE="hidden" NAME="WebGrid" VALUE="11|-1377813985|6/9/96|9:39:55 AM">
<INPUT TYPE="hidden" NAME="Body" VALUE="{BODY BGCOLOR=\FFFFDD\ TEXT=\000066\}">
<INPUT TYPE="hidden" NAME="Line" VALUE="{IMG SRC=\/IMG/RAINBOWSEP.GIF\}{BR}">
<INPUT TYPE="hidden" NAME="Head" VALUE="2|0|3|9|5|8|R1 9|">
<INPUT    TYPE="hidden"    NAME="Labels"    VALUE="RJP|igneous    classifier|classification    of    igneous
    rocks|RJP||characteristic|characteristics|rock|rocks">
<INPUT TYPE="hidden" NAME="C0" VALUE="R1 8 0 10 10 1 9||fine grain size|coarse grain size">
<INPUT TYPE="hidden" NAME="C1" VALUE="R1 9 1 10 10 1 9||high silica content|low silica content">
<INPUT TYPE="hidden" NAME="C2" VALUE="R1 8 2 10 10 1 9||alkali feldspar rich|alkali feldspar poor">
<INPUT TYPE="hidden" NAME="C3" VALUE="R1 8 3 10 10 1 9||light - leucocratic|dark - melanocratic">
<INPUT TYPE="hidden" NAME="C4" VALUE="R1 8 4 10 10 1 9||olivine poor|olivine rich">
<INPUT TYPE="hidden" NAME="E0" VALUE="|0|1|2|1|0|rhyolite|{IMG SRC=\/rocky/rhyolite.jpg\}">
<INPUT TYPE="hidden" NAME="E1" VALUE="|0|4|0|4|1|andesite">
<INPUT TYPE="hidden" NAME="E2" VALUE="|0|6|8|8|6|basalt">
<INPUT TYPE="hidden" NAME="E3" VALUE="|8|0|2|0|0|granite|{IMG SRC=\/rocky/granite.jpg\}">
<INPUT TYPE="hidden" NAME="E4" VALUE="|8|1|1|2|0|granodiorite">
<INPUT TYPE="hidden" NAME="E5" VALUE="|8|3|0|3|0|diorite">
<INPUT TYPE="hidden" NAME="E6" VALUE="|8|5|0|5|0|gabbro|{IMG SRC=\/rocky/gabbro.jpg\}">
<INPUT TYPE="hidden" NAME="E7" VALUE="|8|6|8|7|3|peridotite">
You are considering <B>8</B> rocks and <B>5</B> characteristics in the context of <B>classification of igneous
    rocks</B>
<INPUT TYPE="Submit" NAME="Continue" VALUE="Continue">
<INPUT TYPE="hidden" NAME="CPair" VALUE="1|3">
<IMG  SRC="/IMG/RAINBOWSEP.GIF"><BR>The  characteristics  <B>high silica content--low silica content</B>
    and <B>light - leucocratic--dark - melanocratic</B> are very similar -
click here if you want to enter another rock to distinguish them
<INPUT TYPE="Submit" NAME="CMatch" VALUE="Distinguish">
```

Figure 14 Initial section of HTML showing hidden fields in document in Figure 9

The embedded data in Figure 14 is readily extracted for use in other applications, and WebGrid also provides facilities to export the data in other interchange formats for use in offline programs such as RepGrid and spreadsheets or statistical applications.

4.4 Supporting Multiple Languages and Customization on the Web

It has become standard practice in interactive computer applications to embed all user dialog in some form of "resource file" that can be edited quite separately from the application code. This enables applications to be customized for different purposes and localized for use in different languages by editing the resource file without having to recompile the code.

It is worthwhile taking a similar approach to client-server systems on the web, and putting all user dialog into separate script files that can be edited in a text editor. This allows the server to operate in multiple languages based on a simple parameter embedded in the form with values, "English", "French", "German" and so on. The script files can usually take the form of HTML with embedded macros that are expanded from parameters set up by the code.

Figure 15 shows the WebGrid script generating the HTML which generates the status screen of Figure 7. The backslash character "\" is used to demarcate the script sections and to introduce macro codes. A line containing only a backslash is a section terminator. A line containing a double backslash followed by a letter starts a section named by that letter. In the code the start of a new section causes a case statement jump to a routine associated with that letter.

```
WebGrid Grid Status\
<IMG SRC="http:/\S/WebGrid/WebGrid.gif" ALIGN="left" ALT="WebGrid Icon">
<H1 ALIGN="center">WebGrid <I>Setup Grid</I></H1>\I
\
\\N
You are considering <B>\0</B> \1 and <B>\2</B> \3 in the context of <B>\D</B>
\
\\U
<P>
\#0The name of the person from whom the grid is being elicited\
\#Enter your name\
<BR><B>Name </B><INPUT NAME="User" VALUE="\u"><BR><BR>
\
\\F
The domain about which the grid is being elicited<BR>
<B>Domain </B><INPUT NAME="Domain" VALUE="\d"><BR><BR>
The context or purpuse for eliciting this grid<BR>
<B>Context </B><INPUT NAME="Context" VALUE="\D" SIZE=60><BR><BR>
Singular and plural terms for elements and constructs, and rating scale range<BR>
<B>Element <INPUT NAME="ETerm" VALUE="\e" SIZE=15>
Elements </B><INPUT NAME="ETerms" VALUE="\E" SIZE=15><BR>
<B>Construct <INPUT NAME="CTerm" VALUE="\c" SIZE=15>
Constructs </B><INPUT NAME="CTerms" VALUE="\C" SIZE=15><BR>
<B>Rating scale from 1 to <INPUT NAME="High" VALUE="\R" SIZE=3>
<INPUT TYPE="Radio" NAME="Mode" VALUE="R" \4> Ratings
<INPUT TYPE="Radio" NAME="Mode" VALUE="N" \4> +Names
<INPUT TYPE="Radio" NAME="Mode" VALUE="C" \4> +Categories
<INPUT TYPE="Radio" NAME="Mode" VALUE="V" \4> +Numbers</B>
\
\\E
\I<P>
\#0<b>List on separate lines at least six elements relevant to your context</b>
<TEXTAREA NAME="Elements" ROWS=7 COLS=60></TEXTAREA>\
\#You will be asked to enter your ratings for each \e on each \c in this grid\
\#You will be asked to develop your own \C based on the \E in this grid\
\#You need to enter some \E which you will be asked to rate on each \c in this grid<BR>
<TEXTAREA NAME="Elements" ROWS=7 COLS=60></TEXTAREA>\
\
\\B
<DIV ALIGN="RIGHT">
<P><B>When you are ready click on</B>
\#0<INPUT TYPE="Submit" NAME="Cancel" VALUE="Cancel">\
\#\
<INPUT TYPE="Submit" NAME="OK" VALUE="Done">
</DIV>
\
\\C
<P>\I<P>You can customize WebGrid (using HTML links if you wish)<BR>
<P><B>The following will replace the BODY statement allowing different backgrounds<BR>
<INPUT NAME="Bodying" VALUE="\B" SIZE=60><BR>
<P>The following will replace the horizontal line statement<BR>
<INPUT NAME="Lining" VALUE="\L" SIZE=60><BR>
<P>The following will replace the header at the top of each page<BR>
<TEXTAREA NAME="Heading" ROWS=5 COLS=60>\h</TEXTAREA>
<P>The following notes will appear at the bottom of each page</B><BR>
<TEXTAREA NAME="Trailing" ROWS=5 COLS=60>\H</TEXTAREA>\
\
\\
```

Figure 15 WebGrid script for the status screen of Figure 7

In the script text a backslash followed by a number indicates a parameter to be substituted by some data already set up in the code. A backslash followed by a "#" and a number introduces a list of possible strings to be selected on the basis of a numeric parameter set up by the code. It is simple to develop a macro substitution language that can be used to parameterize the output from a range of applications. The effort is amply repaid by the capability to change the user interface without recompilation, and to customize the application. Figure 16 shows the status screen of Figure 11 generated by a script whose text is in Spanish. The unchanged embedded English text shows where the macro substitutions are occurring.
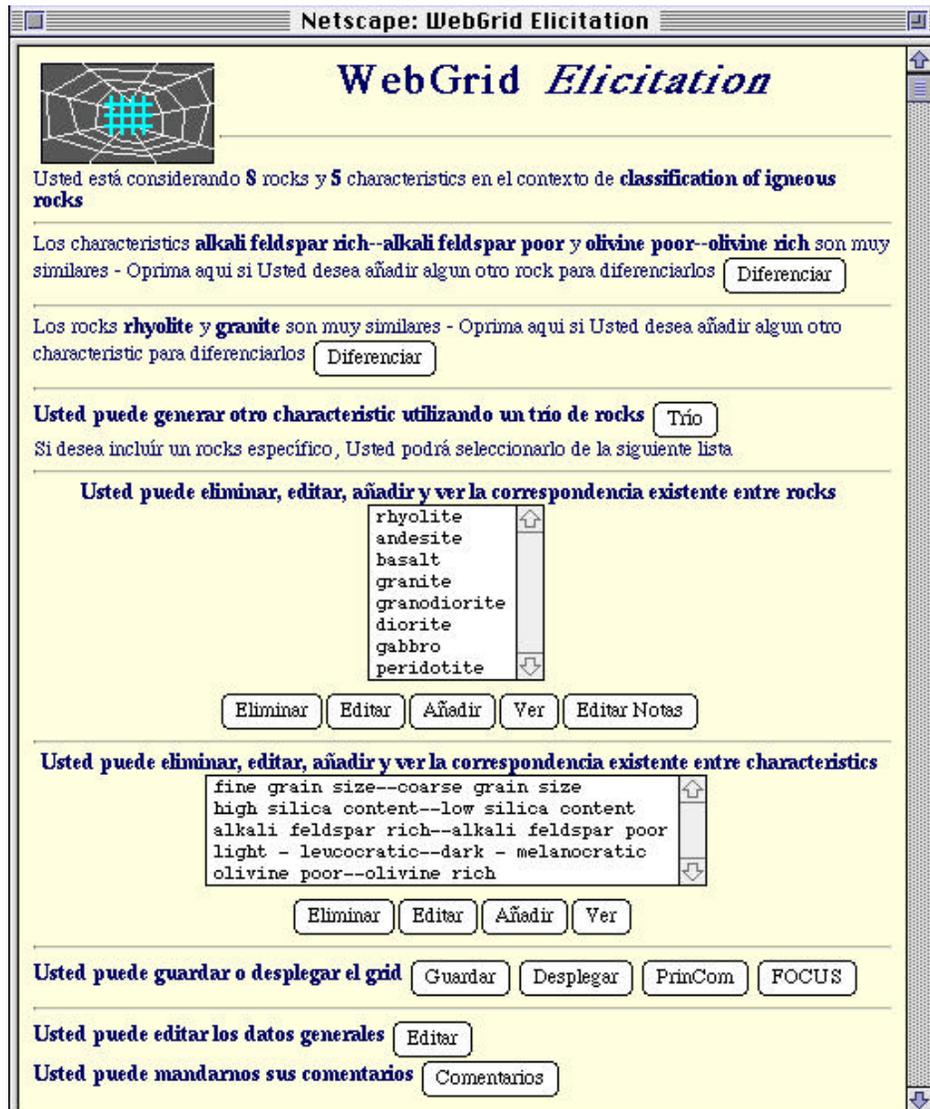
Figure 16 Status screen generated from script with Spanish text

4.5 Encoding Dynamic HTML Data from User for Display, Editing and Hidden Fields

To take full advantage of an application running on the web it is useful to be able to support the user entering HTML data which will be inserted in the output in appropriate places. For example, the fields at the bottom of Figure 7 allow the user to enter HTML data that customizes the user interface for the ensuing interaction.

17

User entered dynamic HTML data has to be translated between three different forms:
- Raw HTML which is received as POST data from the text box and output in other forms as part of the form
- Encoded HTML which is output to the text box for editing purposes and is to be interpreted as character data and not as tags, so that, for example, the "<" character needs to be encoded as the string "<"
- Encoded HTML which is stored in hidden fields and must neither be interpreted as character data nor as tags, so that some alternative character must be substituted for "<", ">" and "&"—WebGrid uses "{", "}" and "\" respectively, and does not allow these in data entry.

It is simple to write routines that translate between these formats and offer the user the opportunity to enter HTML data, such as personal notes and links to images, that is used as part of the application data.

4.6 Group Comparison of Repertory Grid Conceptual Structures

Once knowledge acquisition tools are available on the web it becomes possible to support distributed knowledge acquisition in which experts and knowledge engineers at different sites collaborate in developing a knowledge-based system. The repertory grid system described above has been extended to support the comparison of knowledge structures using an extension of the consensus/conflict/correspondence/contrast methodology described previously (Gaines and Shaw, 1989; Shaw and Gaines, 1989). When a grid has been developed by an expert at one site, another expert at another site can develop a grid based on it, either attempting to rate the elements on the other expert's constructs to evaluate consensus and conflict, or developing her own constructs to evaluate correspondence and contrast.

Figure 17 shows the initial screen for an expert eliciting a grid based on the elements in another expert's grid developed as part of a study of research on soft (reconfigurable) machines from an international intelligent manufacturing systems project (Gaines and Shaw, 1994). The elements are research projects, and the first expert has already developed a grid designed to make explicit the critical dimensions of the research. The second expert will develop another grid with the same elements and compare his conceptual model with that of the first expert. When the second expert has entered his name and clicked on "Done" the server generates screens for triadic elicitation and rating similar to those of Figures 8 and 9, and elicitation proceeds as before.

Figure 18 shows the status screen after six constructs have been elicited from the second expert, and it will be noted that a new "Compare" button is available underneath the analysis buttons in the lower part of the screen. Clicking on this generates the analysis shown in Figure 19 where each construct in the new grid has been matched with the closest corresponding construct in the original grid and the degree of match has been graphed on the right. This enables the experts to see where they have corresponding constructs either providing different terms for the same distinction or indicating some underlying relation, for example a causal link. They can also see the constructs and elements where they do not have correspondence and about which they may have difficulty in communicating.
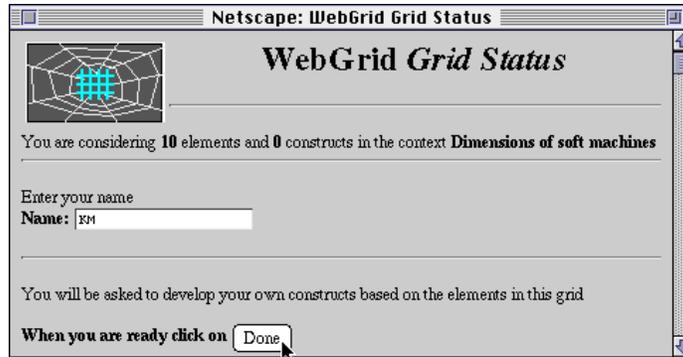
Figure 17 Second expert starting a grid with same elements as first one
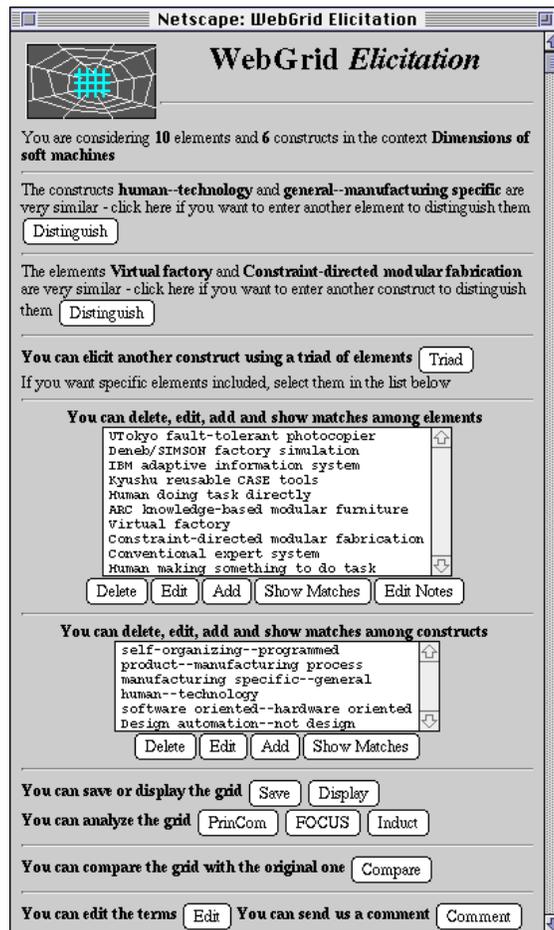
Figure 18 Status screen with compare option

For example, the construct "human—technology" is best matched in the original grid by the construct in the original grid "Conventional system—IMS research target" and the element "Deneb/SIMSON factory simulation" is one which the two experts will find it difficult to discuss because they have no corresponding constructs. Such results can be discussed through email, Internet audio and videophone, or through more conventional means. In some cases the discussion may lead to a consensus on terminology and knowledge structures, in other cases to systems combining different perspectives and in others to systems that offer advice based on different and potentially conflicting sources of expertise (Boose and Bradshaw, 1987).

19
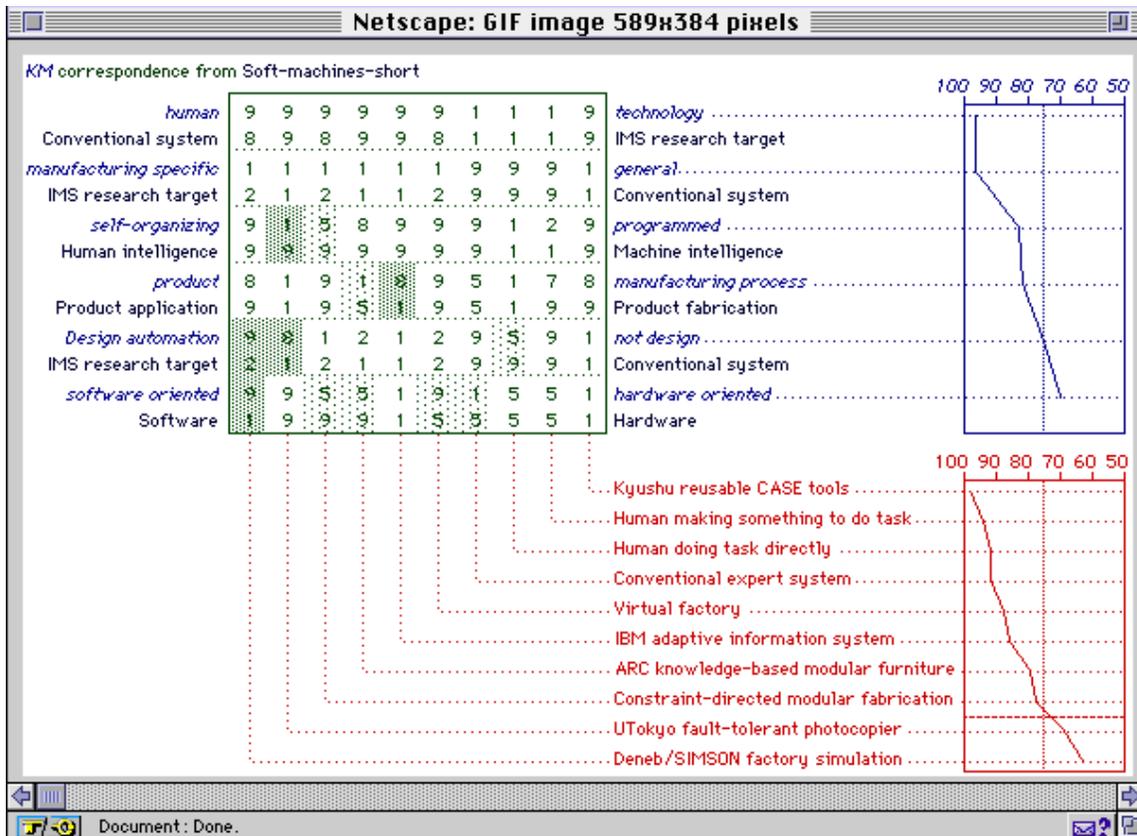
Figure 19 Comparison of grids from two experts

4.7 Integrating Inductive Modeling and Inference with Test Cases

One major objective of many of our previous studies with knowledge elicitation tools has been to integrate them with hypermedia and inference engines to support all phases of knowledge-based system development within a unified framework. In particular our repertory grid tools have been integrated with multimedia tools such as HyperCard and with expert system shells such as Nexpert (Gaines, Rappaport and Shaw, 1992) and Babylon (Gaines and Linster, 1990) in such a way that informal knowledge in hypermedia is made available as annotation to the knowledge acquisition and inference system, and cases tested in the inference system can be edited and posted back to the grid to revise the knowledge model.

The hypermedia annotation facilities of WebGrid have already been illustrated. It also supports inductive knowledge modeling and inference in a unified framework. Figure 20 shows the main screen of WebGrid replicating a repertory grid study of a standard contact lens dataset (Gaines and Shaw, 1993b). The elements consist of clients being prescribed contact lenses, the constructs are attributes relevant to the prescription, and the dataset and its proper inductive model were originally developed by Cendrowska (1987).

Figure 20 Status screen for contact lens data set

When the "Induct" button in Figure 20 is clicked the inference rules shown in Figure 21 are returned, in this example as a pair of EDAGs (Gaines, 1995), one each for the two conclusion constructs specified. The interpretation of the first EDAG is that by default one should prescribe "not soft" unless tear production is "normal" and the client is "not astigmatic" when one should prescribe "soft" unless the client is also "presbyopic" and "myope" when one should again prescribe "not soft." The interpretation of the second is that by default one should prescribe "not hard" unless tear production is "normal" and the client is "astigmatic" and either the client is myope or young when one should prescribe "hard." The controls at the bottom of Figure 21 allow the user to specify that the induction module in WebGrid produce simple production rules or ripple-down rules (Gaines and Compton, 1995), but usually the EDAGs are the most compact knowledge representation.
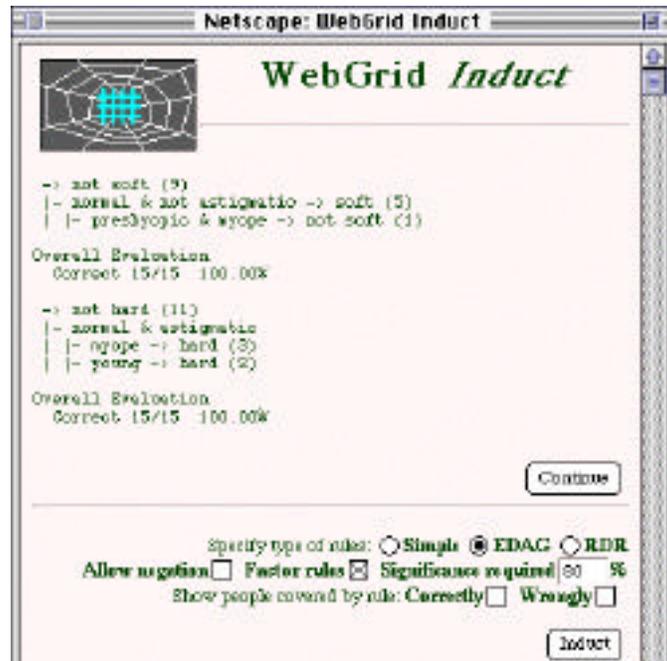
Figure 21 Inference rules returned by Induct

When the "Test" button in Figure 20 is clicked the test case screen shown in Figure 22 is returned. This is familiar to the user because it is very similar to the screen for entering a new element, except that the conclusion constructs have been separated from the data entry ones. Inference has already been done using the rules developed by Induct, and has inferred, as one might expect, that it is open whether "not soft", "soft", "not hard" or "hard" may be inferred.

Figure 23 shows the screen returned when the user enters that the person is "not astigmatic" and clicks on the "Infer" button. The inference system has inferred that "not hard" is true but that "not soft" and "soft" are still open. The system uses proper open-class inference and will, for example, infer "not soft" and "not hard" if "reduced" is entered for tear production.

If the user notes that the result of inference is incorrect, he or she may change the conclusions, enter more data and annotation and click on the "Add" button to add a new case to the grid. Running Induct or Test again generates a new knowledge model taking into account the additional case. Thus, inference with test cases is an integrated facility in knowledge elicitation.

In practice, the capability to iterate through a range of test cases using the screens shown in Figures 22 and 23 is generally highly instructive in understanding the knowledge modeling process. It also provides an "expert system" capability within WebGrid.

22

Figure 22 Test case entry

Figure 23 Inferencewith partial data entered for the test case

23

5 Other Use of the World Wide Web in Knowledge-Based System Research

The World Wide Web has been used in a number of knowledge-based system developments. Rice, Farquhar, Piernot and Gruber (1995, 1996) have developed a web interface to the Ontolingua ontology construction tool, and Farquhar, Fikes, Pratt and Rice (1995) have given examples of its application to collaborative ontology construction for information integration. Swartout, Patil, Knight and Russ (1996) have developed a web-based ontology editing tool Ontosaurus and used it to develop large-scale ontologies. Maurer (1997) has developed CoMo-Kit as a web-based interface to his knowledge modeling system and applied it to tracking dependencies in software engineering. Eriksson (1995) has made some of the PROTÉGÉ-II knowledge-acquisition tools accessible through the web. Skuce (1997) has developed a hybrid knowledge management system that integrates documents, knowledge bases, databases and the web.

The SHADE (SHAred Dependency Engineering) project (McGuire, Kuokka, Weber, Tenenbaum, Gruber and Olsen, 1993) supports reconfigurable exchange of engineering knowledge among computer-aided engineering systems by using agents providing services through the Internet. These agents are generally accessible through the KQML agent-communication language (Finin, Weber, Wiederhold, Genesereth, Fritzson, McKay, McGuire, Shapiro and Beck, 1992) and have been used to support systems operating through the web. For example, Gruber and Gauthier (1993) have used them in providing a Device Modeling Environment on the web, and have demonstrated its use in the simulation of a leak in the Space Shuttle's reaction control system using clickable maps as mimic diagrams.

6 Conclusions

The wide range of technologies developed in artificial intelligence research has the potential to support knowledge-based system development across many disciplines, but access to them is currently not widely available. The technologies, tools and protocols developed for the World Wide Web make it possible to provide knowledge acquisition, representation and inference systems internationally to anyone with access to the Internet. Such knowledge-based technologies integrate naturally with other applications and hypermedia systems to provide a distributed knowledge medium capable of supporting the knowledge processes of professional communities.

This article has illustrated what is possible through examples of the development of knowledge modeling tools operating through the web to provide knowledge acquisition, representation and inference through semantic networks and repertory grids. The web-based tools have been used extensively in undergraduate and graduate courses and have proved both robust in used and effective in enabling students to elicit and compare their conceptual models (Shaw and Gaines, 1995). The systems were developed primarily for locally managed projects, and their availability has only been advertised through specialist list servers for the knowledge acquisition and personal construct psychology communities. However, in the period July-December 1995 WebGrid was accessed from 674 different sites in 30 countries. The web is essentially an anonymous medium and we do not track the activities of outside users. However, occasionally users contact us to discuss their applications and, if the local network goes down, we receive mail from serious users reporting the problem and requesting notification of when the knowledge acquisition programs will be available again. An interesting example of serendipitous use was a graduate student in Holland studying operators' models of nuclear reactors who came across the grid elicitation system on the web in September 1995, used it with his experts to elicit their conceptual models, and included these in an additional chapter in his thesis for examination in October 1995—the web certainly accelerates research processes.

Web technology is improving rapidly, and many of the limitations of the current HTML graphic user interface are disappearing as customized downloadable widgets become available through Sun's Java and Microsoft's ActiveX technologies. The scope for developing new systems through the integration of knowledge-based and other technologies from many different research groups will greatly increase as the web becomes used increasingly to make those technologies widely accessible. The social nature of knowledge processes will also become a significant factor in system development when it is possible to make those systems an integral part of the operation of major professional communities.

In conclusion, we note that the results presented in this article demonstrate that the World Wide Web client server model operating through the Internet can effectively support the porting of existing knowledge acquisition, modeling and representation tools to operate in a distributed, open architecture environment. This opens up new possibilities for collaborative research in which the methodologies and tools developed by one group are made available on the network for use by other research groups world-wide. It enables past studies of the integration of hypermedia, knowledge acquisition and inference tools within a local environment to be generalized to integration on a distributed basis across wide area networks. A major advantage of this approach is that tools can be operated at the site of the originators responsible for them, and their maintenance, upgrading and monitoring can be done very much more effectively than when the software is distributed to other sites.

We envision a collaborative research program internationally in which the first objective is to make tools available across the Internet in interoperable forms, and the second objective is to develop some integrated knowledge management systems that use these tools in a variety of configurations to provide a new generation of knowledge-based systems.

Acknowledgments

URLs

WebGrid can be accessed at **http://tiger.cpsc.ucalgary.ca/WebGrid/**

Concept mapping software can be accessed through
http://ksi.cpsc.ucalgary.ca/local/software/index.html

Related articles on the systems described can be accessed through
http://ksi.cpsc.ucalgary.ca/articles/

Articles by other researchers on web-based knowledge modeling tools may be found in
workshop proceedings accessible through **http://ksi.cpsc.ucalgary.ca/KAW/**

References

Andreessen, M. (1993). NCSA Mosaic Technical Summary. NCSA, University of Illinois. ftp://ncsa.uiuc.edu/Web/Mosaic/Papers/mosaic.ps.Z.

Arnold, K. and Gosling, J. (1996). **The Java Programming Language**. Reading, Massachusetts, Addison-Wesley.

Berners-Lee, T. (1993). World-Wide Web Talk at Online Publishing 1993. CERN, Geneva. http://info.cern.ch/hypertext/WWW/Talks/OnlinePublishing93/Overview.html.

Berners-Lee, T. and Cailliau, R. (1990). WorldWideWeb: Proposal for a Hypertext Project. CERN, Geneva. http://info.cern.ch/hypertext/WWW/Proposal.html.

Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H.F. and Secret, A. (1994a). The World-Wide Web. **Communications ACM 37**(8) 76-82.

Berners-Lee, T., Connolly, D. and Muldrow, K. (1994b). Hypertext Markup Language (HTML) Version 2.0. CERN, Geneva. http://www.hal.com/products/software/olias/Build-html/html-spec.ps.

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. and Bradshaw, J.M. (1987). Expertise transfer and complex problems: using AQUINAS as a knowledge acquisition workbench for knowledge-based systems. **International Journal of Man-Machine Studies 26** 3-28.

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.

Borenstein, N. and Freed, N. (1993). MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for specifying and describing the format of internet message bodies. Internet. RFC 1521.

Cendrowska, J. (1987). An algorithm for inducing modular rules. **International Journal of Man-Machine Studies 27**(4) 349-370.

CPCS (1993). **RepGrid 2 Manual**. Centre for Person Computer Studies, 3019 Underhill Drive NW, Calgary, AB, Canada T2N 4E4.

Eriksson, H. (1995). PROTÉGÉ-II Knowledge-Acquisition Tools. http://camis.stanford.edu/projects/protege/ka-tools.html.

Farquhar, A., Fikes, R., Pratt, W. and Rice, J. (1995). Collaborative ontology construction for information integration. Knowledge Systems Laboratory, Stanford University. ftp://www-ksl-svc.stanford.edu:5915/.

Finin, T., Weber, J., Wiederhold, G., Genesereth, M., Fritzson, R., McKay, D., McGuire, J., Shapiro, S. and Beck, C. (1992). Specification of the KQML Agent-Communication Language. The DARPA Knowledge Sharing Initiative External Interfaces Working Group.

Flores-Mendez, R.A. (1997). Java concept maps for the learning web. **Proceedings of ED-MEDIA'97 : World Conference on Educational Multimedia and Hypermedia**. Charlottesville, VA, Association for the Advancement of Computing in Education.

Ford, K.M., Bradshaw, J.M., Adams-Webber, J.R. and Agnew, N.M. (1993). Knowledge acquisition as a constructive modeling activity. **International Journal of Intelligent Systems 8**(1) 9-32.

Gaines, B.R. (1994). A situated classification solution of a resource allocation task represented in a visual language. **International Journal Human-Computer Studies 40**(2) 243-271.

Gaines, B.R. (1995). 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 Linster, M. (1990). Integrating a knowledge acquisition tool, an expert system shell and a hypermedia system. **International Journal of Expert Systems Research and Applications 3**(2) 105-129.

Gaines, B.R., Rappaport, A. and Shaw, M.L.G. (1992). Combining paradigms in knowledge engineering. **Data and Knowledge Engineering 9** 1-18.

Gaines, B.R. and Shaw, M.L.G. (1980). New directions in the analysis and interactive elicitation of personal construct systems. **International Journal Man-Machine Studies 13** 81-116.

Gaines, B.R. and Shaw, M.L.G. (1989). Comparing the conceptual systems of experts. **Proceedings of the Eleventh International Joint Conference on Artificial Intelligence**. pp.633-638. San Mateo, California, Morgan Kaufmann.

Gaines, B.R. and Shaw, M.L.G. (1992). Documents as expert systems. Bramer, M.A. and Milne, R.W., Ed. **Research and Development in Expert Systems IX. Proceedings of British Computer Society Expert Systems Conference**. pp.331-349. Cambridge, UK, Cambridge University Press.

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

Gaines, B.R. and Shaw, M.L.G. (1993b). Eliciting knowledge and transferring it effectively to a knowledge-based systems. **IEEE Transactions on Knowledge and Data Engineering 5**(1) 4-14.

Gaines, B.R. and Shaw, M.L.G. (1993c). Open architecture multimedia documents. **Proceedings of ACM Multimedia 93**. pp.137-146.

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. (1995a). Concept maps as hypermedia components. **International Journal Human-Computer Studies 43**(3) 323-361.

Gaines, B.R. and Shaw, M.L.G. (1995b). WebMap: concept mapping on the web. **World Wide Web Journal 1**(1) 171-183.

Goldfarb, C.F. (1990). **The SGML Handbook**. Oxford, Clarendon Press.

Gruber, T. and Gautier, P. (1993). Machine-generated explanations of engineering models: a compositional modeling approach. **IJCAI'93: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence**. pp.1502-1508 (http://www-dme.stanford.edu:5051/DME-server/DME-server/Computed/TOP-LEVEL). San Mateo, California, Morgan Kaufmann.

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

Kremer, R.A. (1996). Embedded interactive concept maps in web documents. Maurer, H., Ed. **Proceedings of WebNet96**. pp.273-280. Charlottesville, VA, Association for the Advancement of Computing in Education.

Linster, M. (1994). Special Issue on Sisyphus Problem Solving. **International Journal of Human-Computer Studies 40**(2)

Maurer, F. (1997). CoMo-Kit: Knowledge Based Workflow Management. **Artificial Intelligence in Knowledge Management**. pp.106-109. Menlo Park, AAAI.

McGuire, J.G., Kuokka, D.R., Weber, J.C., Tenenbaum, J.M., Gruber, T.R. and Olsen, G.R. (1993). SHADE: Technology for knowledge-based collaborative engineering. **Journal of Concurrent Engineering: Applications and Research 1**(2)

Rice, J., Farquhar, A., Piernot, P. and Gruber, T. (1995). Stanford KSL Network Services. Knowledge Systems Laboratory, Stanford University. http://www-ksl-svc.stanford.edu:5915/.

Rice, J., Farquhar, A., Piernot, P. and Gruber, T. (1996). Using the web instead of a window system. **Proceedings of CHI'96**. pp.103-117. New York, ACM.

Shadbolt, N.R. (1996). Sisyphus III Proposal (Rocky III). University of Nottingham. http://www.psyc.nott.ac.uk/aigr/research/ka/SisIII/home-page.html.

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. 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. (1989). A methodology for recognizing conflict, correspondence, consensus and contrast in a knowledge acquisition system. **Knowledge Acquisition 1**(4) 341-363.

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.

Shneiderman, B. (1983). Direct manipulation: a step beyond programming languages. **Computer 16**(8) 57-69.

Skuce, D. (1997). Hybrid KM: Integrating documents, knowledge bases, databases and the web. **Artificial Intelligence in Knowledge Management**. pp.149-155. Menlo Park, AAAI.

Slater, P., Ed. (1976). **Dimensions of Intrapersonal Space: Volume 1**. London, John Wiley.

Swartout, B., Patil, R., Knight, K. and Russ, T. (1996). Towards distributed use of large-scale ontologies. Gaines, B.R. and Musen, M.A., Ed. **Proceedings of Tenth Knowledge Acquisition Workshop**. pp.32-1-32-19.