# Contents

1 Introduction .................................................................................................................. 1-1

2 RepServe Operation .................................................................................................... 2-1
   2.1 The RepServe pane in the Rep 5 Manager window ............................................... 2-1
   2.2 RepServe scripting ................................................................................................. 2-3
   2.3 RepServe scripts ..................................................................................................... 2-4
   2.4 WebGrid elicitation and editing scripts ................................................................ 2-4
   2.5 WebGrid analysis scripts ....................................................................................... 2-5
   2.6 WebGrid scripts for saving and accessing saved grids ......................................... 2-6
   2.7 Miscellaneous scripts ............................................................................................ 2-6
   2.8 Alternative scripts ................................................................................................. 2-7
   2.9 Command sequences .............................................................................................. 2-7
   2.10 Contextual help system ......................................................................................... 2-8
   2.11 WebGrid CSS ........................................................................................................ 2-8
   2.10 WebGrid Javascript library .................................................................................... 2-9

3 Using WebGrid .............................................................................................................. 3-1
   3.1 Options page ........................................................................................................... 3-1
   3.2 Main page ................................................................................................................ 3-5
   3.3 Element entry and edit page .................................................................................... 3-7
   3.4 Construct entry and edit pages ................................................................................. 3-10
   3.5 Element and construct delete pages ....................................................................... 3-19
   3.6 Display page ............................................................................................................ 3-20
   3.7 Focus cluster page .................................................................................................. 3-23
   3.8 PrinGrid page .......................................................................................................... 3-25
   3.9 Crossplot page ......................................................................................................... 3-26
   3.10 Compare page ........................................................................................................ 3-28
   3.11 Match page ............................................................................................................ 3-29
   3.12 Save grid page ........................................................................................................ 3-30
   3.13 Cached grids page .................................................................................................. 3-31
   3.14 Register cache page ............................................................................................... 3-32
   3.15 Comment page ....................................................................................................... 3-34
   3.16 Finish page ............................................................................................................ 3-34
   3.17 Contextual help system ......................................................................................... 3-35

4 WebGrid tutorial: “Carol’s life crisis” .................................................................... 4-1
   4.1 The problem situation ............................................................................................. 4-1
   4.2 Developing a grid to help resolve the problem ...................................................... 4-1
   4.3 Possible courses of action as grid elements .......................................................... 4-3
   4.4 Possible consequences of action as grid constructs .............................................. 4-5
   4.5 Rating the possibilities in terms of the consequences ............................................ 4-7
   4.6 Elicitation of consequences by considering similarities and differences in triads of possibilities ...................................................................................... 4-9
   4.7 WebGrid’s main page and analysis of Carol’s conceptual representation .......... 4-13
   4.8 Continuing the elicitation under Carol’s control based on WebGrid’s suggestions .. 4-15
   4.9 Interacting with the clustered grid to assess Carol’s possible courses of action .... 4-16
   4.10 Saving Carol’s grid for later use ............................................................................ 4-19
   4.11 Making a decision directly from the clustered grid .............................................. 4-20
   4.12 Making a decision directly from the mapped grid .............................................. 4-20
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.13 Making the decision from the crossplot of important consequences</td>
<td>4-21</td>
</tr>
<tr>
<td>4.14 Making a decision based on an “ideal possibility”</td>
<td>4-23</td>
</tr>
<tr>
<td>4.15 Summary</td>
<td>4-26</td>
</tr>
<tr>
<td>5 References</td>
<td>5-1</td>
</tr>
</tbody>
</table>
1 Introduction

*RepServe* is a scriptable web server shell embedded as an integral component of *Rep 5*. The shell supports the HTTP protocol for World Wide Web communication with multiple simultaneous client web browsers. It decodes the incoming encoded HTML requests and passes any variables and values posted from forms to a set of scripts written in *RepScript*. The scripts decode the request semantically and generate a response in HTML, passing this back to *RepServe* to send to the requesting client.

*RepScript* is a general-purpose programming language so that the web services provided through *RepServe* can be of any nature. Since *RepScript* has access to the library of *RepGrid* and *RepNet* functionality, it is particularly well-suited to services relating to the use of grids and nets. This combination of a robust HTTP server, an object-oriented, incrementally compiled programming language, and a powerful libraries of conceptual representation tools makes *RepServe* a powerful tool for knowledge management and market research applications.

*RepServe* was developed initially to make *WebGrid* functionality available as part of *Rep 5*, and the sample scripts provided emulate an enhanced version of our original *WebGrid* service. *WebGrid* was first developed in 1994 as a port of *RepGrid* to operate over the World Wide Web (Gaines, 1995; Gaines and Shaw, 1996; Gaines and Shaw, 1997; Gaines and Shaw, 1998; Shaw and Gaines, 1998; Tennison, O’Hara and Shadbolt, 2002). The *WebGrid* server at the University of Calgary (http://tiger.cpsc.ucalgary.ca) has operated on a 24/7 basis since early 1995 making it one of the earliest continuous services on the World Wide Web (Gaines and Shaw, 2007). It has been used for repertory grid elicitation and analysis from over 100,000 sites worldwide and been the basis for experimental data collection for a wide range of studies.

There are major advantages of offering web-based conceptual representation services:

- first, modern web browsers provide well-designed and robust general-purpose user interfaces, readily programmable through HTML which enables the user interface to be easily customized;
- second, they provide excellent typographic and page layout capabilities enabling dynamically generated pages with variable content to be presented attractively;
- third, they provide strong multimedia support enabling sounds, image and videos to be embedded and used, for example, to illustrate elements in a domain in grid elicitation;
- fourth, access to conceptual representation services over the Internet with no requirement for special-purpose software on the client machine facilitates distributed studies based on remote data collection.

The scripts that provide the *WebGrid* functionality are text files that may be edited by *Rep 5* users to extend and customize *WebGrid*, or use as examples on which to base other forms of interaction. The text in the scripts may be translated into other languages to support versions of *WebGrid* in French, German, Dutch, Swedish, and so on. Since the text is in Unicode (UTF-8) it is also possible to support non-Roman languages such as Chinese, Japanese, Arabic, Devanagari, and so on. Provision is made for the support of multiple simultaneous versions of *WebGrid* so that different customizations and translations can all be made accessible on one *RepServe* machine.
RepServe queues requests arriving while the current request is being processed and hence can handle multiple simultaneous users. The number of users that can be served at the same time depends on the acceptable average processing time and the actual processing time determined by the speed of the machine. RepServe is not intended to be a heavy-duty document server such as Apache, but rather a specialist conceptual representation tool that incidentally serves documents as part of providing specialist capabilities requiring significant computation.
2 RepServe Operation

This section describes some of the technical aspects of RepServe operation.

2.1 The RepServe pane in the Rep 5 Manager window

RepServe is managed from the “RepServe” pane in the “Rep 5 Manager” window as shown below.

![Rep 5 Manager window — RepServe pane](image)

The “Port” field specifies the port on which RepServe should listen for HTTP communications. It is set by default under Microsoft Windows to port 80, the standard default port for the World Wide Web. It is set by default under Apple OS X to port 2000 since the apple unix operating system does not allow the port to be set below port 1024 unless the application is running as a root application. Under Windows the port can be changed to any number but should not be below 1024 unless it is set to 80. Under OS X it may be set to any port not below 1024.

The Internet address of the machine is shown in order to monitor the address at which RepServe is listening. The “Active “ check box makes RepServe active when checked, that is listening at the Internet address shown on the specified port. It can be unchecked to stop RepServe listening. The “Remote” check box is shown only for the Enterprise Version and specifies whether requests will be accepted across the Internet. It may be unchecked to restrict requests to come from a client on the same machine as RepServe. The “Browser” button starts the default browser on the local machine and requests the URL on which RepServe is listening, in this case “http://192.168.1.20:2000,” thus fetching the default page “/ServerDocs/index.html.”

The text window at the bottom of the RepServe pane logs the server activity in the web “Common Log Format” as shown in the example below.

![RepServe server activity log in the common log format](image)
The black bar at the top of the vertical scroll bar on the right may be dragged up to hit the top line of RepServe controls and show only the log. It can also be dragged down to show one or two additional lines of controls as shown below.

RepServe script debugging capabilities

The “Window” check box when checked specifies that the activity should be logged in the window. The “File” check box when checked indicates that the activity log should also be appended to the file named “/ServerLogs/Log”.

The “Debug” check box when checked specifies that the common log format should be extended with additional information that may be useful in monitoring the performance of server scripts. This information is specified by a server script (as is the common log format data) and hence may vary. For the supplied WebGrid scripts it comprises the transaction time, the number of active objects, the memory in use, and the UID of the grid being processed.

The “Client” check box when checked specifies that the common log format should be extended with the client identification supplied by the remote web browser. This information may be useful if there are problems with supporting HTML on particular types of browser.

The “Capture” check box when checked specifies that new grids (as contrasted with demonstration grids) should be filed in the directory “/ServerLogs/Capture/” using their UIDs as their file names. This is useful in monitoring remote studies and in supporting users who forget to save their grids locally. It may raise privacy issue in some situations.

The “Clear” button clears the log window and, more importantly, also clears the cache of the scripts that have been dynamically compiled and cached in the operation of the server. This is useful when scripts are being changed and one wishes the changed script to be used rather than that previously cached.

On the bottom row are further options to aid script debugging. The “Sockets” checkbox when checked causes details of each socket process to be written in the log window. The “Request” checkbox when checked causes the raw request from the client to be written in the log window. The “Data” checkbox when checked causes the processed request as pairs of names and values to be written in the log window. The “Reply” checkbox when checked causes the HTML reply to the client to be written in the log window.

RepServe is designed to run unattended. It keeps a record of the state of the controls in its window and sets them up automatically when Rep 5 is started. Hence, most users do not need to access the RepServe pane. Even those developing scripts can usually check their correct
operation by testing them from a web client. However, sometimes it may be useful to monitor the server operation at a more detailed level and, if so, the capabilities are available.

### 2.2 RepServe scripting

When *RepServe* receives a request it calls the script “/ServerScripts/Server.txt” passing it the URL (as the first item of a vector store with an empty name that can be accessed as vGet(0). If it is a POST request from a form it also passes the name/value pairs from the form (in a hash store that can be accessed by a script through the GetPost and CheckPost functions).

The script Server.txt uses the URL passed to determine what scripts should handle the request. It provides the *RepDoc* functionality itself, returning the documents requested by the client, generally stored in “/ServerDocs”. It passes URL’s commencing with “WebGrid” to the script “/ServerScripts/WebGrid/WebGrid.txt” and that script, together with a family of associated scripts support the *WebGrid* functionality.

Note that Server.txt can be changed to provide additional or completely different functionality. *RepServe* is a general-purpose, script-driven web server shell. In practice it is expected that most users of Rep 5 will leave the provided functionality in place, enhance it with other functionality, and customize the provided *WebGrid* functionality. A typical enhancement might be a general questionnaire data collection facility or a link to a description logic inference engine.

The RepScript Manual describes the general features of the *RepScript* programming language together with the additional features available when scripting *RepServe*.

### 2.3 WebDoc, WebNet and WebGrid

Example scripts are provided for three services.

*WebDoc* serves documents having URL’s of the form “http://documentpath” from the directory “/WebDocs/” looking first in the “/User/Documents/Rep 5” or “/User/My Documents/Rep 5” directories and, if the document is not found in the directory where the Rep 5 application resides. The first field in document path must not be “WebGrid” or “WebNet” as these are used as keywords to indicate the other services.

*WebDoc* has a macro substitution capability in that fields at the end of the URL separated by “?” are substituted for tags within the document having the form “<Mn>” where n is a number indexing the string of ?-separated fields, with that after the first “?” being number 1. This capability is used in the *WebGrid* contextual help facilities to pass user-defined terminology to the help documents.

*WebNet* serves nets having the URL’s of the form “http://WebNet/netpath” from “WebDocs” as above, but expects the document to be a net. If the document name has the form “netname.rnet.png” then *WebNet* returns a PNG image of the net which may be embedded in a user-defined document. If the document name has the form “netname.rnet.click” then *WebNet* returns a a document having a link to the net embedded in it (for test purposes and to show the form of an embedded link). When a node in a net embedded in a document is clicked *WebNet* takes the note attached to a node to be a URL and redirects to it.

*WebGrid* is the most complex of the three applications and exposes much of the functionality of *RepGrid* to web clients.

The *WebDoc* and *WebNet* functionality is included in the *Server* script. The *WebGrid* functionality is scripted in a set of modular scripts described in the next section.
2.3 The RepServe scripts

The diagram below shows the flow structure for the RepServe scripts that are normally issued as examples supporting RepDoc, WebNet and WebGrid. The arrows indicate the main flow of calls between scripts.

**RepServe script structure**

The Library script comprises a set of utility routines that can be called from any script and support commonly required computations across all scripts, such as generating HTML headers and encoding the grid data in hidden fields.

The Server script is that called by RepServe when a request is received. It services some requests itself, notably those for RepDoc and RepNet functionality, and passes others to script families providing more complex services, such as those of WebGrid.

2.4 WebGrid elicitation and editing scripts

For ease of development and maintenance the WebGrid service implementation is split across some twenty modular scripts, each providing specific functionality.
When a WebGrid request is received by the Server script it passes it to the WebGrid script which analyzes it and passes it on to an appropriate script. The structure of most of these scripts is based on two routines, one of which writes out the HTML for its associated web page and the other of which analyses the response from that page and calls other scripts to perform the requested service.

The modularity and structure is not necessary. WebGrid could be written as one large script. However, its partition into relatively small modules, each providing routines to generate a specific type of page and to respond to requests from that page, makes the scripts easier to develop, understand, enhance, translate and maintain.

The grid data is stored within hidden fields stored in every WebGrid document. This avoids the need to manage its storage at the server, enables the web clients capability to save web pages to be used to save the grid locally, and makes the client’s “Back” button an effective “Undo” feature since it is always possible to revert to a previous version of the grid.

Control in WebGrid generally passes back to the main page generated by the Main script which lists recommended actions based on analysis of the grid, list the elements and constructs, provides access to the graphic analysis programs and so on. The Main script can transfer control to nearly all the other WebGrid scripts (except Register, WebGridCache,igrd and Cache). It does so either in response to the users clicking on a button in the main page or through a command string that can be entered to control the initial sequence of elicitation, for example triadic elicitation of constructs.

The major options associated with setting up grid elicitation are managed through an options page generated by the Options script. This also supports commencing a new grid and either going through an interactive guided elicitation process or an expedited data entry process.

The Elicit script provides interactive elicitation of elements and significant constructs as an alternative to bulk entry of the elements through the Options script. When the interaction is completed it transfers control to the Main script. The script provided is generic and is readily customized for specific studies.

The Delete script prompts the user to delete either selected elements or selected constructs, and transfers control back to the Main script.

The CAdd script prompts the user to enter a new construct and then transfers control to the ERating script to edit it.

The ERating script allows the user to enter or edit the ratings of all the elements on one or more constructs and then transfers control to the Main script.

The CRating script allows the user to enter or edit the ratings of one or more elements on all the constructs, including entering a new element, and then transfers control to the Main script.

2.5 WebGrid analysis scripts

The WebGrid analysis scripts are entered from the Main script and generally support some form of grid analysis and the graphic and/or textual presentation of the results. The graphic output is a clickable map and the user can select whether the result of clicking on an element or construct is to transfer to the appropriate editor, or to remove that item from the analysis by deselecting it. Each script supports the selection of the parameters controlling the analysis and the color scheme.
for presenting the analysis. The analysis scripts transfer control to the Main, ERating or CRating scripts.

The Display script displays the grid.

The Focus script provides a Focus analysis of the grid.

The PrinGrid script provides a principal components analysis of the grid in two or three dimensions.

The Crossplot script provides a crossplot of the elements on two or three specified constructs.

The Compares script provides a comparison of two grids having the same elements or constructs, one of which must be cached on the server.

The Match script lists the matches among elements and/or those among constructs.

The graphic output from the analysis scripts in the form of PNG file, a link to which is embedded in the document returned. The PNG file is cached in memory rather than written to disk and RepDoc recognizes the requests for such a PNG file, sends it from the cache and then deletes it from the cache, relying on the web client’s caching to keep a copy locally. It also checks the cache for PNG files that have not been requested and removes them after a time period.

2.6 WebGrid scripts for saving and accessing saved grids

WebGrid provides a number of options for saving a grid locally, caching it temporarily at the server for purposes of comparison, or caching it longer term at the server in a password-protected cache directory registered to, and managed by, a user.

When the “Save Grid” button in clicked in the main page the Main script transfers control to the WebGrid.rgrid script which provides various options for saving, caching or downloading the grid. The page generated also informs users how they may save the grid on their local machine and/or download it as a RepGrid file. This script is named “WebGrid.rgrid” because that becomes the value of the form action attribute which is used by the client as the name of the file it downloads.

If the chooses to cache the grid in the general cache directory control is transferred to the script Cache which list the URL’s through which the cached grid may be accessed.

If the user chooses to cache the grid in a registered cache directory control is transferred to the script WebGridCache.rgrid which, if the user’s password is valid, caches the grid in the user’s registered cache directory and displays the contents of that directory together with the means to control it.

The Register script stands alone and is called directly to enable a user to register a cache directory, providing their identification, intended use, and their password to access the cache.

2.7 Miscellaneous scripts

The Comment script provides the opportunity to send comments to the specified email address.

The Finish script provides the opportunity to thank the user for their participation, make sure they know how to analyze and save their data and send comments to the specified email address. This provision is not logically necessary since interaction can be stopped at any time. However, it has proved to useful to users to support them when they feel they have finished their interaction with WebGrid. This script could be modified for remote data collection studies to ask
users for further information or to check that they have rated all the elements on all the constructs.

2.8 Alternative scripts

When interaction with WebGrid commences an alternative script directory may be specified as the location of the scripts to be used by putting its name after a “?” at the end of the URL. For example, the URL “http://myserver/WebGrid/Open/cars/hatchback.rgrid?Spanish” would access the script “/ServerScripts/WebGrid.txt” with the command to open the grid “ServerData/cars/hatchback.rgrid” and transfer to the script “/ServerScripts/Spanish/Main.txt” which might be the Main script translated into Spanish.

All subsequent scripts would be called from the “/ServerScripts/Spanish” directory. However, if a particular script did not exist in that directory then an attempt would be made to call it from the default “/ServerScripts” directory. This enables some scripts to be customized without requiring this to be done for all of them.

2.9 Command sequences

A command sequence may be added to the WebGrid URL after a second “?” and is decoded by the Main script to execute the commands prior to any other action. The commands are encoded as a string of command fields separated by semicolons. The command fields have the format:

command arguments

where the command is currently represented by an alpha character and any arguments are separated by commas.

The command for triadic elicitation is:-

T en1, en2, en3

which elicits a construct from a triad of the three elements whose numbers are specified in en1, en2 and en3. There may be from zero to three elements specified and any that are missing or out of range are replaced by randomly selected elements. Thus the command string:

T 1, 2, 3; T 4, 5, 6; T 1, 3, 5; T 2, 4, 6

can be used to start elicitation with four specified triads. The command strings:

T; T; T and T 1; T 2; T 3

start elicitation with three randomly chosen triads, or three partly specified and partly random, respectively. The command can be preceded with a number specifying it be repeated that number of times, e.g. “3T” is equivalent to “T; T; T”.

The command for construct rating, generally used for given constructs, is:

C n

which shows the construct rating page for construct number n. If n is missing it is taken as “1”. The command can be preceded with a plus sign to generate successive construct rating pages starting with that specified, e.g. “+C” will generate rating pages for all given constructs.

The commands, “D”, “F”, “P” and “X” generate Display, Focus, PrinGrid and Compare analyses, respectively. The command “O origin” sets up the origin of the grid to be as specified to enable grid comparison. These commands are used in the WebGridCache.rgrid script to support the analysis of grids in a cache.
2.10 Contextual help system

One of the facilities provided by the Library script is a contextual help system that allows a clickable “?” icon to be inserted in a WebGrid page, and a specified help document to be opened if the icon is clicked. The help documents are held in the directory “/ServerDocs/WG/Help/” and are always opened in the same window that can be positioned by the user and kept open.

The user terms for “element,” “elements,” “construct” and “constructs,” and the user-defined purpose are passed as parameters in the URL generated for a help document, and the macro-substitution facilities in WebDoc are used to embed them appropriately in the help text. This enables the help messages to be customized to be more readily understood by the user.

2.11 WebGrid CSS

WebGrid pages are linked to the cascading style sheet, “/ServerDocs/WG/WebGrid.css”, which defines a set of standard styles for WebGrid pages. Users may override these by incorporating style definitions in their grid options data. The predefined styles are:

* {font-family: Arial, Verdana, Helvetica, sans-serif; font-size: 12px; line-height: 1.4; color: #008888;}
body {margin: 0 auto; padding: 2px; width: 710px; background: #EEEECC;}
a {color: #006666; background: transparent; text-decoration: none; font-weight: bold;}
a: hover {background: #880000;}
div.a {border: 4px solid #DDDDDD; margin-top: 0px; padding-top: 2px; background: #FFFFEE; text-align: center;}
div.a + div.a {border-top: 0px;}
div.data {background: #FFFFEE;}
table {width: 100%; border-collapse: collapse;}
table.ele, table.con {margin: 0 auto; width: auto;}
table.center, table.items {margin: 0 auto; width: auto; text-align: left;}
table.items {border-width: 2px; border-spacing: 0px; border-style: solid; border-color: #008888;}
table.items th, table.items td {border-width: 1px; border-style: solid;}
table.cmat, table.emat {margin: 4px auto; width: auto; border-width: 2px; border-spacing: 0px; border-style: solid; border-color: #008888;}
table.cmat *, table.emat * {border-width: 1px 0 0 0; padding: 1px 6px; border-style: solid; text-align: center; font-size: 12px;}
table.cmat th, table.emat th {border-width: 2px 0 1px 0; font-size: 13px;}
table.cache, table.cachecon {margin: 0 auto; width: 99%; border-width: 2px; border-style: solid; border-color: #008888;}
table.cache td, table.cache th {border-width: 1px; padding: 2px 3px; border-style: solid; border-color: #BBBBBB;}
table.cache td, table.cache a {font-size: 11px;}
table.cache td {text-align: right;}
table.cache td.a {text-align: left;}
table.cache tr.a0 {background: #FFFFFF;}
table.cache tr.a1 {background: #FFEEEE;}
table.cache tr.a2 {background: #C6EFF7;}
table.cache tr.a3 {background: #C6EFF7;}

2.12 WebGrid Javascript library

WebGrid pages are linked to the Javascript library, “/ServerDocs/WG/WebGrid.js” which defines some common functions used in several WebGrid pages. These are:

```javascript
function bitSet(num, bit) { return num | bit; }
function bitClear(num, bit) { return num & ~bit; }
function addHidden(item, value) {
    var el = document.createElement('input');
    el.type = 'hidden';
    el.value = value;
    item.appendChild(el);
}
```


el.name=item;
el.value=value;
document.grid.appendChild(el); // append to form after type set to hidden
return el;
}

function setGridValue(item,value) {
    var els=document.getElementsByName(item);
    if (els.length==0) {
        addHidden(item,value);
    } else {
        els[0].value=value;
    }
}

function setswatch(el,id,bord) {
    if (!el) return;
    var s=el.value;
    for (var i=s.length; i<6; i++) {s="0"+s;}
    el.value=s;
    s='#'+s;
    var els=document.getElementById(id+'x');
    els.style.backgroundColor=s;
    if (bord) els.style.borderColor=s;
}

function initswatches() {
    var s;
    for (var i=0; i<initswatches.arguments.length; i++) {
        s=arguments[i];
        setswatch(document.getElementById(arguments[i]),s,1);
    }
}

function help(screen,link) {
    w=window.open('/WG/Help'+screen+'.html#'+link,'wghelp','resizable=yes,width=500,height=400');
    w.focus();
}

function showImage(imurl,title,ww,hh) {
    ww=ww+20;
    hh=hh+20;
    w=window.open('','_blank',toolbar=0,location=0,directories=0,status=0,menubar=0,resizable=1,width=ww+20,height=hh);
    w.document.write('<html><head><title>'+title+'</title></head><body><img src='+imurl+'></body></html>');
    w.document.close();
    w.focus();
}

function changeVisible(id,vis) {
    document.getElementById(id).style.display=(vis)?'block':'none';
}

function switchVisible(id,vis,bit){
    changeVisible(id,vis);
    changeVisible(id+'x',!vis);
    var el=document.grid._Control;
    var n=(el)?el.value:0;
n=(vis)?bitClear(n,bit):bitSet(n,bit);
setGridValue('_Control',n);
}

function selInit(selid) {
    var sel=document.getElementById(selid);
    if (!sel) return;
    var opt=sel.options;
    var len=opt.length;
    var n=0;
    for (var i=0; i<len; i++) if (opt[i].selected) n++;
    var dis=n==0;
    for (var i=1; i<arguments.length; i++) {
        var el=document.getElementById(arguments[i]);
        if (el) el.disabled=dis;
    }
}

function selAll(selid,b) {
    var sel=document.getElementById(selid);
    if (!sel) return;
    var opt=sel.options;
    var len=opt.length;
    for (var i=0; i<len; i++) opt[i].selected=b;
}

function doBtn(name) {
    var el=addHidden(name,"");
    var elh=addHidden("*","*");
    document.grid.submit();
    el.parentNode.removeChild(el);
    elh.parentNode.removeChild(elh);
}
3 Using WebGrid

This section illustrates and discusses each of the web pages generated by *WebGrid*. The same examples will be used as those in the *RepGrid* manual to facilitate comparison between the stand-alone and web versions of the grid tools.

3.1 Options page

The options page shown when entering or eliciting a new grid is shown below.

![WebGrid options page]

*WebGrid options page*

The top section provides text boxes for the entry of: the name of the person generating the grid; a note which will be placed in parentheses after the name in the analysis heading in order to identify the grid; a purpose that will be used to remind the user of her or his intentions as the grid is being elicited and will be shown in the analysis heading; annotation which will not be shown but can be used for additional notes; and a list of initial elements. The check box below when checked indicates that an existing grid is to be entered and leads to a rapid data entry sequence rather than an elicitation. The “Update” button refreshes the page taking into account any changes in items entered or customization. The “Done” button transfers to the main page.
The three sections below each have outliner arrows that are initially closed. Clicking on the “Grid Parameters” arrow opens the section allowing various parameters to be entered or edited as shown below.

**WebGrid options page Grid Parameters section**

The text boxes on the first line allow the singular and plural forms of the terms to be used for elements to be defined, for example, person/people, car/cars. The text boxes on the second line allow the singular and plural forms of the terms to be used for constructs to be defined, for example, quality/qualities, property/properties. The text boxes on the third line set the range of the default rating scale. The check boxes on the fourth line specify what meta values will be allowed in rating scales. The check boxes on the fifth line specify what types of rating scales will be allowed in construct definitions.

Clicking on the “Grid Items” arrow opens the section showing any grid items that have been defined and providing a blank one for the entry of an additional item.

**WebGrid options page Grid Items section**

If multiple items are to be entered the “Update” button should be clicked between item entries. Clicking on the “WebGrid Customization” arrow opens the section allowing the web pages to be customized.
The checkbox at the top if checked specifies that the HTML note entered for an element will be inserted in the page when the element is being used in elicititation. This enables text, images and audio clips to be inserted to remind users of the features of an element they are considering.

The first text box allows a heading to be entered which will replace the standard WebGrid heading at the top of each page. This enables the WebGrid pages to be customized with an application-dependent heading, typically a logo and title.

The second text box allows text to be inserted at the end of the <head> section of the document, typically a style sheet and/or javascript. This enables the styling of the pages to be customized, including adding, removing or moving controls.

The third text box allows text to be inserted at the end of the <body> section of the document, typically javascript or a link to an associated document. This enables the enhancement of the functionality provided.

Since WebGrid pages all link a default CSS style sheet defining types for its major visual elements an ID is generated for each page and each major sub-element, the use of these customization capabilities can radically change the appearance and operation of WebGrid interaction without requiring any change to the underlying scripts.

All of the parameters, items and customization are stored with the grid.

An example of a customized options page is shown below.
Example of customized WebGrid options page
3.2 Main page

The main page giving access to other WebGrid functionality is shown below.

WebGrid main page

The five top sections show options generated by WebGrid based on the grid data and suggesting further elicitation of elements and constructs. The two sections below each have outliner arrows that are initially closed. Clicking on the “Situations” arrow opens the section showing the elements (which have been termed “situations” on the options page) as shown below.

WebGrid main page elements section
The element names are shown in a list box where none, one or more may be selected by clicking upon them (holding down the CMD key for multiple selections). The buttons at the side may be clicked to add another element, delete those selected, edit each of those selected, edit just the note attached to each of those selected, sort the elements to bring those selected to the top (appropriate selections and a sequence of clicks can achieve any desired order), or select no elements. The check box below when selected causes the selected elements to be used in the pairs or triads used in the elicitation suggestions above.

Clicking on the “Qualities” arrow opens the section showing the constructs (which have been termed “qualities” on the options page) as shown below.

- **WebGrid main page constructs section**

  The construct identifiers are shown in a list box where none, one or more may be selected by clicking upon them (holding down the CMD key for multiple selections). The buttons at the side may be clicked to add another construct, delete those selected, edit each of those selected, edit just the note attached to each of those selected, sort the constructs to bring those selected to the top (appropriate selections and a sequence of clicks can achieve any desired order), or select no constructs.

  The section below shows five icon buttons which when clicked generate different analyses of the grid. If the grid has been generated from another one that has been cached then a sixth icon button, “Compare,” will also be shown.

  ![WebGrid main page constructs section](image)

- **WebGrid main page analysis section**

  The bottom section provides five buttons to access the options page, and so on, or turn the contextual help system on or off.

  ![WebGrid main page analysis section](image)

- **WebGrid main page bottom section**

  ![WebGrid main page bottom section](image)
3.3 Element entry and edit page

The element entry and edit page has a variety of headings and preambles dependent on the reason an element is being entered or edited. The simplest is that shown below which is displayed when the “Add” button in the elements section of the main page is clicked.

![WebGrid 5 Add situation](image)

**WebGrid element addition and rating page**

The text box at the top allows an element name to be entered, and that the bottom allows it to be annotated with text that may contain HTML markup including links to images or sounds that can be incorporated in pages where the element is involved in construct elicitation. The menus in the center section allow the element to be rated on the constructs.

As shown below, a similar page is generated when the “Edit” button in the elements section of the main page is clicked showing the selected elements and their ratings, in several sections if more than one is selected.
As shown below, another variant is generated if there is a prompt on the main page to add an element to reduce a construct match, such as “The qualities flexible—rigid and self-organised—staff-organised are very similar. Do you want to enter another situation to distinguish them?” and the “Add situation” button is clicked.
**WebGrid element addition to reduce construct match page**

The radio buttons at the top allow the way in which the new element goes against the existing construct match to be entered. Clicking on one of them triggers javascript to set up the appropriate ratings as shown below, leaving the user to enter the element name and the other ratings, and to adjust those WebGrid has entered if desired.
3.4 Construct entry and edit pages

The construct entry page has a variety of headings and preambles dependent on the reason an construct is being entered or edited. The simplest is that shown below which is displayed when the “Add” button in the constructs section of the main page is clicked.

![WebGrid construct addition page](image)

*WebGrid construct addition page*

The two text boxes at the top allows the construct pole names to be entered, and that the bottom allows the construct to be.

As shown below, another variant of the construct addition page is generated if there is a prompt on the main page to add construct to reduce an element match, such as “The situations lecture and tutorial are very similar. Do you want to enter another quality to distinguish them?” and the “Add quality” button is clicked.

![WebGrid construct addition to reduce element match page](image)

*WebGrid Construct addition to reduce element match page*
The poles are now shown associated with the elements to be rated at either end of the construct, and if the user enters pole names such as “one way” for lecture and “interactive” for tutorial and clicks on “Done” then the page below is generated where the ratings of the elements to be distinguished have been preset appropriately. The user can enter ratings of the remaining elements on the new construct, and edit the present ones if they wish. They can also adjust the pole names in the light of consideration of all the elements if they wish, and click on “Update” to see the effect of the revisions before they click on “Done” to indicate they have finished entering the new construct.

WebGrid 5
Rate situation

A similar construct rating page is generated when the button to add a new construct is clicked in any variant of the construct addition page.

A similar sequence of pages is generated if there is a prompt on the main page to add a construct to represent the difference between a pair of two elements, such as “Can you think of a quality that distinguishes between the two situations, video tape and programmed text?” and the “Add quality” button is clicked.

The construct rating page is also generated when the “Edit” button in the constructs section of the main page is clicked. As shown below, it shows the selected constructs and their ratings, in several sections if more than one has been selected.
As shown below, another variant of the construct addition page is generated if there is a prompt on the main page to add a construct to represent the similarities and differences between a triad of three elements, such as “Can you think of a quality that distinguishes between the three situations, library, practical and video tape, such that two are alike and differ from the third?” and the “Add quality” button is clicked.
WebGrid triadic construct elicitation page

The radio buttons at the top allow the way in which two of the elements are similar and different from the third to be entered. Clicking on one of them and, entering the appropriate pole names and any annotation, and clicking on “Add quality” triggers javascript to set up the appropriate ratings as shown in the two pages below.
WebGrid construct rating page after triadic elicitation

The pages above are generated if none of the “Data Types” check boxes in the options page are checked and WebGrid defaults to basic grid elicitation with default rating scales. If the “Ratings” check box is checked more advanced options for a rating scale are added as shown below.

WebGrid construct addition page with all ratings options
Users can change the rating scale range, add categories naming intermediate points or sub-ranges, give the construct a name, a weight, a level relative to other constructs and specify whether it is an output to be anticipated. The weight is used in clustering and the level/output in inductive modeling.

When the “Categories” text box is used to label intermediate points or intervals on the rating scale, these labels appear in the popup menus when elements are rated on the construct. This supplements the normal labeling of the extreme rating values by the construct pole names. The first screen below exemplifies the way in which such categories are entered, and the second the way they appear in a popup menu.

![WebGrid construct addition page with categories specified for ratings](image-url)
WebGrid construct rating page with categories showing in a popup menu

If all the “Data Types” check boxes in the options page are checked then, as shown below, additional options are given to specify that the construct is a rating scale, an integer in a specified range, a floating point number in a specified range, or a set of ordered or unordered categories. If only some “Data Types” check boxes are checked then only the appropriate options will be given.
When a categorical scale is defined the construct editing page has a popup menu of categories with which to rate each element similar to those for numeric ratings. When a numeric rating scale is specified the construct editing page has an edit field for each element in which may be entered the numeric values. An example of the construct addition page and associated editing page for a numeric construct is shown below.

*WebGrid* construct addition page with all data type options
Enter another quality relevant to the context of exploring the nature of learning situations.

Enter a phrase for the way in which some of the situations are different

Enter a phrase for the way in which the others are alike

Click on a button to select the type of quality from the options below

- Rating scale from 1 to 5 (in range -100 to +100)
- Integer from to (e.g. 0 to 100, or -250 to 1000)
- Number from 0.0 to 10.0 (e.g. 0.0 to 100.0, or -250.00 to 1000.00)
- Categories Ordered (list categories on separate lines in box below)

Categories:
- 0.4 low
- 1.5 normal
- 4.1 high
- 4.10 very high

One category on each line
Categories for scales have the scale point or interval first e.g. "4 high", "1.2 low"

Name the quality and specify its weight in clustering, its level, and whether it is an output to be anticipated

Name: student's time (hours) Weight: 100 Level: 10 Output

You may also annotate the quality with a note that can contain HTML tags and links. The time involved may affect the appropriateness of a learning situation

WebGrid construct addition page with a numeric construct specified
WebGrid construct rating page for the numeric construct

3.5 Element and construct delete pages

Clicking the “Delete” button in the elements or constructs section takes the user to the delete page for the elements or constructs respectively, asking that the deletion of the selected items be confirmed or cancelled as shown below, and returning to the main page when that is done.
3.6 Display page

When the “Display” icon is clicked in the main page the grid data is displayed, typically as below.

WebGrid display page

If the “Continue” button is clicked WebGrid returns to the main page. If the “Copy Image” button is clicked the graphic display is copied into a separate window as shown below.
WebGrid display page with image copied to another window

This capability to keep a copy of the results is common to all of the analyses with graphic output, enabling intermediate results to be kept and saved.

The graphic output is a clickable image, and clicking in may be used either to edit the data or to deselect it so that it is not used in the analysis. The radio buttons under the graphic output allow one of the two modes to be selected. In the edit mode if, for example, the element name “film” is clicked WebGrid displays the element editing page for “film” as shown below.

WebGrid element editing page shown after an element name is clicked in edit mode
In the deselect mode if the element name “film” is clicked WebGrid shows the display page with the element “film” omitted as shown below.

**WebGrid display page shown after an element name is clicked in deselect mode**

These capabilities for interactive editing and re-analysis with a reduced grid are common to all of the analyses with graphic output. The edit mode enables ratings to be adjusted easily if the results of analysis suggest that some are inappropriate. The deselect mode enables the sensitivity of the analysis to particular elements or constructs to be investigated.

At the center bottom of the display there is an outliner arrow to access the “Display Parameters” section and clicking it opens the section enabling the form of the display to be specified as shown below.

**WebGrid display page “Display Parameters” section**

3-22
There are three sub-sections. The “Plot” check box when checked causes the graphic display to be produced. The “Title,” “Numbers” and “Shading” check boxes when checked cause the display to be titled, the elements and construct numbers to be shown and the ratings to have a background shading according to their value, respectively. The radio buttons under “Rows” determine whether the rows in the display will be the elements or the constructs. The check boxes under “Notes” determine whether the annotation of the elements and/or constructs will be shown in parentheses after their names.

The “Style” sub-section top row allows the font and font size used in the display to be changed, and the display to be shown in black and white. The next two rows enable the colors used in the display to be specified in hexadecimal RGB form. The color swatches enable the colors specified to be seen. Changing the hexadecimal specification and keying “return” causes the lower half of the swatch to show the new color so that it can be checked before use.

The “Use” sub-section specifies whether only selected elements and constructs will be displayed. The “All” button at the top selectts all elements and constructs to be analyzed (equivalent to selecting all of them on the main page). If any of the “Data Types” check boxes on the main page are checked this sub-section also gives the options to use element and construct weights in analyses.

Clicking on the “Display” button causes the graphic output to be produced again with the new parameters.

Similar facilities are provided for all the graphic analysis and they are coupled such that opening the parameter section opens all the analysis parameter sections, and changing the style changes all the analysis styles so that there is consistency across the results.

In the following sections on the other analysis pages only those features that differ from those for the display page will be discussed in detail.

### 3.7 Focus cluster page

When the “Cluster” icon is clicked in the main page Shaw’s (1980) Focus cluster analysis of the grid data is displayed, typically as below. See the RepGrid manual for more details of the analysis and its parameters.
In the top row of the “Plot” sub-section of the “Focus Cluster Parameters” section, the “Cutoff” value specifies how low a match is to be shown in the element and construct trees, the “Power” value specifies the Minkowski metric used in the cluster distance measure, and the “Interior” check box if checked specifies that items may be matched to interior items of a cluster, not just its edges. In the second row the radio buttons under “Tree” determine whether the tree associated with the columns should be at the top or on the right side.
The “Text” sub-section provides check boxes to determine whether a textual form of the Focus cluster results are presented and, if so, whether element and/or construct matches, links and sorts are shown.

3.8 PrinGrid page

When the “Map” icon is clicked in the main page a PrinGrid Map principal components analysis grid data is displayed, typically as below. See the RepGrid manual for more details of the analysis and its parameters.

WebGrid PrinGrid map page
The “Axes,” “Dimensions,” “Variance” and “Fit” check boxes in the top row of the “Plot” subsection determine whether the component axes should be shown, whether the construct lines should be shown, whether the percentage variance in each component should be shown, and whether the construct pole positions will be scaled to have the same maximum position as the element positions, respectively. The X, Y and Z popup menus and “Reverse” check boxes in the second row specify what component should be plotted on each axes and whether the direction of the axis should be reversed. The “3D” check box on the third row if checked specifies a three-dimensional plot with axes rotated as specified by the three values to its right.

The “Components,” “Situation loadings” and “Quality loadings” in the “Text” sub-section specify whether, if a textual version of the analysis is output, the percentage variance for each component and the loadings on the elements and constructs should be shown.

3.9 Crossplot page

When the “Crossplot” icon is clicked in the main page a crossplot of the elements on the selected is displayed—a 3D plot is shown below. See the RepGrid manual for more details of the analysis and its parameters.
WebGrid Crossplot page

The X, Y and Z popup menus and “Reverse” check boxes in the “Plot” sub-section determine which constructs are used on each axis and whether the ratings are reversed.

3.10 Compare page

When one grid is derived from another grid that has been cached then a “Compare” icon appears in the analysis section of the main page. When this is clicked in the main page a comparison of
the two grids is presented as shown below where Mary’s exchange grid with Arthur is compared with Arthur’s original.

**WebGrid 5**

*Compare Mary (Arthur (after class discussion) exchange) with Arthur (after class discussion)*

Compare Mary (Arthur (after class discussion) exchange) consensus with Arthur (after class discussion) 74.60 “exploring the nature of learning situations”

Grids have common elements and/or common constructs may be compared, and the “Compare Parameters” provide the same degree of control over the comparison as they do in the standalone *RepGrid* program, and the *RepGrid* manual should be consulted for details.

**WebGrid Compare page**

Grids have common elements and/or common constructs may be compared, and the “Compare Parameters” provide the same degree of control over the comparison as they do in the standalone *RepGrid* program, and the *RepGrid* manual should be consulted for details.
The capability to deselect items and show how the analysis changes with fewer items can be used to implement Shaw’s (1980) Core procedure for determining the stable core of a conceptual representation from grids elicited at different times or from different people. For example, the comparison below has resulted from the least matched elements and least matched constructs in the comparison above being successively deselected.

**WebGrid Compare page after lowest matched items have been deselected**

### 3.11 Match page

When the “Match” icon is clicked in the main page tables of matches between the elements and between the constructs are displayed as shown below.
### Matches Between Situations

<table>
<thead>
<tr>
<th>Situations</th>
<th>Match %</th>
</tr>
</thead>
<tbody>
<tr>
<td>library</td>
<td>85.7</td>
</tr>
<tr>
<td>informal</td>
<td>interaction</td>
</tr>
<tr>
<td>programmed</td>
<td>82.1</td>
</tr>
<tr>
<td>text video</td>
<td></td>
</tr>
</tbody>
</table>

### Matches Between Qualities

<table>
<thead>
<tr>
<th>Qualities</th>
<th>Match %</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable content -- specific content</td>
<td>94.4</td>
</tr>
<tr>
<td>like -- dislike</td>
<td></td>
</tr>
<tr>
<td>flexible -- rigid</td>
<td>86.1</td>
</tr>
<tr>
<td>like -- dislike</td>
<td></td>
</tr>
<tr>
<td>flexible -- rigid</td>
<td>83.3</td>
</tr>
<tr>
<td>variable content -- specific content</td>
<td></td>
</tr>
<tr>
<td>flexible -- rigid</td>
<td>80.6</td>
</tr>
<tr>
<td>self-organised -- staff-organised</td>
<td></td>
</tr>
<tr>
<td>self-organised -- staff-organised</td>
<td></td>
</tr>
<tr>
<td>small group -- large group</td>
<td></td>
</tr>
<tr>
<td>involvement -- remoteness</td>
<td></td>
</tr>
<tr>
<td>self-organised -- staff-organised</td>
<td></td>
</tr>
<tr>
<td>involvement -- remoteness</td>
<td></td>
</tr>
<tr>
<td>flexible -- rigid</td>
<td></td>
</tr>
</tbody>
</table>

### WebGrid Match page

The “Match Parameters” section provides the same options as they do in the standalone RepGrid program, and the RepGrid manual should be consulted for details.

#### 3.12 Save grid page

When the “Save Grid” button is clicked at the bottom of the main page a page presenting the user with a number of options for saving the grid is displayed as shown below.
WebGrid 5
Save Grid

You are considering 9 situations and 7 qualities in the context of exploring the nature of learning situations.

Saving and restoring your data to continue session later

All of your data is stored in this document.
Save it on your local computer as an HTML source file.
Use the "Save As..." option in the "File" menu of your browser.
You may continue the interaction by opening this file and clicking on "Continue". Check that you can save your data before any major use of WebGrid.

Continue

Downloading your data for use in other programs

You may download your data as a Rep 6 grid file that can be saved locally, opened in Rep 6, or imported into a spreadsheet as tab-separated data.

Download

Caching a grid for use by others

You may temporarily cache the grid on the WebGrid server for use by others.

Cache

Caching a grid in an allocated cache directory

Specify your cache directory if you have been allocated one.

Access directory
Cache in directory

WebGrid save grid page

The top section explains how to save the grid and the session by simply saving the page using the “Save As...” command in the web client. Each page contains the grid data and a link to the server such that when it is reopened WebGrid interaction can continue. If the page is saved as HTML rather than a web archive it can be opened in RepGrid as if it were a grid file. Clicking the “Continue” button return the user to the main page.

Clicking the “Download” button in the second section downloads the grid data in a grid file named “WebGrid.rgrid” that can be renamed appropriately and opened in RepGrid.

To support grid comparison and remote data collection WebGrid supports the capability to cache grids at the server in a sub-directory of the directory “/ServerData/Cache/” and to make these accessible through the web. Clicking on the “Cache” button in the third section caches a copy of the grid and generates the page shown below which gives URL’s for accessing the cached grid for various purposes.
3.13 Cached grids page

If a user has registered their own cache an associated cache management page may be accessed through a URL of the form:

http://serveraddress:serverport/WebGrid/Cache/cachename

Below is shown a cache management page where the cache name is 121D7851647.
WebGrid show cache page

The table at the top of the page shows the four grids in the cache. The radio buttons in the first two columns allow a grid to be selected as “A” and another as “B.” Clicking on the grid name displays the grid. Clicking on “Remove A” removes the grid. This button only appears for password-protected caches, not for caches that are not protected so as to make the data generally available. Clicking on the “Download A” button downloads the grid as a grid file with the name “WebGridCache.rgrid” that may be renamed appropriately. Clicking on “Refresh table” reloads the page so that one can see if any more grids have been cached.

The four buttons on the next row provide the capability to open grid A and run various forms of analysis. The four buttons below provide the capability to open a copy of the grid, an exchange copy with ratings set to be open, a copy with only the elements, or a copy with only the constructs. These copies contain a pointer back to the grid from which they are derived and when the user edits them the edited grid is also automatically written to the cache to support remote data collection. Generally one wants to make a URL available to others so WebGrid does not cache the new grid immediately it is created thus allowing the URL to be copied without affecting the cache.

The “Commands” field below allows a sequence of commands to be added to the URL of a derived copy, for example, to specify some triads for elicitation in an “Elements” copy. The “Path” allows a script path to be specified in the URL so that customized scripts can be used supporting the remote data collection process.

3.14 Register cache page

As shown below page to register a cache may be accessed through the URL:

http://serveraddress:serverport/WebGrid/Cache
WebGrid 5
Register Cache

Registering a Cache Directory for Your Data

WebGrid allows you to register a cache directory in which you may save grid data.

You will have password-protected access to your cache management page.

It enables you to view the grids in your cache, analyze and compare them, and load them in various forms for further use.

You will be able to supply your clients with URL's enabling them to access copies of one or more grids in your cache.

(If when they modify these grids the results are automatically stored in your cache.)

This facility enables WebGrid to support distributed research studies.

Registration Information

Fill in the form below.

The data collected is private to you and will not be made available to others.

We will use it only if we need to contact you.

Your name

Email

Telephone

Institution

Some notes on your intended use, particularly approximate timescales if known

Cache Name, Account and Password

Please supply an account name and password of at least 6 characters that you will use to access your cache directory.

Account name

Password

Enter password again

When you click on the button below WebGrid will send you the name of your cache directory and a URL to access it.

Register

WebGrid register cache page

This page collects data about the registrant that may be useful to the WebGrid server owner in managing caches created by otherwise anonymous users together with an account name and password supplied by the registrant to give them protected access to the cache.

One possible customization is to remove the cache facilities if they are not required or to require the WebGrid server owner’s approval for registering a cache.
3.15 Comment page

The comment page is displayed when the “Comment” button at the bottom of the main page is clicked. It is intended to allow users to send messages to the server operator. The email address shown is that specified in the RepServe pane of the “Rep 5 Manager” window.

It could be customized in various ways: minimally, change the email address to direct the comments appropriately; add a text box for comments, the contents of which are appended to a comments file; remove the “Comments” button on the main page if the capability to comment is not required.

3.16 Finish page

The finish page is displayed when the “Finish” button at the bottom of the main page is clicked. It is intended to allow users to take a definite action when their grid elicitation is complete and to remind them of how to save their grid. It needs customization to support particular applications of WebGrid
3.17 Contextual help system

The 🔄 icons that appear next to the various options on WebGrid pages provide access to a “help” window that provides more detail on the associated options. As shown below, when the mouse is moved over a 🔄 icon the cursor changes to 🔄 indicating that that help will be provided when it is clicked.

WebGrid cursor changing to “?” as user mouses over a contextual help “?”

The “help” window for the top item on the main page is shown below. It provides a short explanation of the page and of the options offered by the two buttons next to the help icon.
WebGrid contextual help window

The help window can be resized and moved to a convenient location so that it is always accessible. It will reopen if closed and will come to the front if hidden.
4 WebGrid tutorial: “Carol’s life crisis”

This is an example of using a grid to support a personal decision-making process based on Chapter 5 of *Think Again* (Shaw and McKnight, 1981). It is based on a generic customization of the *Elicit* script which provides interactive elicitation of elements and significant constructs and then transfers to the main page with a command sequence that leads the user to rate the elements on the constructs they have entered and generate more constructs through triadic elicitation.

4.1 The problem situation

Carol is a middle-aged housewife who is in a rut. The worst part is, she knows she is in a rut but can’t think clearly enough to get out of it. Her husband works long hours as a successful self-employed businessman and her son has gone away to school, so she sits at home thinking, and her thoughts seem to go round and round in circles. Carol has arrived at what we might call a crisis point in her life. She feels it is time for a change, but is not sure in which direction she ought to go.

4.2 Developing a grid to help resolve the problem

One day, Carol’s thoughts and feelings are so pressing that she decides she cannot continue without sorting them out. Her mind is so jumbled that she cannot just sit down and think out her problems. She decides to try using a grid so that she can at least look at her problem objectively. She hopes that the exercise may help break her out of her rut.

Carol’s rut is one born of habit, but it has not always been that way. Before Carol and her husband John were married, Carol had a job as personal assistant to the director of an engineering firm. Her job meant long hours. But every day was different, and often she was taken to dinner by her boss on business occasions. Indeed, it was on one such occasion that she met John who, at the time, was starting his own business.

After a brief romance, Carol and John were married. A year later their son was born. Carol had given up her job shortly after they were married, and once their son was born, she found that mothering was a full-time job. John spent all his waking hours working to build up his business.

Now, eighteen years later, Carol’s son has just left home for university and Carol now realizes that she has spent all that time serving her family’s needs. John is still working all day, every day, even though his business is now thriving. Work has become a habit to John; he fears his business would collapse if he worked less. Now the only times Carol meets people are either when she shops or when John brings business associates home for dinner.

Carol feels that there is not a “mental” solution to her problem—she must do something. The trouble is she doesn’t know what to do. She decides to use *WebGrid* to help her consider her problem and starts a new grid as shown below.
She enters her purpose as “to think about possible courses of action to improve my life”, the entities relevant to her purpose as “possibilities”, and the distinctions relevant to it as “consequences” as shown below.
WebGrid then asks her to list significant possibilities as shown below.

![WebGrid IV: Possibility Elicitation](image)

### 4.3 Possible courses of action as grid elements

The first possibility that occurs to Carol is that she could get a job again. Her typing and shorthand used to be very good, and she is sure it would take her very little time to brush up on them. The local university often has vacancies for secretarial staff, so she is fairly sure she could get a job.

Upon reflection, Carol decides that she really has two possibilities here. She knows that there are often part-time secretarial jobs available at the university, so she lists her first two possibilities as “take a full-time job” and “take a part-time job.” Any job would give her some money that she could spend as she wished. Obviously, a full-time job would give her more money than a part-time job, but a part-time job would leave her with more free time.

Thinking about the university leads Carol to another possibility. Although she is now middle-aged she knows that it might be possible for her to become a student. She had always encouraged her son to get as good an education as possible because she felt that she had not been given such an opportunity. However, when she took her son to one of his college interviews, she was surprised to find that a few of his fellow interviewees were closer to her age than his! Talking to them while she waited for her son, she discovered that it was not uncommon for social science departments to accept mature students. Her son was majoring in sociology, but she thought that she would prefer psychology, having always maintained an interest in people and their behavior. She would like to study the subject formally, although she now finds the idea of a full-time program a little daunting. As her third possibility she lists “take a psychology course.”

The doubt Carol feels about taking such a course leads her to her fourth element. She knows there are many night school classes that she could attend. They could involve not only academic but leisure interests, for example, guitar playing, dress making, and so forth. Rather than make her mind up now, Carol simply lists “take an evening class” as her fourth possibility.

Carol misses the daily contact with her son, something she has had for the past eighteen years. With John working such long hours, she sometimes does not see another human being from one day to the next. In short, she is often lonely.

At her age, Carol knows that she is a little old to have another baby. She is still physiologically able to give birth, but she knows that the risk of having a handicapped child is greater at her age. However, she could easily foster a child. This would give her contact with a child again, but it would also mean continuing in a serving role. She has not discussed the idea with her husband yet, but the idea is appealing enough for her to include it as her fifth possibility as shown below.
These possibilities seem somewhat mundane and Carol then begins to think of more exotic ones, clicking on “Add more possibilities” to enter them.

Another way she can think of to counteract her loneliness relates to the lack of contact she has with her husband. A few weeks ago, John brought Mike, a business associate, home to dinner. During the course of the evening, Mike (a single man) was very flattering to Carol, particularly when John was out of the room. On the basis of this encounter, Carol is fairly confident that she could at least have an affair with Mike if she wanted to. She is not sure whether such an involvement would be only a temporary answer to her problem. She has no way of knowing at this stage whether Mike would be interested in a permanent relationship and is not sure herself whether she wants to leave her husband, but she decides to include “have an affair” as her sixth element. Because of her uncertainty about staying with John, she decides to include “leave husband” as her seventh element.

Carol decides to include the possibility of emigrating as her eighth element. She has a married sister in England and has enjoyed holidays there without John. She knows John would not leave the United States, so going to England would mean leaving him. However, it would be a different type of separation than if she simply moved out of their house and stayed in the same neighborhood. The latter would be a possibility if Mike were interested in a long-term relationship.

As a final element, Carol decides to include “carry on as before.” She is prepared to believe that her present worries may just fade away, that she is just overreacting to small changes and that she will adapt to her new situation. However, she is not very confident that she can adapt. The pressure she feels seems to demand action, but she doesn’t want to rush into any action she might later regret.
4.4 Possible consequences of action as grid constructs

*WebGrid* now gives Carol the opportunity to think about the important consequences that might result from her different possible courses of action.

Carol thinks about what would be improvements in her life and realizes she wants more independence and enters this as an important consequences.

She also wants more excitement to make life less humdrum. She clicks on “Add more consequences” and describes this consequence.
She cannot think of any other important consequences immediately and clicks on “Done” to let WebGrid continue.

WebGrid lists the possibilities she has entered and the important consequences she wishes to consider, and offers her the opportunity to add more before she begins to rate the possibilities in terms of their consequences.

She is content with what she has done and clicks “Done” to move on to the next phase of the elicitation.
4.5 Rating the possibilities in terms of the consequences

*WebGrid* asks her to rate all the possibilities she has entered on her first consequence dimension.

Carol clicks on the popup menu next to “take a full-time” job and rates this as “5 gain in independence” as it will very much help her achieve this objective.
She goes on to rate each of the other possibilities in terms of this consequence.

She clicks on “Update” and WebGrid sorts the possibilities according to her ratings so that she can see which possibilities offer greater independence and decide whether she has rated them appropriately or whether she wishes to change any ratings.
What she has done makes sense to her so she clicks on “Done” and WebGrid asks her to rate her possibilities her second consequence, that they might be humdrum or exciting.

She is content with her ratings and clicks on “Done”.

4.6 Elicitation of consequences by considering similarities and differences in triads of possibilities

WebGrid presents her with three of her possibilities, “take a psychology course”, “take a full-time job” and “take a part-time job”, asking her in what way two are alike and different from the third.
The first thing that occurs to her is that both the job possibilities are alike in that they would provide her with some money, but taking the course differs because it would actually cost her money. She enters this information.

Carol then clicks on “Add consequence”, and WebGrid asks her to rate her possibilities in terms of the consequences, “cost more money—make more money”, having already rated the three possibilities she has already considered.
Carol rates the other possibilities and changes the rating on “take a part-time job” because she will make less money part-time.

She clicks on “Done” and WebGrid presents her with another three of her possibilities, “take an evening class”, “foster a child” and “have an affair.” In this case, the evening class and the affair possibilities seem similar because Carol knows that she could stop them both if she found that they were not what she wanted. On the other hand, fostering seems irreversible. It would be emotionally unfair to the fostered child to change her mind once the process had begun.
Again, WebGrid asks her to rate the remaining possibilities in terms of these consequences.

She clicks on “Done” and WebGrid presents her with another three of her possibilities, “leave husband,” “emigrate,” and “carry on as before.” She knows that both leaving her husband and emigrating would disrupt John’s life, but if she simply carries on as before, she is fairly sure that he will do the same.
Again, WebGrid asks her to rate the remaining possibilities in terms of these consequences.

4.7 WebGrid's main page and analysis of Carol’s conceptual representation

When Carol clicks on “Done”, WebGrid takes her to its main page where it has already begun to analyze her conceptual representation of her possibilities and their consequences.
The top five sections suggest various actions she might take next. The first section gives her the option to have WebGrid choose what she does next, or to provide her with other choices. The second section notes that the possibilities “take a psychology course” and “take an evening class” are very similar in their consequences and asks whether she wants to enter another consequence that might distinguish them. The third section notes that the consequences “no gain in independence—gain in independence” and “wouldn’t disrupt John—would disrupt John” are very similar, and asks whether she wants to enter another possibility to distinguish them. The fourth section asks whether she can think of a consequence that distinguishes between the three possibilities, “take a part-time job”, “take a full-time job” and “have an affair”. The fifth section asks whether she can think of a consequence that distinguishes between the two possibilities “take a full-time job” and “have an affair.

All of these questions and suggestions are based on the possibilities, consequences and ratings that Carol has entered, and indicate aspects of her conceptual representation that will help her to think about possible courses of action to improve her life.

The sixth section shows an outliner arrow. If she clicks on it the possibilities she has entered will be shown, giving her the opportunity to add, delete, edit or sort possibilities.
The seventh section also shows an outliner arrow. If she clicks on it the consequences she has entered will be shown, giving her the opportunity to add, delete, edit or sort consequences.

The eighth section has a set of five icons illustrating various ways in which she can view and interpret her conceptual representation based on her grid content, and the ninth and final section offers various additional options.

4.8 Continuing the elicitation under Carol’s control based on WebGrid’s suggestions

Carol realizes from the questions in the fourth and fifth sections that she has not considered the consequences in terms of physical or emotional strain and chooses to add it as a consequence that distinguishes between the two possibilities “take a full-time job” and “have an affair.”
She then rates all her other possibilities in terms of this consequence.

4.9 Interacting with the clustered grid to assess Carol’s possible courses of action

By the time she has reached this point, Carol feels that she is really getting somewhere. Rather than go on adding more possibilities and consequences, she decides to look at a conceptual representation based on her grid and make some decisions on that basis.

She clicks on the “Cluster” icon which provides a Focus analysis she has found useful before that sorts her possibilities to bring similar ones together, and sorts her consequences to bring similar ones together. She likes this approach because it does not change anything she has entered and she can see the relations between her ratings and the structure represented.
Carol can see from the sorted grid that her possibilities cluster into three groups based on their consequences: emigrating and leaving her husband have similar consequences; taking courses or jobs have similar consequences; and, to a lesser extent, having an affair, carrying on as before or fostering a child have similar consequences. When she looks at the grouping of consequences, reversibility, excitement, independence and disrupting John group together.

The consequence “make more money—cost more money” does not seem to her to be important. She doesn’t feel that her present predicament would be solved by gaining money or that an alternative should be ruled out because it would cost money. She clicks on it to remove it from consideration and the representation changes a little.

Looking at the consequence “reversible—irreversible” Carol decides that she does not want to do anything irreversible. She knows that she is going through a hard time, but she doesn’t want to overreact and do something she might regret later. Therefore, she clicks on the possibilities that seem irreversible “foster a child” and “leave husband” to remove them from consideration.

Looking at the consequence “would disrupt John—wouldn’t disrupt John,” Carol feels that she would not mind if John were slightly disrupted. After all, why should she be the only one to suffer all the time? But she doesn’t want John to be greatly disrupted because he has worked so hard to give them both a high standard of living. With this in mind she decides to remove the possibilities “emigrate” and “take a Psychology course.”
Looking at the consequence “humdrum—exciting” Carol realizes that the only really humdrum possibility left is “carry on as before.” She is fairly sure that she does not want to carry on as before, so Carol removes this possibility.

Looking at the consequence “physical strain—emotional strain,” Carol is fairly sure that she does not want any more emotional strain, although physical strain does not worry her much. As a result she removes the possibility “have an affair.”

Carol is left with the possibilities “take a full-time job,” “take a part-time job,” and “take an evening class.” Looking at these three possibilities in terms of her initial significant consequences “no gain in independence—gain in independence” and “humdrum—exciting” Carol can see that all three are roughly equivalent.

She could continue her interpretation but decides that she already has clarified many of her thoughts and feelings and has reduced her possible courses of action. Rather than carry on with
the exercise, Carol decides that she would like to talk to John about the three possibilities. It is possible that he is so wrapped up in his work that he simply is not aware of her problems, so talking about these things would be a good way of introducing him to the problems.

Brimming with optimism and clear-headed, Carol feels that if she decides to take an evening class, perhaps she could take a psychology course and pursue her interest in people and their problems.

4.10 Saving Carol’s grid for later use

Carol is pleased with her grid and wants to keep a copy to review in future. She clicks “Continue” to return to the main WebGrid page, and then clicks “Save Grid” to keep a copy of her grid on her computer.

All she need do is use the “Save As...” command in the “File” menu of her browser to file her data on her disk. At any time she can re-open this file and it will automatically reconnect to WebGrid and let her review her possibilities, consequences and ratings again, edit them, add to them as further possibilities and consequences emerge, and generally keep track of whether she is actually improving her life.

If her relations with John improve she may someday share her thoughts about her life crisis with him, and maybe that will help them both. Meanwhile, sharing them with a friendly and supportive computer system has helped her clarify her options and encouraged her to take action to alleviate her problems.
4.11 Making a decision directly from the clustered grid

The scenario above follows as closely as possible that described in *Think Again* (Shaw and McKnight, 1981) where Carol develops a grid manually and has no tools for its analysis. Her sequential interaction with the grid in removing some possibilities is valuable in its own right as it encourages her to assess each of her possible courses of action in terms of what she sees as their likely consequences. However, it is also possible to arrive at much the same conclusions directly from WebGrid’s graphic analysis of her conceptual representation.

The cluster of possibilities at the bottom, having an affair, carrying on as before or fostering a child, can be seen as similar in that, although they would not disrupt John, they give Carol no gain in independence. The cluster at the top, emigrating or leaving her husband, do improve her independence but she does not see them as exciting and they do involve emotional strain. That leaves her with the cluster of taking courses or job in the middle, all of which she sees as reversible, exciting, gaining independence and not involving emotional strain. Out of those, she sees taking a psychology course as most disruptive to John and might prefer the others because of that, leaving her with same outcome as before. Thus, it is possible to come to conclusion based primarily on interpretation of the conceptual representation created by using Focus to sort and cluster the grid.

4.12 Making a Decision Directly from the Mapped Grid

Carol used the Cluster analysis option to view and interact with her conceptual representation because she likes to see her ratings as she thinks about her representation of her problem. Others may prefer the Map analysis option next to it on WebGrid’s main screen that treats the consequences as dimensions in space and plots a map of the options.
Carol can see from the mapped grid that her possibilities cluster into same three groups as before: emigrating and leaving her husband have similar consequences, irreversible and disruptive to John; taking courses or jobs have similar consequences, reversible, physical strain, exciting and gain in independence; and having an affair, carrying on as before or fostering a child have similar consequences, being humdrum, no gain in independence and not disrupting John.

The options of emigrating and leaving her husband seem unattractive because they are irreversible and disruptive so Carol clicks on them to remove them.

The option to foster a child seems humdrum and that to carry on as before is also humdrum and has no gain in independence. Carol removes both.
The option to have an affair involves no gain in independence and emotional strain so Carol removes it.

This leaves the job and course options in a very small map with little variation. Interestingly enough, the main source of variation left is the consequence of making money or costing money which, as already noted, is not important to her.

**4.13 Making the decision from the crossplot of important consequences**

Carol can also use the “Crossplot” analysis to display her possibilities plotted in relation to her most important consequences, “gain in independence” and “emotional strain”.
The three clusters and her primary considerations in choosing those in the top right quadrant are clearly apparent in this plot which provides a simple rationale for Carol’s choice of possibilities on which to base her future plans.

4.14 Making a decision based on an “ideal possibility”

Another technique commonly used when a grid has been developed to help in making a choice between various possibilities is to add an “ideal possibility” that represents the best possible choice, rate it on the consequences, and see what are the actual possibilities that most closely match it. Carol does this by clicking on the “Add” button by the side of her list of possibilities.

She enters “ideal course of action” as a new possibility, and rates it on six consequences she has already entered: as giving her a gain in independence, being exciting, neutral as to costing or making money, reversible”, neutral as to its effect on John, and preferring physical strain to emotional strain.

She clicks on “Add possibility” and returns to the main screen with the new possibility selected.
She clicks on the “Matches” icon and is shown the matches of the other possibilities to the new one (note that the options chosen are to match the selected possibility with all the others, with the cut off set to zero so that all matches show).
It is apparent that the highest three matches are those she had come to consider as most appropriate in the previous approaches discussed above. If she had clicked on the “Cluster” icon she would reach the same conclusion from the clusters shown.
And the same cluster of possibilities is apparent is she had click on the “Map” icon.

4.16 Summary

This example has shown how developing a grid can support a significant decision-making process. A number of approaches to interpreting the grid have been illustrated. It is not expected that all of them will be used or that they will be equally attractive and meaningful. What is important is that they all lead to much the same interpretation, showing that it is the grid, not the method of interpretation, that supports the decision. Users should be encouraged to explore different approaches and choose those they find most meaningful.

Note that much the same interpretation may be derived directly from clusters, maps and crossplots as from the interactive process of removing possibilities and consequence dimensions that was first described. However, direct interpretation may not give the same feeling for the underlying basis of the clustering as does the interaction. It is often more important that those making significant decisions understand the basis on which they are making them than that they are shown the ‘right’ choice to make relative to the way they construe the issues.

Note also that it may be ‘obvious’ that the more extreme options are not going to be chosen but it is important that they be considered and that their roles within the space of all options be made apparent. Sometimes those extreme options may become the appropriate choices despite their negative consequences, and it can then be important when choosing them to prepare to manage those consequences.
5 References

Some of the reports cited in the manual are available through http://repgrid.com


