10 Appendix: Elicit Scripts

The sample scripts for elicitation are in English and emulate Shaw's (1980) PEGASUS program. Facilitators may wish to edit them to make them more appropriate to particular communities and to translate them into other languages. The scripts are listed here to illustrate what is involved.

For many purposes there will be no need to change the programming and modifying the quoted text which constitutes the dialog will be sufficient. The programming language is fully documented in the ***RepScript Manual*** and editable in the Rep Plus ***Script Editor*** which automatically lays it out and colours it to make the syntactic form more apparent.

10.1 Script: Elicit Grid.repscript

The **Elicit Grid** script sets up the text styles, shows the welcome message, and passes control to the ***Main*** script at the ***Initial*** entry point.

```
// PEGASUS-style (Shaw 1980) conversational elicitation with feedback from matches

// set up standard text styles
StyleAdd("Text",1,0,12,RGB(0,100,0),"") // 12 pt green
StyleAdd("Center",2,0,12,RGB(0,100,0),"") // 12 pt green centred
StyleAdd("CenterBold",2,1,12,RGB(0,100,0),"") // 12 pt green centred bold
StyleAdd("HeadBig",2,1,18,RGB(255,0,0),"") // 18 pt red centred bold
StyleAdd("Head",2,1,14,RGB(255,0,0),"") // 14 pt red centred bold
StyleAdd("CenterBoldBig",2,1,14,RGB(0,100,0),"") // 14 pt green centred bold

// initialize appearance
SetBackColor(RGB(240,240,255))
StyleSet("Text")

// Commence flow of interactive dialog
Select Case vGet
Case ""
  UndoSave("Elicit Grid")
  SetMessage("")
  hSet("true","Initial") // set a flag to tell other code we are in initial phase
  if gGet("Context")="" or gGet("Name")="" or gGetI("ne")<6 then
    TextClear
    Output("PEGASUS Plus"+EOL+EOL,"HeadBig")
    Output("Program Elicits Grids and Sorts Using Similarities"+EOL+EOL,"Head")
    Output("This is a program to elicit a Repertory Grid. ")
    Output("A repertory grid is a technique devized by George Kelly to help you explore the
        dimensions of your thinking. ")
    Output("Please read carefully everything that is shown, and make sure you understand what you
         have to do. ")
    Output(EOL+EOL)
  end if
  ScriptFlow("/Elicit/Main/Initial") // normal entry
Case "*" // re-entry to script window when grid edited elsewhere
  TextClear
  Output("Continue Repertory Grid Elicitation"+EOL+EOL,"Head")
  ScriptFlow("/Elicit/Main/Initial")
End Select
```

10.2 Script: Elicit/Main.repscript

The **Main** script controls the flow of elicitation, passing control to other scripts which take action and then return to the Main script.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

sub ClearMatches()
  dim i,n As integer
  n=gGetI("MatchC","80","0","M","MV") //get rid of construct matches in existing grid
  for i=0 to n-1
    hSet(vGetI(i,"MV"),vGet(i,"M"),"MatchC")
  next
  n=gGetI("MatchE","80","0","M","MV") //get rid of element matches in existing grid
  for i=0 to n-1
    hSet(vGetI(i,"MV"),vGet(i,"M"),"MatchE")
  next
end sub

Select case ScriptState

Case "Initial" // initial entry point
// main ScriptFlow of control
 hEmpty("MatchE")
 hEmpty("MatchC")
 ClearMatches
 ScriptFlow("CheckName","Context")

Case "Context"
  ScriptFlow("CheckContext","InitialElements")

Case "InitialElements"
  SetMessage(gGet("Name")+" is considering """+gGet("Context")+"""")
  ScriptFlow("GetInitialElements","CheckExchange")

Case "CheckExchange"
  if status=3 or status=5 then // exchange or constructs
    ScriptFlow("Constructs/RateAnyOpen","Options") // don't test for matches
  else
    ScriptFlow("Triads","CheckMatches") // new grid or elements
  end if

Case "CheckMatches"
  ScriptFlow("Match/DoMatch","Options")

Case "Options"
  hRemove("Initial") // unset flag
  TextClear
  Output("Select an option"+EOL+EOL,"Head")
  Output("Your grid has "+str(ne)+" "+ten+" and "+str(nc)+" "+tcn+", and you may chose what to do
       next. ")
  Output("Click on one of the options below to select it."+EOL+EOL)
  if nc<gGetI("LimitC") and ne>2 then Output("T\Elicit another "+tc+" from a triad"+EOL,"
      CenterBold")
  Output("E\List and edit your "+tes+EOL,"CenterBold")
  Output("C\List and edit your "+tcs+EOL,"CenterBold")
  Output("X\Finish now"+EOL,"CenterBold")
  ScriptWait("OptionsClick",kClick)
Case "OptionsClick"
  select case InCode
  case "T"
    ScriptFlow("AddConstruct/DoTriadChosen","CheckMatches")
  case "E"
    ScriptFlow("Elements/List","CheckMatches")
  case "C"
    ScriptFlow("Constructs/List","CheckMatches")
  case "X"
    ScriptFlow("Constructs/RateAnyOpen","Halt") // Check for unrated given constructs
  case "$"
    ScriptFlow("Options")
  else
    Alert("Not editable","The item in which you clicked cannot be edited")
    ScriptWait("OptionsClick",kClick)
  end select

Case "Halt"
  TextClear
  Output(EOL+"User has finished"+EOL,"Head")
  Halt("")

Case "CheckName"
  if gGet("Name")="" then
    hSet("false","Initial")
    Output("    What is your name or identification: ")
    ScriptWait("InputName",kText)
  else
    ScriptFlow
  end if
Case "InputName"
  gSet("Name",Input)
  Output(EOL+EOL)
  ScriptFlow

Case "CheckContext"
  if gGet("Context")="" then
    hSet("false","Initial")
    Output("State your purpose"+EOL+EOL,"Head")
    Output("You must decide on a purpose for doing the grid and keep this in mind when you chose
        the "+tes+"--")
    Output("the things you are going to think about during the program. These "+tes+" will then
        be used to elicit "+tcs+".")
    Output(EOL+EOL)
    Output("If you make a typing error press the delete key as many times as you want to eraze a
        character, then carry on.")
    Output(EOL+EOL)
    Output("    What is your purpose? ")
    ScriptWait("InputContext",kText)
  else
    ScriptFlow
  end if
Case "InputContext"
  gSet("Context",Input)
  Output(EOL+EOL)
  ScriptFlow

Case "GetInitialElements"
  if ne<6 then
    if hGet("Initial")="false" then TextClear // if we collected name or context
    hSet("false","Initial")
    Output("Name six or more "+tes+EOL+EOL,"Head")
    Output("You must choose a set of "+tes+" keeping in mind why you want to do this grid. ")
    if tes="Elements" and status<>5 then // only output following text if type of elements has
        not been specified and not a "Constructs" copy
      Output("They could be people, events, pieces of music, pictures, books or what you want, ")
      Output("but whatever you chose they must be of the same type and each must be well known to
           you. ")
      Output("Try to choose specific things. ")
    end if
    Output("Now type each one after each colon. ")
    Output("Do not forget to press return after each. ")
    Output("When you have entered as many "+tes+" as you wish, just press return alone to
        continue")
    Output(EOL+EOL)
    ScriptFlow("GetElements")
  else
    ScriptFlow
  end if

Case "GetElements"
  Output("  What is "+te+" "+str(ne+1)+": ")
  ScriptWait("InputElement",kText)
Case "InputElement"
  Output(EOL)
  if Input="" then
    if ne>=6 or status=5 then // "Constructs" copy allowed to avoid triadic elicitation
      Output(EOL)
      ScriptFlow
    else
      Alert("Not enough "+tes,"You have entered only "+str(ne)+" "+tes+". You need at least 6 to
          elicit a grid.")
      ScriptFlow("GetElements")
    end if
  else
    hEmpty("X")
    hSet(Input,"Name","X")
    gSet("NewE","X")
    ScriptFlow("GetElements")
  end if

Case "Triads"
  hSet("","E") // select random triads
  select case nc-sGetI(gGet("OpenC")) // don't include any given constructs that are totally
      unrated
  case 0
    ScriptFlow("AddConstruct/DoTriad","Triads")
  case 1
    TextClear
    Output("How to think about "+tcs+EOL+EOL,"Head")
    Output("Now you have one "+tc+" you know what to do. ")
    Output("You may think of "+tcs+" as lines along which each of your "+tes+" has a place in
        relation to all the other "+tes+". ")
    Output("Please do not use "+tcs+" which do not apply to all your "+tes+". ")
    Output("An example of this is redhead--blond, as it is impossible to rate a person with black
         hair on this "+tc+". ")
    Output("One pole must be in some sense what the other is not, and they must divide your "+tes
        +" into two approximately equal groups, ")
    Output("so please try to avoid "+tcs+" where nearly all the "+tes+" are at one end. ")
    Output("An example might be ""extremely tall--not extremely tall"""+EOL+EOL)
    ScriptFlow("AddConstruct/DoTriadNoClear","Triads")
  case 2
    ScriptFlow("AddConstruct/DoTriad","Triads")
  case 3
    ScriptFlow("AddConstruct/DoTriad","Triads")
  else
    ScriptFlow
  end select

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

10.3 Script: Elicit/Elements.repscript

The **Elements** script lists, adds and rates elements, and allows members of a triad to be chosen.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

dim i,j,jj,n As integer, lhp,rhp,v,e,e1,e2,s,t As string, b As Boolean

Select case ScriptState

Case "List"
  TextClear
  Output("List of "+tes+EOL+EOL,"Head")
  for j=0 to ne-1
    n=gGetI("OpenE",str(j))
    if n=0 then s="" else s=" ("+str(n)+" "+tcs+" not rated)"
    Output("E"+str(j)+"\"+"    "+gGet("E",str(j))+s+EOL)
  next
  Output(EOL)
  if ne<gGetI("LimitE") then Output("A\Click here to add another "+te+"."+EOL,"CenterBold")
  Output("$\Click in an item to edit it, or here if you have finished editing."+EOL,"CenterBold")
  ScriptWait("ListClick",kClick)
Case "ListClick"
  select case Left(incode,1)
  case "$"
    ScriptFlow
  case "A"
    ScriptFlow("AddElement","List")
  case "E"
    hSet(Right(incode,len(incode)-1),"E")
    ScriptFlow("EditElement/DoEditElement","List")
  else
    Alert("Not editable","The item in which you clicked cannot be edited")
    ScriptWait("ListClick",kClick)
  end select

Case "AddElement"
  TextClear
  Output("Add another "+te+EOL+EOL,"Head")
  Output("  What is "+te+" "+str(ne+1)+": ")
  ScriptWait("AddElementIn",kText)
Case "AddElementIn"
  if Input="" then
```

```
    Output(EOL)
    ScriptFlow
  else
    hSet(ne,"E")
    hEmpty("X")
    hSet(Input,"Name","X")
    gSet("NewE","X")
    Output(EOL+EOL+"Give your "+te+", "+Input+", a rating on each "+tc)
    Output(" by entering a number or clicking to use a popup menu."+EOL+EOL)
    ScriptFlow("CRatingLoop")
  end if

Case "CRating" // Entry after an element match
  TextClear
  Output("Rate your "+te+" on the "+tcs+EOL+EOL,"Head")
  Output("Give your "+te+", "+gGet("E",hGet("E"))+", a rating on each "+tc)
  Output(" by entering a number or clicking to use a popup menu."+EOL+EOL)
  ScriptFlow("CRatingLoop")
Case "CRatingLoop"
  j=hGetI("E")
  vCountSet(0,"Sort") // set up a vector of unrated constructs
  for i=0 to nc-1
    v=gGet("V",str(i),str(j))
    if v<>"?" then Output("    "+gGet("C",str(i))+": "+v+EOL) else vPush(i,"Sort")
  next
  ScriptFlow("CRatingLoopOpen")
Case "CRatingLoopOpen"
  if vOK("Sort") then
    ScriptFlow("CRatingLoopOut")
  else
    Output(EOL)
    ScriptFlow("EditElement/DoEditElement")
  end if
Case "CRatingLoopOut"
  i=vPopI("Sort")
  hSet(i,"C")
  Output("    "+gGet("C",str(i))+": ")
  ScriptWait("CRatingLoopIn",kCMenu)
Case "CRatingLoopIn"
  Output(EOL)
  i=hGetI("C")
  j=hGetI("E")
  s=NthField(Input," ",1)
  gSet("V",str(i),str(j),s)
  if gGet("V",str(i),str(j))<>s then
    Call gGet("C",str(i),"X")
    s=hGet("Range","X")
    Alert ("Value not appropriate","Enter a value in the range "+sGet(s,1)+" to "+sGet(s,2))
    ScriptFlow("CRatingLoopOut")
  else
    ScriptFlow("CRatingLoopOpen")
  end if

Case "ChooseTriad"
  hRemove("E")
  ScriptFlow("ReListTriad")
Case "ReListTriad"
  TextClear
  Output("Choose a triad of "+tes+EOL+EOL,"Head")
  Output("Press the return key to have the program select a triad. ")
  Output("Or, you may select up to three "+tes+" yourself. If you select less than three the
      others will be selected at random."+EOL+EOL)
  s="Triad: "
  e=hGet("E")
  jj=sGetI(e)
  for j=1 to jj
    t=gGet("E",sGet(e,j))
    if j=1 then s=s+t else s=s+", "+t
  next
  Output(s+EOL+EOL)
  e=hGet("E")
  for j=0 to ne-1
    if sFind(e,str(j))=0 then
      Output("E"+str(j)+"\"+"    "+gGet("E",str(j))+EOL)
    end if
  next
  Output(EOL)
  Output("$\Click in your chosen "+te+" to select it, or here if you have finished selecting."+
      EOL,"CenterBold")
  ScriptWait("TriadClick",kClick)
Case "TriadClick"
  select case Left(incode,1)
  case "$"
    ScriptFlow
  case "E"
    e1=hGet("E")
    e2=Right(incode,len(incode)-1)
    j=sGetI(e1)
    if j=0 then hSet(e2,"E") else hSet(sMake(e1,e2),"E")
    if j>1 then
      ScriptFlow
    else
      ScriptFlow("ReListTriad")
    end if
  else
    Alert("Not editable","The item in which you clicked cannot be edited")
    ScriptWait("TriadClick",kClick)
  end select

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

10.4 Script: Elicit/EditElement.repscript

The **EditElement** script supports editing an element name and ratings.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

dim n,i,j As integer, s,v As string, ok As Boolean

Select case ScriptState

Case "DoEditElement"
```

```
  TextClear
  j=hGetI("E")
  s=gGet("E",str(j))
  Output("Edit "+te+" """+s+""""+EOL+EOL,"Head")
  OutPut("E\    Name of "+te+": "+s+EOL+EOL)
  for i=0 to nc-1
    Output("C"+str(i)+"\    "+gGet("C",str(i))+": "+gGet("V",str(i),hGet("E"))+EOL)
  next
  Output(EOL)
  if hGet("Initial")="" then
  Output("D\Click here to delete the "+te+EOL,"CenterBold")
  end if
  Output("$\Click on the "+te+" name or the "+tc+" to edit it, or here when you have finished
      editing"+EOL,"CenterBold")
  ScriptWait("EditClick",kClick)
Case "EditClick"
  s=Incode
  Select Case Left(s,1)
  case "D"
    ScriptFlow("EDelete")
  case "E"
    ScriptFlow("EOut")
  case "C"
    hSet(Right(Incode,len(Incode)-1),"C")
    ScriptFlow("CRatingOut")
  case "$"
    ScriptFlow // return as specified by caller
  else
    Alert("Not editable","The item in which you clicked cannot be edited")
    ScriptWait("EditClick",kClick)
  end select

Case "EDelete"
   j=hGetI("E")
   s=gGet("E",str(j))
   ok=Confirm("OK to delete "+s+"?","Deleting the "+te+", "+s+", is irreversible. Click on OK if
       you really want to delete it.")
   if ok then
     gSet("RemoveE",str(j))
     ScriptFlow
   else
     ScriptFlow("DoEditElement")
   end if

Case "EOut"
  TextClear
  s=gGet("E",hGet("E"))
  Output("Edit name of "+te+" """+s+""""+EOL+EOL,"Head")
  Output("    Name of "+te+": ")
  ScriptWait("EIn",kText)
  Output(s)
Case "EIn"
  if Input="" then
    ScriptFlow("DoEditElement")
  else
    Output(EOL)
    hSet(Input,"Name","X")
    gSet("E",hGet("E"),"X")
    ScriptFlow("DoEditElement")
end if

Case "CRatingOut"
  TextClear
  i=hGetI("C")
  j=hGetI("E")
  Output("Edit rating for "+te+" """+gGet("E",str(j))+""""+EOL+EOL,"Head")
  Output("    "+gGet("C",str(i))+": ")
  ScriptWait("CRatingIn",kCMenu)
  OutputSelect(gGet("V",str(i),str(j)))
Case "CRatingIn"
  Output(EOL)
  if Input="" then
    ScriptFlow("DoEditElement")
  else
    i=hGetI("C")
    j=hGetI("E")
    s=NthField(Input," ",1)
    gSet("V",str(i),str(j),s)
    Call gGet("C",str(i),"X")
    if gGet("V",str(i),str(j))<>s then
      s=hGet("Range","X")
      Alert ("Value not appropriate","Enter a value in the range "+sGet(s,1)+" to "+sGet(s,2))
      ScriptFlow("CRatingOut")
    else
      ScriptFlow("DoEditElement")
    end if
  end if

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

10.5 Script: Elicit/Constructs.repscript

The **Constructs** script lists and edits constructs.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

dim i,k,n As integer, s,c As string

Select case ScriptState

Case "List"
  TextClear
  Output("List of "+tcs+EOL+EOL,"Head")
  for i=0 to nc-1
    n=gGetI("OpenC",str(i))
    if n=0 then s="" else s=" ("+str(n)+" "+tes+" not rated)"
    Output("C"+str(i)+"\"+"    "+gGet("C",str(i))+s+EOL)
  next
  Output(EOL)
  if nc<gGetI("LimitC") then Output("A\Click here to add another "+tc+"."+EOL,"CenterBold")
  Output("$\Click in an item to edit it, or here if you have finished editing."+EOL,"CenterBold")
  ScriptWait("ListClick",kClick)
```

```
Case "ListClick"
  select case Left(incode,1)
  case "$"
    ScriptFlow
  case "A"
    hRemove("E") // not from a triad or pair so make sure no elements
    ScriptFlow("AddConstruct/AddConstruct","List")
  case "C"
    hSet(Right(incode,len(incode)-1),"C")
    ScriptFlow("EditConstruct/DoEditConstruct","List")
  else
    Alert("Not editable","The item in which you clicked cannot be edited")
    ScriptWait("ListClick",kClick)
  end select

Case "RateAnyOpen"
  if gGetI("Open")>0 then
    hSet("true","RateOpen") // stop option to delete
    ScriptFlow("RateOpen","RateAnyOpen")
  else
    hRemove("RateOpen")
    ScriptFlow
  end if

Case "RateOpen"
  for i=0 to nc-1
    k=gGetI("OpenC",str(i))
    if k>n then
      hSet(i,"C")
      n=k
    end if
  next
  if n>0 then
    ScriptFlow("AddConstruct/RateConstruct")
  else
    ScriptFlow
  end if

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

10.6 Script: Elicit/AddConstruct.repscript

The **AddConstruct** script supports adding constructs, directly, from triads and matches.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

function MakeTriad() As Boolean
```

```
  // enter with 0 or more prescribed elements in E
  // select other elements at random to make the number up to 3 and return in E
  // return false if not enough elements to make a triad
  dim e,s As string
  if ne<3 then return false
  do
    e=hGet("E")
    if sGetI(e)>=3 then return true
    s=str(GetRandom(0,ne-1))
    if sFind(e,s)=0 then
      if e="" then e=s else e=e+TAB+s
      hSet(e,"E")
    end if
  loop
end function

// main program
dim i,j,n As integer, lhp,rhp,v,s,e,b As string

Select case ScriptState

Case "RateConstruct"
  TextClear
  i=hGetI("C")
  Output("Rate "+tes+" on """+gGet("C")+""""+EOL,"Head")
  ScriptFlow("ERating")

Case "AddConstruct"
  TextClear
  Output("Add another "+tC+EOL+EOL,"Head")
  ScriptFlow("LHP")

Case "DoTriadChosen"
  ScriptFlow("Elements/ChooseTriad","DoTriad")

Case "DoTriad"
  TextClear
  Output("Elicit "+tC+" from a triad"+EOL+EOL,"Head")
  ScriptFlow("Triad")

Case "DoTriadNoClear"
  ScriptFlow("Triad")

Case "Triad"
  if MakeTriad then
    e=hGet("E")
    Output("Can you choose two of this triad of "+tes+" which are in some way alike and different
         from the other one?"+EOL+EOL)
    for i=1 to 3
      s=sGet(e,i)
      Output(s+"\"+gGet("E",s)+EOL,"CenterBoldBig")
    next
    Output(EOL)
    Output("$\Click in the "+te+" which is different, or here if you cannot do this."+EOL,"
        CenterBold")
    ScriptWait("TriadClicked",kClick)
  else
    Halt("Not enough elements for triad"+EOL)
  end if
Case "TriadClicked"
  If InCode="$" then
    Output(EOL)
    ScriptFlow
  else
    e=hGet("E")
    if sGet(e,2)=Incode then // make sure the one clicked is first
      hSet(sMake(sGet(e,2),sGet(e,1),sGet(e,3)),"E")
    elseif sGet(e,3)=Incode then
      hSet(sMake(sGet(e,3),sGet(e,1),sGet(e,2)),"E")
    end if
    TextClear
    Output("Name the poles of your "+tC+EOL+EOL,"Head")
    Output("Now I want you to think what you have in mind when you separate the pair from the
        other one. ")
    Output("Just type one or two words for each pole to remind you what you are thinking or
        feeling when you use this "+tC+".")
    Output(EOL+EOL)
    Output("X\"+"Or click here if you cannot do this"+EOL+EOL,"CenterBold")
    ScriptFlow("LHP")
  end if

Case "LHP"
  // enter with 0 to 3 element numbers stored in "E"
  e=hGet("E")
  s=sGet(e,2)
  if s<>"" then
    s=gGet("E",s)
    e=sGet(e,3)
    if e<>"" then s=s+", "+gGet("E",e)
    s=" {"+s+"}"
  end if
  Output("    Left pole rated "+gGet("MinR")+s+": ")
  ScriptWait("LHPIn",kText)
Case "LHPIn"
  Output(EOL)
  if InCode="X" or Input="" then
    Output(EOL)
    ScriptFlow // return, no construct added
  else
    hSet(Input,"L") // hold LHP in "L"
    ScriptFlow("RHP")
  end if
Case "RHP"
  s=sGet(hGet("E"),1)
  if s<>"" then s=" {"+gGet("E",s)+"}"
  Output("    Right pole rated "+gGet("MaxR")+s+": ")
  ScriptWait("RHPIn",kText)
Case "RHPIn"
  Output(EOL)
  if InCode="X" or Input="" then
    Output(EOL)
    ScriptFlow // return, no construct added
  else
    i=nc
    hSet(i,"C") // hold construct number in "C"
    hEmpty("X")
    hSet("R","Type","X")
    hSet(hGet("L"),"LHP","X")
    hSet(Input,"RHP","X")
    gSet("NewC","X")
    e=hGet("E")
    b="true"
    for n=1 to 3
      s=sGet(e,n)
      if s<>"" then gSet("EndV",str(i),s,b)
       b="false"
    next
    ScriptFlow("ERating")
  end if
```

```
Case "ERating"
  TextClear
  i=hGetI("C")
  Output("Rate the "+tes+" on your "+tC+" """+gGet("C",str(i))+""""+EOL+EOL,"Head")
  Output("According to how you feel about them, please assign to each of the following "+tes+" a
      rating from ")
  Call gGet("C",str(i),"X")
  s=hGet("Range","X")
  Output(sGet(s,1)+" ("+hGet("LHP","X")+") to "+sGet(s,2)+" ("+hGet("RHP","X")+")")
  Output(" by entering a number or clicking to use a popup menu.")
  Output(EOL+EOL)
  Call gGet("SortV",str(i),"Sort","false")
  ScriptFlow("ERatingLoop")
Case "ERatingLoop"
  if vOK("Sort") then
    j=vExtractI("Sort")
    hSet(j,"E")
    v=gGet("V",str(hGetI("C")),str(j))
    if v<>"?" then
      Output("    "+gGet("E",str(j))+" "+v+EOL) // show already set values
      ScriptFlow("ERatingLoop")
    else
      ScriptFlow("ERatingLoopOut")
    end if
  else
    Output(EOL)
    ScriptFlow("EditConstruct/DoEditConstruct") // offer option to edit
  end if
Case "ERatingLoopOut"
  Output("    "+gGet("E",hGet("E"))+": ")
  ScriptWait("ERatingLoopIn",kCMenu)
Case "ERatingLoopIn"
  Output(EOL)
  i=hGetI("C")
  j=hGetI("E")
  s=NthField(Input," ",1)
  gSet("V",str(i),str(j),s)
  if gGet("V",str(i),str(j))<>s then
    s=hGet("Range","X")
    Alert ("Value not appropriate","Enter a value in the range "+sGet(s,1)+" to "+sGet(s,2))
    ScriptFlow("ERatingLoopOut")
  else
    ScriptFlow("ERatingLoop")
  end if

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

10.7 Script: Elicit/Match.repscript

The **Match** script finds and displays element and construct matches, and gives the option to add constructs and elements to reduce the match.

```
dim ne,nc,status As integer, te,tes,ten,tc,tcs,tcn As string
ne=gGetI("ne")
nc=gGetI("nc")
status=gGetI("Status")
```

```
te=gGet("E")
tes=gGet("Es")
if ne=1 then ten=te else ten=tes
tc=gGet("C")
tcs=gGet("Cs")
if nc=1 then tcn=tc else tcn=tcs

dim i,j,n,i1,i2,j1,j2,v As integer, match As double, e,e1,e2,s As string, b As Boolean

Select case ScriptState

Case "DoMatch" // find any matches not already checked
  b=false
  if ne>3 and ne<gGetI("LimitE") then // test for C matches
    n=gGetI("MatchC","80","0","M","MV")
  //print str(n)+EOL
  //print vDump("M")+EOL
  //print vDump("MV")+EOL
  //print hDump("MatchC")+EOL
    for i=0 to n-1
      s=vGet(i,"M") // get match pair
      if not hCheck(s,"MatchC") then // new match
        i1=sGetI(s,1) // get first construct
        hSet(i1,"C1") // hold number in C1
        Call gGet("C",str(i1),"X1") // hold specification in X1
        i2=sGetI(s,2) // get second construct
        hSet(i2,"C2") // hold number in C2
        Call gGet("C",str(i2),"X2") // hold specification in X2
        v=vGetI(i,"MV") // get match value
        hSet(v,"M") // hold match value in M
        hSet(v,s,"MatchC") // record match pair in MatchC
        b=true // found a new construct match
        ScriptFlow("ShowCMatch","DoMatch")
       exit
      end if
    next
  end if
  if not b and nc>3 and nc<gGetI("LimitC") then // test for E matches
    n=gGetI("MatchE","80","0","M","MV")
    for i=0 to n-1
      s=vGet(i,"M") // get match pair
      if not hCheck(s,"MatchE") then // new match
        hSet(s,"E") // hold matching E numbers
        v=vGetI(i,"MV") // get match value
        hSet(v,"M") // hold match value in M
        hSet(v,s,"MatchE") // record match pair in MatchE
        b=true // found a new element match
        ScriptFlow("ShowEMatch","DoMatch")
       exit
      end if
    next
  end if
  if not b then ScriptFlow // no matches

Case "ShowCMatch"
  TextClear
  Output("Break "+tc+" match"+EOL+EOL,"Head")
  Output("The two "+tcs+" you called"+EOL)
  Output(hGet("Identifier","X1")+EOL+hGet("Identifier","X2")+EOL,"Center")
  Output("are matched at the "+hGet("M")+"% level.")
  Output(" This means that most of the time you are saying "+hGet("LHP","X1")+" you are also
      saying "+hGet("RHP","X2"))
  Output(", and most of the time you are saying "+hGet("RHP","X1")+" you are also saying "+hGet("
      LHP","X2")+"."+EOL+EOL)
  Output("Think of another "+te+" which is either:-"+EOL+EOL)
  Output("1\    "+hGet("LHP","X1")+" and "+hGet("RHP","X2")+EOL,"CenterBold")
  Output("or"+EOL,"Center")
  Output("2\    "+hGet("LHP","X2")+" and "+hGet("RHP","X1")+EOL+EOL,"CenterBold")
  Output("$\Click on the appropriate combination, or here if you cannot do this"+EOL,"CenterBold
      ")
  ScriptWait("BreakCMatch",kClick)
Case "BreakCMatch"
  Select Case Incode
  case "1"
    TextClear
    Output("Add a "+te+" to reduce a match between "+tcs+EOL+EOL,"Head")
    Output("X\"+"Or click here if you cannot do this"+EOL+EOL,"CenterBold")
    Output("  What is the "+te+" that is """+hGet("LHP","X1")+""" and """+hGet("RHP","X2")+""":
        ")
    ScriptWait("InElement",kText)
  case "2"
    TextClear
    Output("Add a "+te+" to reduce a match between "+tcs+EOL+EOL,"Head")
    Output("X\"+"Or click here if you cannot do this"+EOL+EOL,"CenterBold")
    Output("  What is the "+te+" that is """+hGet("RHP","X1")+""" and """+hGet("LHP","X2")+""":
        ")
    ScriptWait("InElement",kText)
  case "$"
    ScriptFlow // return as specified by caller
  else
    Alert("Not understood","The item in which you clicked is not an option")
    ScriptWait("BreakCMatch",kClick)
  end select
Case "InElement"
  if InCode="X" or Input="" then
    Output(EOL)
    ScriptFlow
  else
    j=ne
    hSet(j,"E")
    hEmpty("X")
    hSet(Input,"Name","X")
    gSet("NewE","X")
    b=InCode="1"
    gSet("EndV",hGet("C1"),str(j),BooStr(not b))
    gSet("EndV",hGet("C2"),str(j),BooStr(b))
    ScriptFlow("Elements/CRating")
  end if

Case "ShowEMatch"
  s=hGet("E")
  e2=gGet("E",sGet(s,1))
  e1=gGet("E",sGet(s,2))
  TextClear
  Output("Break "+te+" match"+EOL+EOL,"Head")
  Output("The two "+tes+" "+e1+" and "+e2+" are matched at the "+hGet("M")+"% level. ")
  Output("This means that so far you have not distinguished between them."+EOL+EOL)
  Output("Think of another "+tc+" which separates "+e1+" from "+e2+"."+EOL+EOL)
  Output("X\"+"Or click here if you cannot do this"+EOL+EOL,"CenterBold")
  ScriptFlow("AddConstruct/LHP")

Case "Compile"
  ScriptFlow

else
  Halt("Flow error--no label "+ScriptState+EOL)

end select
```

11 Bibliography

Adler, J. E. and Rips, L. J. (2008). *Reasoning: Studies of Human Inference and Its Foundations*. Cambridge University Press, Cambridge.

Algom, D. and Fitousi, D. (2016). Half a century of research on garner interference and the separability–integrality distinction. *Psychological Bulletin*, 142(12):1352–1383.

Attneave, F. (1950). Dimensions of similarity. *American Journal of Psychology*, 43(4):516–556.

Balme, D. M. (1987). Aristotle's use of division and differentiae. In Gotthelf, A. and Lennox, J. G., editors, *Philosophical issues in Aristotle's biology*. Cambridge University Press, Cambridge.

Bellman, L. (2012). *Auktoriserade fastighetsvärderares syn på värdering: tankemönster om kommersiella fastigheter*. Licentiate thesis.

Boose, J. H. and Gaines, B. R. (1988). *Knowledge Acquisition Tools for Expert Systems*. Academic Press, London.

Buchanan, B. G. and Shortliffe, E. H. (1984). *Rule-based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.

Caputi, P. (2011). *Personal Construct Methodology*. Wiley, Chichester, UK.

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

Corbridge, C., Rugg, G., Major, N. P., Shadbolt, N. R., and Burton, A. M. (1994). Laddering: technique and tool use in knowledge acquisition. *Knowledge Acquisition*, 6(3):315–341.

Davies, S. (1991). *Definitions of Art*. Cornell University Press, Ithaca, NY.

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap Methods and Their Application*. Cambridge University Press, Cambridge.

Dey, S. and Lee, S.-W. (2017). Reassure: Requirements elicitation for adaptive socio-technical systems using repertory grid. *Information and Software Technology*, 87:160–179.

Efron, B. and Tibshirani, R. (1993). *An introduction to the bootstrap*. Chapman & Hall, New York.

Emmons, O. (1939). *Linearity In Factor Analysis*. PhD thesis.

Evans, J. S. B. T., Newstead, S. E., and Byrne, R. M. J. (1993). *Human Reasoning: The Psychology of Deduction*. Lawrence Erlbaum, Hove.

Feixas, G. and Cornejo, J. M. (1996). *Manual de la técnica de rejilla mediante el programa RECORD ver. 2.0*. Paidós, Barcelona.

Fortune, S. (1987). A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2:153–174.

Fransella, F., Bell, R. C., and Bannister, D. (2004). *A Manual for Repertory Grid Technique*. Wiley, Chichester, UK.

Fromm, M. (2004). *The Repertory Grid Interview*. Waxman, Munster.

Frontier, S. (1976). Étude de la décroissance des valeurs propres dans une analyse en composantes principales: Comparaison avec le moddèle du bâton brisé. *Journal of Experimental Marine Biology and Ecology*, 25(1):67–75.

Gaines, B. R. (1995). Porting interactive applications to the web. In *4th International World Wide Web Conference Tutorial Notes*, pages 199–217. O'Reilly, Sebastopol, CA.

Gaines, B. R. (2015). Universal logic as a science of patterns. In Koslow, A. and Buchsbaum, A., editors, *The Road to Universal Logic: Festschrift for 50th Birthday of Jean-Yves Béziau*, pages 145–189. Birkhäuser, Basel.

Gaines, B. R., Chen, L. L.-J., and Shaw, M. L. G. (1997). Modeling the human factors of scholarly communities supported through the Internet and World Wide Web. *Journal American Society Information Science*, 48(11):987–1003.

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. (1996). Webgrid: Knowledge modeling and inference through the World Wide Web. In Gaines, B. and Musen, M., editors, *Proceedings of Tenth Knowledge Acquisition Workshop*, pages 65–1–65–14.

Gaines, B. R. and Shaw, M. L. G. (1997). Knowledge acquisition, modeling and inference through the World Wide Web. *International Journal of Human-Computer Studies*, 46(6):729–759.

Gaines, B. R. and Shaw, M. L. G. (1998). Developing for web integration in Sisyphus-IV: WebGrid-II experience. In Gaines, B. and Musen, M., editors, *Proceedings of Eleventh Knowledge Acquisition Workshop*.

Gaines, B. R. and Shaw, M. L. G. (2012). Computer aided constructivism. In Caputi, P., Viney, L. L., Walker, B. M., and Crittenden, N., editors, *Constructivist Methods*, pages 183–222. Wiley, New York.

Gärdenfors, P. (2000). *Conceptual Spaces: The Geometry of Thought*. MIT Press, Cambridge, MA.

Gaut, B. (2000). "Art" as a cluster concept. In Carroll, N., editor, *Theories of Art Today*, pages 25–44. University of Wisconsin Press, Madison, WI.

Gill, M. L. (2010). Division and definition in plato's sophist and statesman. In Charles, D., editor, *Definition in Greek philosophy*, pages 172–199. Oxford University Press, Oxford.

Gower, J., Lubbe, S., and Le Roux, N. (2011). *Understanding Biplots*. John Wiley, Chichester, UK.

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–338.

Gower, J. C. and Hand, D. J. (1995). *Biplots*. Chapman & Hall, London.

Greenacre, M. J. (2007). *Correspondence Analysis in Practice*. Chapman & Hall, London.

Hardman, D. and Macchi, L. (2003). *Thinking: Psychological Perspectives on Reasoning, Judgment, and Decision Making*. Wiley, Hoboken, NJ.

Hayes-Roth, F., Waterman, D. A., and Lenat, D. B. (1983). *Building Expert Systems*. Addison-Wesley, Reading, Massachusetts.

Heckmann, M. and Bell, R. C. (2016). A new development to aid interpretation of hierarchical cluster analysis of repertory grid data,. *Journal of Constructivist Psychology*, 29(4):368–381.

Hennig, C. M., Meilă, M., Murtagh, F., and Rocci, R. (2016). *Handbook of cluster analysis*. Chapman & Hall/CRC, Boca Raton.

Honey, P. (1979). The repertory grid in action: how to use it to conduct an attitude survey. *Industrial and Commercial Training*, 11(11):452–459.

Jackson, D. A. (1993). Stopping rules in principal components analysis: A comparison of heuristical and statistical approaches. *Ecology*, 74(8):2204–2214.

James, W. (1890). *The Principles of Psychology*. Macmillan, London.

Jankowicz, D. (2004). *The Easy Guide to Repertory Grids*. Wiley, Chichester, UK.

Kaipainen, M., Zenker, F., Hautamäki, A., and Gärdenfors, P. (2019). *Conceptual Spaces: Elaborations and Applications*. Synthese Library 405. Springer,, Cham.

Kaldis, B. (2008). The question of platonic division and modern epistemology. In Dillon, J. M., Zovko, M.-E., and Doner, J. F., editors, *Platonism and Forms of Intelligence*. Akademie, Berlin.

Kelly, G. A. (1938). The assumption of an originally homogeneous universe and some of its statistical implications. *Journal of Psychology*, 5:201–208.

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

Kelly, G. A. (1969). A mathematical approach to psychology. In Maher, B., editor, *Clinical Psychology and Personality: The Selected Papers of George Kelly*, pages 94–113. Wiley, New York.

Kirkcaldy, B., Pope, M., and Siefen, G. (1993). Sociogrid analysis of a child and adolescent psychiatric clinic. *Social Psychiatry and Psychiatric Epidemiology*, 28(6):296–303.

Kolodner, J. L. (1993). *Case-based Reasoning*. Morgan Kaufmann, San Mateo, CA.

Korenini, B. (2014). Consistent laddering: A new approach to laddering technique. *Journal of Constructivist Psychology*, 27(4):317–328.

Korzybski, A. (1951). The role of language in the perceptual processes. In Blake, R. R. and Ramsey, G. V., editors, *Perception, an Approach to Personality*, pages 170–205. Ronald Press, New York.

Leach, C., Freshwater, K., Aldridge, J., and Sunderland, J. (2001). Analysis of repertory grids in clinical practice. *British Journal of Clinical Psychology*, 40:225–248.

Mireaux, M., Cox, D. N., Cotton, A., and Evans, G. (2007). An adaptation of repertory grid methodology to evaluate australian consumers' perceptions of food products produced by novel technologies. *Food Quality and Preference*, 18(6):834–848.

Nosofsky, R. (1985). Overall similarity and the identification of separable-dimension stimuli: A choice model analysis. *Perception & Psychophysics*, 38(5):415–432.

Okabe, A., Boots, B. N., and Sugihara, K. k. (1992). *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley, NY.

Peres-Neto, P. R., Jackson, D. A., and Somers, K. M. (2005). How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, 49(4):974–997.

Pope, M. L. and Denicolo, P. M. (2001). *Transformative Professional Practice: Personal Construct Approaches to Education and Research*. Whurr, London.

Rad, A., Wahlberg, O., and Öhman, P. (2013). How lending officers construe assessments of small and medium-sized enterprise loan applications: a repertory grid study. *Journal of Constructivist Psychology*, 26(4):262–279.

Récanati, F. (2013). *Mental Files*. Oxford University Press, Oxford.

Reynolds, T. J. and Gutman, J. (1988). Laddering theory, method, analysis, and interpretation. *Journal of Advertising Research*, 28(1):11–31.

Rosch, E. (1978). Principles of categorization. In Rosch, E. and Lloyd, B. B., editors, *Cognition and Categorization*, pages 27–48. Lawrence Erlbaum, Hillsdale, NY.

Rosch, E. and Lloyd, B. B. (1978). *Cognition and Categorization*. Lawrence Erlbaum, Hillsdale, NY.

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

Shaw, M. L. G. (1981). *Recent Advances in Personal Construct Technology*. Academic Press, London.

Shaw, M. L. G. and Gaines, B. (1992). Kelly's 'Geometry of psychological space' and its significance for psychological modeling. *New Psychologist*, pages 23–31.

Shaw, M. L. G. and Gaines, B. R. (1989). Comparing conceptual structures: consensus, conflict, correspondence and contrast. *Knowledge Acquisition*, 1(4):341–363.

Shaw, M. L. G. and Gaines, B. R. (1996a). Requirements acquisition. *Software Engineering Journal*, 11(3):149–165.

Shaw, M. L. G. and Gaines, B. R. (1996b). Webgrid: Knowledge elicitation and modeling on the web. In Maurer, H., editor, *Proceedings of WebNet96*, pages 425–432. Association for the Advancement of Computing in Education, Charlottesville, VA.

Shaw, M. L. G. and Gaines, B. R. (1998). Webgrid II: Developing hierarchical knowledge structures from flat grids. In Gaines, B. and Musen, M., editors, *Proceedings of Eleventh Knowledge Acquisition Workshop*.

Shaw, M. L. G. and Gaines, B. R. (2005). Expertise and expert systems: emulating psychological processes. In Fransella, F., editor, *The Essential Practitioner's Handbook of Personal Construct Psychology*, pages 87–94. Wiley, Chichester, UK.

Shaw, M. L. G. and McKnight, C. (1981). *Think Again: Personal Problem-solving and Decision-making*. Prentice-Hall, Englewood Cliffs, NJ.

Shepard, R. N. (1964). Attention and the metric structure of the stimulus space. *Journal of Mathematical Psychology*, 1(1):54–87.

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

Slater, P. (1977). *Dimensions of Intrapersonal Space: Volume 2*. John Wiley, London.

Spencer, H. (1862). *First Principles*. Williams and Norgate, London.

Stenning, K. and Lambalgen, M. v. (2008). *Human Reasoning and Cognitive Science*. MIT Press, Cambridge, MA.

Stephenson, W. (1953). *The Study of Behavior: Q-technique and its Methodology*. University of Chicago Press, IL.

Tan, F. B., Tung, L.-L., and Xu, Y. (2009). A study of web-designers' criteria for effective business-to-consumer (b2c) websites using the repertory grid technique. *Journal of Electronic Commerce Research*, 10(3):155–177.

Thurstone, L. L. (1935). *The Vectors of Mind: Multiple-factor Analysis for the Isolation of Primary Traits*. University of Chicago science series. University of Chicago Press, Chicago.

Tofan, D., Avgeriou, P., and Galster, M. (2014). Validating and improving a knowledge acquisition approach for architectural decisions. *International Journal of Software Engineering and Knowledge Engineering*, 24(4):553–589.

Torgerson, W. S. (1958). *Theory and Methods of Scaling*. Wiley, New York.

Tversky, A. (1977). Features of similarity. *Psychological Review*, 84(4):327–352.

Wason, P. C. (1968). Reasoning about a rule. *Quarterly Journal of Experimental Psychology*, 20(3):273–281.

Whitehead, A. N. (1929). *Process and Reality: An Essay in Cosmology*. Free Press, New York.

Yorke, D. M. (1978). Repertory grids in educational research: Some methodological considerations. *British Educational Research Journal*, 4(2):63–74.

Yorke, D. M. (1983). Straight or bent? an inquiry into rating scales in repertory grids. *British Educational Research Journal*, 9(2):141–151.

Young, S. M., Edwards, H. M., McDonald, S., and Thompson, J. B. (2005). Personality characteristics in an xp team: a repertory grid study. *SIGSOFT Software Engineering Notes*, 30(4):1–7.

Zenker, F. and Gärdenfors, P. (2015). *Applications of Conceptual Spaces: The Case for Geometric Knowledge Representation*. Springer, Cham.

Most of our publications cited in the references are freely available on the web