Visualisation using Empirical Modelling principles and tools

Meurig Beynon

Computer Science, The University of Warwick, Coventry CV4 7AL

Abstract

This paper surveys the way in which Empirical Modelling (EM) principles and tools have been deployed for a variety of different kinds of visualisation. The illustrative examples referred to in the paper are displayed in Figure 1. They relate to three principal themes: visualising affect, visualisation in support of cognition, and mathematical visualisation. These are representative of various different ways in which EM can be exploited, in each of which visualisation plays a central role.

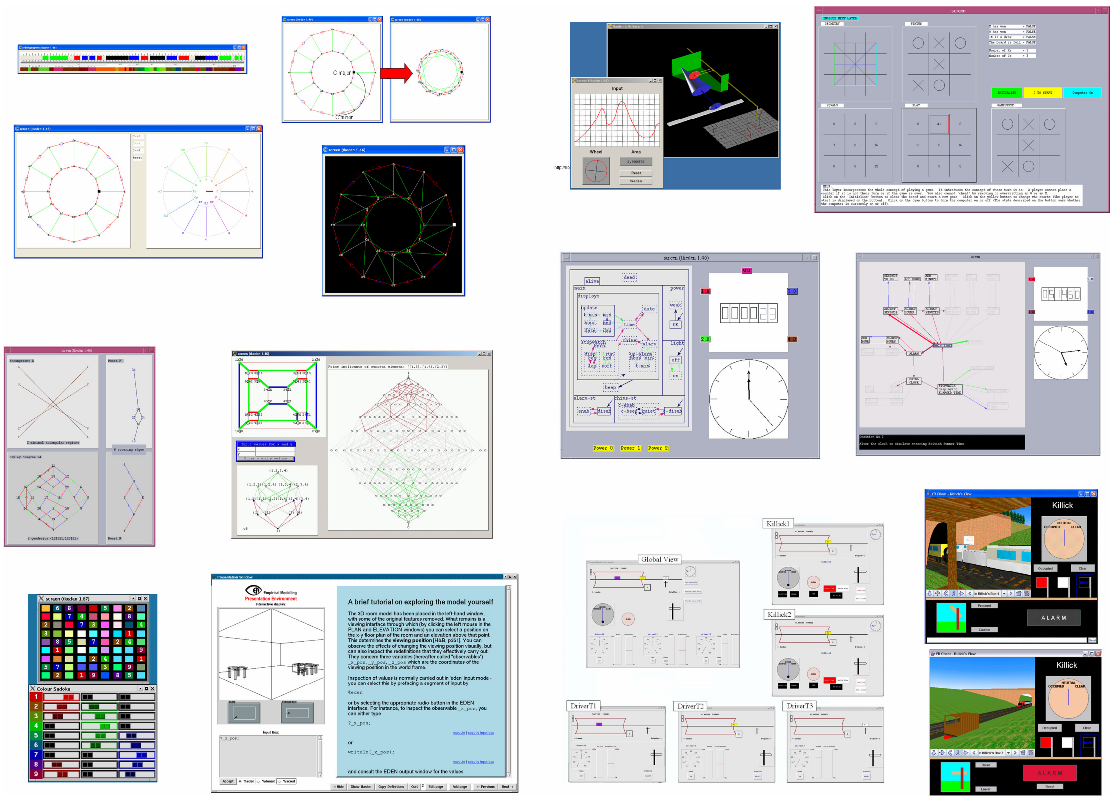
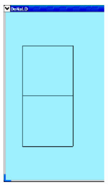
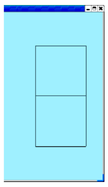


Figure 1: Examples of visualisations generated using EM principles and tools

1. Introduction

Empirical Modelling (EM) has been developed as a new conceptual framework for computing (see the EM website [22]). This conceptual framework is motivated by a shift in emphasis in computer use that has taken place over the fifty years since the first high-level programming languages were developed. Whilst the significant uses of the computer as a classical computational device continue to flourish and expand, the use of computer-related technology as a means of generating experiences, in which computer-supported visualisation has a central role, points to potential of a quite different kind. And though visualisation in the scientific domain may relate to formal representations and processes that can be regarded as well-suited to abstract computational interpretation, other forms of computer-assisted visualisation – such as are prominent in recreational applications of computing in digital photography and video, and in the emerging field of humanities computing – demand altogether different semantic priorities. Whereas the relationship between interaction with the computer and human interpretation in traditional calculation is preconceived and precise, meanings can be more subjective and emergent in modern interactions with computers, as in the engagement between artists and their media and instruments, and between experimental scientists and unfamiliar phenomena [13].





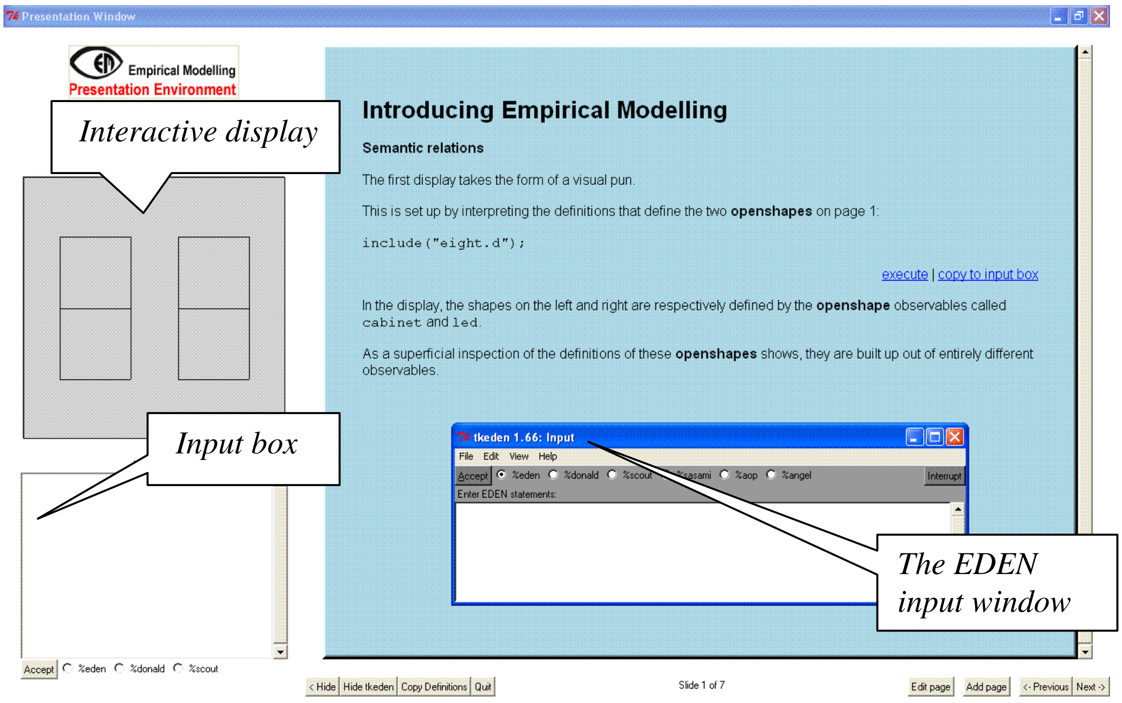
```
%donald
## DoNaLD: a 'definitive notation'
## that supports 2D line-drawing

openshape cabinet
within cabinet {
  int width, length
  # the size of the filing cabinet
  point NW, NE, SW, SE
  # the 4 corners of the cabinet
  line N, S, E, W
  # the 4 edges of the cabinet

  N = [NW, NE]
  S = [SW, SE]
  W = [NW, SW]
  E = [NE, SE]

  width, length = 300, 300

  SW = {600, 200}
  SE = SW + {width, 0}
  # the other 3 corners are ...
  NW = SW + {0, length}
  # ... relative to SW corner
  NE = NW + {width, 0}

  ##### cabinet/drawer #####
  openshape drawer
  # the filing cabinet has a drawer
  within drawer {
   real open
   # 1 = open; 0 = close
   real length
   # the length of the drawer
   line N, S, W, E
   # the 4 edges of the drawer

   length = ~/length * open
   open = 1.0
   # the drawer is open initially
   N = [~/NW + {0, length},
        ~/NE + {0, length}]
   S = [~/NW, ~/NE]
   W = [~/NW + {0, length}, ~/NW]
   E = [~/NE + {0, length}, ~/NE]
  }
}
```

```
openshape led
within led {
  int digit
  point p1, p2, p3, p4, p5, p6
  # 6 points
  line l1, l2, l3, l4, l5, l6, l7
  # 7 segments
  boolean
  on1, on2, on3, on4, on5, on6, on7
  # status of the 7 segments

  digit = 8
  # initially display all segments

  p1 = {100, 800}
  p2 = {100, 500}
  p3 = {100, 200}
  p4 = {400, 800}
  p5 = {400, 500}
  p6 = {400, 200}

  on1 = digit != 1 && digit != 4
  on2 = digit != 0 && digit != 1
                 && digit != 7
  on3 = digit != 1 && digit != 4
                 && digit != 7
  on4 = (digit == 0 || digit >= 4)
                 && digit != 7
  on5 = digit == 0 || digit == 2
       || digit == 6 || digit == 8
  on6 = digit != 5 && digit != 6
  on7 = digit != 2

  l1 = if on1 then [p1, p4]
                 else [p1, p1]
  l2 = if on2 then [p2, p5]
                 else [p2, p2]
  l3 = if on3 then [p3, p6]
                 else [p3, p3]
  l4 = if on4 then [p1, p2]
                 else [p1, p1]
  l5 = if on5 then [p2, p3]
                 else [p2, p2]
  l6 = if on6 then [p4, p5]
                 else [p4, p4]
  l7 = if on7 then [p5, p6]
                 else [p5, p5]
}
```

Listing 1: Families of definitions for a filing cabinet floorplan and an LED digit

In EM, files of definitions serve to specify the current state of an artefact. These definitions resemble the definitions of cells in a spreadsheet, in that each cell is associated with some external observable (such as a profit or cost), and the definitions express dependencies whereby changing the values of one observable directly affects the values of all other dependent observables. In the principal tool that we have developed for EM – the EDEN interpreter – definitions can express dependencies between observables of many different kinds. For instance, Listing 1 displays the definitions that are used to specify two line drawings that appear to be identical (as shown in the image that appears above the listing and in the Interactive display window in Figure 2), but that admit two quite different interpretations [5,23*k*].

The essential idea behind EM is that the semantics of an artefact can be expressed informally and implicitly with reference to the family of meaningful interactions by way of redefinitions of observables that it admits. The EM Presentation Environment depicted in Figure 1 is an example of such an artefact. It is constructed by entering definitions into the EDEN input window, making use of various 'definitive (definition-based) notations', associated with the radio buttons in the input window, of which %donald – whose use is illustrated in Listing 1 – is one.



Introducing Empirical Modelling

Semantic relations

The first display takes the form of a visual pun.

This is set up by interpreting the definitions that define the two **openshapes** on page 1:

```
include("eight.d");
```

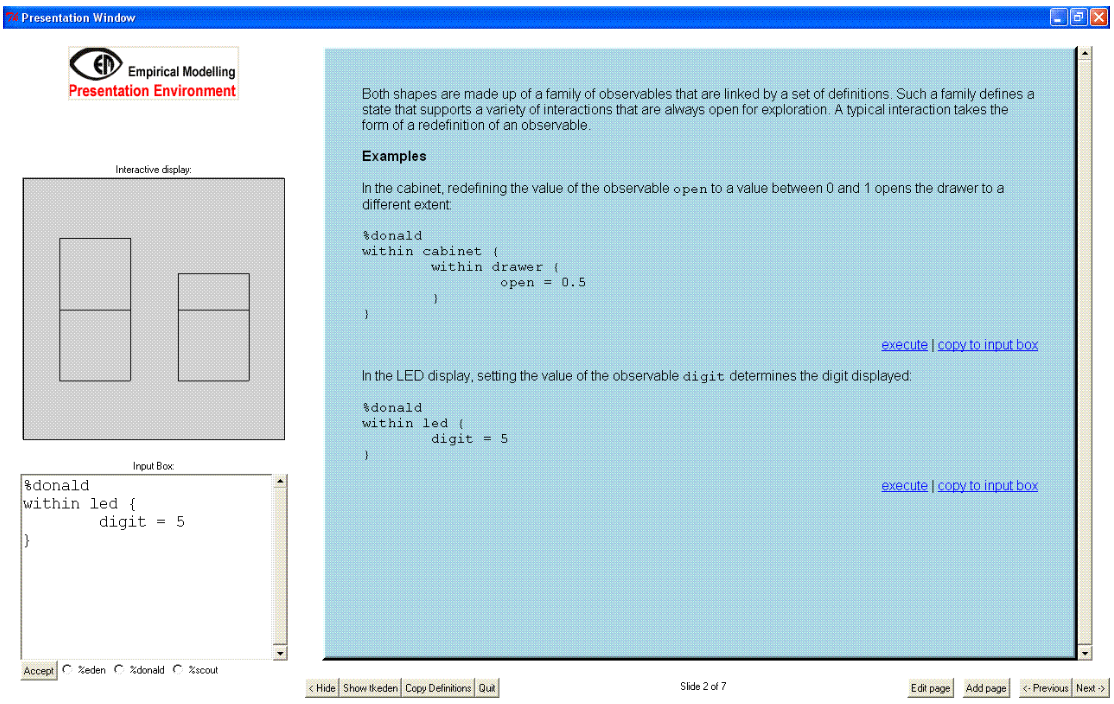
execute | copy to input box

In the display, the shapes on the left and right are respectively defined by the **openshape** observables called `cabinet` and `led`.

As a superficial inspection of the definitions of these **openshapes** shows, they are built up out of entirely different observables.

Figure 2: Interpretations of simple line drawings in the EM presentation environment

Figures 3 and 4 (from [23*l*]) show the EM presentation environment in use. The EDEN input window has been hidden using the Hide tkeden button, and only a limited part of its functionality is available via the Input box. The EDEN redefinitions embedded in the presentation pages serve to highlight how simple interactions can draw out the two different interpretations for the line drawing in Figure 2. In Figure 3, the observable open which determines the extent to which the filing cabinet drawer is open, has been set to the value 0.5 by selecting the first of the execute options on the presentation page. The redefinition of the observable digit has been also copied to the Input box ready for execution. In Figure 4, the observable digit has been assigned to 5, and the width of the cabinet redefined to 250, rather than 300 (cf. Listing 1). These simple redefinitions serve to distinguish the two possible interpretations of the visually identical line drawings in Figure 2. The distinctive feature of such visualisation is the representation of latent characteristic transformations that are perceived as atomic.


Presentation Window
Empirical Modelling Presentation Environment
Interactive display:
Input Box:
Accept ○ %eden ○ %donald ○ %scout
< Hide | Show tkeden | Copy Definitions | Quit
Slide 2 of 7
Edit page | Add page | <- Previous | Next ->

```
%donald
within led {
        digit = 5
}
```

Both shapes are made up of a family of observables that are linked by a set of definitions. Such a family defines a state that supports a variety of interactions that are always open for exploration. A typical interaction takes the form of a redefinition of an observable.

Examples

In the cabinet, redefining the value of the observable `open` to a value between 0 and 1 opens the drawer to a different extent:

```
%donald
within cabinet {
        within drawer {
                open = 0.5
        }
}
```

execute | copy to input box

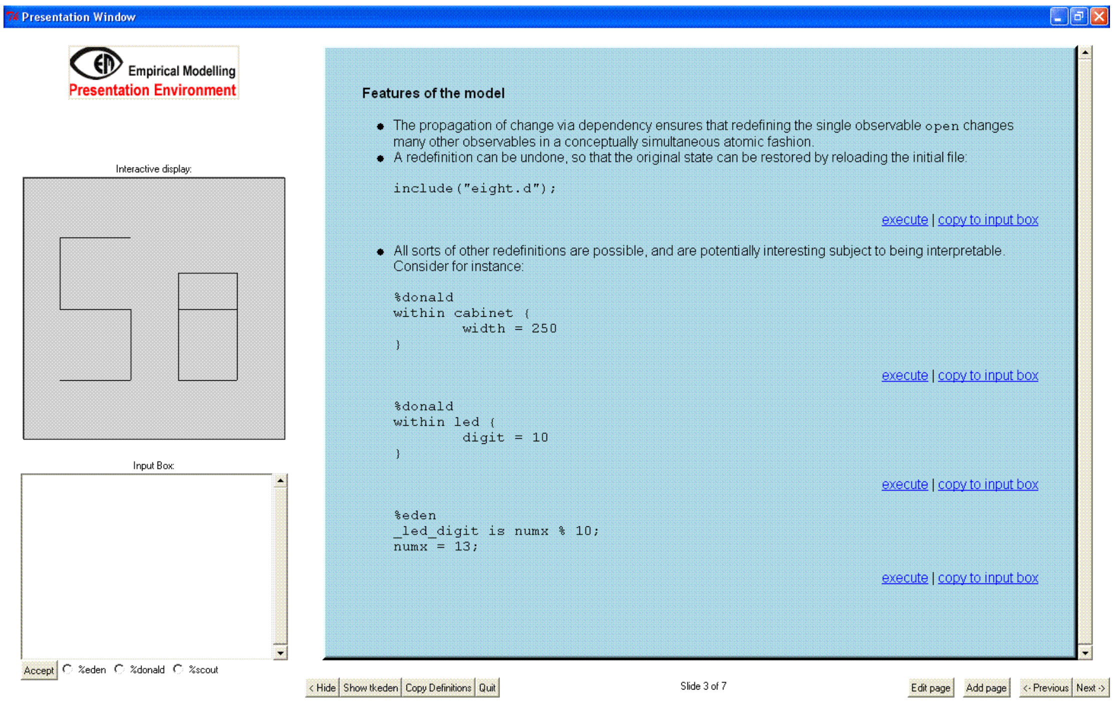
In the LED display, setting the value of the observable `digit` determines the digit displayed:

```
%donald
within led {
        digit = 5
}
```

execute | copy to input box

Figure 3: Simple examples of redefinitions that disclose the semantics of drawings

The most significant aspect of the EDEN environment is the open-ended scope for redefinition. In this respect, an EM artefact is quite unlike a program, where there is a preconceived repertoire of meaningful interactions and interactions outside that repertoire are prohibited by design. In EM, open-ended and exploratory interaction is actively encouraged, and it is appropriate to attempt to do things that probe the boundary of what is meaningful. Other definitions in Figure 4 illustrate this: digit is first set to 10, and the LED is then adapted to display the value of digit modulo 10.


Presentation Window
Empirical Modelling Presentation Environment
Interactive display:
Input Box:
Accept ○ %eden ○ %donald ○ %scout
< Hide | Show tkeden | Copy Definitions | Quit
Slide 3 of 7
Edit page | Add page | <- Previous | Next ->

Features of the model

- The propagation of change via dependency ensures that redefining the single observable `open` changes many other observables in a conceptually simultaneous atomic fashion.
- A redefinition can be undone, so that the original state can be restored by reloading the initial file:

```
include("eight.d");
```

execute | copy to input box

- All sorts of other redefinitions are possible, and are potentially interesting subject to being interpretable. Consider for instance:

```
%donald
within cabinet {
        width = 250
}
```

execute | copy to input box

```
%donald
within led {
        digit = 10
}
```

execute | copy to input box

```
%eden
_led_digit is numx % 10;
numx = 13;
```

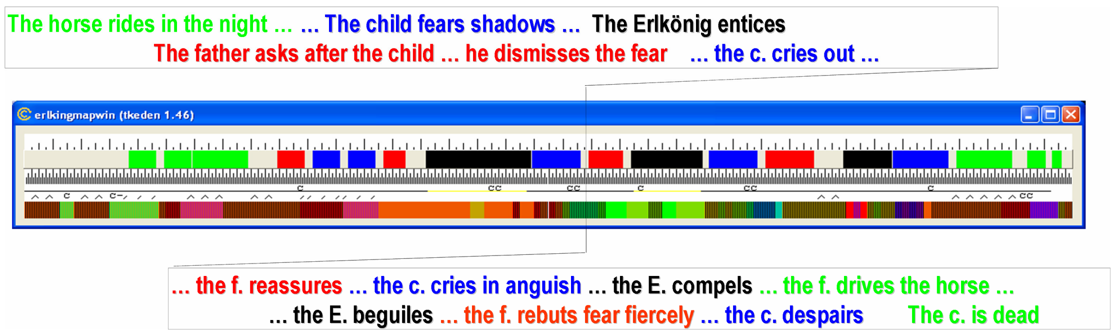
execute | copy to input box

Figure 4: Redefinitions that probe and extend the semantics of the line drawings

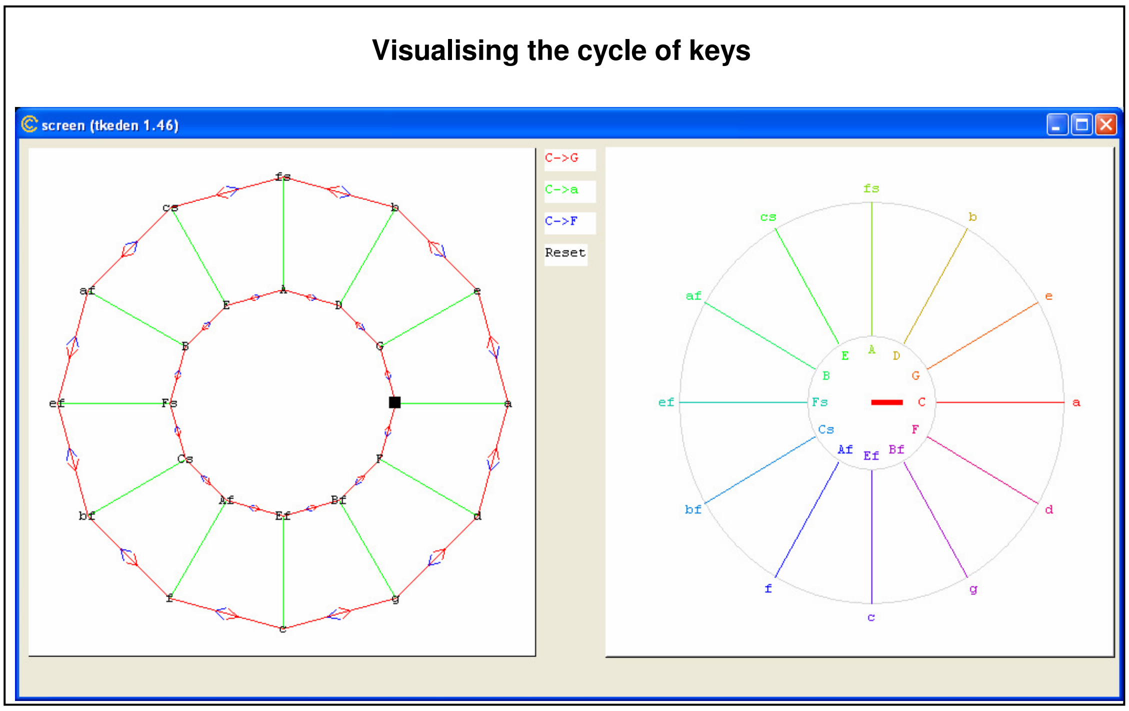
2. Illustrative examples of visualisation using EM principles and tools

A brief review of the visualisations gathered together in Figure 1 highlights a variety of characteristics of EM and its potential applications. This review proceeds from the top left hand corner of Figure 1 clockwise. In broad terms, the visualisations displayed address subjects that are progressively more amenable to formal objective treatment. All may be viewed as concerned with a common theme of 'sense-making'.

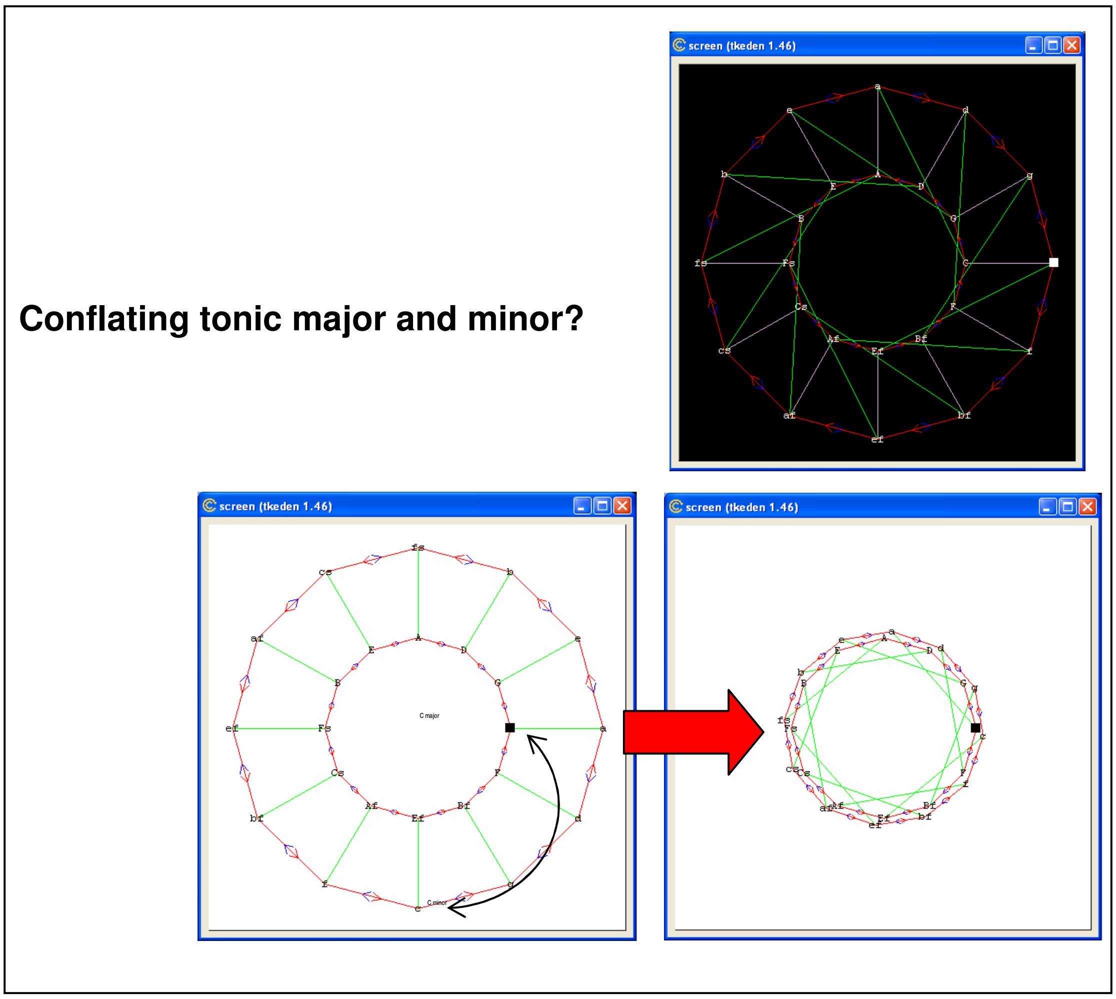
The four images at the top left are extracted from a visualisation developed in an attempt to express something of the author's subjective response to Schubert's famous setting of Goethe's ballad Erlkönig [16,17]. The first of these visualisations (here annotated with text) can be interpreted as a map of the song, made up of different layers. These layers lay out the metrical structure of the song, the roles of the different characters in the ballad that are to be played by the singer, the distinctive musical elements that feature in the accompaniment and the underlying harmonic texture.


The horse rides in the night ... ... The child fears shadows ... The Erlkönig entices
The father asks after the child ... he dismisses the fear ... the c. cries out ...
erlkingmapwin (tkeden 1.46)
... the f. reassures ... the c. cries in anguish ... the E. compels ... the f. drives the horse ...
... the E. beguiles ... the f. rebuts fear fiercely ... the c. despairs The c. is dead

The colours that are used to depict the harmony in the colour ribbon at the bottom of the above map are modelled by a visualisation of the cycle of keys, linked by dependency to a colourwheel. This visualisation applies to any classical tonal composition, and can also be exercised independently of the specific Erlkönig model.

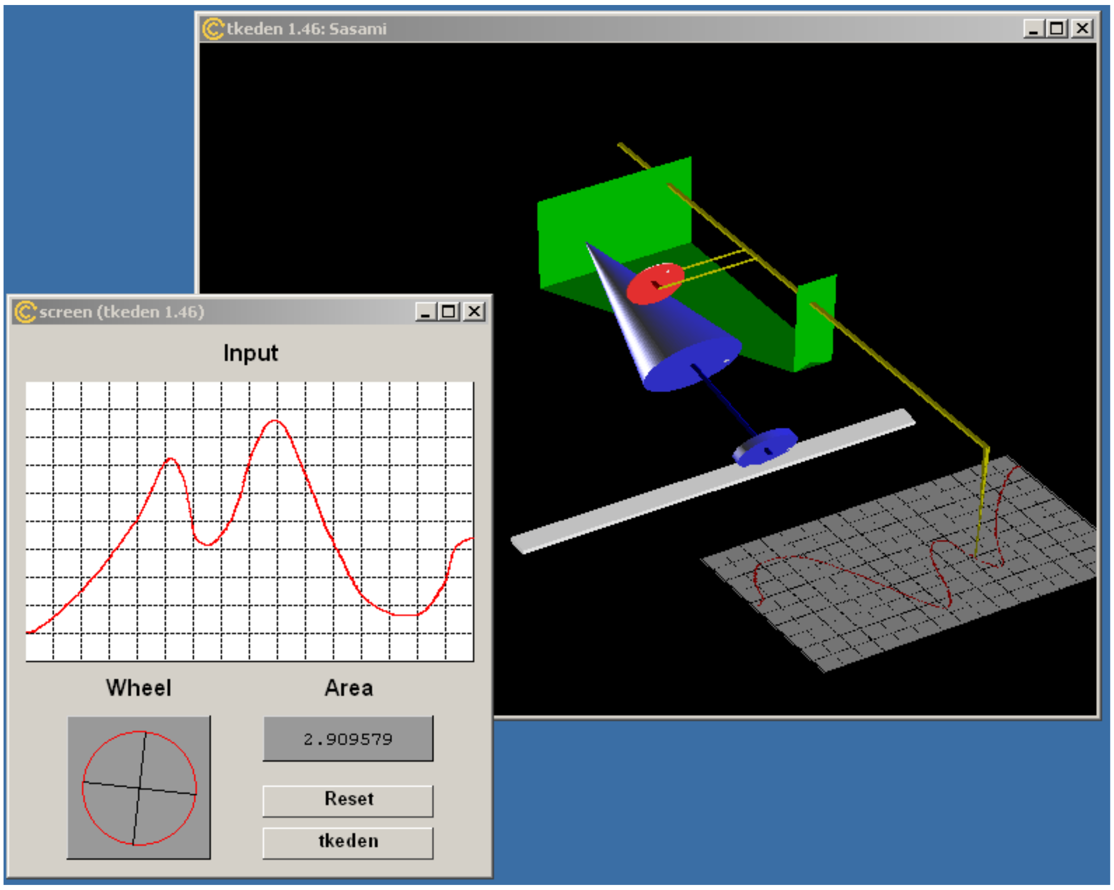


A distinctive feature of Schubert's musical idiom is a predisposition to using tonic major and minor tonalities in close juxtaposition. This characteristic is represented in Erlkönig in a highly dramatic way, whereby harmonic analysis indicates a conflation of tonic major-minor in certain passages. This type of treatment of tonality cannot be satisfactorily represented by conventional use of the cycle of keys. This motivated experimentation with two kinds of visualisation. One of these involved modifying the cycle of keys so as to reflect a much more direct connection between tonic major and minor, the other (adopted in the final model) a physical distortion of the cycle of keys that is deemed to take effect dynamically as the harmonic ambiguity arises in the song.



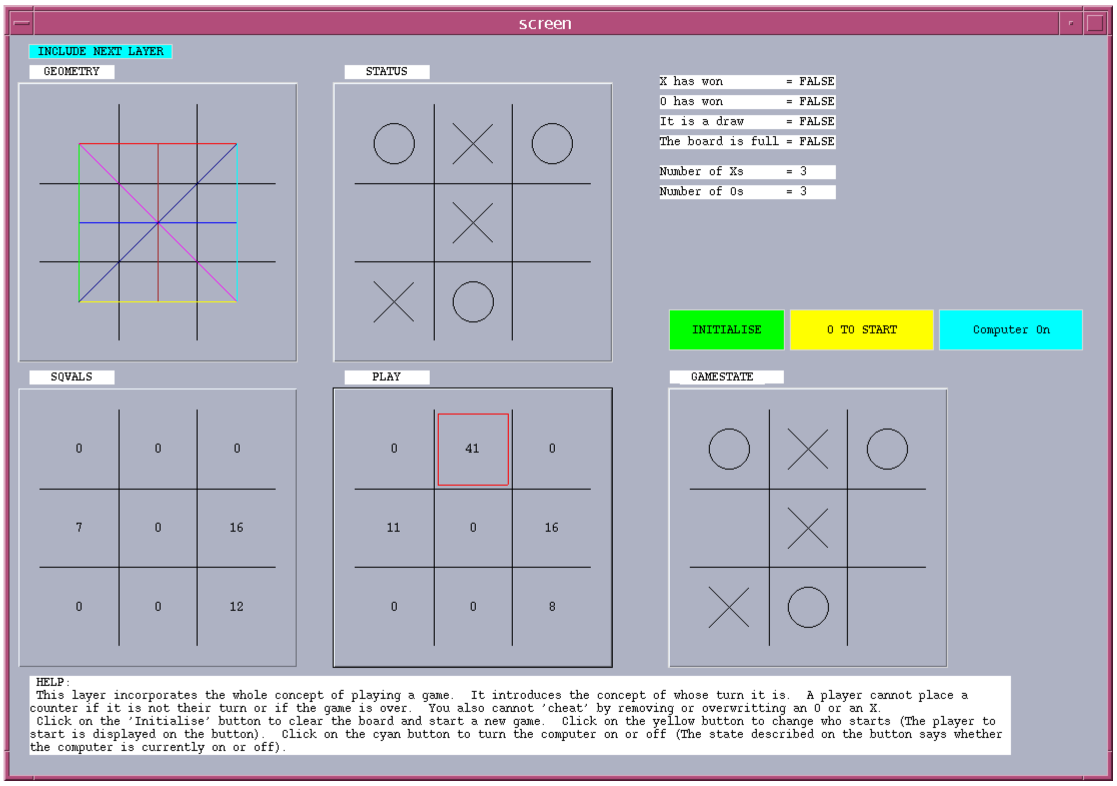
The visualisations developed in conjunction with this study of Erlkönig illustrate several features of EM application. The use of families of definitions is well-suited to supporting the intimate association between human activity, as in the crafting of the map and the experimentation in the representation of major-minor conflation, and computer activity, as in the automatic maintenance of relationships between internal values and their visual representations. Other significant features are: the use of a special-purpose definitive notation, originally developed with the representation of group-graphs, also known as Cayley Diagrams, in mind (in the cycle of keys); the highly subjective and open-ended nature of certain aspects of the model (such as the choice of colours in the colourwheel, and the interpretation of the harmonic idiom); the way in which static and dynamic aspects of the visualisation are blended (as when synchronising the transformation of the cycle of keys with a performance of the song).

The four images at the top right of Figure 1 are broadly concerned with visualisation that serves a cognitive function. The first of these images was developed by Charles Care, who used EM principles and tools to construct several models of planimeters – analogue scientific instruments widely used prior to the advent of computers for measuring area [21]. (For details, see planimeterCare2005 in the EM archive [23*a*].)



An important feature of Care's model is the way in which relationships between physical observables that are maintained in the actual device by mechanical linkages can be faithfully represented using dependencies. This made it possible to construct the visualisation of the planimeter in a way that reflected Care's emerging understanding of how the component mechanisms worked in isolation and contributed to the functionality of the complete device. The close connection established in this way between the virtual and the actual device proved to be helpful not only in enabling Care to use planimeters in the Science Museum without the assistance of a curator, but also in disclosing issues concerning their design (e.g. in respect of sensitivity to errors in manual tracing of areas) that would not normally be exposed through a virtual construction.

Another example of the application of EM principles and tools to support a cognitive analysis is represented in a visualisation developed to express the nature of the perception and intelligence that is needed to be able to play even such a simple game as noughts-and-crosses [8, 23*b*]. The visualisation itself is built up layer-by-layer in a manner that – it may be surmised – reflects the way in which ability to play noughts-and-crosses relies on progressively more sophisticated capacities for observation and interpretation, ranging from a simple ability to perceive and interpret spatial



```
X has won          = FALSE
0 has won          = FALSE
It is a draw       = FALSE
The board is full  = FALSE

Number of Xs       = 3
Number of 0s       = 3
```

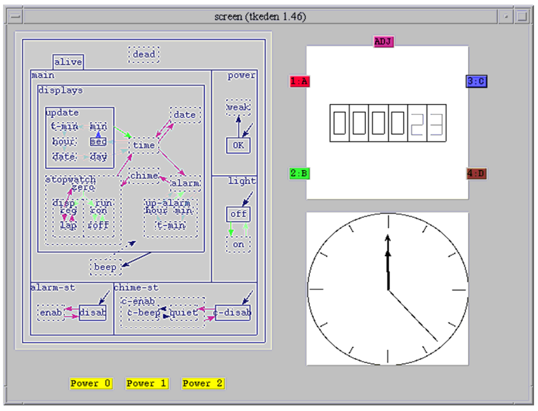
0	0	0
7	0	16
0	0	12

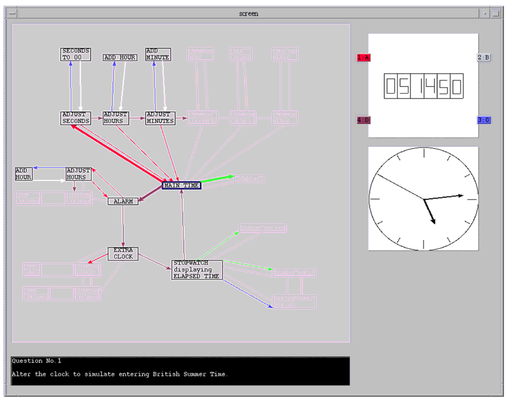
0	41	0
11	0	16
0	0	8

```
HELP:
 This layer incorporates the whole concept of playing a game.  It introduces the concept of whose turn it is.  A player cannot place a
counter if it is not their turn or if the game is over.  You also cannot 'cheat' by removing or overwritting an 0 or an X.
 Click on the 'Initialise' button to clear the board and start a new game.  Click on the yellow button to change who starts (The player to
start is displayed on the button).  Click on the cyan button to turn the computer on or off (The state described on the button says whether
the computer is currently on or off).
```

configurations of lines and symbols to knowledge of game playing conventions and strategies. A significant consequence of building up families of definitions to describe a visualisation in an incremental fashion is that each extension inherits pre-existing redefinitions. In the model of noughts-and-crosses, this means that actions such as modifying the winning lines, directly changing the configuration of noughts and crosses on the board, or revising the conventions that determine when the game is won and whose turn it is to play etc remain at all times open to the model maker. In this way, the visualisation can be seen as simultaneously supporting the roles of many agents, such as game observers, designers and players, both manual and automated.

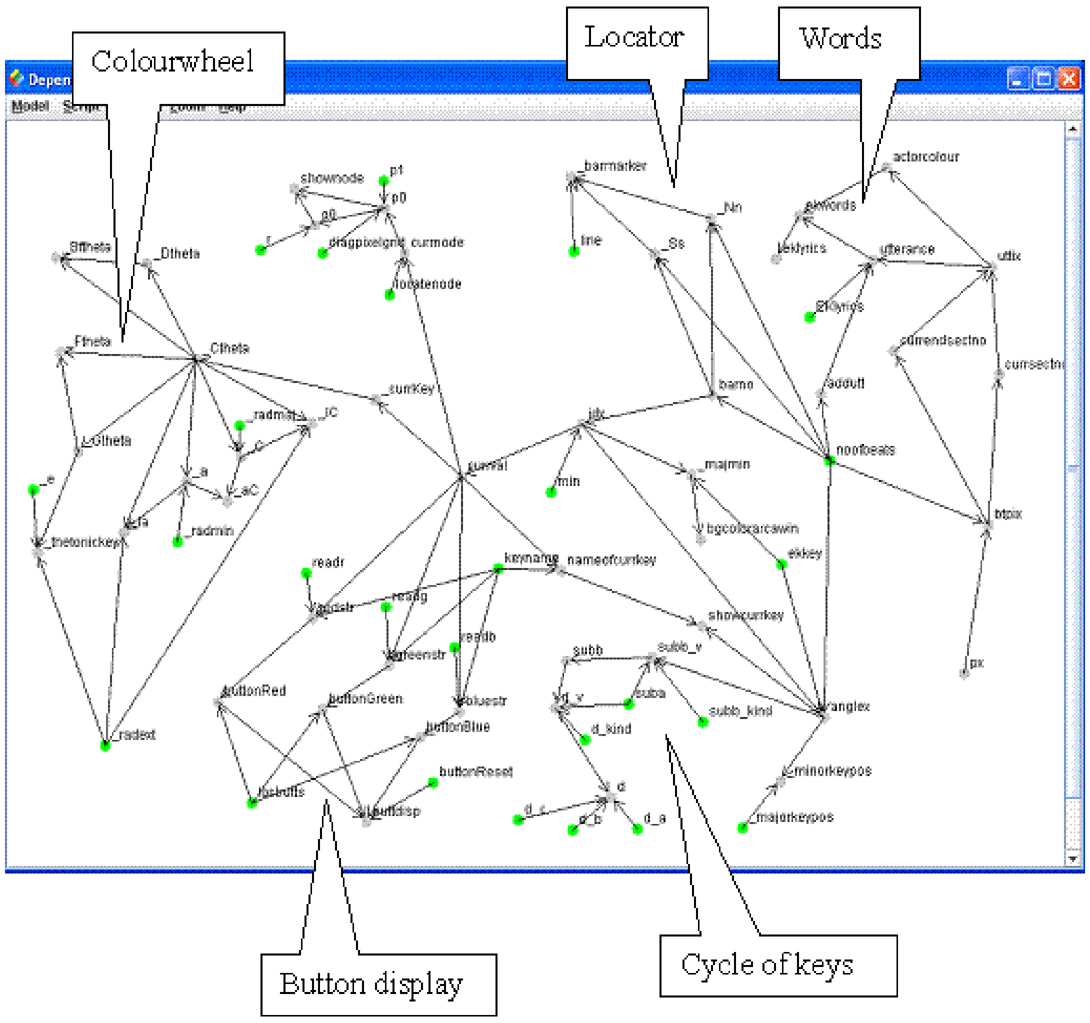
The importance of visualisation in supporting design has been remarked by Harel [27]. The technique that he developed to address this issue, the *statechart* [26], is now a standard ingredient of the Unified Modelling Language (UML). The visualisations of the current state and status of a digital watch depicted below incorporate a statechart devised by Harel to represent the display modes of typical digital watch [25]:



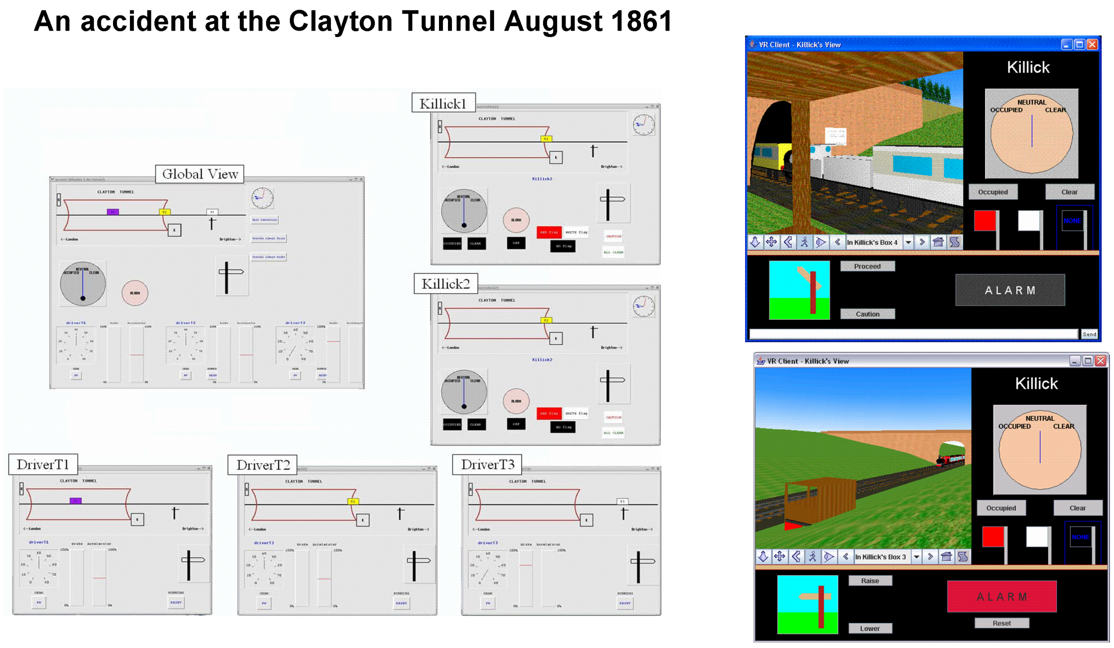


```
Question No.1

Alter the clock to simulate entering British Summer Time.
```

A variant of Harel's statechart appears explicitly in the visualisation on the left [23*c*]. The family of definitions used to specify this was adapted by Chris Roe (cf. [12]) in the visualisation on the right [23*d*] in order to convey his more informal understanding of how the modes of his personal digital watch were configured. Harel's motivation for devising statecharts was that they were capable of succinctly expressing much richer state relationships than could be apprehended by inspecting the equivalent finite state machine diagram. Though families of definitions do not admit such direct visual representation of states and transitions as statecharts, they have much greater expressive power as a means for representing state. This can be gauged from the fact the digital watch visualisations require no more than a few hundred definitions and that these subsume Harel's statechart specification of modes for the watch interface alongside a visual representation of the state of the actual watch, together with an "old-fashioned" analogue clock, and embrace a far more comprehensive model of the mechanisms of the watch that takes full account of its functionality. Families of definitions are also amenable to a limited form of visual representation using dependency graphs. For instance, the graph below, drawn using Wong's Dependency Modelling Tool [36], depicts the structure of a representative subset of the dependencies underlying the Erlkönig model discussed above.

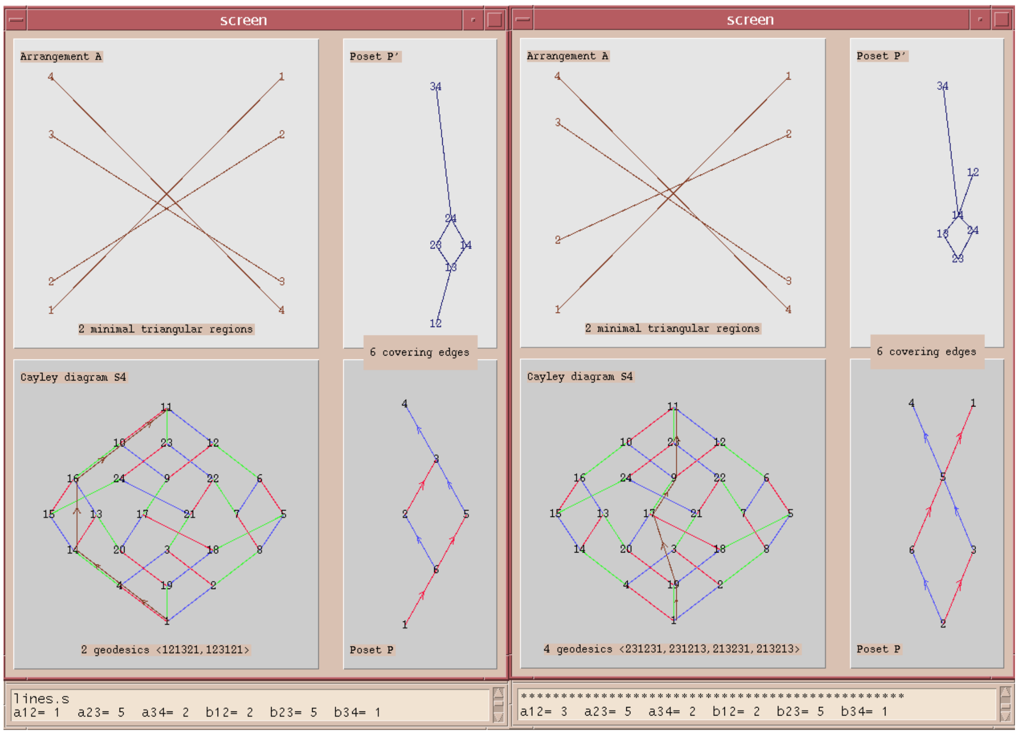


The previous visualisations have been primarily concerned with the perspective of an individual observer. The use of definitions for visualisation lends itself to another kind of activity, in which the primary goal is to reconcile the perspectives of many different individual observers from the point of view of a putative objective external observer. The images below, taken from two different models relating to railway operation in the vicinity of the Clayton Tunnel on the occasion of a serious accident that took place in August 1861, illustrate some of this potential.



The cluster of six images on the left hand side above comprises screenshots taken from a distributed model [23*e*] in which each separate image reflects the viewpoint of one of the railway personnel. The largest of these images depicts what can be construed as a global view of the current state. This distributed model has been exercised using a small team of school-children, each child seated at a different workstation, and playing the role of a signalman or driver (the role of signalman Killick being divided into two separate parts for this purpose). The two images on the right are snapshots from another model [23*f*], in which there is a more visually realistic representation of Killick’s views from his signal box. The contrast between the two styles of visualisations is instructive. Though we might suppose that the more visually realistic model better conveys the current state of affairs as viewed by Killick, this is not in every respect a justified supposition. Whilst the two dimensional visualisation does not convey any sense of what is involved in looking out in different directions from the signal box, and has a crude representation of train visibility – whereby a train disappears from view the moment it enters the tunnel, it may (for instance) be better able to express the idea that Killick has in his mind expectations about where the next train to pass through the tunnel is currently located, even though the train is not yet visible from the box (cf. [14, 10]).

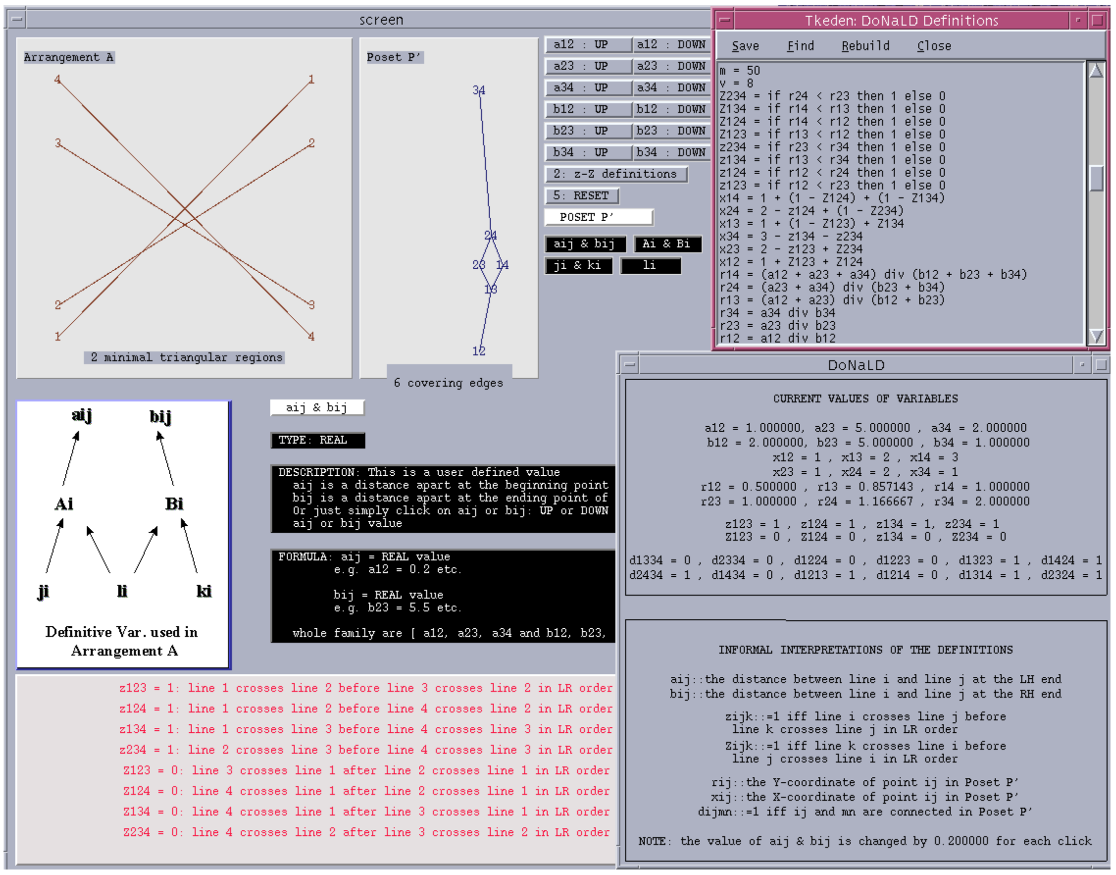
The final cluster of four images, in the bottom left hand corner of Figure 1, is associated with visualising abstract mathematical relationships of various kinds. Viewed in clockwise order from the top left, they comprise two visualisations developed in connection with mathematical research [6], a visualisation developed to explain 3D to 2D projection mapping, and a visual aid to Sudoku solving.



```
lines.s
a12= 1   a23= 5   a34= 2   b12= 2   b23= 5   b34= 1
```

```
**************************************************
a12= 3   a23= 5   a34= 2   b12= 2   b23= 5   b34= 1
```

The top left image in the bottom left cluster [23*g*] depicts a configuration of 4 lines, together with three different geometric representations of the combinatorial pattern exhibited by their intersection. This pattern, which is interpreted as expressing the permutation (14)(23) as a product of transpositions, depends on the intervals between the endpoints of the lines as specified by the ratios a12:a23:a34 and b12:b23:b34.



```
m = 50
v = 8
Z234 = if r24 < r23 then 1 else 0
Z134 = if r14 < r13 then 1 else 0
Z124 = if r14 < r12 then 1 else 0
Z123 = if r13 < r12 then 1 else 0
z234 = if r23 < r34 then 1 else 0
z134 = if r13 < r34 then 1 else 0
z124 = if r12 < r24 then 1 else 0
z123 = if r12 < r23 then 1 else 0
x14 = 1 + (1 - Z124) + (1 - Z134)
x24 = 2 - z124 + (1 - Z234)
x13 = 1 + (1 - Z123) + Z134
x34 = 3 - z134 - z234
x23 = 2 - z123 + Z234
x12 = 1 + Z123 + Z124
r14 = (a12 + a23 + a34) div (b12 + b23 + b34)
r24 = (a23 + a34) div (b23 + b34)
r13 = (a12 + a23) div (b12 + b23)
r34 = a34 div b34
r23 = a23 div b23
r12 = a12 div b12
```

```
DESCRIPTION: This is a user defined value
  aij is a distance apart at the beginning point
  bij is a distance apart at the ending point of
  Or just simply click on aij or bij: UP or DOWN
  aij or bij value
```

```
FORMULA: aij = REAL value
        e.g. a12 = 0.2 etc.

        bij = REAL value
        e.g. b23 = 5.5 etc.

  whole family are [ a12, a23, a34 and b12, b23,
```

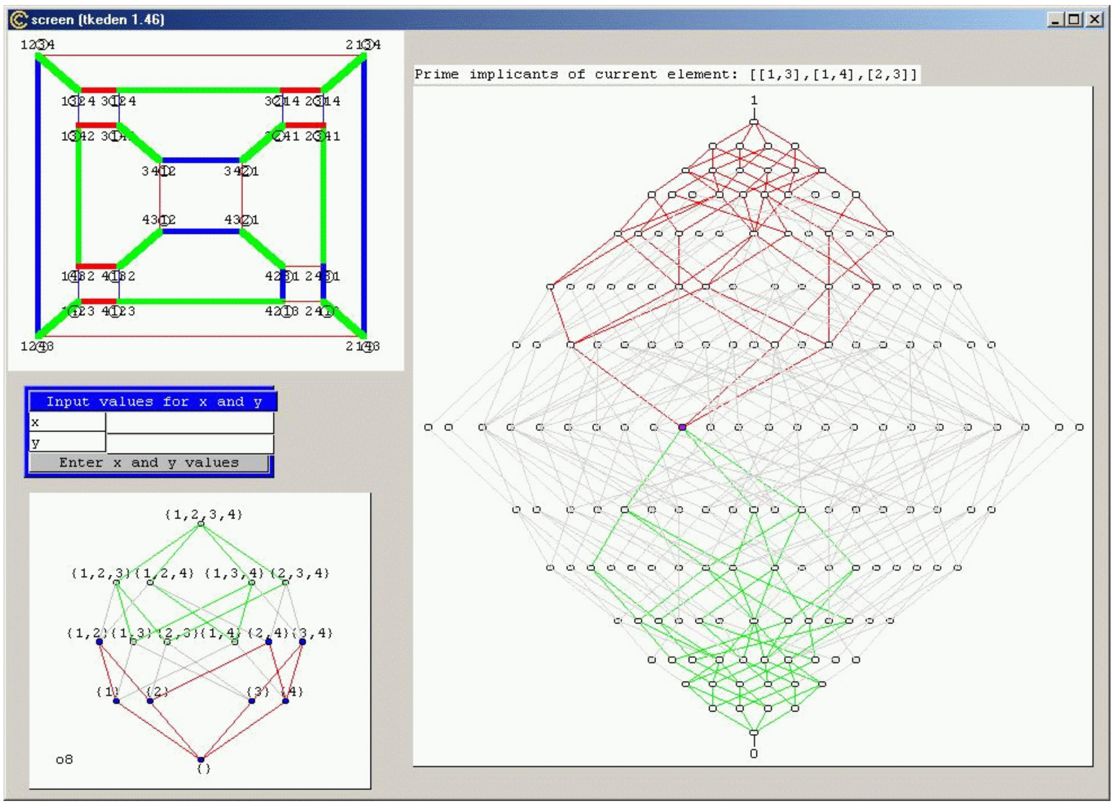
```
z123 = 1: line 1 crosses line 2 before line 3 crosses line 2 in LR order
z124 = 1: line 1 crosses line 2 before line 4 crosses line 2 in LR order
z134 = 1: line 1 crosses line 3 before line 4 crosses line 3 in LR order
z234 = 1: line 2 crosses line 3 before line 4 crosses line 3 in LR order
 Z123 = 0: line 3 crosses line 1 after line 2 crosses line 1 in LR order
 Z124 = 0: line 4 crosses line 1 after line 2 crosses line 1 in LR order
 Z134 = 0: line 4 crosses line 1 after line 3 crosses line 1 in LR order
 Z234 = 0: line 4 crosses line 2 after line 3 crosses line 2 in LR order
```

```
                 CURRENT VALUES OF VARIABLES

         a12 = 1.000000, a23 = 5.000000 , a34 = 2.000000
         b12 = 2.000000, b23 = 5.000000 , b34 = 1.000000
                  x12 = 1 , x13 = 2 , x14 = 3
                  x23 = 1 , x24 = 2 , x34 = 1
        r12 = 0.500000 , r13 = 0.857143 , r14 = 1.000000
        r23 = 1.000000 , r24 = 1.166667 , r34 = 2.000000

             z123 = 1 , z124 = 1 , z134 = 1, z234 = 1
             Z123 = 0 , Z124 = 0 , z134 = 0 , Z234 = 0

d1334 = 0 , d2334 = 0 , d1224 = 0 , d1223 = 0 , d1323 = 1 , d1424 = 1
d2434 = 1 , d1434 = 0 , d1213 = 1 , d1214 = 0 , d1314 = 1 , d2324 = 1
```

```
            INFORMAL INTERPRETATIONS OF THE DEFINITIONS

       aij::the distance between line i and line j at the LH end
       bij::the distance between line i and line j at the RH end

             zijk::=1 iff line i crosses line j before
              line k crosses line j in LR order
             Zijk::=1 iff line k crosses line i before
              line j crosses line i in LR order

           rij::the Y-coordinate of point ij in Poset P'
           xij::the X-coordinate of point ij in Poset P'
         dijmn::=1 iff ij and mn are connected in Poset P'

  NOTE: the value of aij & bij is changed by 0.200000 for each click
```

By enriching the family of definitions used to define the configuration of lines, the internal structure and significance of the associated dependencies can be revealed, as in the enhanced visualisation above [35]. An extract from the original family of definitions is shown in the top right corner of the enhanced visualisation. This exploits observables with relatively complex interpretations as predicates. For instance, the observable z123 has the value 1 if *the line 1 crosses line 2 before line 2 crosses line 3 in left-to-right order*. New definitions are introduced to maintain a dependency such that the truth of this predicate is asserted as and when the observable z123 has the value 1. A particularly interesting feature of using families of definitions as a state representation is that it is robust even when singular conditions prevail, as happens – for instance – if lines 1 and 3 cross line 2 in the same point. This is typically in contrast to the behaviour of visualisations that rely on procedural specifications, which are prone to fail catastrophically when computations return undefined values, and have no elegant means to express the notion that such values can arise as a mathematical phenomenon rather than merely stem from the limitations of arithmetic.

The lines configuration visualisation was constructed in connection with mathematical research linked to the study of *combinatorially piecewise linear maps*: an unusual representation for monotone Boolean expressions discovered by the author [1]. (A Boolean expression is monotone if it only uses the connectives **AND** and **OR**.) The visualisation below [23*h*] was developed to illustrate the connection between four different representations for monotone Boolean expressions in four variables.

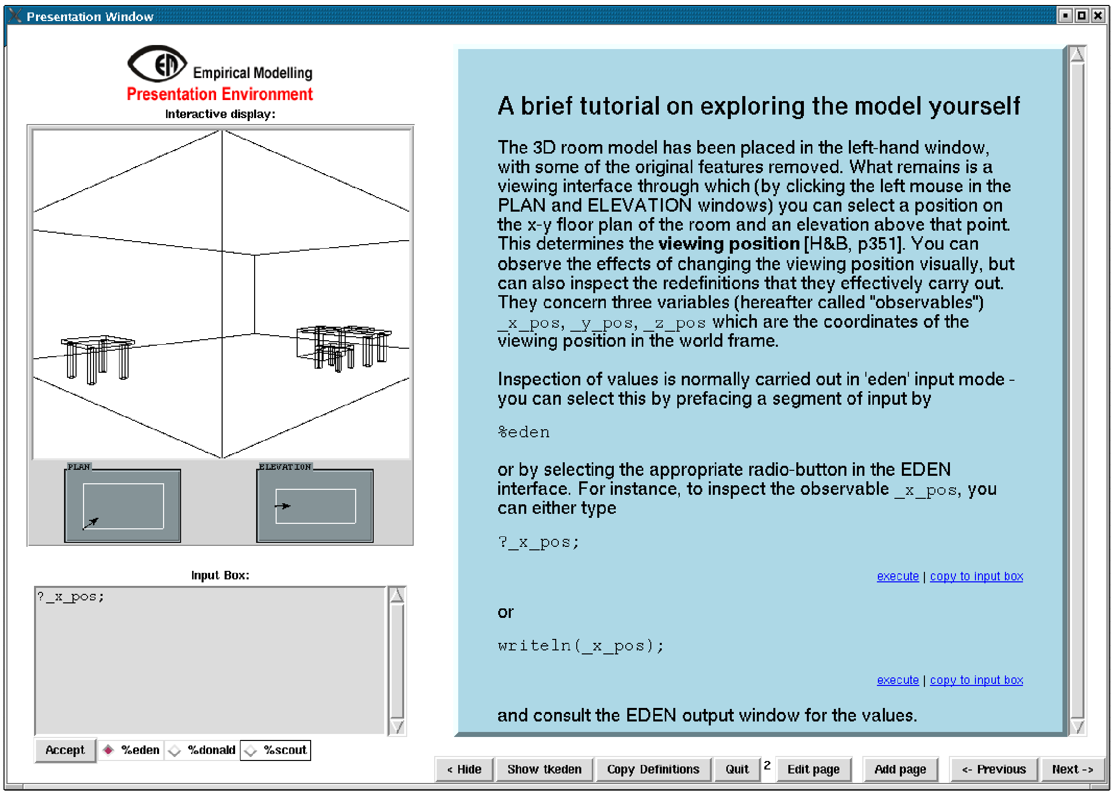


Prime implicants of current element: [[1,3],[1,4],[2,3]]

The image on the right identifies a monotone Boolean expression (MBE) in four variables x,y,z,t with an element of the free distributive lattice on four generators FDL4. The MBE itself is identified with the prime implicants in its disjunctive normal form, as encoded in the list above the image of FDL4. The specific MBE pictured in this visualisation can be identified as (x AND z) OR (x AND t) OR (y AND z). The image at the bottom left depicts the decreasing subset of the power set of {1,2,3,4} defined by the zero set of this MBE. The image at the top left depicts the associated combinatorially piecewise linear map. Significant features of this visualisation are: the part-manual part-automated process that was used to convert a static computer-based visualisation of FDL4 constructed by John Buckle [20] (the first visualisation of the structure of this lattice) into a dynamic representation; the way in which the representation of combinatorially piecewise linear maps re-uses the group graph component that appears in the line configuration model, redefining its vertices to obtain a planar layout; the efficacy of dependency as a basic mechanism for dealing with the equivalence between different representations that is common in mathematics.

Using EM principles and tools to construct visualisations to support mathematical research has potential merits over other approaches. Because the construction of visualisations is semantically guided, the end result is a family of definitions of observables many of which are meaningful within the underlying domain. In principle, this makes it possible to extend and adapt models in a far more open-ended fashion in response to new insights or exploratory designs. More investment in the tools would ideally be required to exploit these advantages fully. Model building to support research is time-consuming, work-intensive and demands specialist knowledge of tools. Because of the primitive nature of the observable, dependency and agency concepts, there is no direct support for object-orientation and only limited facility for semi-automated definition. This favours the construction of visualisations that are typically subtle but small scale.

The advantages of the intimate connection between domain learning and model making are at present more easily appreciated in a teaching context. An application of EM for visualisation in teaching computer graphics [23*i*] is shown in the image below.



```
?_x_pos;
```

A brief tutorial on exploring the model yourself

The 3D room model has been placed in the left-hand window, with some of the original features removed. What remains is a viewing interface through which (by clicking the left mouse in the PLAN and ELEVATION windows) you can select a position on the x-y floor plan of the room and an elevation above that point. This determines the **viewing position** [H&B, p351]. You can observe the effects of changing the viewing position visually, but can also inspect the redefinitions that they effectively carry out. They concern three variables (hereafter called "observables") `_x_pos`, `_y_pos`, `_z_pos` which are the coordinates of the viewing position in the world frame.

Inspection of values is normally carried out in 'eden' input mode - you can select this by prefacing a segment of input by

```
%eden
```

or by selecting the appropriate radio-button in the EDEN interface. For instance, to inspect the observable `_x_pos`, you can either type

```
?_x_pos;
```

execute | copy to input box

or

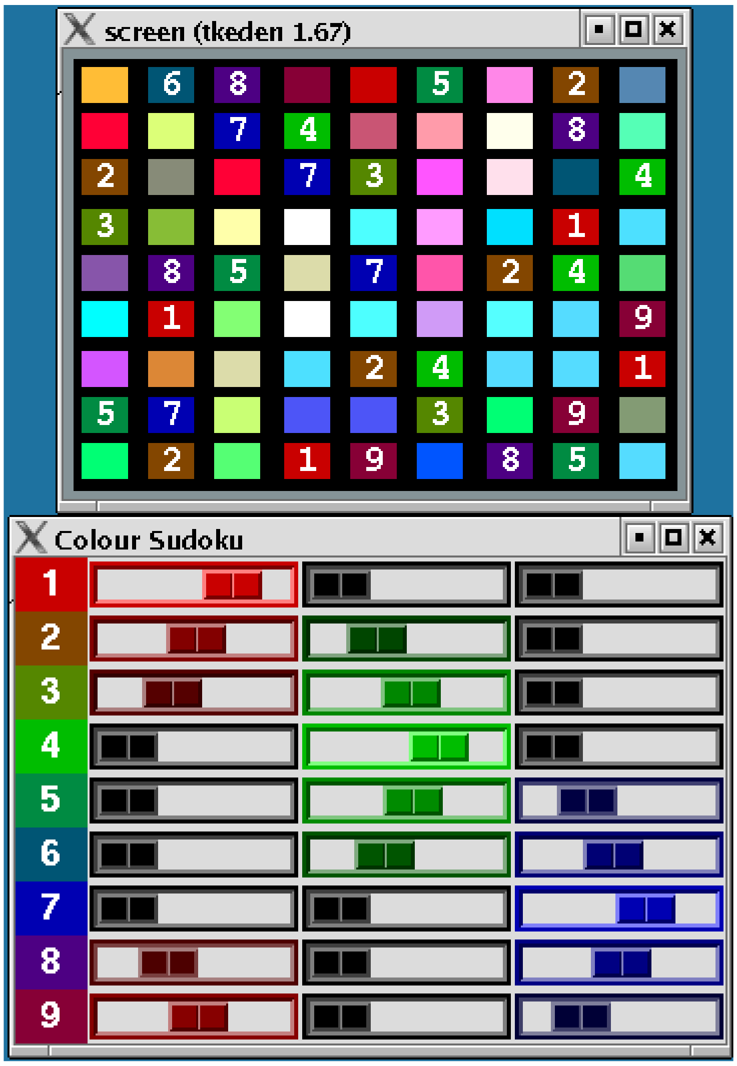
```
writeln(_x_pos);
```

execute | copy to input box

and consult the EDEN output window for the values.

In this application, the EM Presentation Environment is being used to introduce the concept of projection from 3D to 2D. The 3D line drawing of the room in the Interactive Display window is defined by applying a projection function, explicitly defined by the modeller, to map points with 3 spatial coordinates to points on the 2D display screen that can be directly represented using the built-in 2D line drawing notation (cf. Figure 2 and Listing 1). Within the presentation environment, interaction through simple redefinitions of parameters associated with the projection function serves to illustrate a wide range of properties of projection. It allows the learner to trace the exposition of 3D to 2D projection as it is appears in a standard graphics textbook, such as Hearn and Baker [28] Chapter 5, complementing the static images in the textbook with animations of activities such as panning round a scene, adding construction lines to identity vanishing points, and transforming the projection function from a perspective to an orthogonal projection. The first few slides of the presentation introduce the elementary knowledge of the underlying EM tools that is needed for this purpose. These are sufficient to give a learner without specialist knowledge of EM principles and tools the means not only to exercise the standard interactions embedded in the presentation, but to carry out independent exploration.

The final example of visualisation [23*j*], at the bottom left corner of Figure 1, makes use of colour to convey combinatorial information in Sudoku puzzle solving.



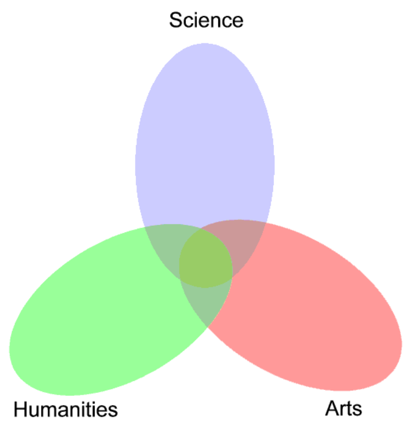
The principle behind the visualisation is simple. Following a common strategy that is used by many Sudoku solvers, there is a list of possible digits associated with each empty square in the grid, comprising digits that do not appear in the enclosing row, column or 3-by-3 region. The content of this list, rather than being recorded as a set of digits (e.g. pencilled into the square, and later refined by a deductive process of elimination), is instead reflected in a visual colour encoding of the square. To define the encoding, a different colour is assigned to each of the nine digits, and a set of digits is represented by blending the appropriate colours. The darker squares in the grid are then those for which there are fewer possibilities by the simple criterion of eliminating digits that already appear in the same row, column or region (a black square indicating that an error has been made). This can make it possible to infer the content of a square by visual inspection and colour matching – for instance identifying that the entry in the fourth square of the top row above is necessarily 9. Modelling with dependency ensures that the colours of other squares are updated when this 9 is entered into the grid. As an additional aid to solution, the RGB components used to define the colours associated with the nine digits can be controlled through manipulating the sliders in the Colour Sudoku interface. Setting the colour associated with digit 2 to white, for instance, will highlight all the squares in which 2 can possibly be entered by the naïve basic criterion for elimination. From this, it becomes immediately apparent that the ninth column is the only location in the eighth row where the digit 2 can be entered.

3. Discussion

The illustrative examples discussed above (all of which are available for download at [23]) exhibit the wide variety and scope of applications for EM principles and tools in visualisation. Virtues of the approach that are illustrated in many of these examples include extensibility, re-usability, potential for exploiting and integrating many visualisations in parallel, and means to trace the relationships that connect visual data to internal and abstract values. The use of a family of definitions for visualisation is convenient because (unlike a sequence of instructions) it admits a direct static interpretation as a set of current values for observables. It is also semantically powerful because (unlike a set of explicit values) it expresses dependencies amongst observables that can be interpreted as latent atomic transformations accessible to a variety of different state-changing agents according to their role and status. The way in which the observables, dependencies and agency associated with an EM artefact are to be interpreted is crucial in shaping its possible applications.

In a scientific context, the aspiration is for observables, dependencies and agency whose nature is rigorously prescribed. The measurable entities, the manner in which these are measured, the kind of interaction with these entities that is considered legitimate, the law-like relationships between entities that are respected in this interaction, and the way in which the effects of interactions are interpreted are ideally all well-understood and consistent with expectation and theory. In this setting, visualisation typically expresses the view of the objective observer. In the context of artistic creation, an entirely different regime for observation and interaction prevails. The significant observables are only partially preconceived, never precisely prescribed and identified, and their interrelationships emerge through skilful and experimental interaction and interpretation. Visualisation is first and foremost a personal expression – a record of what is found to be satisfying and meaningful in the experience of the artist. Within the humanities, the subject matter is drawn from the realm of the arts, but the emphasis in modelling is upon the inter-subjective. The visual expression of entities and the relationships amongst them serves to externalise a personal understanding, and helps to expose it to the many different viewpoints and interpretations from which it can be critically evaluated. It is in this attempt to accommodate a second-person perspective that humanities helps to bridge the gap between the artist's imagination and the scientist's reality.

In one of the group sessions that followed the presentation of the visualisations in Figure 1 at the recent 3DVisA workshop, Hugh Denard highlighted the relationship between the epistemological stances of Science, Humanities and the Arts as a key issue in understanding the role and potential of visualisation. The following simple Venn diagram served as a visual cue for the ensuing discussion:



Denard's observation about the need in general to reconcile visualisations reflecting many different disciplinary perspectives is vividly illustrated in his research into creating virtual models of historical buildings. As remarked by Drew Baker during the discussion, there is enormous diversity and scope in the "cloud of interpretations" that is relevant to bringing intellectual transparency to the creation of such a model. The factors to be accommodated include hard physical facts, speculative scholarship subject to be challenged and revised and guesswork based on intuition and artistic licence. The need here is to bridge first-, second- and third-person perspectives in a way that draws on the arts, humanities and science.

Denard placed Empirical Modelling in the intersection between Science and the Humanities in his diagram. This is appropriate in so far as EM is deemed to involve *making a model*, and thereby having some referent in mind. It is apparent that such a model can take a precise mathematical form in relation to a scientific application where there is an underlying theory, but can also fulfil the role of modelling in humanities computing as identified by Willard McCarty [32] Chapter 1, where the interpretation is always evolving, and reflects the tension between objective rationalisation and subjective intuition.

In fact, EM has an even more ambitious objective: that of supporting a holistic approach that can draw upon the epistemological stances of all three disciplines [11,17]. It is helpful here to recognise that the sharp association of science and the arts with objective and subjective experience is in many respects misleading. New science itself begins in the personal domain, and – in the view of the philosopher of science David Gooding [24] – relies in its pre-theoretic experimental practices on the construction of interactive artefacts ("construals") that tacitly capture provisional and tentative understandings. In a complementary way, the lifework of a productive artist – despite the highly personal character of each individual creation – can be seen as part of a higher level creative activity, in which an expressive language is being developed that – whilst it can never admit a formal objective interpretation – is accessible to those who become sufficiently well-acquainted with the entire oeuvre. The distinctive feature of the more informal and primitive representational stance that is found in art and in experimental science is that the meaning of the artefact can only be mediated through live interaction. What potentially bring objectivity to the interpretation of the artefact are the reliable patterns of interaction and interpretation that develop around it.

There are many ways in which the word 'reliable' can be understood according to context. In science, the concern is for well-defined experimental settings and procedures, where observations and interactions predictably lead to essentially the same outcomes. In classical computing, there is generic interaction and observation associated with a computational device, complemented by standard encodings and interpretative conventions to address algorithmic problem solving. In music, there are conventions of performance that become ritualised within different traditions, and some apparent consensus about the appropriateness of music to particular occasion and mood. What these diverse manifestations of reliable interaction and interpretation have in common is a promise of moment-by-moment management of personal experience that can to a greater or lesser degree be controlled so as to become familiar. Observation, dependency and agency are the basic ingredients of the conceptual framework within which this shaping of experience can be ventured through EM.

Quite apart from the convergence to an ever closer correspondence with an external referent that is characteristic of model refinement, the sculpting of reliable experience can – and in many contexts necessarily must – take on a different character. This is evident for instance in software development, where the term 'modelling' is routinely - and to some extent legitimately - applied, even though there is something inherently problematic about the idea that the preliminary fragments of a specification can be deemed to be a model of an actual software product that emerges only after several significant further stages of development (cf. [31] for a related discussion). In spite of its name, EM entails something broader than *model making* in its strict and proper sense. A critical examination of Listing 1, for instance, shows that the family of definitions that purports to model the floor plan of a filing cabinet does not accommodate the possibility of totally withdrawing the drawer, nor reflect the fact that the dimensions of the drawer itself are independent of whether or not it is open. What is more, if we set out to model the physical process of opening the drawer, it becomes necessary to engage with the interpretation of redefinitions in a quite pragmatic empirical manner – potentially considering such issues as how accurately the motion of the drawer can be displayed on the discrete screen display, how fast an actual filing cabinet drawer can be opened and shut, and the speed at which the redefinitions of the drawer location can be evaluated and displayed. By such considerations, we are led to a totally different perception of the role of the underlying EDEN interpreter, and a new perspective from which it can no longer be viewed merely as an abstract computational device. In this context, reliable interaction and interpretation is concerned with the extent to which we can develop a suitable *instrument* for simulating the motion of a drawer.

Denard's diagram of the disciplines is a useful setting within which to interpret the way that the broad concept of "EM for visualisation" illustrated in this paper first emerged, then evolved, over the last twenty years. Such a historical account helps to give a more complete picture of what EM for visualisation entails than can be appreciated from the most accessible illustrative examples alone – which are naturally biased towards model making applications. It also provides a useful guide for a reader who wishes to explore EM itself in more detail. (Note that there is something anachronistic about referring to "EM" throughout this account, since the term was first introduced fifteen years ago.)

Since EM was initially conceived in a computer science setting, it was natural for it to be at first associated with a style of programming, so-called 'definitive programming', in which families of definitions were used to represent state [2,3]. Under this interpretation, the notions of observable, dependency and agent most naturally inherited the static objective character appropriate to a scientific framework. In this context, EM could be seen as closely resembling traditional uses of spreadsheets, and as exploiting dependency in interface design (as has been done to great effect in Amulet [33]) or in design patterns such as *model-view-controller* [34]. What gave a distinctive character to EM, alien to abstract computational theory, was its emphasis on the importance of the experiential human interpretation of observables, and the recognition that observables and dependencies might elude formal specification, being subject to different modes of interpretation according to the observing agent and the context for interaction (cf. [7]). This established an important connection between EM and visualisation from the first.

The broader significance of this connection only became clear when the role played by observables, dependency and agency in shaping the semantics of geometric symbols was explicitly identified in connection with a short presentation for which the digit/filing-cabinet visual pun displayed in Figure 2 was devised [5]. Though the discussions of visualisation at that time (see e.g. [6]) were framed with reference to the idea of "definitive programming" [2,3], it was already apparent that it was more appropriate to regard EM as a *modelling* rather than a *programming* activity in general [4]. The adoption of the term "Empirical Modelling" was associated with the recognition that the modelling activity was rooted in observation and experiment, and highlighted its affinity with building construals [9], where the referent itself is being conceived in the model making process. The significant connections subsequently made with William James's Radical Empiricism [29, 15] licensed an altogether different view, in which those aspects of EM that go beyond building an artefact to reflect state-as-experienced are regarded as complex and sophisticated constructs. It is this step that gives peculiar centrality to the theme of EM for visualisation, endorsing the notion of a neutrality in representation that potentially situates EM activity at the common intersection of all three disciplines in Denard's diagram (cf. [11,17,18]). What then determines the characteristics of an EM artefact (and whether indeed it should be deemed to be a program, a model, a construal or an instrument) is the interactive and interpretative activity that comes to be associated with it. In Jamesian terms, there is no absolute distinction to be made between the experience the artefact offers to the human interpreter no matter how our interpretation of it migrates between the different disciplines – whether we adopt the perspective of science, arts or the humanities in relation to it is a pragmatic matter of classification of experience [29:141] subject only to the constraints imposed by the obligations and imperatives of sense-making. Finally – echoing an issue that was raised in our discussion of Denard's diagram – there remains the question of how such activity relates to social studies. Following the rehabilitation of the vexed notion of so-called 'social constructivism' ventured by Bruno Latour in his paper "The Promises of Constructivism" [30], we are led to view the negotiation of possible kinds of agency, observables and dependencies within EM as a process of construction, for which we have recently proposed the term *constructivist computing* [19].

Acknowledgements

I am indebted to John Buckle, Charles Care, Richard Cartwright, Zhan En Chan, Carlos Fischer, Simon Gardner, Antony Harfield, Mike Joy, Karl King, Andrew MacDonald, Chris Roe, Jaratsri Rungrattanaubol, Pi Hwa Sun and Allan Wong for their role in creating the visualisations in this paper, and to Ashley Ward and Steve Russ for their significant practical and theoretical contributions to its overall theme. I am grateful to Hugh Denard for some stimulating ideas, and above all to Julie Tolmie for her inspiration and enthusiastic encouragement.

References

1. Beynon, W.M. 1974. Combinatorial aspects of piecewise-linear maps JLMS (2) 7, 719-727
2. Beynon, W.M. 1989. A definitive programming approach to the implementation of CAD software. Intell. CAD Systems II: Implementation Issues, Springer-Verlag, 126-45.
3. Beynon, W.M., Yung, Y.P. 1990. Definitive interfaces as a visualisation mechanism. Proc Graphics Interface '90, Canadian Information Processing Soc, 285-292.
4. Beynon, W.M., Russ, S.B., Yung, Y.P. 1990. Programming as Modelling: New Concepts and Techniques. Proc ISLIP'90, Computing & Info Science Dept, Queen's Univ, Canada 1990.
5. Beynon, W.M., Cartwright, A.J., Russ, S.B., Yung, Y.P. 1990. Programming Paradigms and the Semantics of Geometric Symbols (abstract). Proc. W/S "Visual Interfaces to Geometry" at CHI'90, Seattle, April 1990.
6. Beynon, W.M., Yung, Y.P., Atkinson, M.D., Bird, S.R. 1991. Programming Principles for Visualisation in Mathematical Research. Proc. Compugraphics '91: 1st Int Conf on Computational Graphics & Visualisation Techniques, 288-298.
7. Beynon, W.M., Yung, Y.P., Cartwright, A.J., Horgan, P.J. 1992. Scientific visualisation: experiments and observations. Proc. Eurographics W/S: Visualization in Scientific Computing, 157-173.
8. Beynon, W.M. and Joy, M.S. 1994. Computer Programming for Noughts-and-Crosses: New Frontiers. In Proc. PPIG'94, Open University, 27-37.
9. Beynon, W.M. 1994. Agent-oriented Modelling and the Explanation of Behaviour. Proc. International W/S "Shape Modeling Parallelism, Interactivity and Applications", Dept. of Computer Software, TR 94-1-040, Univ. Aizu, Japan, 54-63.
10. Beynon, W. M., Sun, P.-H. 1999. Computer-mediated communication: a Distributed Empirical Modelling perspective, Proc CT'99, San Francisco.
11. Beynon, W.M. 1999. Empirical Modelling and the Foundations of Artificial Intelligence. Computation for Metaphors, Analogy and Agents, LNAI 1562, Springer, 322-364.
12. Beynon, W.M., Roe, C.P, Ward, A.T., Wong, K.T.A. 2001. Interactive Situation Models for Cognitive Aspects of User-Artefact Interaction. Proc Cognitive Technology: Instruments of Mind, University of Warwick, August 2001, LNAI 2117, Springer-Verlag, 356-372
13. Beynon, W.M. & Russ, S.B. 2004. Redressing the past: liberating computing as an experimental science. Computer Science Research Report 421, University of Warwick, 2006. Also at url: http://www.nesc.ac.uk/esi/events/Grand_Challenges/gcconf04/submissions/26.pdf (accessed 15th August 2007)

14. Beynon, W.M. 2005. Computational Support for Realism in Virtual Environments. In Proc 11th International Conference on Human-Computer Interaction (HCII 2005): Volume 10 - Internationalization, Online Communities and Social Computing: Design and Evaluation, Las Vegas, NV, 22-27 July 2005.
15. Beynon, W.M. 2005. Radical Empiricism, Empirical Modelling and the nature of knowing. In Dror, Itiel, E. (ed.) Cognitive Technologies and the Pragmatics of Cognition: Special issue of Pragmatics and Cognition, 13:3, 615-646.
16. Beynon, W.M. 2006. Mathematics and Music - Models and Morals. In Conference Proceedings, Bridges London: Mathematical Connections in Art, Music, and Science (eds. Sarhangi and Sharp), Tarquin Books, 437-444.
17. Beynon, W. M., Russ, S. B., McCarty, W. 2006. Human Computing: Modelling with Meaning. Literary and Linguistic Computing 21(2), 141-157.
18. Beynon, W.M., Klein, R.R., Russ, S.B. 2006. Humanities' Computings (extended abstract). In Digital Humanities 2006: 1st International Conference of the Alliance of Digital Humanities Organisations, Conference Abstracts, Paris-Sorbonne, France, July 2006, 17-20.
19. Beynon, W.M., Harfield, A.J. 2007. Lifelong Learning, Empirical Modelling and the Promises of Constructivism. Journal of Computers, Volume 2, Issue 3, 43-55.
20. Buckle, J.F. 1990. Computational aspects of lattice theory. PhD Thesis, Computer Science, University of Warwick.
21. Care, C. 2006. The Analogue Computer as a Scientific Instrument, Computer Science Research Report 420, University of Warwick.
22. The EM website: http://www.dcs.warwick.ac.uk/modelling (accessed 15th August 2007)
23. The EM archive: http://empublic.dcs.warwick.ac.uk/projects (accessed 15th August 2007)

a. planimeterCare2005
b. oxoGardner1999
c. digitalwatchFischer1999
d. digitalwatchRoe2001
e. claytontunnelSun1999
f. claytontunnelChanHarfield2005
g. linesBeynon1991
h. mbf4Beynon2003
i. graphicspresHarfield2007
j. sudokucolourHarfield2007
k. cabinetdigitBeynon1990
l. cabinetdigitpresBeynon2007

24. Gooding, D. 1990. Experiment and the Making of Meaning: Human Agency in Scientific Observation, Kluwer.
25. Harel, D. 1987. Algorithmics: the spirit of computing. Addison-Wesley.
26. Harel, D. 1987. Statecharts: A visual formalism for complex systems. Sci. Comput. Program. 8, 3 (Jun. 1987), 231-274.
27. Harel, D. 1988. On visual formalisms. Commun. ACM 31, 5 (May. 1988), 514-530.
28. Hearn, D, Baker, M.P. 2004. Computer Graphics with OpenGL, Prentice Hall.
29. James, W. 1912. Essays in Radical Empiricism. (Reprinted from the original 1912 edition by Longmans, Green and Co, New York) London: Bison Books. 1996
30. Latour, B. 2006. The promises of constructivism, In Idhe, D. (ed) Chasing Technoscience: Matrix of Materiality.
31. Loomes, M., Nehaniv, C.L. 2001. Fact and Artifact: Reification and Drift in the History and Growth of Interactive Software Systems. Cognitive Technology 2001, 25-39
32. McCarty, W. 2005. Humanities Computing, London: Palgrave.
33. Myers, B., et al. 1997. The Amulet Environment: New Models for Effective User Interface Development, IEEE Transactions on Software Engineering, Vol 23(6), 346-365
34. Reenskaug, T.M.H. Model-View-Controller: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html (accessed 15th August 2007)
35. Rungrattanaubol, J. 2002. A treatise on Modelling with definitive scripts, PhD Thesis, Computer Science, University of Warwick.
36. Wong, K.T.A. 2003. Before and Beyond Systems: An Empirical Modelling Approach, PhD Thesis, Computer Science, University of Warwick.