Programming from an Empirical Modelling perspective

From modelling with definitive scripts to programming:

- representing state in programming
- behaviour of programs
- the semantics of programs

Programming from an Empirical Modelling perspective

EM in the first instance models *state* …

… many varieties of state in programming

States relevant to programming ...

- state within the executing program
- external state: what is visible?
- state in respect of interaction
- state in program development
- state significant in the external world

Diverse representations are required:

- *state within the executing program*
  - Program variables, machine locations
- *external state: what is visible?*
  - Graphics / display techniques
- *state in respect of interaction*
  - Statechart, message sequence diagram

Diverse representations required …

- *state in program development*

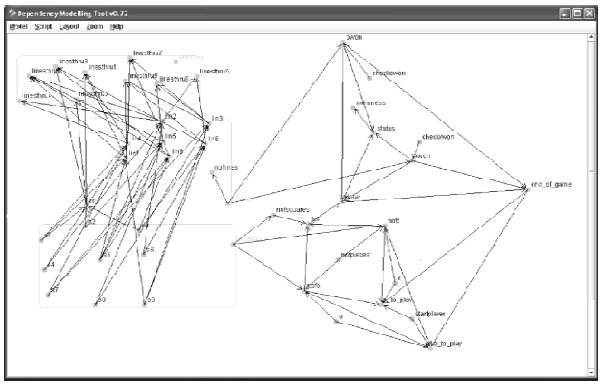
  UML diagrams, prototypes
- *state significant in the external world*

  apprehended by the human interpreter

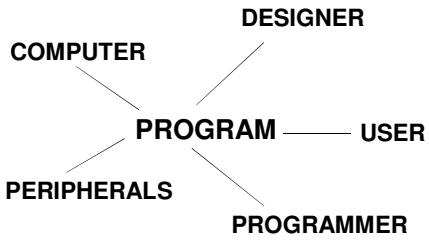
cf. Brian Cantwell-Smith on semantics …

States within oxoJoy1994

Definitive scripts express …
- internal state – contents of squares
- visible state – appearance of the board
- interaction state: whose turn is it?
- state of development
- state of mind of the player: which square?

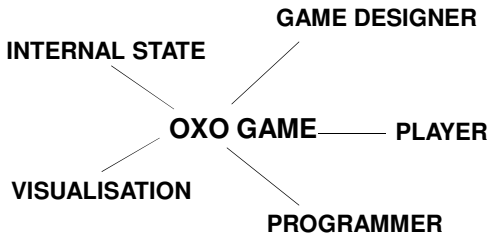


Dependencies in the oxoJoy1994 model



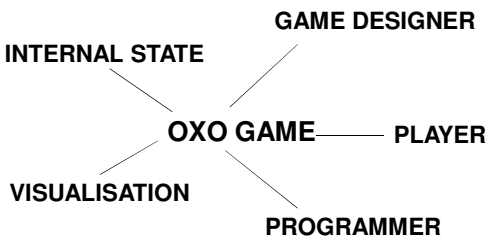
Diverse relations / representations in a traditional program

… compare this with the OXO laboratory



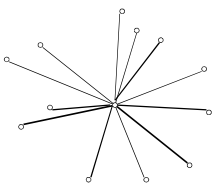
… all relations mediated by definitions

… Behaviour as programmed state change



Static and dynamic elements of state

Definitive scripts as “furry blobs”



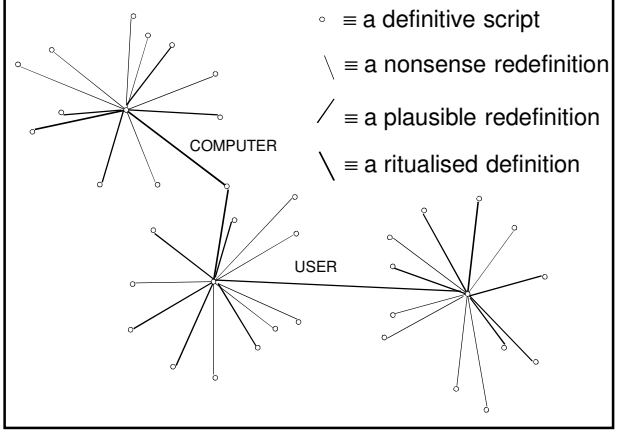
◦ ≡ a definitive script

\ ≡ a nonsense redefinition

/ ≡ a plausible redefinition

\ ≡ a ritualised definition

Plausible : *could* open the desk drawer
– note continuous spectrum of redefinitions
Ritualised : door *automatically* closes after being opened
Nonsense : opening the drawer makes the room smaller



Modelling with definitive scripts:

… a holistic view of state that integrates and conflates all the different perspectives

in contrast to

Programming-in-the-wild:

… an eclectic model of state in which many different strategies for representation and interpretation are jumbled up together

Two emphases

- Empirical Modelling encourages us to consider programming in a holistic way, using similar principles to deal with the entire process of development from conception to customisation and use
- It also has a means to represent the specific activity that is captured by a traditional program (a "pseudo-program")

Traditional programming

Requirements capture and specification	specification	Program design implementation maintenance	user interface	**Use** affordances interface culture
Identifying agency in the machine-like components and in the human context for use *Framing goals for the design protocols for interaction and interpretation* *e.g. devise UML*		*constructing and programming the machine-like components* *designing program by identifying objects and functions* *technical interface development* *e.g. writing Java code*		*human factors study* *interface design* *empirical studies of use* *prototyping* *e.g. goals, operators, methods (GOMS) evaluation*

Empirical Modelling

Requirements capture and specification	Program design implementation maintenance	**Use** affordances interface culture
develop scripts in isolation as "furry blobs" that represent the observables and dependencies associated with putative machine-like components and human interactions and interpretations	*identify and document reliably reproducible sequences of redefinition / chains of "furry blobs" that correspond to programmable automatable machine behaviours and ritualisable human behaviours and interfaces*	*exercise, explore, customise, revise and adapt sequences of redefinition and interpretation to reflect emerging and evolving patterns of interaction and interpretation; extend and augment observables to support additional functionalities combining scripts*

Objects and dependencies

- An **object** corresponds to a particular way of associating observables: grouping together observables according to whether they exist concurrently
- A **dependency** links observables according to how they are linked in change: whether making a change to the value of one observable necessarily entails changing others

Object model vs. account of observation

An account of observation is in some respects a more primitive concept than an object model: it entails fewer preconceptions about what might be observed …

"**Definitive scripts are neutral wrt agent's views & privileges**"

Is the DoNaLD room an object in the class-based OOP sense? 2

Circumscription creates objects

BUT

a definitive script merely reflects observed latent transformations

Comprehending / designing an object = knowing / determining everything we can do with it

BUT

definitive script doesn't circumscribe the family of transformations that we can apply

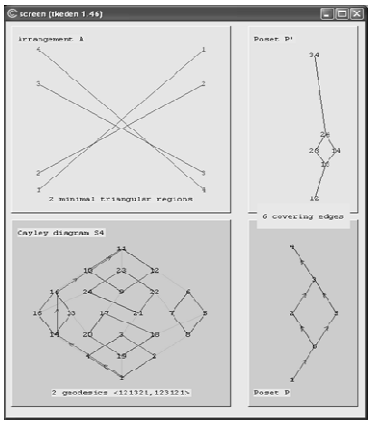
From logic to experience

- the computer enables us to use logical constructs to specify relationships that admit reliable interpretations and support robust physical realisations
- human skill and discretion plays a crucial role in crafting ritualisable experiences
- NB classical computer science doesn't take explicit account of robust physical realisations or ritualisable experience

From experience to logic?

- open-ended interaction with what is experienced is a means to representing with a high degree of realism and subtlety (cf. the strained representation of observables in the Miranda 3D OXO)
- mathematical concepts such as abstract lines as “realised” in this fashion

linesBeynon1991



The linesBeynon1991 script …

```
real sc,size
point O,Oa,Ob
O,Oa,Ob=(500,500),O-(size,size),O+(size,size)
size,sc = 380.0,2*size
real a12,a23,a34,b12,b23,b34
point A1,A2,A3,A4,B1,B2,B3,B4
line l1,l2,l3,l4
# a12 and b12 determine the distances between the L and R ends of lines 1 and 2
a12=1.0
a23=5.0
a34=2.0
b12=2.0
b23=5.0
b34=1.0
# A1 and B1 are the left and right endpoints of line 1
A1=Oa
A2=Oa+(0, a12 div (a12+a23+a34))*sc
A3=Ob+(0,(a12+a23) div (a12+a23+a34))*sc
A4=Ob+(0,(a12+a23+a34) div (a12+a23+a34))*sc
B1=Ob
B2=Ob-(0,(b12) div (b12+b23+b34))*sc
B3=Ob-(0,(b12+b23)) div (b12+b23+b34))*sc
B4=Ob-(0,(b12+b23+b34) div (b12+b23+b34))*sc
l1=[A1,B1]
l2=[A2,B2]
l3=[A3,B3]
l4=[A4,B4]

label j1,j2,j3,j4,k1,k2,k3,k4
j1 = label("1",A1)
j2 = label("2",A2)
j3 = label("3",A3)
j4 = label("4",A4)
k1 = label("1",B1)
k2 = label("2",B2)
k3 = label("3",B3)
k4 = label("4",B4)

# this is the DoNaLD script that defines Figure 1(b) in the viewport "POSET"
viewport POSET

# r12 determines the LR position of the line 1,2 intersection
# r12 is to be the crossing index of the line 1,2 intersection
#       eg this is 1 if lines 1 and 2 are the top pair at their pt of Xn
real r12,r23,r34,r13,r24,r14
int a12,a23,a34,a13,a24,a14
r12,r23,r34 = a12 div b12 , a23 div b23, a34 div b34
r13 = (a12+a23) div (b12+b23)
r24 = (a23+a34) div (b23-a34)
r14 = (a12+a23+a34) div (b12+b23+b34)

# z123 = 1 if line 1 crosses line 2 before line 3 crosses line 2 in LR order
int z123,z124,z134,z234
int Z123,Z124,Z134,Z234
z123 = if r12<r23 then 1 else 0
z124 = if r12<r24 then 1 else 0
z134 = if r13<r34 then 1 else 0
z234 = if r23<r34 then 1 else 0
Z123 = if r13<r12 then 1 else 0
Z124 = if r14<r12 then 1 else 0
Z134 = if r14<r13 then 1 else 0
Z234 = if r24<r23 then 1 else 0

# s12 calcs the crossing index for lines 1 and 2
# "by default" this is 1 but is incremented  if line 3 or 4 crosses line 1
# before line 2 crosses line 1 in LR order
x12 = 1+Z123+Z124
x13 = 1+(1-Z123)+Z134
x14 = 1+(1-Z124)+(1-Z134)
x23 = 2-z123-z234
x24 = 2-z124+(1-Z234)
x34 = 3-z134-z234
```

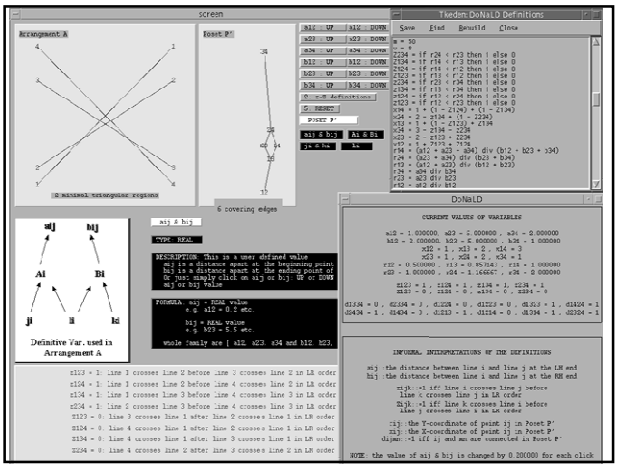
```
# these are the points of the poset of intersections
# v is a vertical, m a global magnification factor
int v,m
v,m=8,50
point orig,p12,p23,p34,p13,p24,p14
p12 = orig+[v*12,r12*v]*m
p23 = orig+[v*23,r23*v]*m
p34 = orig+[v*24,r24*v]*m
p34 = orig+[v*34,r34*v]*m
p14 = orig+[v*14,r14*v]*m
p13 = orig+[v*13,r13*v]*m
orig = [400,400]

# Line l1213 occurs in the poset if the intersection of lines 1 and 2
# and the intersection of lines 1 and 3 corresponds to a covering edge
# Line l1213 is present if the parameter d1213 evaluates to 1
# it otherwise contracts to the origin
line l1213,l1214,l1314,l2324,l1223,l1224,l1334,l2334,l1323,l1424,l2434,l1434
int d1213,d1214,d1314,d2324,d1223,d1224,d1334,d2334,d1323,d1424,d2434,d1434
l1214 = [p12*p1214,p14*d1214]
l1323 = [p13*d1323,p23*d1323]
l1224 = [p12*d1224,p24*d1224]
l1424 = [p14*d1424,p24*d1424]
l2324 = [p23*d2324,p24*d2324]
l1334 = [p13*d1334,p34*d1334]
l1213 = [p12*d1213,p13*d1213]
l1434 = [p14*d1434,p34*d1434]
l1314 = [p13*d1314,p14*d1314]
l1223 = [p12*d1223,p23*d1223]
l2434 = [p24*d2434,p34*d2434]
l2334 = [p23*d2334,p34*d2334]

# d1213 is 1 if the crossing index of lines 1 and 2 differs from that of
# lines 1 and 3  and line 4 doesn't cross line 1 between its points
# of intersection with lines 2 and 3
d1334 = if !(x13=x34) && ((r23<r13)^(r23<r34)=0) then 1 else 0
d2334 = if !(x23=x34) && ((r13<r23)^(r13<r34)=0) then 1 else 0
d1224 = if !(x12=x24) && ((r23<r12)^(r23<r24)=0) then 1 else 0
d1223 = if !(x12=x23) && ((r24<r12)^(r24<r23)=0) then 1 else 0
d1323 = if !(x13=x23) && ((r34<r13)^(r34<r23)=0) then 1 else 0
d1424 = if !(x14=x24) && ((r34<r14)^(r34<r24)=0) then 1 else 0
d2434 = if !(x24=x34) && ((r14<r24)^(r14<r34)=0) then 1 else 0
d1434 = if !(x14=x34) && ((r24<r14)^(r24<r34)=0) then 1 else 0
d1213 = if !(x12=x13) && ((r14<r12)^(r14<r13)=0) then 1 else 0
d1214 = if !(x12=x14) && ((r13<r12)^(r13<r14)=0) then 1 else 0
d1314 = if !(x13=x14) && ((r12<r13)^(r12<r14)=0) then 1 else 0
d2324 = if !(x23=x24) && ((r12<r23)^(r12<r24)=0) then 1 else 0
```

```
int numcovedge
numcovedge = d1213+d1223+d1224+d1214+d1314+d1334+d1323+d2324+d2334+d2434+d1434+df 424

int gl213,gl214,gl314,gl2324,gl1223,gl1224,gl1334,gl2334,gl1323,gl1424,,gl2434,gl1434
gl 213 = if (((p12.1-p13.1)^(p12.2-p13.2)) < 0) then 0 else 1
gl 214 = if (((p12.1-p14.1)^(p12.2-p14.2)) < 0) then 0 else 1
gl 314 = if (((p13.1-p14.1)^(p13.2-p14.2)) < 0) then 0 else 1
gl2324 = if (((p23.1-p24.1)^(p23.2-p24.2)) < 0) then 0 else 1
gl 223 = if (((p12.1-p23.1)^(p12.2-p23.2)) < 0) then 0 else 1
gl 224 = if (((p12.1-p24.1)^(p12.2-p24.2)) < 0) then 0 else 1
gl 334 = if (((p13.1-p34.1)^(p13.2-p34.2)) < 0) then 0 else 1
gl2334 = if (((p23.1-p34.1)^(p23.2-p34.2)) < 0) then 0 else 1
gl 323 = if (((p13.1-p23.1)^(p13.2-p23.2)) < 0) then 0 else 1
gl 424 = if (((p14.1-p24.1)^(p14.2-p24.2)) < 0) then 0 else 1
gl2434 = if (((p24.1-p34.1)^(p24.2-p34.2)) < 0) then 0 else 1
gl 434 = if (((p14.1-p34.1)^(p14.2-p34.2)) < 0) then 0 else 1

int r1213,r1214,r1314,r2324,r1223,r1224,r1334,r2334,r1323,r1424,r2434,r1434
r1213 = if (r12-r13) < 0) then 0 else 1
r1214 = if (r12-r14) < 0) then 0 else 1
r1314 = if (r13-r14) < 0) then 0 else 1
r2324 = if (r23-r24) < 0) then 0 else 1
r1223 = if (r12-r23) < 0) then 0 else 1
r1224 = if (r12-r24) < 0) then 0 else 1
r1334 = if (r13-r34) < 0) then 0 else 1
r2334 = if (r23-r34) < 0) then 0 else 1
r1323 = if (r13-r23) < 0) then 0 else 1
r1424 = if (r14-r24) < 0) then 0 else 1
r2434 = if (r24-r34) < 0) then 0 else 1
r1434 = if (r14-r34) < 0) then 0 else 1
```

```
int Ul 213,Ul 214,Ul 314,U2324,Ul 223,Ul 224,U1334,U2334,Ul 323,Ul 424,U2434,Ul 434
Ul 213 = g1213 * r1213 * d1213
Ul 214 = g1214 * r1214 * d1214
Ul 314 = g1314 * r1314 * d1314
U2324 = g2324 * r2324 * d2324
Ul 223 = g1223 * r1223 * d1223
Ul 224 = g1224 * r1224 * d1224
Ul 334 = g1334 * r1334 * d1334
U2334 = g2334 * r2334 * d2334
Ul 323 = g1323 * r1323 * d1323
Ul 424 = g1424 * r1424 * d1424
Ul 434 = g1434 * r1434 * d1434
U2434 = g2434 * r2434 * d2434

int ul 213,ul 214,ul 314,u2324,ul223,ul 224,ul1334,u2334,ul 323,ul 424,,u2434,u1434
ul 213 = g1213 * (1-r1213) * dl213
ul 214 = g1214 * (1-r1214) * dl214
ul 314 = g1314 * (1-r1314) * dl314
u2324 = g2324 * (1-r2324) * d2324
ul 223 = g1223 * (1-r1223) * dl223
ul 224 = g1224 * (1-r1224) * dl224
ul 334 = g1334 * (1-r1334) * dl334
u2334 = g2334 * (1-r2334) * d2334
ul 323 = g1323 * (1-r1323) * dl323
ul 424 = g1424 * (1-r1424) * dl424
ul 434 = g1434 * (1-r1434) * dl434
u2434 = g2434 * (1-r2434) * d2434

int vl 213,vl 214,vl 314,vl2324,vl 223,vl 224,vl1334,vl2334,vl 323,vl 424,vl2434,vl 434
vl 213 = (1-gl213) * (1-r1213) * dl 213
vl 214 = (1-gl214) * (1-r1214) * dl 214
vl 314 = (1-gl314) * (1-r1314) * dl 314
vl2324 = (1-gl2324) * (1-r2324) * dl2324
vl 223 = (1-gl223) * (1-r1223) * dl 223
vl 224 = (1-gl224) * (1-r1224) * dl 224
vl 334 = (1-gl334) * (1-r1334) * dl 334
vl2334 = (1-gl2334) * (1-r2334) * dl2334
vl 323 = (1-gl323) * (1-r1323) * dl 323
vl 424 = (1-gl424) * (1-r1424) * dl 424
vl 434 = (1-gl434) * (1-r1434) * dl 434
vl2434 = (1-gl2434) * (1-r2434) * dl2434

int Vl 213,Vl 214,Vl 314,V2324,Vl 223,Vl 224,Vl 334,V2334,Vl 323,Vl 424,V2434,Vl 434
Vl 213 = (1-gl213) * r1213 * dl 213
Vl 214 = (1-gl214) * r1214 * dl 214
Vl 314 = (1-gl314) * r1314 * dl 314
V2324 = (1-gl2324) * r2324 * dl2324
Vl 223 = (1-gl223) * r1223 * dl 223
Vl 224 = (1-gl224) * r1224 * dl 224
Vl 334 = (1-gl334) * r1334 * dl 334
V2334 = (1-gl2334) * r2334 * dl2334
Vl 323 = (1-gl323) * r1323 * dl 323
Vl 424 = (1-gl424) * r1424 * dl 424
Vl 434 = (1-gl434) * r1434 * dl 434
V2434 = (1-gl2434) * r2434 * dl2434

%donald
int in12, in23, in24, in34, in14, int 3
in13 = Ul314+Ul1334+Ul323+ul1213+vl213+Vl 334+Vl 323+Vl 314
in23 = U2324+U2334+ul223+ul1323+vl1323+vl 223+V2324+V2334
in24 = U2434+ul2324+ul1224+ul1424+V2434+v2324+vl1224+vl424
in34 = ul1334+ul2334+ul1434+ul2434+vl1334+vl2334+vl1434+vl2434
in14 = Ul424+Ul434+ul1214+ul1314+Vl424+Vl 434+vl 214+vl 314
in12 = Ul213+Ul214+Ul223+Ul224+Vl224+Vl223+Vl213+Vl214

int Vl 213,Vl 214,Vl 314,V2324,Vl 223,Vl 224,Vl 334,V2334,Vl 323,Vl 424,V2434,Vl 434
Vl 213 = (1-gl213) * r1213 * dl 213
Vl 214 = (1-gl214) * r1214 * dl 214
Vl 314 = (1-gl314) * r1314 * dl 314
V2324 = (1-gl2324) * r2324 * dl2324
Vl 223 = (1-gl223) * r1223 * dl 223
Vl 224 = (1-gl224) * r1224 * dl 224
Vl 334 = (1-gl334) * r1334 * dl 334
V2334 = (1-gl2334) * r2334 * dl2334
Vl 323 = (1-gl323) * r1323 * dl 323
Vl 424 = (1-gl424) * r1424 * dl 424
Vl 434 = (1-gl434) * r1434 * dl 434
V2434 = (1-gl2434) * r2434 * dl2434

%donald
int in12, in23, in24, in34, in14, int 3
in13 = Ul314+Ul1334+Ul323+ul1213+vl213+Vl 334+Vl 323+Vl 314
in23 = U2324+U2334+ul223+ul1323+vl1323+vl 223+V2324+V2334
in24 = U2434+ul2324+ul1224+ul1424+V2434+v2324+vl1224+vl424
in34 = ul1334+ul2334+ul1434+ul2434+vl1334+vl2334+vl1434+vl2434
in14 = Ul424+Ul434+ul1214+ul1314+Vl424+Vl 434+vl 214+vl 314
in12 = Ul213+Ul214+Ul223+Ul224+Vl224+Vl223+Vl213+Vl214
```

Interesting comparisons …

- the lines script as not object-oriented – most of its core observables are associated with relationships that cannot be identified with any single object
- the lines script as resembling a functional programming script in its homogeneity (“all definitions”), but associated with directly accessible external observables …

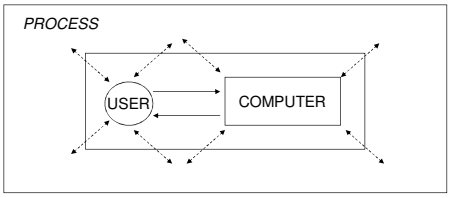


Features of the lines model …

- directly accessible external observables: z123 = 1 means that line 1 crosses line 2 before line 3 crosses line 2 in L-to-R order
- the ideal geometry as associated with a mode of interaction with the model (subject to being able to enhance the accuracy of arithmetic indefinitely on-the-fly)

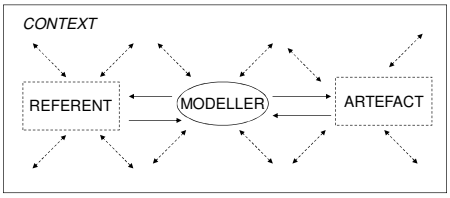
Programming from two perspectives

- a program is conceived with reference to how its behaviour participates in a wider process with functional objectives: states emerge as the side-effects of behaviours
- a computer artefact is developed so as to reflect the agency within an environment: the artefact and environment evolve until (possibly) program-like processes emerge



Conventional programs as embedded in *processes* of interaction with the world

Programs are understood in relation to processes in their surrounding environment



Artefacts and their referents as sculpted out of open interaction with the world

States of the referent and the artefact are connected through experience of interacting with the referent and the artefact

… but this presents some philosophical challenges …

An EM perspective on programming … … some problematic issues

In focusing on current state-as-experienced, we have some problems to resolve:

- Behaviour raises questions about agency: what is the status of a “computer” action?
- How do we deal with state-as-experienced in semantic terms?
- How do we make science of activities in which human interpretation is so critical?