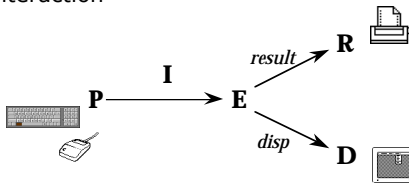


the PIE model

- 'minimal' model of interactive system
- focused on external observable aspects of interaction



PIE model - user input

- sequence of commands
- commands include:
 - keyboard, mouse movement, mouse click



- call the set of commands C
- call the sequence P
 - P = seq C

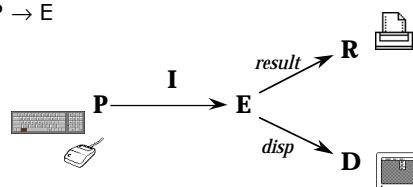
PIE model - system response

- the 'effect'
- effect composed of:
 - ephemeral display
 - the final result
 - (e..g printout, changed file)
- call the set of effects E

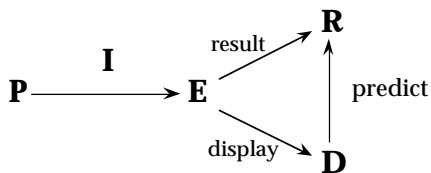


PIE model - the connection

- given any history of commands (P)
- there is some current effect
- call the mapping the interpretation (I)
 - I: P → E



properties - WYSIWYG



$\exists \text{ predict} \in (D \rightarrow R) \text{ s.t. } \text{predict} \circ \text{display} = \text{result}$

- but really not quite the full meaning

state

- avoided to focus external aspects ...
 - ... but often easier to think about!
- but can add it back
 - either 'create' it from I and E
 - or assume E is actually a state

creating the state!

- given a PIE
- say two command sequences as equivalent if they are indistinguishable in the future:
 $p \sim q \text{ iff } \forall r \in H: I(pr) = I(qr)$
- the quotient set of P is a minimal state allowing the same effects

N.B. change notation use H (history) instead of P for sequence of commands

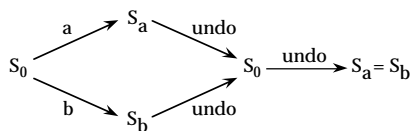
property of state

- for E to be a state it needs a state update function – call it 'doit':
 $doit : E \times C \rightarrow E$
- this needs to 'agree' with I:
 $\forall p \in H, c \in C: doit(I(p),c) = I(pc)$
- if E acts as a state we'll call it S with the initial state $s_0 = I(<>)$

proving things – undo

$\forall c: c \text{ undo} \sim \text{null} ?$

only for $c \neq \text{undo}$



lesson

- undo is no ordinary command!
- other meta-commands:
 back/forward in browsers
 history window

undo as meta-command

- need to think of ordinary commands C plus augmented commands A:
 $C^a = C \cup A \quad H^a = \text{seq } C^a$
- also augmented system state:
 S^a
- and behaviour:
 $doit^a: S^a \times C^a \rightarrow S^a$
 $I^a: H^a \rightarrow S^a$

two state (flip) undo

- system keeps two copies of state:
 $S^a = S \times S$
- ordinary commands update state:
 $doit^a((s_{save}, s), c) = (s, doit(s, c))$
- undo (redo) flips states:
 $doit^a((s_{save}, s), \text{undo}) = (s, s_{save})$

the real system inside

- the augmented system still needs to be the old system inside!
- link new and old with projection:
proj: $S^a \rightarrow S$
- proj(s) is the old state 'inside'

projected state of flip undo ..

$$S^a = S \times S$$

- projected state simply second component:

$$\text{proj}^a ((s_{\text{save}}, s)) = s$$

properties of flip undo

- undo really reverses undo:
undo undo \sim null (strong-uu)
- undo reverses ordinary commands on projected (original) state
c undo \sim_{proj} null (weak-cu)

$$\forall s^a \in S^a : \text{proj}(\text{doit}^a (s^a, c \text{ undo})) = \text{proj}(s^a)$$

stack (multistep) undo/redo

- augmented state is a stack of states:

$$S^m = S^+ \times \text{Nat}$$

$$s^{m_0} = \langle \langle s_0 \rangle, 1 \rangle$$

$$\text{proj}^m : S^m \rightarrow S$$

where $\text{proj}^m(\langle \text{hs}, n \rangle) = \text{hs}[n]$

stack (multistep) undo/redo

- obvious (!) update:

$$\text{doit}^m(\langle \text{hs}, n \rangle, \text{undo}) = \langle \text{hs}, n-1 \rangle$$

if $n > 1$ - otherwise nothing

$$\text{doit}^m(\langle \text{hs}, n \rangle, \text{redo}) = \langle \text{hs}, n+1 \rangle$$

if $n < \text{length}(\text{hs})$

$$\text{doit}^m(\langle \text{hs}, n \rangle, c) = \langle \langle \text{hs}[1..n], \text{doit}(\text{hs}[n], c) \rangle, n+1 \rangle$$

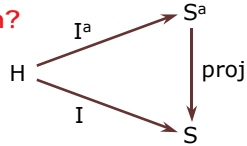
properties of multistep undo/redo

- redo really reverses undo:
undo redo \sim null (strong-cu)
- undo reverses commands on projected (original) state
c undo \sim_{proj} null (weak-cu)
- the only way to satisfy these ...
Prove It !

the real system inside (2)

- behaviour?
- if no A commands ever used, identical:
 $\forall h \in H : \text{proj}(I^a(h)) = I(h)$

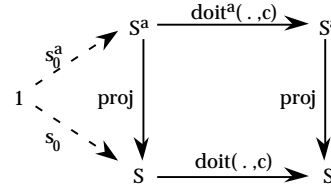
enough?



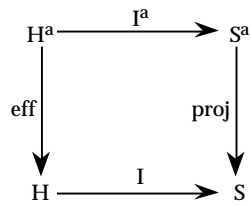
conservativeness of state

$$\text{proj}(s_0^a) = s_0$$

$$\forall c \in C, s \in S^a : \text{proj}(\text{doit}^a(s, c)) = \text{doit}(\text{proj}(s), c)$$



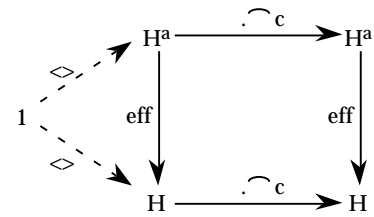
encapsulation



conservativeness of effective history

$$\text{eff}(\diamond) = \diamond$$

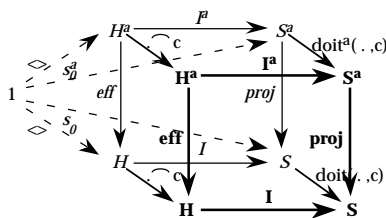
$$\forall c \in C, h \in H^a : \text{eff}(h \wedge c) = \text{eff}(h) \wedge c$$



the cube

$$\text{eff}(\diamond) = \diamond$$

$$\forall c \in C, h \in H^a : \text{eff}(h \wedge c) = \text{eff}(h) \wedge c$$



full details ...

R. Mancini (1997).
Modelling Interactive Computing by exploiting the Undo.
 Dottorato di Ricerca in Informatica, IX-97-5,
 Università degli Studi di Roma "La Sapienza"