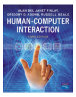
harnessing the power of formalism for understanding interaction

Alan Dix

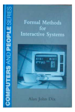
Lancaster University, UK
www.hiraeth.com/alan/tutorials/formal

sources

- Upside down ∀s and algorithms - computational formalisms and theory. In *HCI Models, Theories, and Frameworks: Toward an Multidisciplinary Science*. John Carroll (ed.). Morgan Kaufman, 2003. pp. 381-429
- A. Dix, J. Finlay, G. Abowd and R. Beale (2004). Chapters 16, 17 & 18. In *Human-Computer Interaction, third edition*. Prentice Hall.



- A. J. Dix (1991). Formal Methods for Interactive Systems. Academic Press



for i

green letters tumble against black glass and dim pizza filled rooms tremble with heavy intonations, fingers drum whilst a single screen reflects a bespectacled face on plastic rimmed spectacles, seeing clearly four eyes doubled and redoubled by interactions of photons, words form from the void within

for i =

it is done

language is the ultimate formalisation tying patterns of electrical and chemical activation, spaghetti wrapped neurons, discretised to token sounds, virtuosity to vocabulary; in writing digitised as fingers scratch ink upon parchment or softly caress smooth worn key tops

before I continue let us reflect, for i can only tell my story, but the words are our own, for eye to eye and voice to voice the tokens were formed, even though sheet to sheet or screen to screen we use them now

and we have found ways to bare our soul and transport our listeners through simple words, and to, in turn, reflect and talk about the talking, formalising the understandings we have about words in words

the hard edged symbols cut upon stone, dark text stamped from lead, and pixelated poetry touch our very heart

is it surprising that silicon and liquid crystal should be no less richly understood

outline

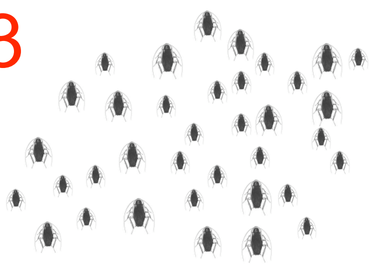
- setting the scene
  - what is formal?
  - types of formalism
  - first examples
  - placemat maths
- models of systems
  - dialogue notations
  - modelling state
  - generic models of interaction
- why do it?
  - it works! - a formal methods success story
  - formal futures - ubiquity and physicality

what is formal?

- dinner jacket and bow tie?
  - outward appearance of things – the form
- in maths and computing …
  - representations (diagrams, formulae, etc.)
    - analysed and manipulated separate from meaning
  - how?
    - faithfully encapsulate significant aspects of meaning

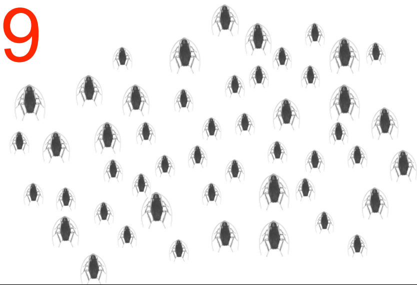
counting cockroaches - first night

213



counting cockroaches - second night

279



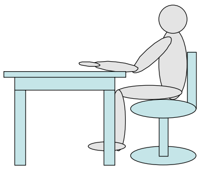
which night had more?

- second night: 279 > 213
- how can you be certain?
  - count faithfully represents significant feature
- but not everything …
  - cockroaches on first night may be:
  - bigger, different colour, more friendly

representing things absent

- symbols, icons, words
  - stand for things not present
- simulated screen shots
  - represent the unrealised designs
    (N.B. no dynamics – limited meaning)
- counting cockroaches
  - keep in a jam jar? disrupts the world
  - numbers make the impossible possible

placeholders



- homunculus – any person
  - not just someone, anyone
- maths: $\forall$ n: n+1 > n
  - saying an infinite amount
- counting: 279 > 213
  - cockroaches, apples, llamas

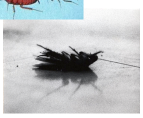
abstraction

- increasing abstraction
  - screenshot – one screen
  - storyboard – single sequence of interaction
  - navigation diagram – potential paths
- and further ...
  - work on UNDO
  - any system with particular properties ...

forcing you to think

when you count cockroaches
you have to decide
what counts as a cockroach

baby or adult



live or dead

the myth of informality

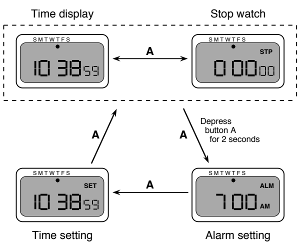
- spiritus mundi
  - formality, precision
    = reductionism, positivism = BAD
- focus (rightly) on
  - context, situatedness, contingency
- BOTH needed
  - the world is rich and complex
  - but computers are formal (as is language)
  - key is choosing the right absteractions
  - and knowing what is left out

early examples

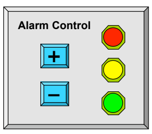
formalism in action

digital watch – user instructions

- two main modes
- limited interface
  - 3 buttons
- button A
  changes mode
- state transition
  network (STN)

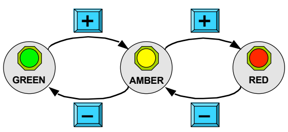


example - nuclear control



- what happens if we press '+' in red mode?

N.B. question from form only

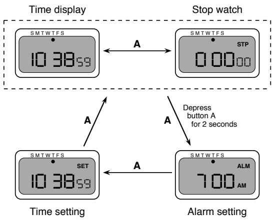


digital watch – user instructions

"depress button A
for 2 seconds"

so ...

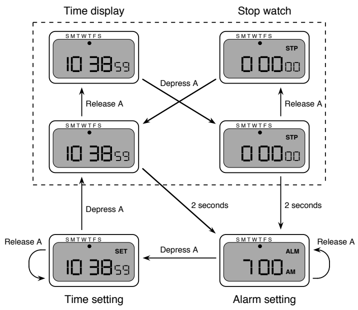
- time important
- distinguish depress A and release A



designer's instructions

and ...

that's just
one button



lessons

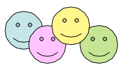
- formal analysis
  – ask questions based on *form* of diagrams
- early analysis
  – catch problems even before prototyping
- lack of bias
  – usually test what we expect, analysis breaks this
- alternative perspective
  – different representations show different things
- forcing deisgn decisions
  – did watch designer make these decsions or programmer?

using formalism in HCI

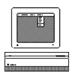
from cognitive models
to placemats

what to model

- users
  - cognitive models
  - task models



- system
  - behaviour
  - architectural structure



- world
  - domain models

notations

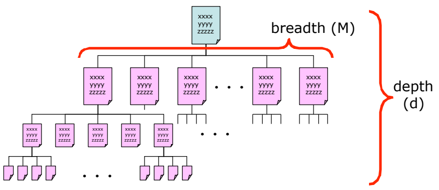
- graphical
  - digital watch STNs, Petri Nets, CTT, UML
- textual
  - production rules (used in UIMS and cog. models)
  - mathematical formulae, process algebras
- plain old sums
  - back of the envelope/placemat calculations

placemat math - menu sizes

- on-screen menus
  - e.g. web site navigation
- how many items per screen?
- frequent misapplication of Miller 7±2
- but how many is right?

placemat math (ii)

- menu tree has N items
- number of items per screen = M (breadth)
- depth (d) = $\log_2(N) / \log_2(M)$



placemat math (iii)

T_{total} – time to find an item

$= (T_{display} + T_{select}) \times d$

$T_{display}$ – time to display screen (fixed)

T_{select} – time to select menu item

$= A + B \log(M)$ (Fitts' Law)

$T_{total} = (T_{display} + A + B \log(M)) \times \log(N) / \log(M)$

cancel

$= ((T_{display} + A) \times \log(N)) / \log(M) + B \log(N)$

best menu size?

$T_{total} = ((T_{display} + A) \times \log(N)) / \log(M) + B \log(N)$

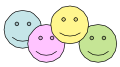
- larger M means shorter total time
- the bigger the better!

N.B. other factors

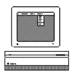
- visual search (linear if not expert)
- error rates
- minimum selectable size
- effective organisation of menu items

what to model

- users
  - cognitive models
  - task models



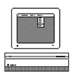
- system
  - behaviour
  - architectural structure



- world
  - domain models

what to model

- users
  - cognitive models
  - task models
- system
  - behaviour
  - architectural structure



- world
  - domain models

types of system model

- dialogue – main modes
- full state definition
- abstract interaction model

$\left.\begin{array}{l}\text{dialogue – main modes}\\ \text{full state definition}\end{array}\right\}$ specific system

abstract interaction model — generic issues

dialogue notations

what to do when

what is dialogue?

- conversation between two or more parties
  - usually cooperative
- in user interfaces
  - refers to the *structure* of the interaction
  - syntactic level of human–computer 'conversation'
- levels
  - lexical – shape of icons, actual keys pressed
  - syntactic – order of inputs and outputs
  - semantic – effect on internal application/data

structured human dialogue

- human-computer dialogue very constrained
- some human-human dialogue formal too …

Minister: do you *man's name* take this woman …
Man: I do
Minister: do you *woman's name* take this man …
Woman: I do
Man: With this ring I thee wed
(places ring on womans finger)
Woman: With this ring I thee wed *(places ring ..)*
Minister: I now pronounce you man and wife

lessons about dialogue

- wedding service
  - sort of script for three parties
  - specifies order
  - some contributions fixed – "I do"
  - others variable – "do you *man's name* …"
  - instructions for ring
    concurrent with saying words "with this ring …"
- if you say these words are you married?
  - only if in the right place, with marriage licence
  - syntax not semantics

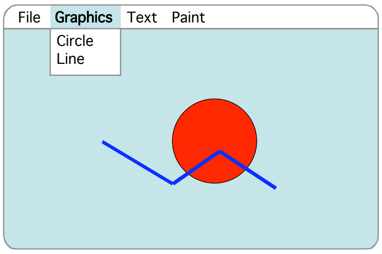
… and more

- what if woman says "I don't"?
- real dialogues often have alternatives:

Judge: How do you plead guilty or not guilty?
Defendant: *either* Guilty *or* Not guilty

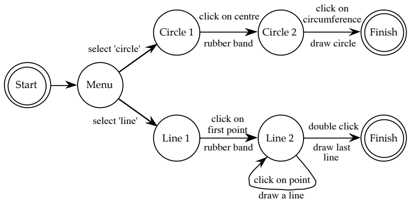
  - the process of the trial depends on the defendants response
- focus on normative responses
  - doesn't cope with judge saying "off with her head"
  - or in computer dialogue user standing on keyboard!

a simple graphics package



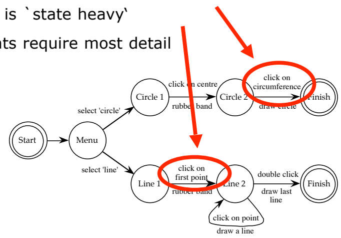
state transition networks (STN)

- circles - states
- arcs - actions/events



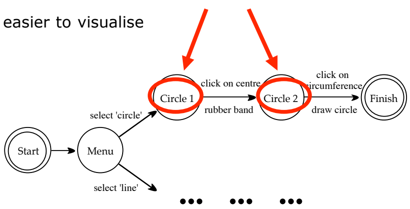
state transition networks - events

- arc labels a bit cramped because:
  - notation is `state heavy`
  - the events require most detail



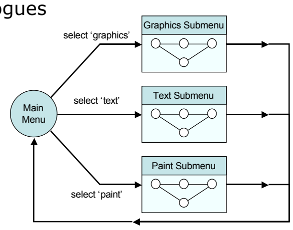
state transition networks - states

- labels in circles a bit uninformative:
  - states are hard to name
  - but easier to visualise



hierarchical STNs

- managing complex dialogues
- named sub-dialogues



action properties

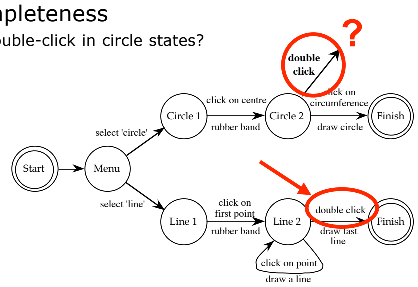
- completeness
  - missed arcs
  - unforeseen circumstances
- determinism
  - several arcs for one action
  - deliberate: application decision
  - accident: production rules
- nested escapes
- consistency
  - same action, same effect?
  - modes and visibility

state properties

- reachability
  - can you get anywhere from anywhere?
  - and how easily
- reversibility
  - can you get to the previous state?
  - but NOT undo
- dangerous states
  - some states you don't want to get to
    e.g. digital watch: time/alarm set, button press for 2 secs

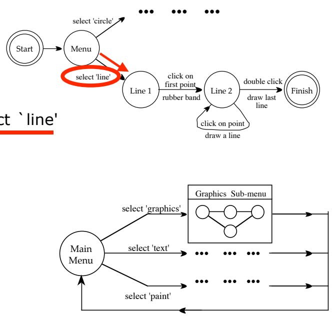
checking properties (i)

- completeness
  - double-click in circle states?



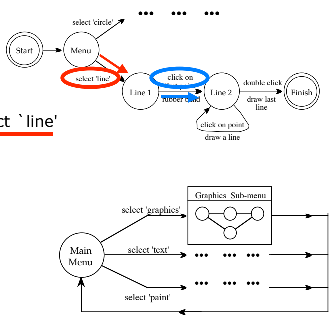
checking properties (ii)

- Reversibility:
  - to reverse select `line'

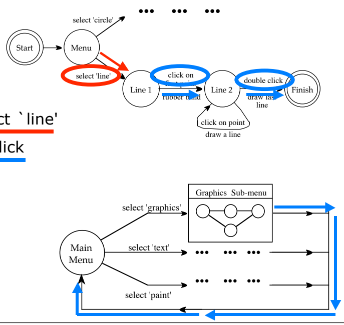


checking properties (ii)

- Reversibility:
  - to reverse select `line'
  - click

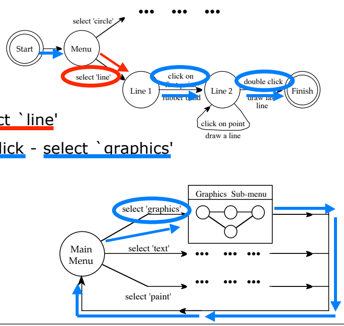


checking properties (ii)



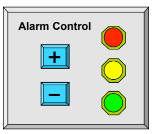
- Reversibility:
  - to reverse select `line'
  - click - double click

checking properties (ii)

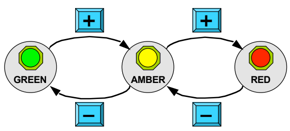


- Reversibility:
  - to reverse select `line'
  - click - double click - select `graphics'
  - (3 actions)
- N.B. not undo

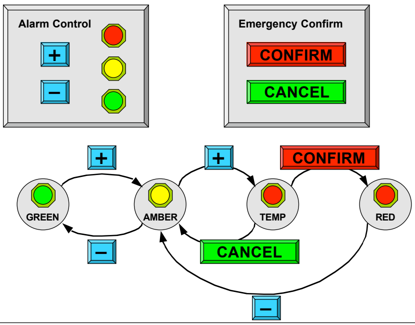
example - nuclear control



- missing arcs
- dangerous state?



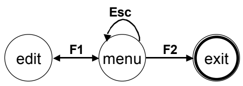
revised STN



dangerous states

- word processor: two modes and exit

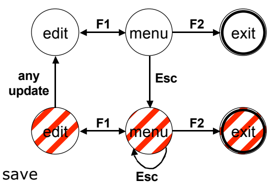
F1 - changes mode
F2 - exit (and save)
Esc - no mode change



but … Esc resets autosave

dangerous states (ii)

- exit with/without save ⇒ dangerous states
- duplicate states - semantic distinction



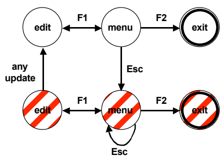
F1-F2 - exit with save
F1-Esc-F2 - exit with no save

lexical Issues

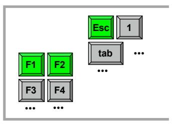
- visibility
  - differentiate modes and states
  - annotations to dialogue
- style
  - command - verb noun
  - mouse based - noun verb
- layout
  - not just appearance ...

layout matters

- word processor - dangerous states

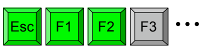


- old keyboard - OK



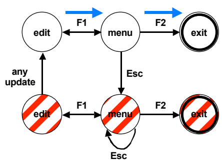
layout matters

- new keyboard layout



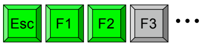
intend F1-F2 (save)

finger catches Esc



layout matters

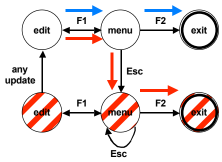
- new keyboard layout



intend F1-F2 (save)

finger catches Esc

F1-Esc-F2 - disaster!

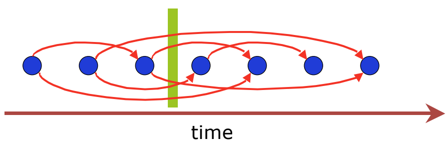


modelling state

looking within

what is state

that in the present
of that in the past
which affects that of the future



modelling state

- describe state using variables
- types of variables:
  - basic type:
    x: Nat – non-negative integer {0,1,2,...}
  - individual item from set:
    shape: {circle, line, rectangle}
  - subset of bigger set:
    selection: **set** Nat – set of integers
  - function (often finite):
    objects: Nat → shape
  - user defined:
    Point = x, y: Real – e.g. (1.79,-3.2)

stages

iterative define:

state	–	what needs to be remembered
invariants	–	what is always true
initial state	–	how it starts
actions	–	what can happen to the state (need to relate this to keys etc.)
display	–	what the user sees (hears etc.)

use scenarios to check they are what you want

four function calculator

- formal description of the state
- define the effect of the following actions:

type_digit(d)	–	user presses single digit
equals	–	user presses '=' button
op(p)	–	user presses '+','–', '*' or '/' button

N.B. will not be right first time ... spot the mistakes

calculator state - first attempt

state

total: Nat – running total (accumulator)
disp: Nat – number currently displayed

no invariants

initial state

total = 0
disp = 0

display

disp – more complex calculator may show formulae

calculator actions - first attempt

type_digit(d)
add d to the end of disp
total unchanged

equals
do last operation “+,-,*,” to disp and total
...

what is it!

calculator state - second attempt

state
total: Nat – running total (accumulator)
disp: Nat – number currently displayed
pend_op: {+,–,*,/,none} – pending operation

initial state
total = 0
disp = 0
pend_op = none

calculator actions - second attempt

type_digit(d)
add d to the end of disp
total and pend_op unchanged

equals
do pend_op to disp and total
put result in both disp and total
set pend_op to none

op(o)
do pend_op to disp and total
put result in both disp and total
put o into pend_op

calculator - scenario

- user types: 1 + 2 7 = – 3
- start after 1 + 2

action	total	disp	pend_op
	1	2	+
type_digit(7)	1	27	+
equals	28	28	none
op(–)	28	28	–
type_digit(3)	28	283	–

!!!

calculator state - third attempt

state

total: Nat – running total (accumulator)
disp: Nat – number currently displayed
pend_op: {+,–,*,/,none} – pending operation
typing: Bool – true/false flag

- added 'typing' flag
  - user in the middle of typing a number

calculator actions - third attempt

type_digit(d)

if typing then add d to the end of disp
otherwise clear disp and put d in it
also set typing to true
total and pend_op unchanged

equals and op(o):

- as before except both set typing to false

calculator - scenario revisited

- user types: 1 + 2 7 = – 3
- start after 1 + 2

action	total	disp	pend_op	typing
	1	2	+	yes
type_digit(7)				
	1	27	+	yes
equals				
	28	28	none	no
op(–)				
	28	28	–	no
type_digit(3)				
	28	3	–	yes

defining state

two problems:

- too little state
  - elements missing from specification
  - may be deliberate
    - e.g. dialogue level spec.
- too much state
  - too many states, too complex state
  - may be deliberate
    - redundancy, extensibility

too little state

- forgotten elements
  - e.g. 'typing' flag for calculator
- checking:
  - dialogue state
    - can you work out current dialogue state?
  - action specification
    - do you have enough information?
  - implicit global variables (see also later)
    - suggest state missing

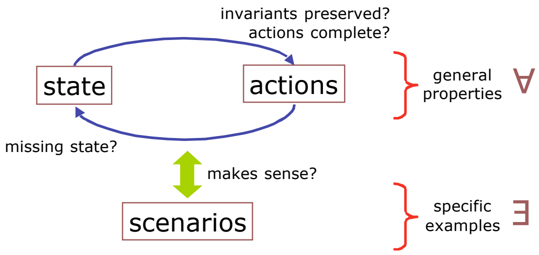
too much state

- unreachable states
  - too few actions (see later)
  - constraints
  - states are not orthogonal
- spare variables: constant/functional dependent
- dependent state
  - e.g. first point of line, number being typed
- indistinguishable states
  - what is observable?

defining actions

- framing problems
  - = too little in result state
- unreachable states – insufficient actions
- using ‘global’ variables
  - implicit in operation definition
- beware extreme cases
  - (e.g. empty document, cursor at end of line)

internal and external consistency



interaction models

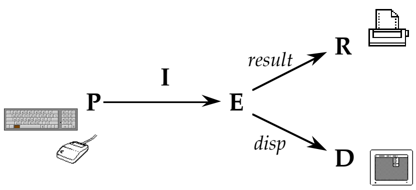
talking generally

interaction models

- generic models of classes of system
- mainly to aid understanding of general issues
- e.g. undo and 'back' button

the PIE model

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

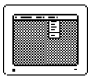


PIE model - user input

- 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

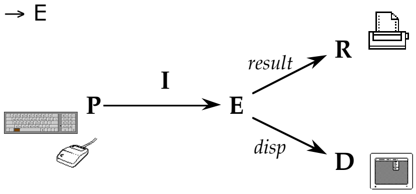


- the ‘effect’
- effect composed of:
  ephemeral display
  the final result
  - (e..g printout, changed file)

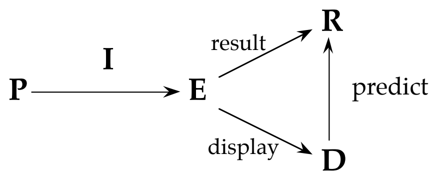
- call the set of effects E

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



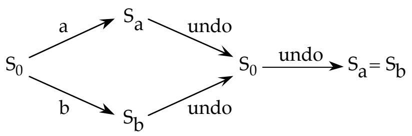
$\exists$ predict $\in$ (D $\rightarrow$ R) s.t. predict o display = result

- but really not quite the full meaning

proving things – undo

$\forall$ c : c undo ~ null ?

only for c $\neq$ undo



lesson

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

undo and history

work with Roberta Mancini, Univ. of Rome

used generic framework based on PIE ...
... the cube

proved uniqueness of certain kinds of undo

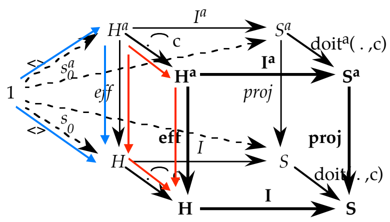
analysis of ‘back’ button and history in hypertext and web browsers

- N.B. ‘back’ was different in them all!

the cube

- $eff(\langle\rangle) = \langle\rangle$
- $\forall\, c \in C, h \in H^a : eff(h\hat{}c) = eff(h)\hat{}c$

...



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"

A. Dix and R. Mancini (1997).
Specifying history and backtracking mechanisms.
In Formal Methods in Human-Computer Interaction, Eds. P. Palanque and F. Paterno. London, Springer-Verlag. pp. 1-24.

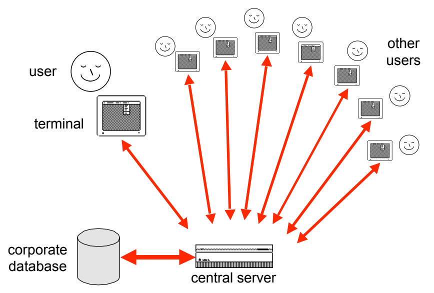
formal methods in HCI

a success story

problem

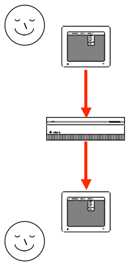
- context
  - mid 80s
  - local authority DP dept
- transaction processing
  - vast numbers of users
  - order processing, pos systems etc.
  - COBOL!
- existing programs … didn't work

TP physical architecture

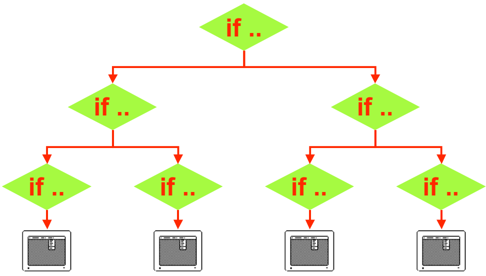


what happens

user edits form
message goes to TP engine
passed to application module
which processes the message
and prepares new screen
which is sent to the user
....



structure of programs



why?

program is trying to work out
what is happening!

- standard algorithm
  - program counter implicit
- TP, web, event-based GUI
  - need explicit dialogue state

mixed up state

- many users – one application module

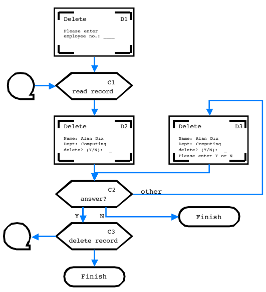
  user A starts multi-screen search list
  application stores value 'next_record'
  user B starts multi-screen search list
  application overwrites value 'next_record'
  user A selects 'next screen' ...
  ... and gets next screen of B's search!

state is hard

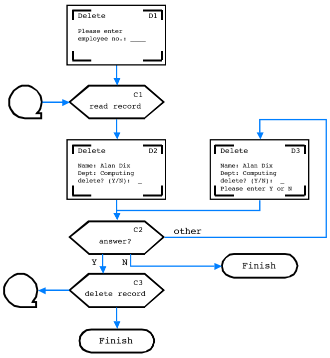
- recent MSc course
  - CS and psych
  - exercise – state of 4 function calculator
  - difficult for both
- why?
  - in real life state is implicit
  - in standard CS state is implicit too!

solution?

- flowchart!
- not of program
  ... but of dialogue
- a formal dialogue specification!

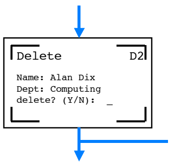


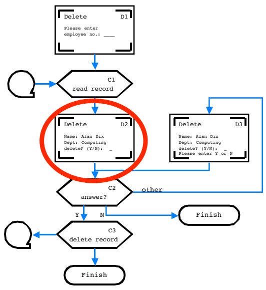
fu



details …

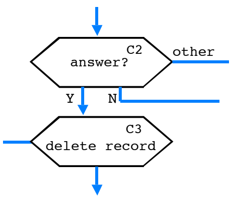
- miniature screen sketch

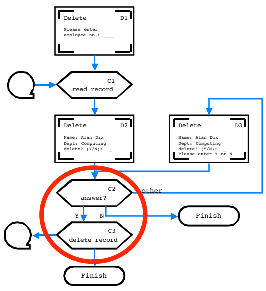




details …

- minimal internal details





and then ...

- hand transformation to boiler plate code
- store 'where next' for each terminal
  - in 'session' data
- code starts with big 'case'
- do processing
- set new 'where next' ... send screen

lessons

useful	– addresses a real problem!
communication	– mini-pictures and clear flow easy to talk through with client
complementary	– different paradigm than implementation
fast pay back	– quicker to produce application (at least 1000%)
responsive	– rapid turnaround of changes
reliability	– clear boiler plate code less error-prone
quality	– easy to establish test cycle
maintenance	– easy to relate bug/enhancement reports to specification and code

λλλ

formal futures

ubiquity and physicality

changing nature of the interface

- ubiquitous computing
  computers everywhere!
- many somple systems
  + complex interactions
- sounds like a job for ....
  formalism

an example ...

- understanding the tangible
- the physical world
  - we live in it
  - we are good at it!
  - we understand it
- properties of physicality
  - directness of effect – push and it moves
  - locality of effect – here and now
  - visibility of state – small number of relevant parameters

study the old to design the new

– work with Masitah Ghazali

- look at ordinary consumer devices
  - washing machine, light switch, personal stereo
- why?
  - we are used to using them ourselves
  - they have been 'tested' by the marketplace
  - they embody the experience of designers

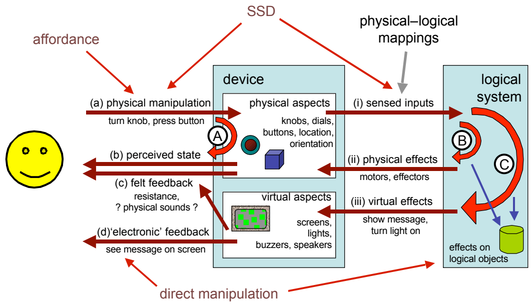
half empty?

- not the first …
  - Norman – DOET/POET
  - Thimbleby – FSM for video, microwave
- often used as HCI strawman
  - emphasise for design flaws
- we are looking for the good lessons
  - how mundane devices exploit physicality

models of AR & tangiblity

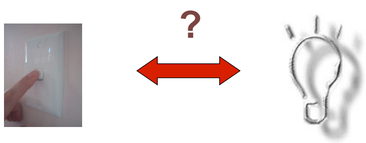
- Ullmer and Ishii – MCRpd
  - architectural interaction model
- Benford et al. – sensible/sensable/desirable
  - exploring design space
- Koleva et al. – TUI framework
  - 'coherence' between the physical and digital

physical-logical connections

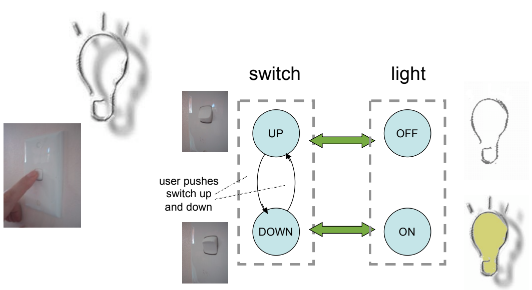


fluidity

- ‘naturalness’ of device–logical mapping



device & logical states



exposed state

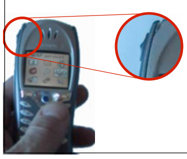
+ several **visible states of device**
+ **one-to-one** mapping to logical state
+ separate issue: is mapping clear?

hidden state

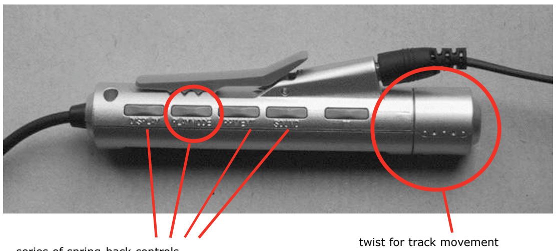
+ when **no exposed state**
+ may rely on **semantic feedback**
+ poor ‘fixes’ … LEDs, separate display
+ but sometimes necessary: too many logical states, variable number of logical states, limited space
+ transitions become more important: natural felt bumps … haptic feed back

inverse actions

+ speaker dial – exploits **natural physical inverse actions**: turn left/right
+ especially important if the user does not have a perfect knowledge of the **physical-logical mapping** unknown or mode-dependent
+ semantic **feedback** essential
+ issues: delays, obvious inverse?



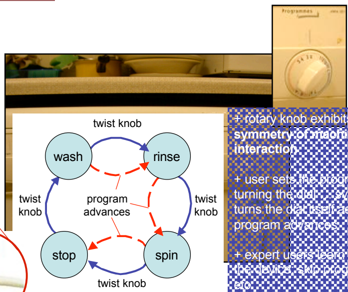
spring-back controls



series of spring-back controls
each cycle through some options
–natural inverse back/forward

twist for track movement
pull and twist for volume
– spring back
– natural inverse for twist

compliant interaction



+ rotary knob exhibits **symmetry of machine–system interaction**

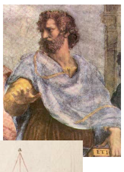
+ user sets the program by turning the dial … system also turns the dial itself as the program advances

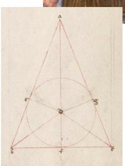
+ expert users learn to fine tune the device: skip programmes etc a brief history of formalism

from Aristotle
to Alan Turing

first steps

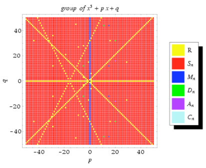
- Aristotle (384 BC - 322 BC)
  - foundations of logic
- Euclid (325 BC - 265 BC)
  - axiom, theorem and proof

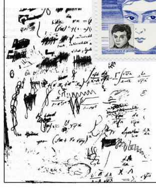


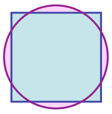


breakthrough

- Evariste Galois (1811–1832)
  - solving the quintic
  - proving the impossible
  - formalising groups



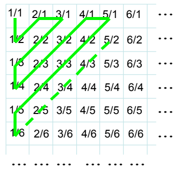




I have not time.
I have not time.
I have not time.
I have not time.
I have not time.
I have not time.
I have not time.
I have not time.

babel grows

- Georg Cantor (1845–1918)
  - foundations of set theory
  - mathematics of the infinite



- James Clerk Maxwell (1831–1879)
  - Maxwell's equations
  - unifying electricity and magnetism
  - the theory of everything

$$
\begin{aligned}
\nabla\cdot\mathbf{E} &= \frac{\rho}{\epsilon_0}\\
\nabla\times\mathbf{E} &= -\frac{\partial\mathbf{B}}{\partial t}\\
\nabla\cdot\mathbf{B} &= 0\\
\nabla\times\mathbf{B} &= \mu_0\mathbf{J}+\epsilon_0\mu_0\frac{\partial\mathbf{E}}{\partial t},
\end{aligned}
$$

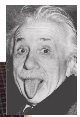
the cracks form

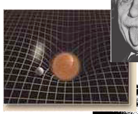
The next line is true.
The last line was false.

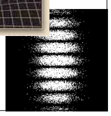
- self-reference
  - all Cretans are liars
    Epimenides the Cretan (6th century BC)
  - the Russell Paradox
    - the set that doesn't contain itself

      { X | X ∉ X }

- uncertainty at the centre
  - Einstein's relativity
  - quantum mechanics



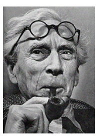




$$i\hbar\frac{\partial\Psi}{\partial t} = \hat{H}\Psi(x,t)$$

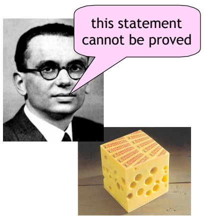
battling on

- Bertrand Russell (1872–1970)
  - Principia Mathematica
    (with Whitehead)
  - reducing mathematics to logic
  - the proof of all things

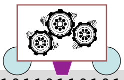


the end comes

- Kurt Gödel (1906–1978)
  - incompleteness theorem
  - mathematics is full of holes



- Alan Turing (1912–1954)
  - formal foundations of computation
  - inherent limitations of computation



0101101011011010110101101101011011010110101101101

… but

- I still expect my change
  to add up at the supermarket