Formal Methods in HCI:

Moving Towards

an Engineering Approach

Alan J. Dix

HCI Group, Dept. of Computer Science

University of York

Heslington, YORK YO1 5DD UK

alan@minster.york.ac.uk

Overview

Formal models in HCI

- why they're good ...
- and why they're bad!

Analysing dialogue descriptions

- existing part of interface design
- automatic analysis
- bridging the semantic/lexical gap

Status/event analysis

- formal underpinning
- naïve psychology
- engineering level of expertese

Formal Models of Interactive Systems

Why use formal methods

Everyone else is

- not a silly reason!
- interface can get 'left out' of the software engineering process

Intellectual control

- interfaces are complex
- context dependent → not modular
- orthogonality

Understanding

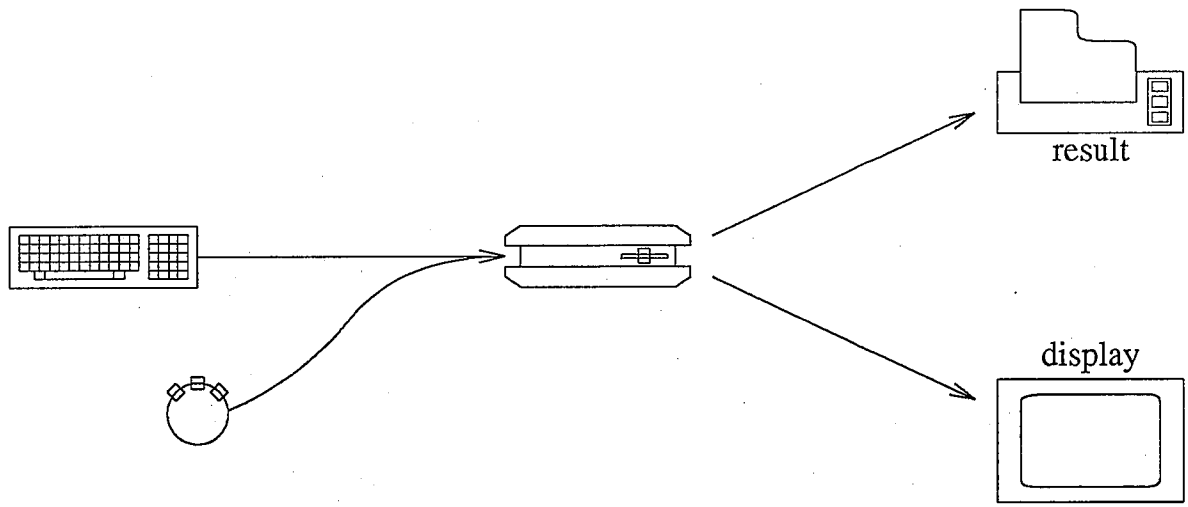
- generalisable knowledge
- specific results

But ...

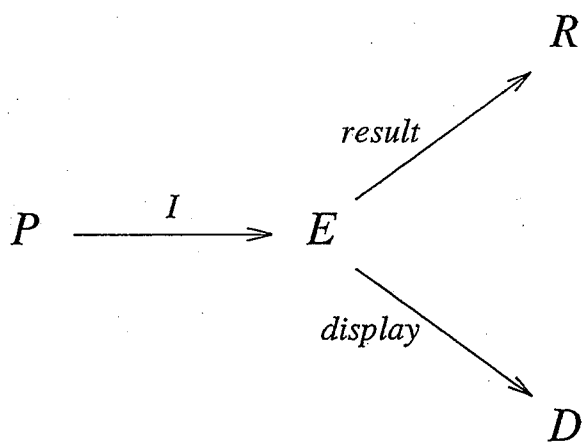
- requires considerable expertese

The PIE model

A black-box model



More formally ...



$P == \text{seq}\, C$

$I : P \rightarrow E$

$display : E \rightarrow D$

$result : E \rightarrow R$

$doit : E \times P \rightarrow E$

Reachability and undo

Reachability — getting from one state to another.

$$\forall e, e' \in E \bullet \exists p \in P \bullet doit(e, p) = e'$$

Too weak

Undo — reachability applied between current state and last state.

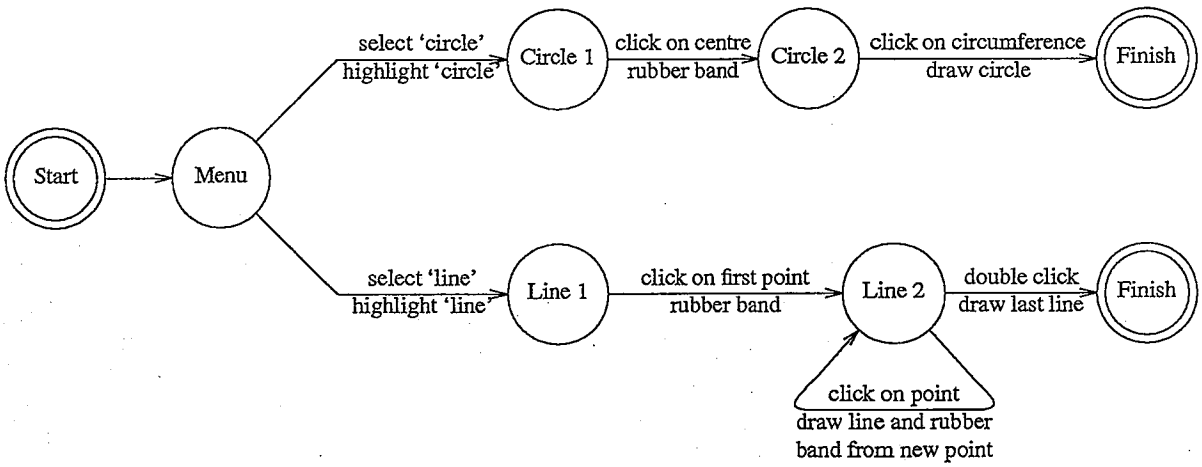
$$\forall c \in C \bullet doit(e, c \frown undo) = e$$

Impossible except for very simple system with at most two states!

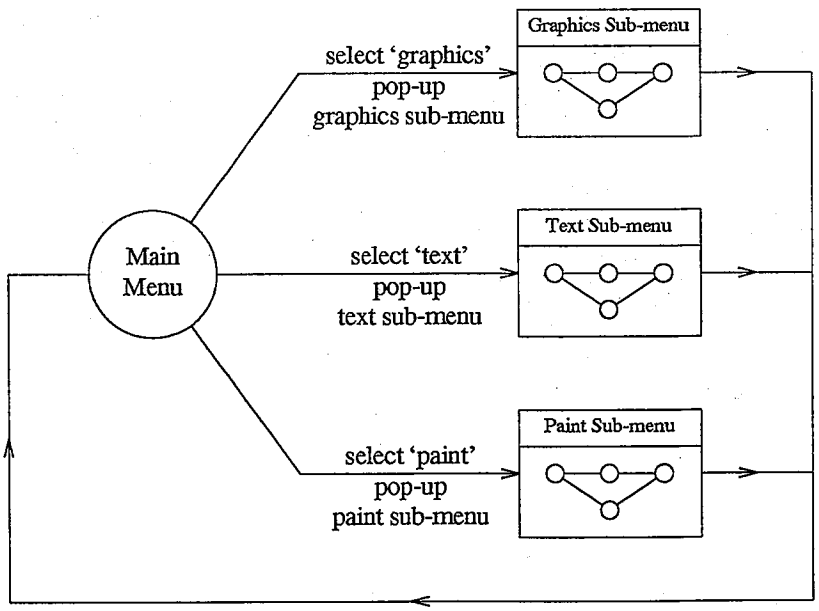
Better models of *undo* treat it as a special command to avoid this problem

Dialogue Analysis

State transition networks



circles – states, arcs – actions/events



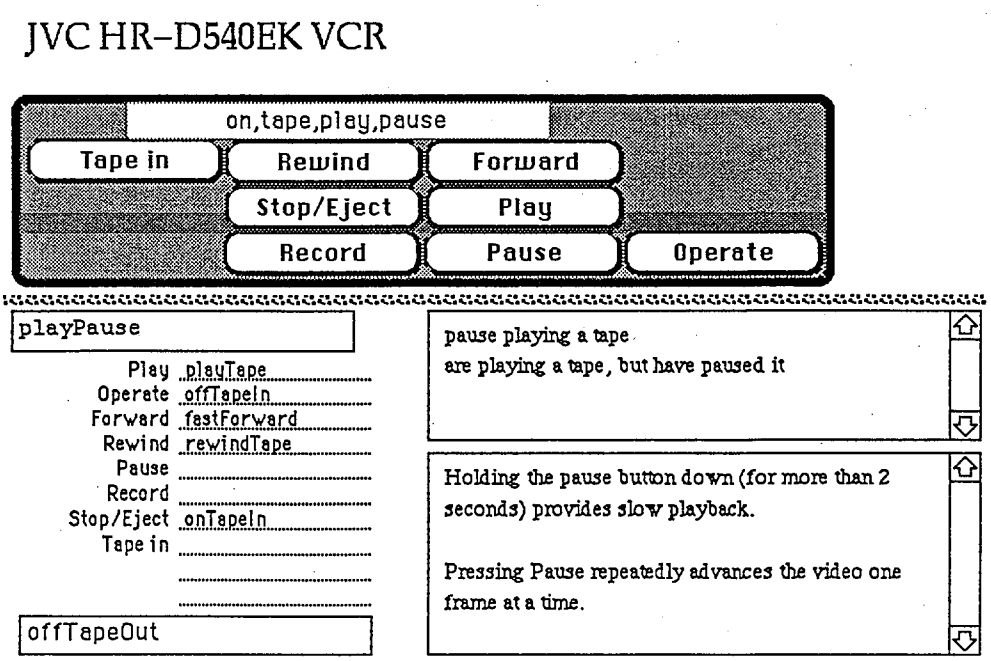
Dialogue Descriptions
Why are they used?

- UIMS
- Paper specifications

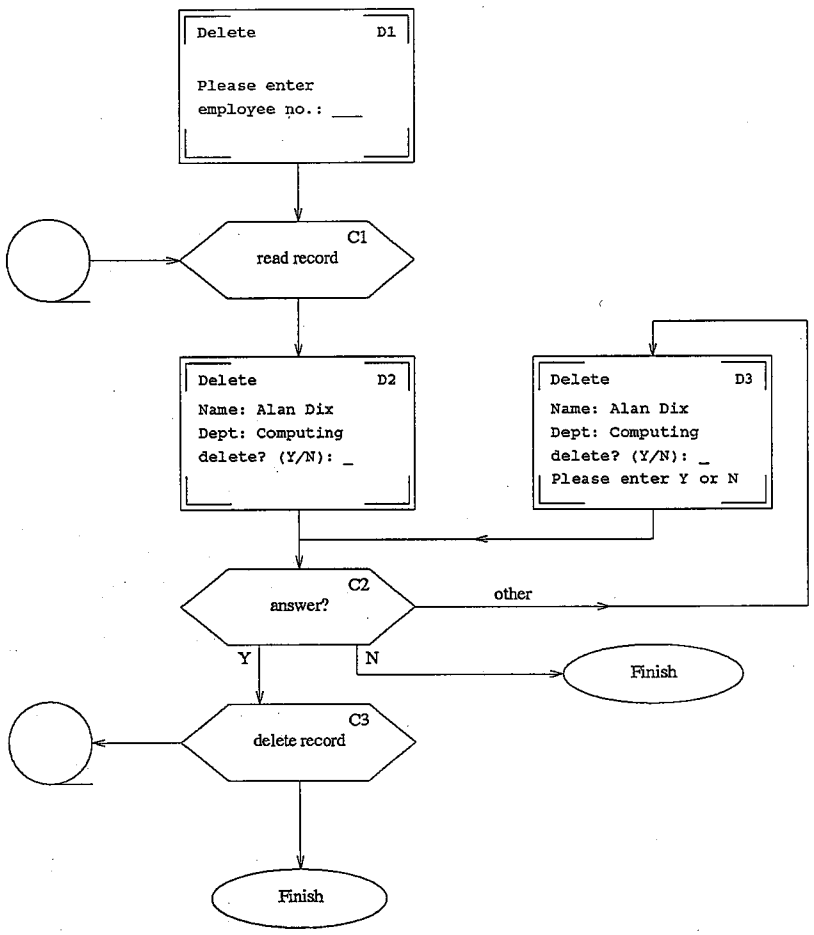
  even flowcharts!

- Documentation
- Prototyping tools

  e.g., Hyperdoc



Flowcharts



boxes – process/event **not** state

1000% productivity gain!

orthogonal to implementation

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

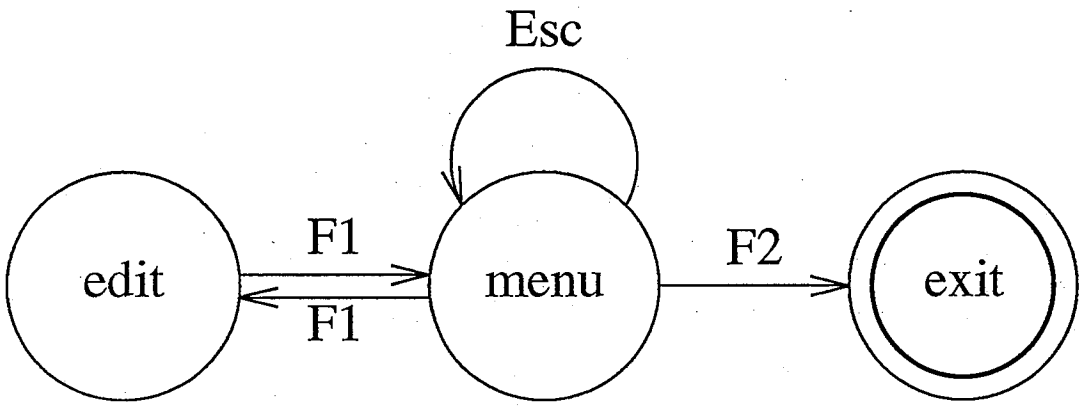
dangerous states

- some states you don't want to get to

Dangerous states (i)

Word processor: two modes and exit

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



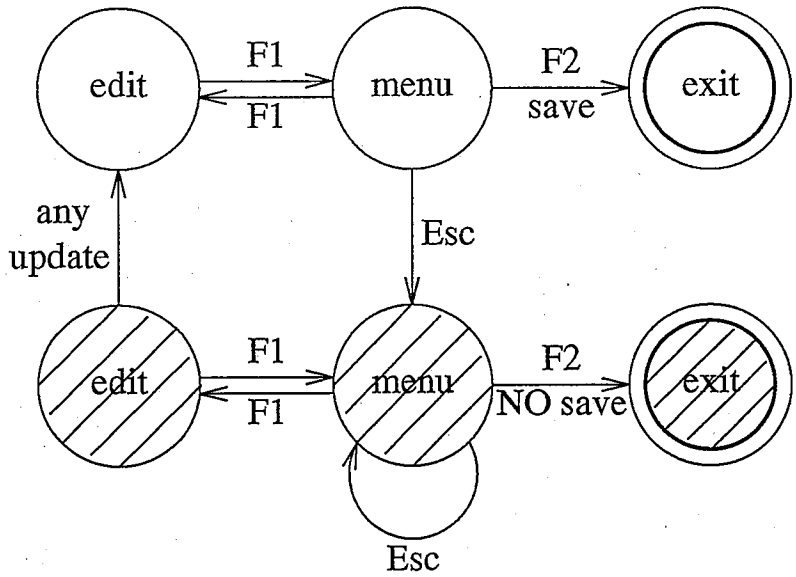
but...

Esc resets autosave

exit with/without save → dangerous states

duplicate states – semantic distinction

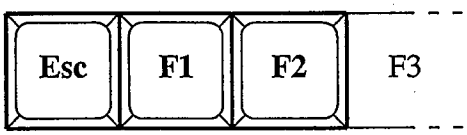
Dangerous states (ii)



F1-F2 – exit with save

F1-Esc-F2 – exit *no* save

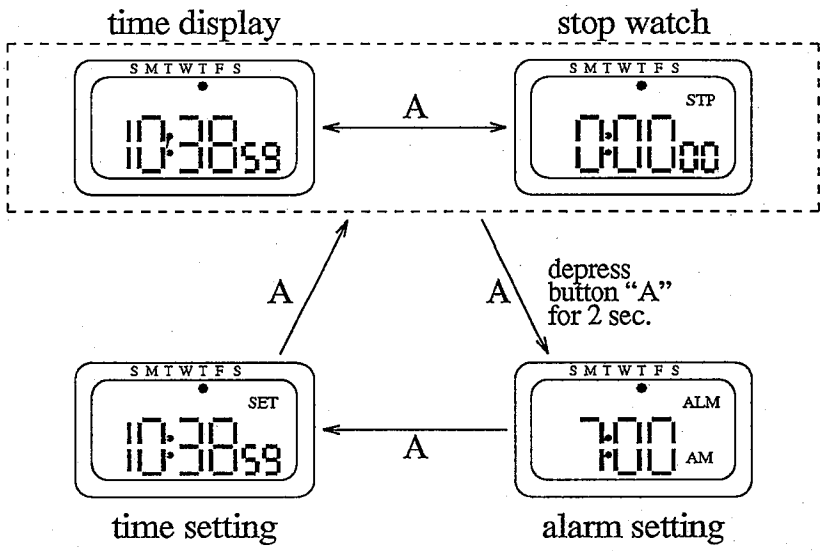
actual layout ...



Digital watch – Users instructions

limited interface – 3 buttons

button A moves between main modes



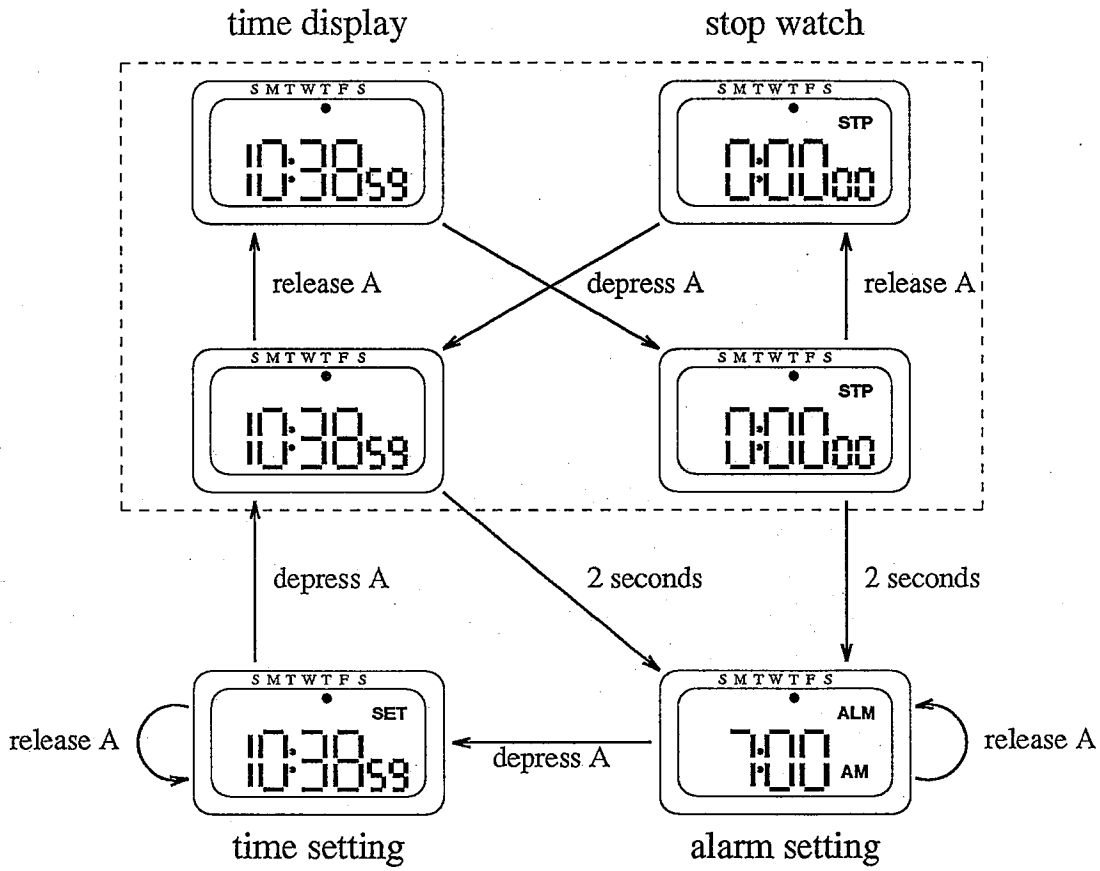
dangerous states

- guarded by two second hold

completeness

- distinguish depress A from release A
- what do they do in all modes?

Digital watch – Designers instructions



and that's only one button!

Status/Event Analysis

Status/event analysis

semi-formal technique

"engineering" level analysis

based on formal models

uses naïve psychology

clocks and calendars as example

status – analogue watch face

event – an alarm

Properties of events

status change event

- the passing of a time

actual and perceived events

- usually some gap

polling

- glance at watch face
- status change becomes perceived event

granularity

- birthday – days
- appointment – minutes

Naïve psychology

Predict where the user is looking

mouse – when positioning

insertion point – intermittantly when typing

screen – if you're lucky

Immediate events

audible bell – when in room (and hearing)

peripheral vision – movement or large change

Closure

lose attention (inc. mouse)

concurrent activity

Example – screen button widget (i)

screen button often missed, ...
but, error not noticed

a common widget, a common error: Why?

Closure

mistake likely – concurrent action

not noticed – semantic feedback missed

Solution

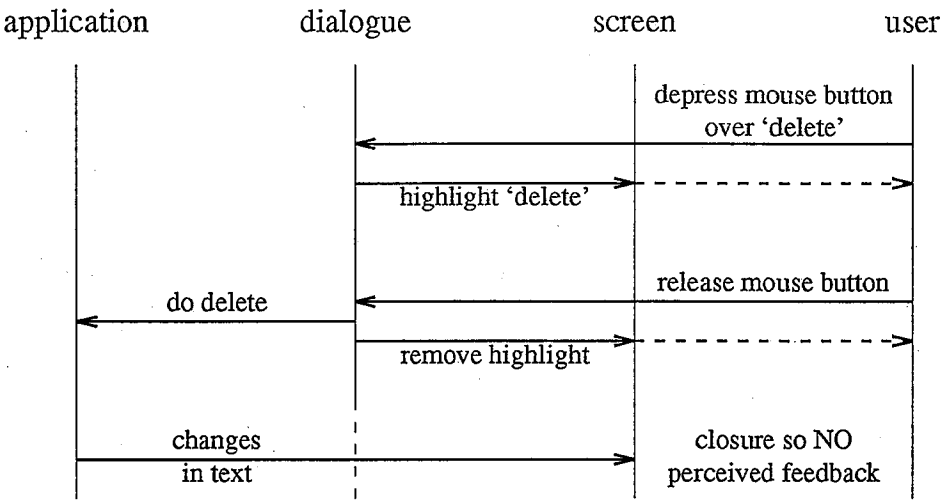
widget feedback for application event

a *perceived event* for the user

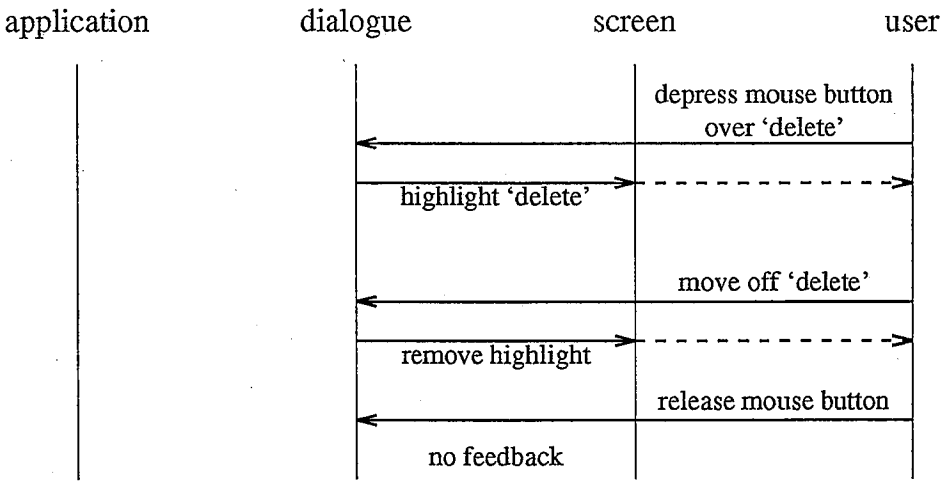
N.B., an expert slip – testing doesn't help

Screen button widget (ii)

a HIT



or a MISS



Summary

Formal models

- powerful and successful
- require formal expertise

Dialogue descriptions

- often there already
- both hand and automatic analysis

Status/event analysis

- formal concepts + naïve psychology

Engineering approach

- packaging up formal methods for the practitioner