Dialogue Notations and Design

Overview

Dialogue Notations

- Diagrammatic
  - state transition networks
  - JSD diagrams
  - Flow charts
- Textual
  - formal grammars
  - production rules
  - CSP

Dialogue Analysis

- Semantics and dialogue
- Properties of dialogue
- Presentation and lexical issues

Example

- Digital watch
State transition networks

circles – states
arcs – actions/events

Arc labels a bit cramped because:

- notation is ‘state heavy’
- the events require most detail
Hierarchical STNs

managing complex dialogues

named sub-dialogues
Flowcharts

familiar to programmers

boxes – process/event not state

Flowcharts

familiar to programmers

boxes – process/event not state
JSD diagrams

- for tree structured dialogues
  - less expressive
  - greater clarity

```
Personnel
Record
System

login

transaction

logout

add employee record

change employee record

display employee record

delete employee record
```
Concurrent dialogues (i)

Text Style

- **bold**
- **italic**
- **underline**

three toggles – individual STNs

- **Bold**
  - No
  - Bold
  - Click over bold

- **Italic**
  - No
  - Italic
  - Click over italic

- **Underline**
  - No
  - Underline
  - Click over underline
Concurrent dialogues (ii)

Text Style

- bold
- italic
- underline

bold and italic combined

- Neither
- click over bold
- click over italic

- Italic Only
- click over bold

- Bold Only
- click over italic
- Bold Italic

example
Concurrent dialogues (iii)

- combinatorial explosion
  - $n$ toggles
  - $2^n$ states
Textual – Grammars

Regular expressions

sel-line click click* double-click

- compare with JSD
  - same computational model
  - different notation

BNF

expr ::= empty
| atom expr
| `( expr `)` expr

- more powerful than regular exp. or STNs

Still NO concurrent dialogue
Production rules

if condition then action

Good for concurrency

Event based production rules

Sel-line → first
C-point first → rest
C-point rest → rest
D-point rest → <draw line>

Bad for sequence

Bad at state!
Propositional Production System

State based

Attributes:

**Mouse:** \{ mouse-off, select-line, 
    click-point, double-click \}

**Line-state:** \{ menu, first, rest \}

Rules (feedback not shown):

\[
\begin{align*}
\text{select-line} & \quad \rightarrow \quad \text{mouse-off first} \\
\text{click-point first} & \quad \rightarrow \quad \text{mouse-off rest} \\
\text{click-point rest} & \quad \rightarrow \quad \text{mouse-off} \\
\text{double-click rest} & \quad \rightarrow \quad \text{mouse-off menu}
\end{align*}
\]

Bad at events!
CSP and process algebras

used in Alexander’s SPI, and Agent notation
good for sequential dialogues

\[
\begin{align*}
\text{Bold-tog} &= \text{select-bold?} \rightarrow \text{bold-on} \\
&\quad \rightarrow \text{select-bold?} \rightarrow \text{bold-off} \\
&\quad \rightarrow \text{Bold-tog}
\end{align*}
\]

\[
\begin{align*}
\text{Italic-tog} &= \ldots \\
\text{Under-tog} &= \ldots
\end{align*}
\]

and concurrent dialogue

\[
\begin{align*}
\text{Dialogue-box} &= \text{Bold-tog} \parallel \text{Italic-tog} \parallel \\
\text{Under-tog}
\end{align*}
\]

but causality unclear
Dialogue Notations — Summary

Diagrammatic
- STN
- JSD
- Flow charts

Textual
- grammars
- production rules
- CSP

Issues
- event base vs. state based
- power vs. clarity
- model vs. notation
- sequential vs. concurrent
Semantics – Alexander’s SPI (i)

Two part specification:

EventCSP – pure dialogue order

EventISL – target dependent semantics

Centralised dialogue description

Tolerable syntactic/semantic trade-off
Semantics – Alexander’s SPI (ii)

EventCSP

\[
\begin{align*}
\text{Login} &= \text{login-mess} \rightarrow \text{get-name} \rightarrow \text{Passwd} \\
\text{Passwd} &= \text{passwd-mess} \rightarrow (\text{invalid} \rightarrow \text{Login} \\
&\phantom{=} \land \text{valid} \rightarrow \text{Session})
\end{align*}
\]

EventISL

\begin{align*}
\text{event:} & \quad \text{login-mess} \\
\text{prompt:} & \quad \text{true} \\
\text{out:} & \quad "\text{login:}" \quad \text{event:} \quad \text{get-name} \\
\text{uses:} & \quad \text{input} \\
\text{set:} & \quad \text{user-id} = \text{input} \quad \text{event:} \quad \text{valid} \\
\text{uses:} & \quad \text{input, user-id, passwd-db} \\
\text{when:} & \quad \text{passwd-id} = \text{passwd-db(user-id)}
\end{align*}
Event loop for word processor

Distributed dialogue description

Terrible syntactic/semantic trade-off

```plaintext
switch ( ev.type ) {
    case button_down:
        if ( in_text ( ev.pos ) ) {
            mode = selecting;
            mark_selection_start(ev.pos);
        }
        ...
    case button_up:
        if ( in_text ( ev.pos ) && mode == selecting ) {
            mode = normal;
            mark_selection_end(ev.pos);
        }
        ...
    case mouse_move:
        if ( mode == selecting ) {
            extend_selection(ev.pos);
        }
        ...
} /* end of switch */
```
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
Checking properties (i)

Completeness:

double-click in circle states?

Reversibility:

to reverse select ‘line’ from graphics Menu

click – double click – select ‘graphics’

(3 actions)

N.B. not undo
Checking properties (ii)

- Select 'circle' and highlight 'circle'.
- Circle 1: Click on centre, rubber band.
- Circle 2: Click on circumference, draw circle.
- Finish:

- Select 'line' and highlight 'line'.
- Line 1: Click on first point, rubber band.
- Line 2: Click on point, draw line and rubber band from new point.
- Double click, draw last line. Finish:

- Select 'graphics' and pop-up graphics sub-menu.

- Select 'text' and pop-up text sub-menu.

- Select 'paint' and pop-up paint sub-menu.
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
Dangerous states (ii)

exit with/without save → dangerous states

duplicate states – semantic distinction

F1-F2 – exit with save

F1-Esc-F2 – exit *no* save
Lexical issues

visibility
- differentiate modes and states
- annotations to dialogue

style
command - verb noun
mouse-based - noun verb

layout
dangerous states (previous slide)
old keyboard - OK
new keyboard - disaster
Dialogue Analysis — Summary

Semantics and dialogue
- attaching semantics
- distributed/centralised dialogue description
- maximising syntactic description

Properties of dialogue
- action properties: completeness, determinism, consistency
- state properties: reachability, reversibility, dangerous states

Presentation and lexical issues
- visibility
- style
- layout

N.B. not independent of dialogue
Digital watch – User’s 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 – Designer’s instructions

and that’s only one button!