Chapter 16

Dialogue Notations and Design

Dialogue Notations and Design

- Dialogue Notations
  - Diagrammatic
    - state transition networks, JSD diagrams, flow charts
  - Textual
    - formal grammars, production rules, CSP

- Dialogue linked to
  - the **semantics** of the system – **what it does**
  - the **presentation** of the system – **how it looks**

- Formal descriptions can be analyzed
  - for inconsistent actions
  - for difficult to reverse actions
  - for missing actions
  - for potential miskeying errors

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 of computer language**
  - **Lexical** – shape of icons, actual keys pressed
  - **Syntactic** – order/structure 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 woman's 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 license
  - **syntax** not **semantics**

… and more

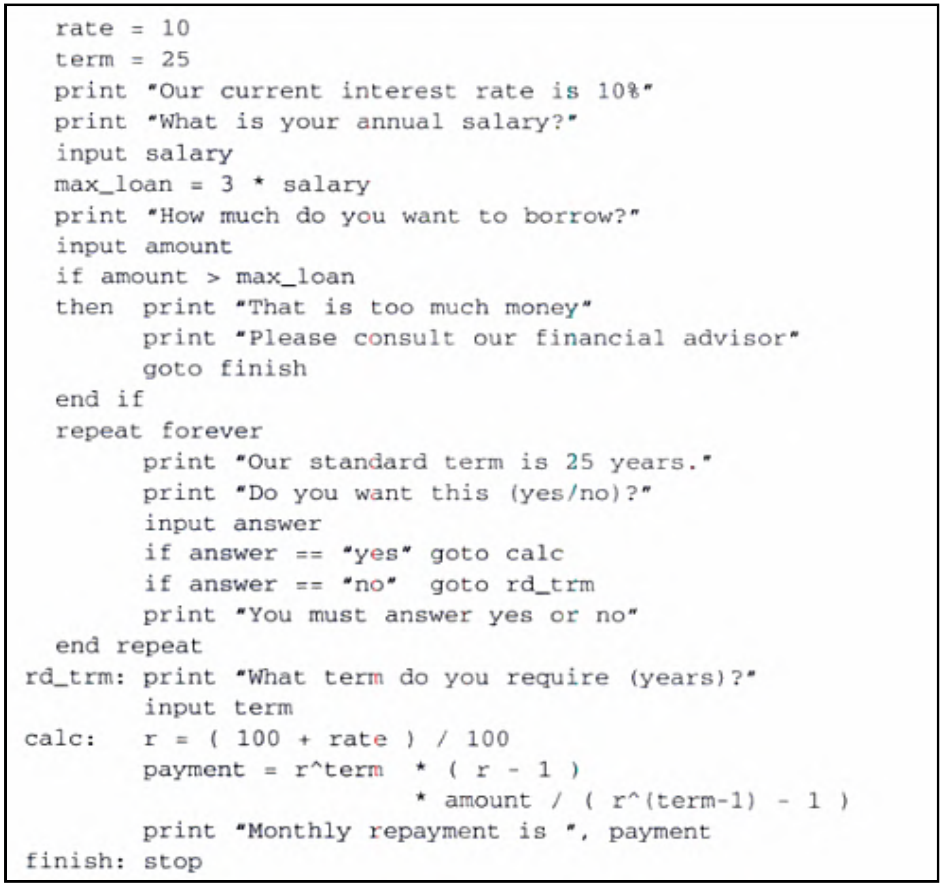
- What if woman says “I don’t”?
- Real dialogues often have **alternatives**:

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

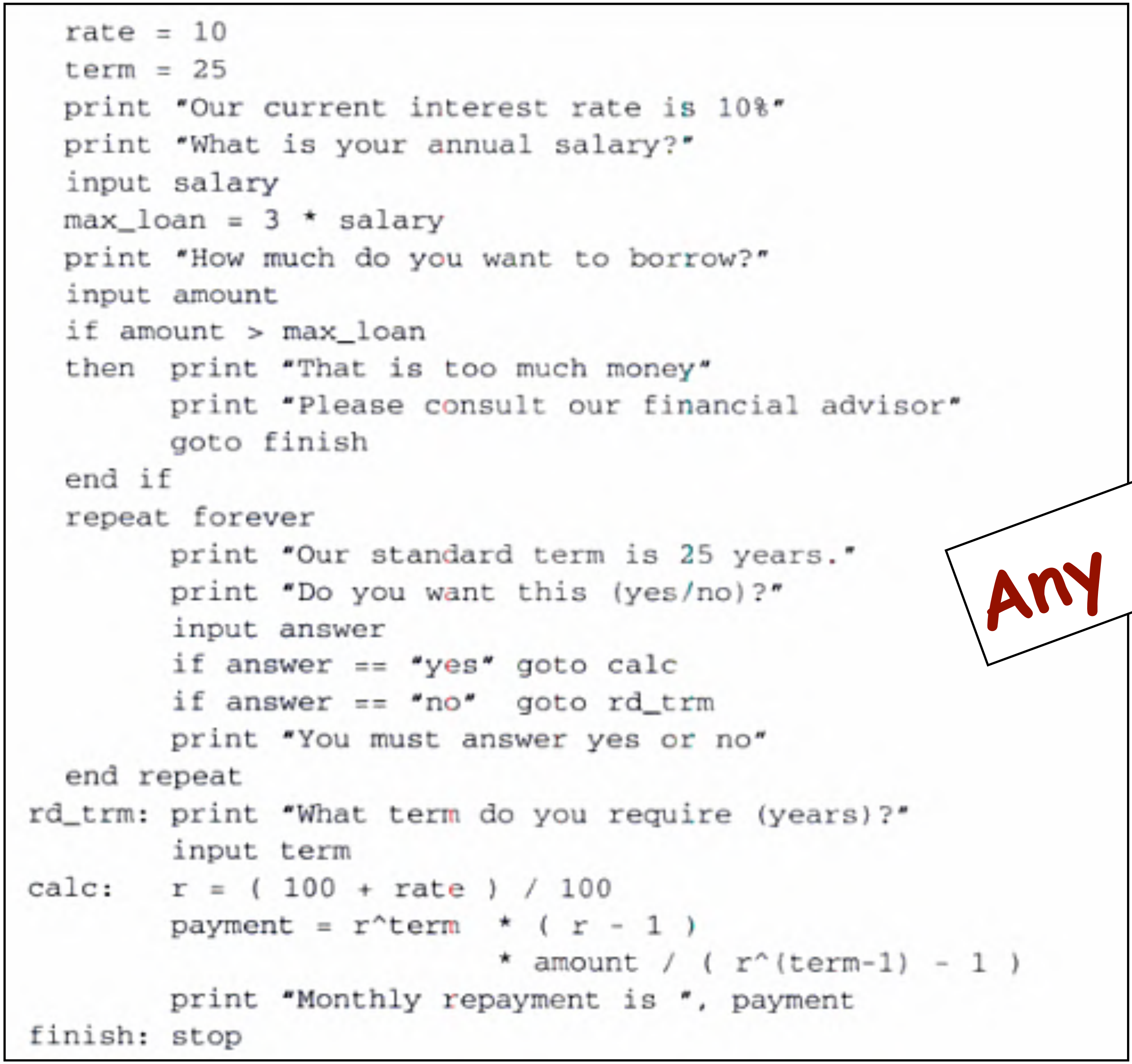
  - the process of the trial depends on the defendants response
- Focus on **normative responses**
  - doesn’t cope with judge saying “off with her head”
  - or in computer dialogue user standing on keyboard!

Dialogue design notations – pseudo code

Why not?



```
    rate = 10
    term = 25
    print "Our current interest rate is 10%"
    print "What is your annual salary?"
    input salary
    max_loan = 3 * salary
    print "How much do you want to borrow?"
    input amount
    if amount > max_loan
    then  print "That is too much money"
          print "Please consult our financial advisor"
          goto finish
    end if
    repeat forever
          print "Our standard term is 25 years."
          print "Do you want this (yes/no)?"
          input answer
          if answer == "yes" goto calc
          if answer == "no"  goto rd_trm
          print "You must answer yes or no"
    end repeat
rd_trm: print "What term do you require (years)?"
        input term
calc:   r = ( 100 + rate ) / 100
        payment = r^term  * ( r - 1 )
                          * amount / ( r^(term-1) - 1 )
        print "Monthly repayment is ", payment
finish: stop
```



```
   rate = 10
   term = 25
   print "Our current interest rate is 10%"
   print "What is your annual salary?"
   input salary
   max_loan = 3 * salary
   print "How much do you want to borrow?"
   input amount
   if amount > max_loan
   then  print "That is too much money"
         print "Please consult our financial advisor"
         goto finish
   end if
   repeat forever
         print "Our standard term is 25 years."
         print "Do you want this (yes/no)?"
         input answer
         if answer == "yes" goto calc
         if answer == "no"  goto rd_trm
         print "You must answer yes or no"
   end repeat
rd_trm: print "What term do you require (years)?"
        input term
calc:   r = ( 100 + rate ) / 100
        payment = r^term  * ( r - 1 )
                          * amount / ( r^(term-1) - 1 )
        print "Monthly repayment is ", payment
finish: stop
```

Any problem?

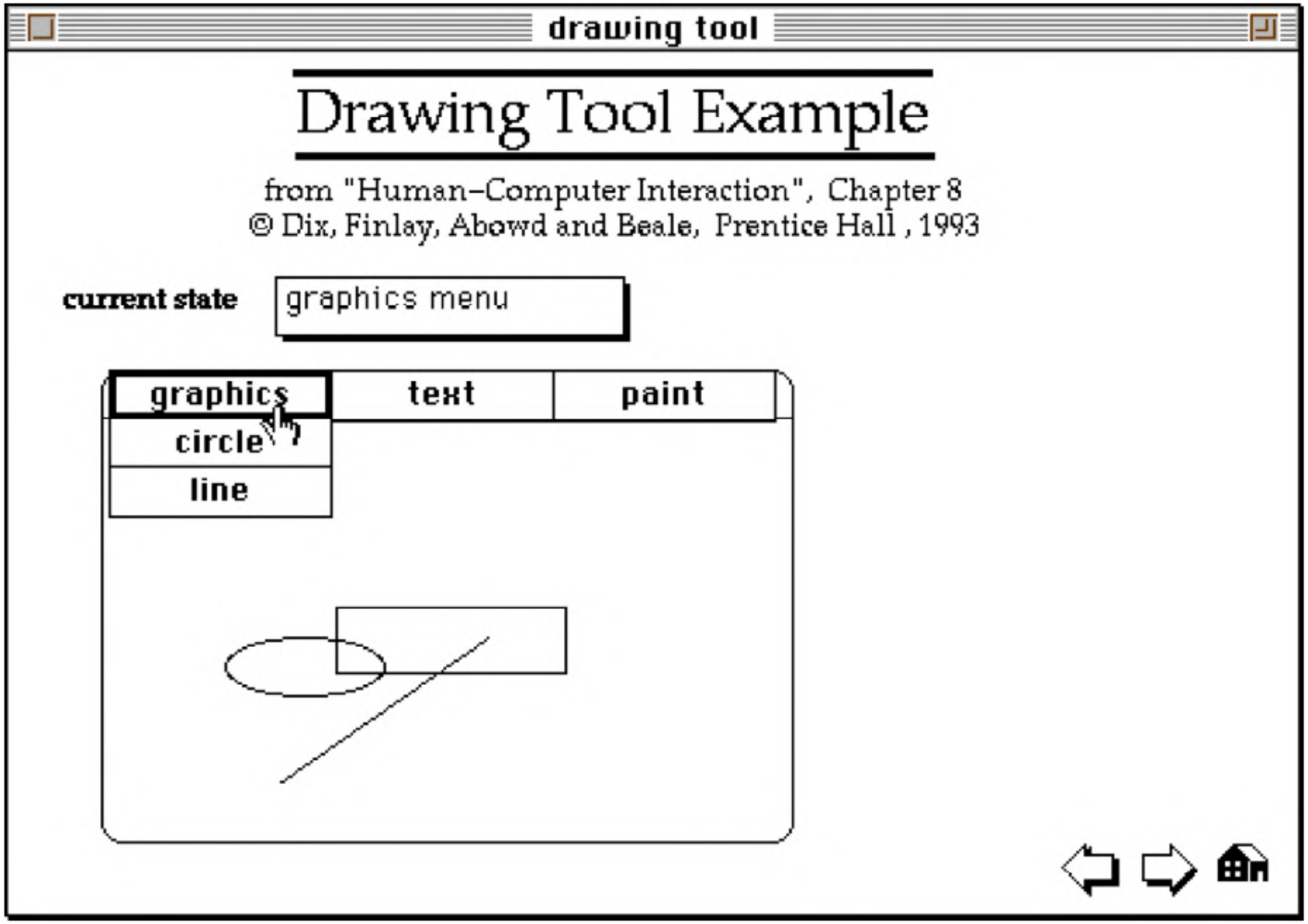
Dialogue design notations

- **Dialogue gets buried in the program**
- **In a big system can we:**
  - analyze the dialogue:
    - can the user always get to see current shopping basket
  - change platforms (e.g. Windows/Mac)
  - dialogue notations helps us to
    - analyze systems
    - **separate** lexical/syntactical from semantic
- **… and before the system is built**
  - notations help us **understand proposed designs**

Graphical Notations

State-Transition Nets (**STN**)
Petri Nets, State Charts
Flow Charts, JSD diagrams

State Transition Networks (STN)



drawing tool

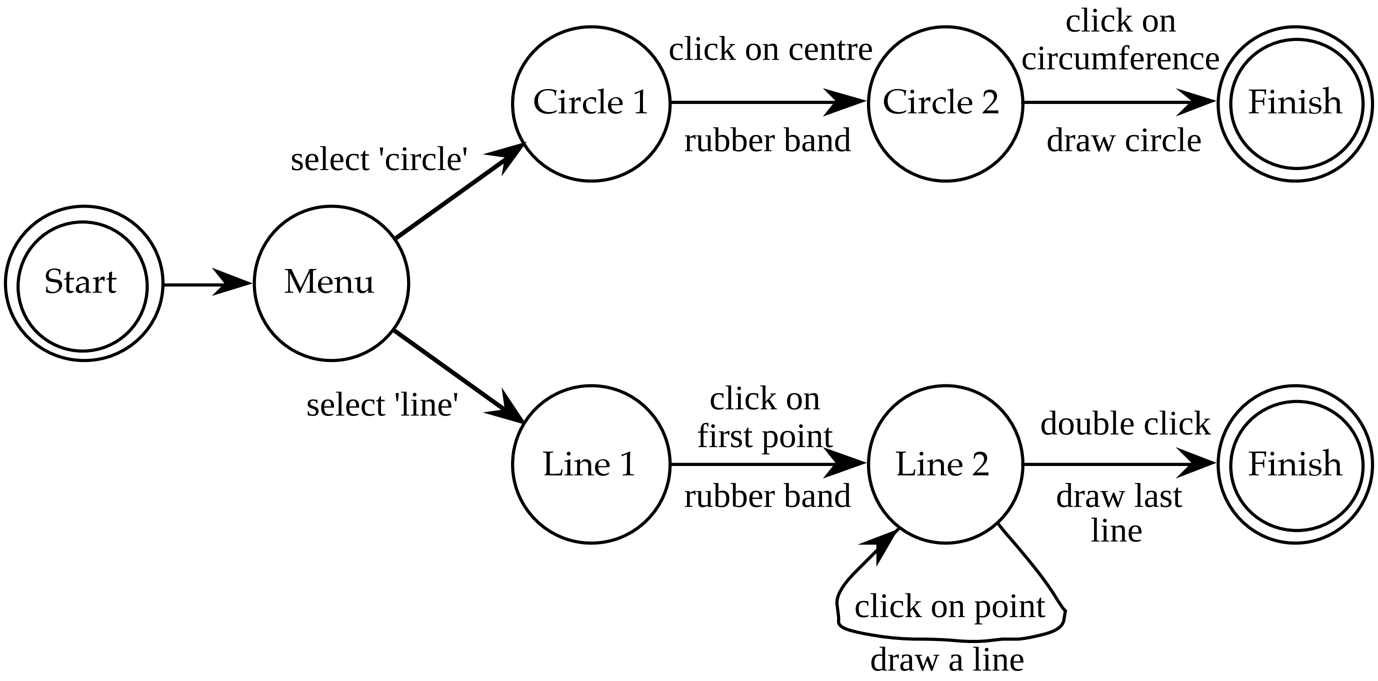
Drawing Tool Example

from "Human–Computer Interaction", Chapter 8
© Dix, Finlay, Abowd and Beale, Prentice Hall , 1993

current state: graphics menu

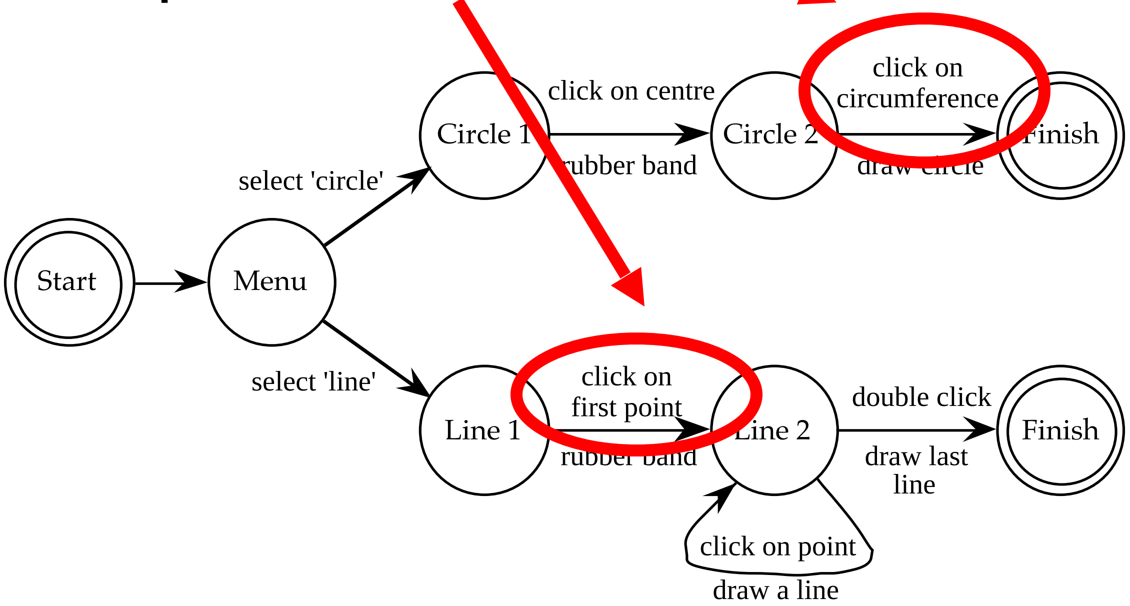
State Transition Networks (STN)

- circles - states
- arcs - actions/events



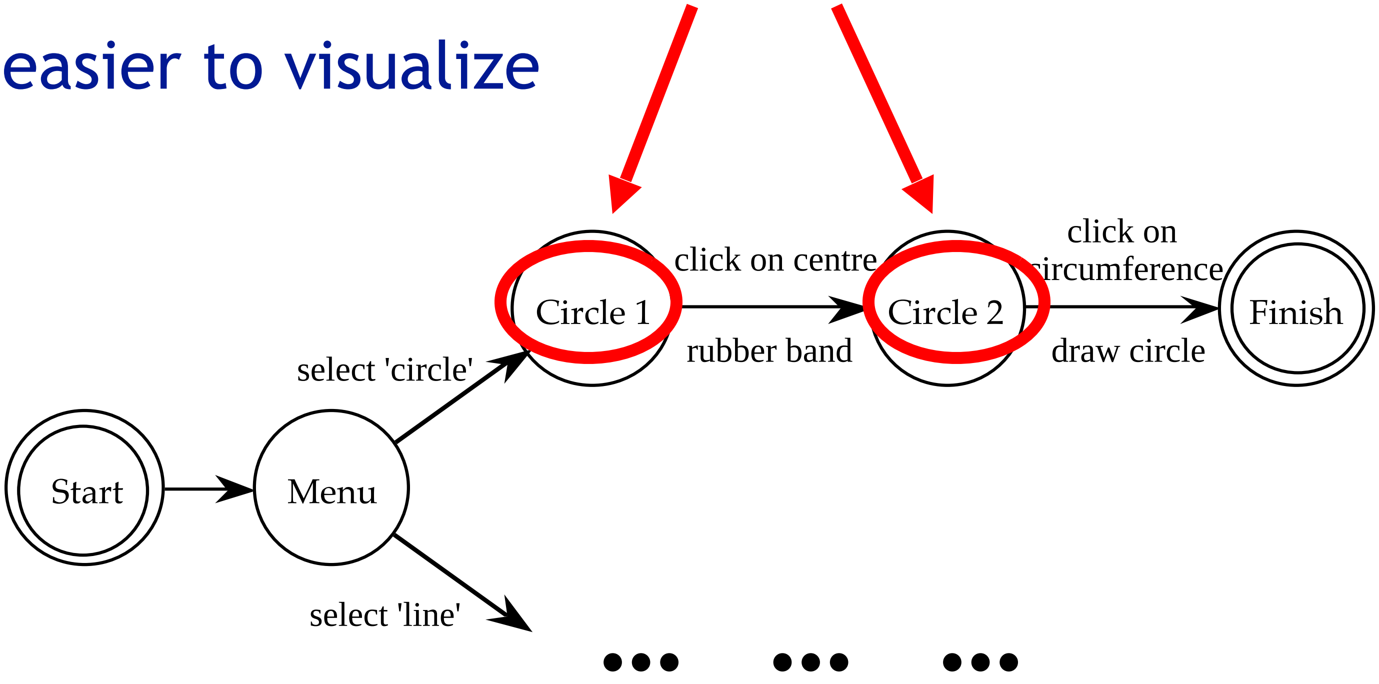
State Transition Networks - events

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



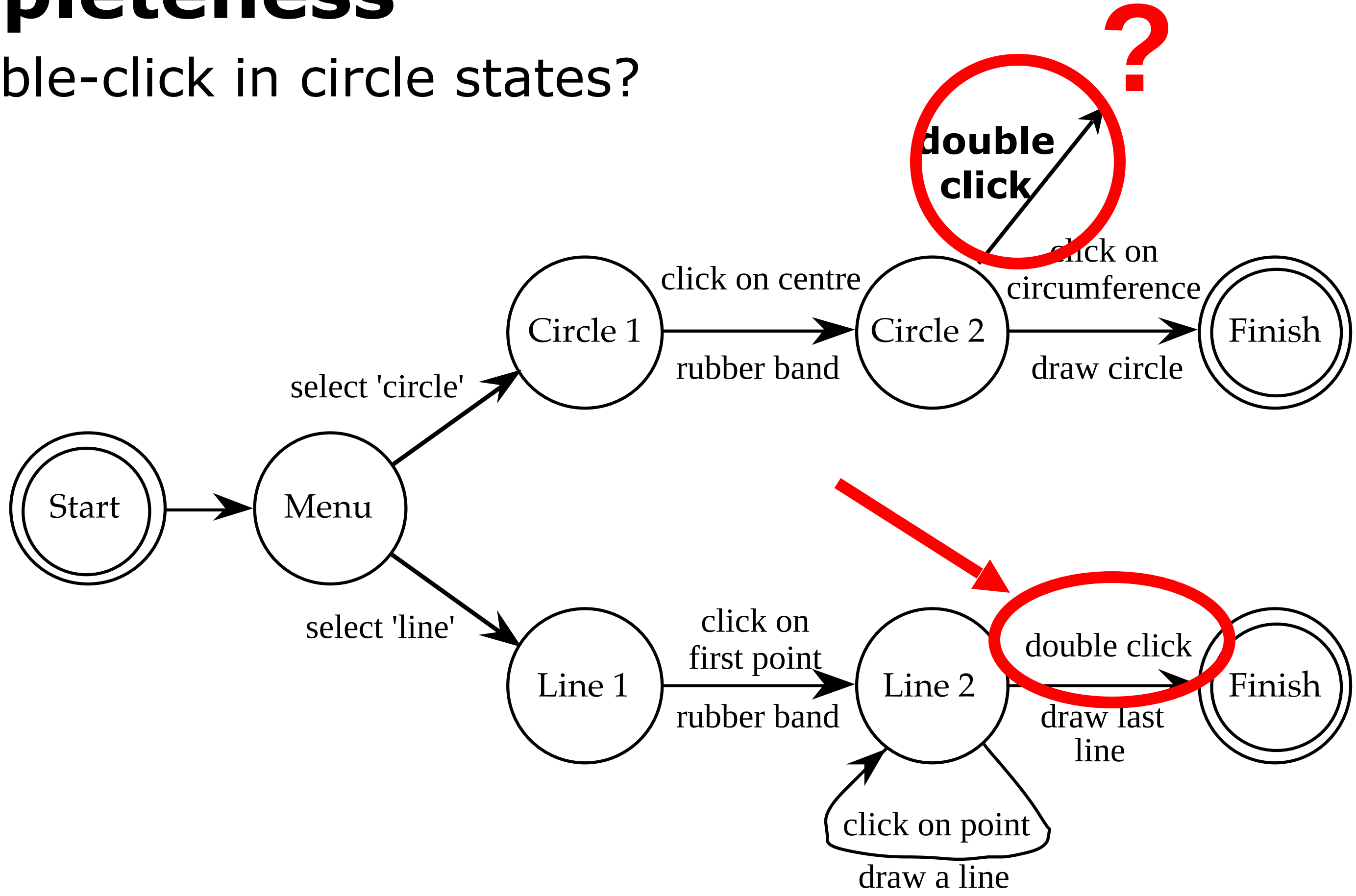
State Transition Networks - states

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



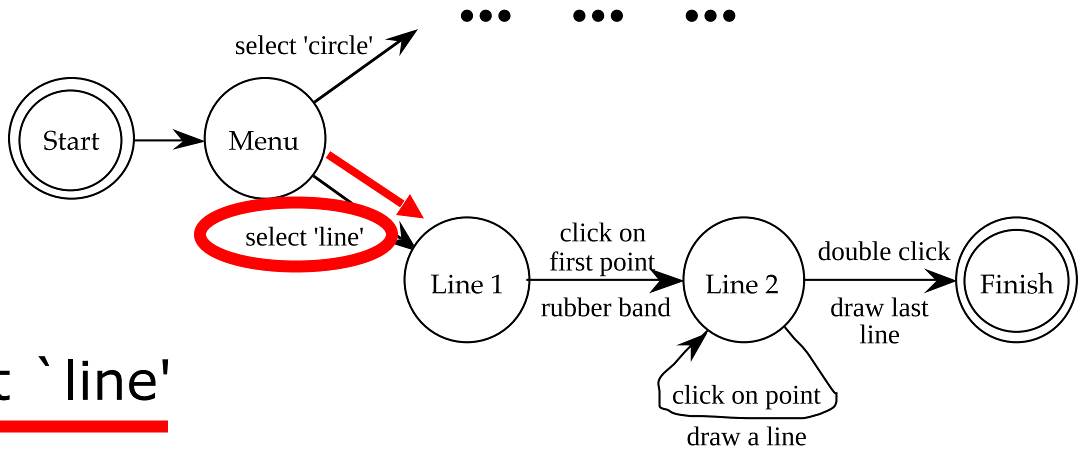
Checking properties (i)

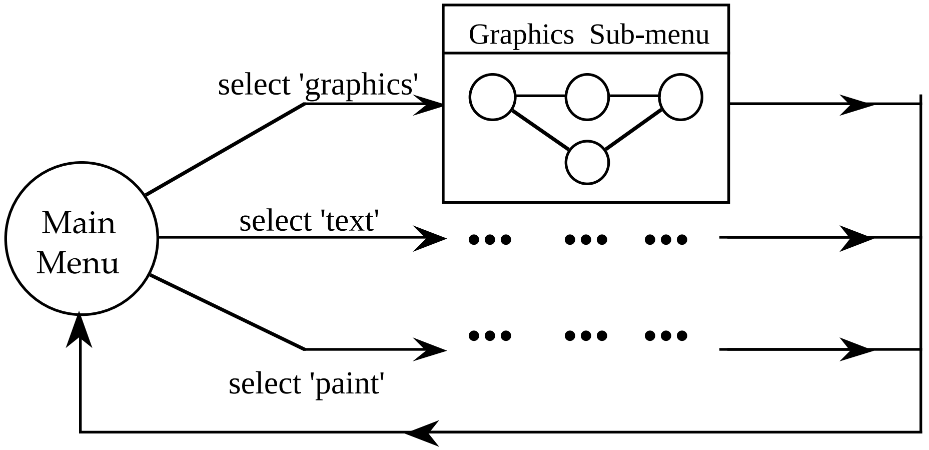
- **Completeness**
  - double-click in circle states?



Checking properties (ii)

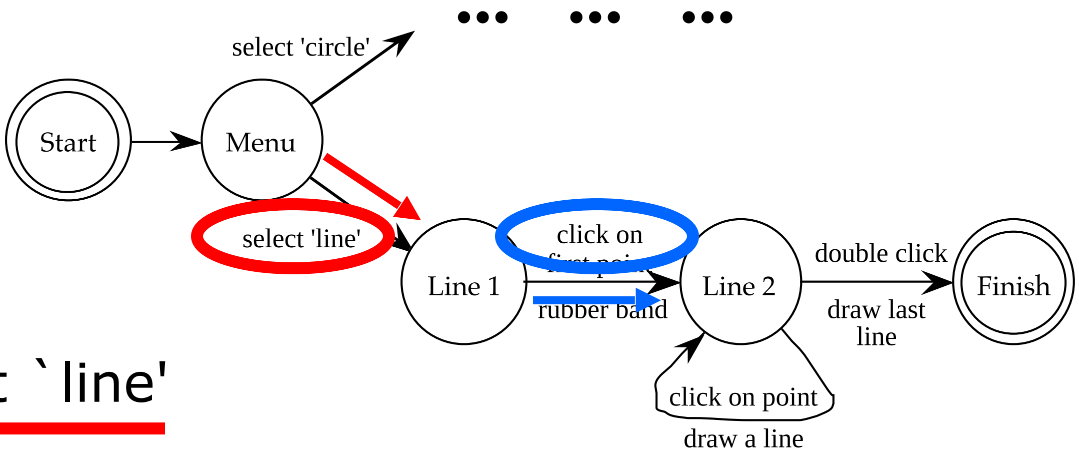
- **Reversibility**:
  - to reverse select `line'

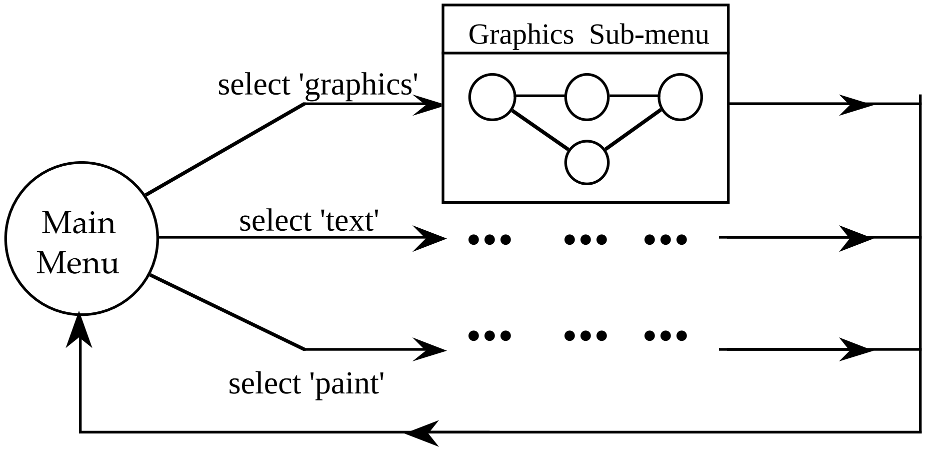




Checking properties (ii)

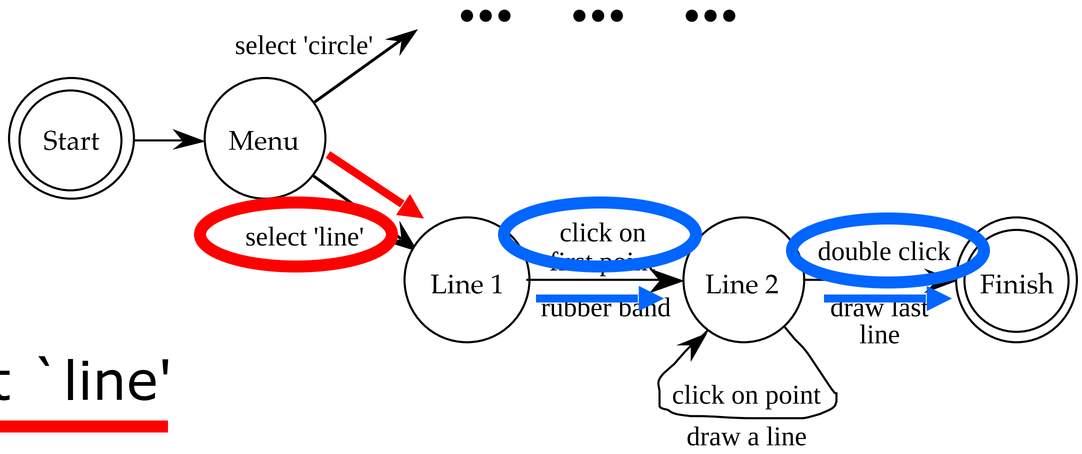
- Reversibility:
  - to reverse select `line'
  - click

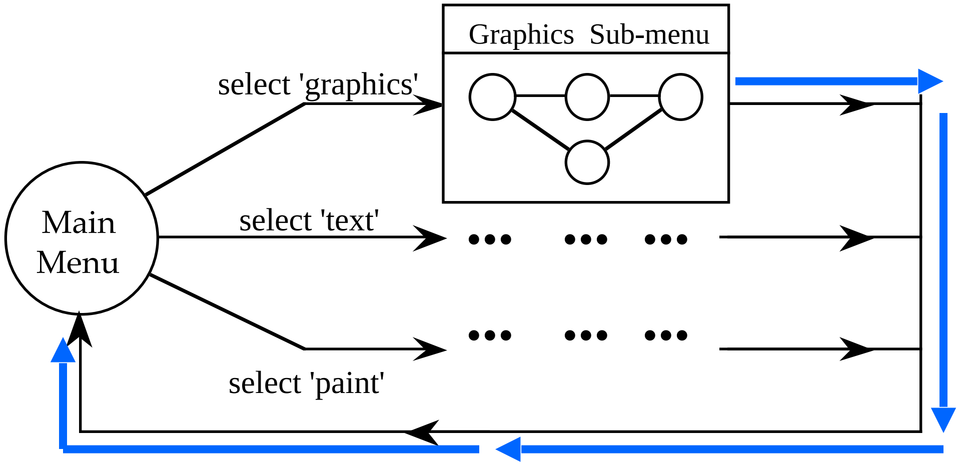




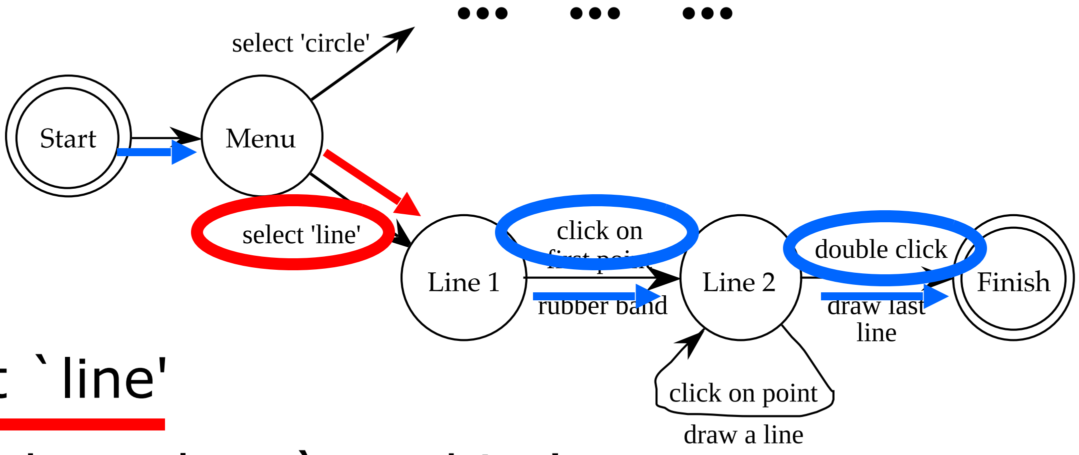
Checking properties (ii)

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

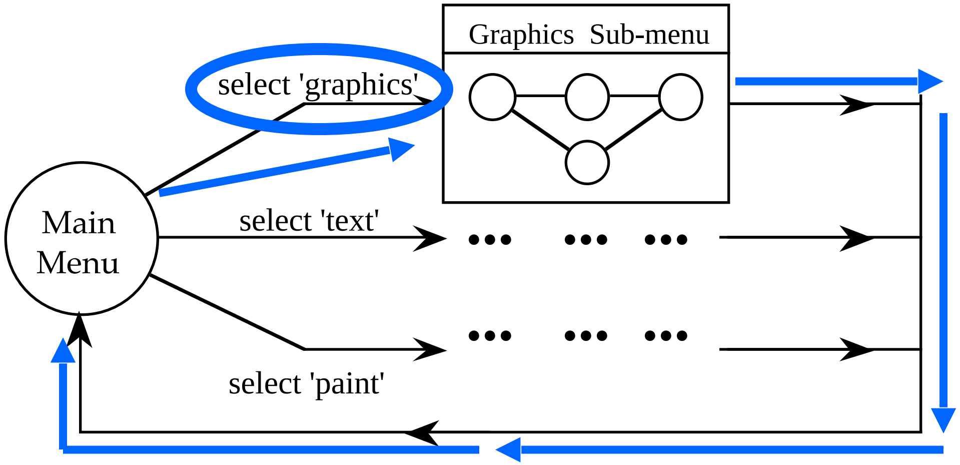




Checking properties (ii)

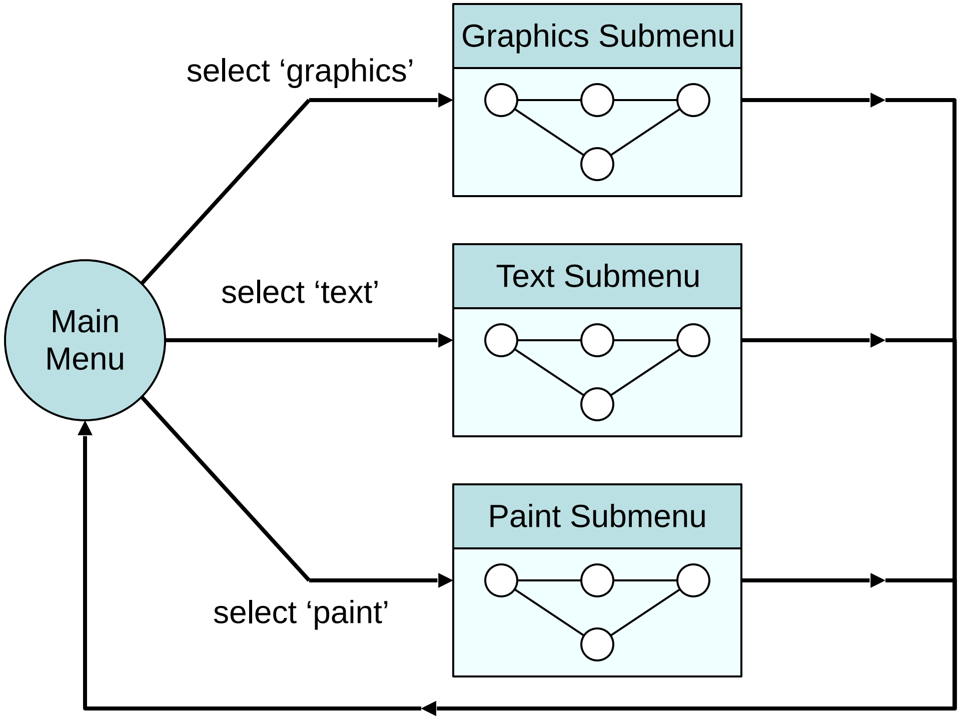


- Reversibility:
  - to reverse select `line'
  - click - double click - select `graphics'
  - (3 actions)
- N.B. not undo



Hierarchical STNs

- Managing complex dialogues
- Named sub-dialogues



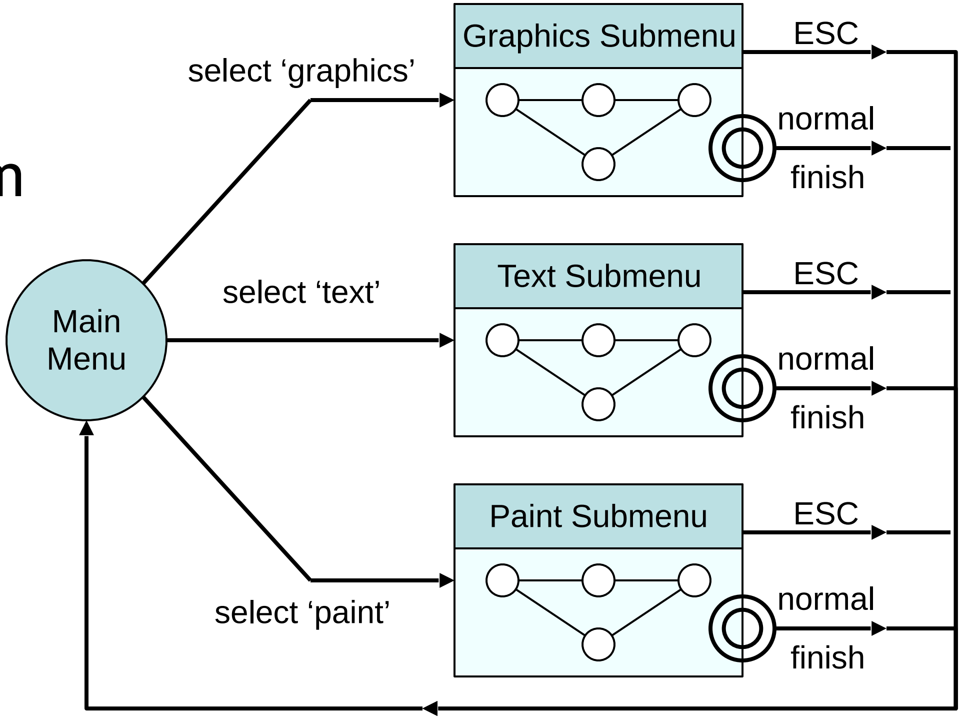
Escapes

- **'back' in web, escape/cancel keys**
  - similar behavior everywhere
  - end up with spaghetti of identical behaviors
- **try to avoid this**

e.g. on high level diagram

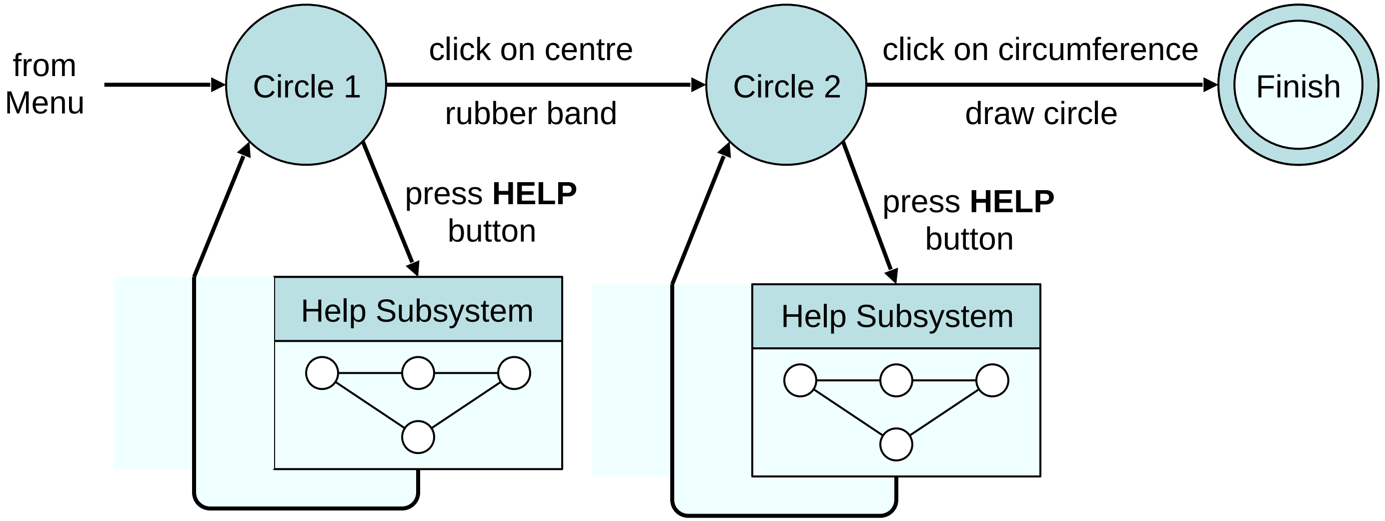
'normal' exit for each submenu

plus separate escape arc active 'everywhere' in submenu



help menus

- Similar problems
  - nearly the same everywhere
  - but return to same point in dialogue
  - could specify on STN … but very messy
  - usually best added at a ‘meta’ level



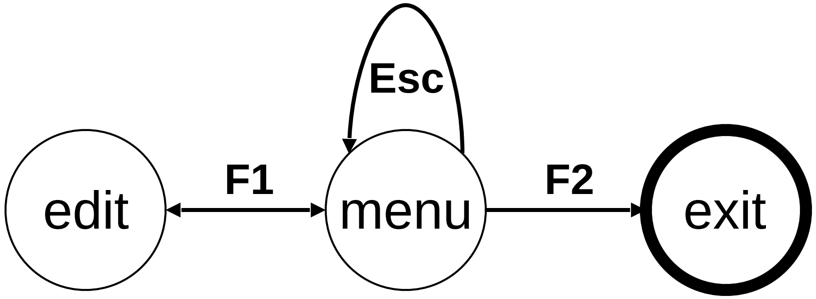
State properties

- 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

- Word processor: two modes and exit

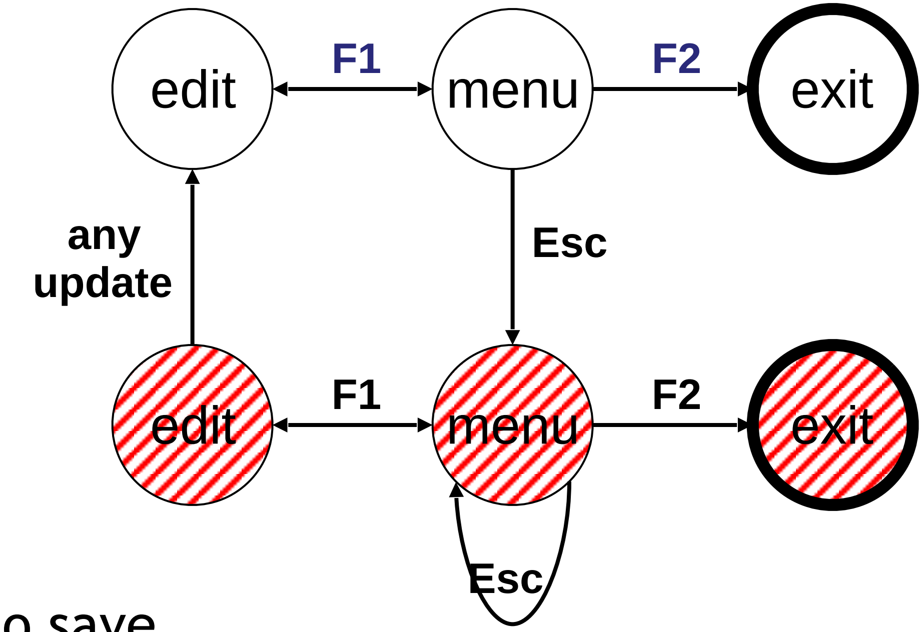
F1	-	changes mode
F2	-	exit (and save)
Esc	-	no mode change



but ... Esc resets autosave

Dangerous States (ii)

- Exit with/without save ⇒ dangerous states
- Duplicate states - semantic distinction

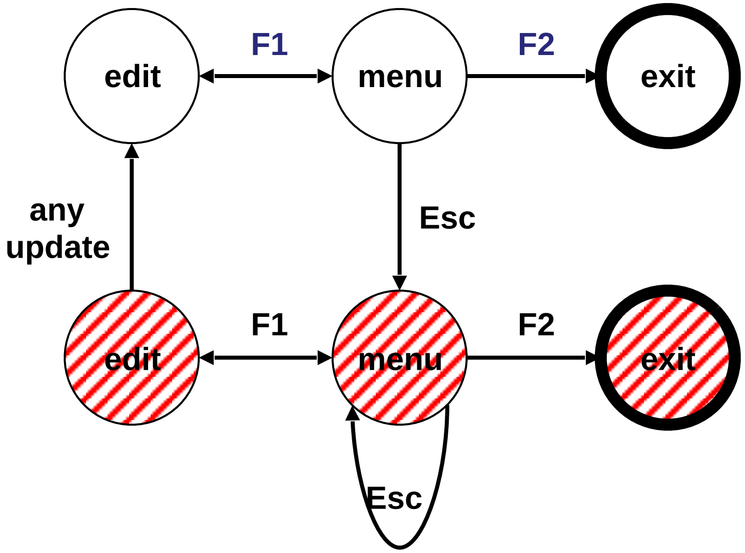


F1-F2 - exit with save

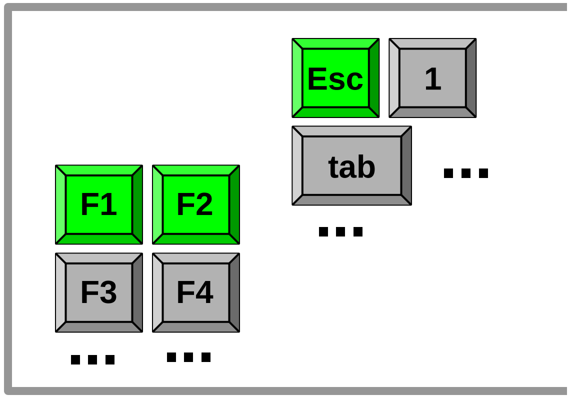
F1-Esc-F2 - exit with no save

layout matters

- Word processor - dangerous states

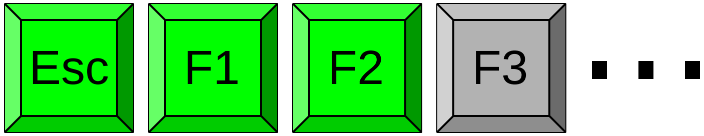


- Old keyboard - OK



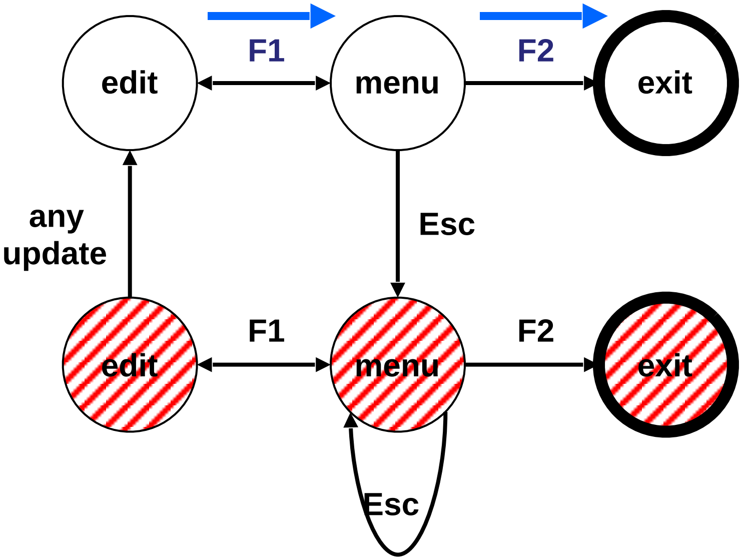
layout matters

- **but** with new keyboard layout



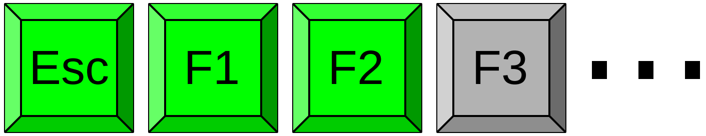
intend F1-F2 (save)

finger catches Esc



layout matters

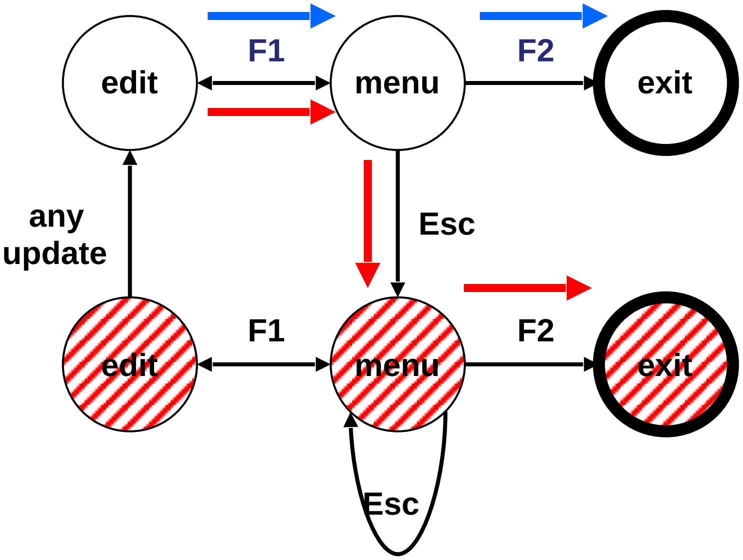
- **but** with new keyboard layout



intend F1-F2 (save)

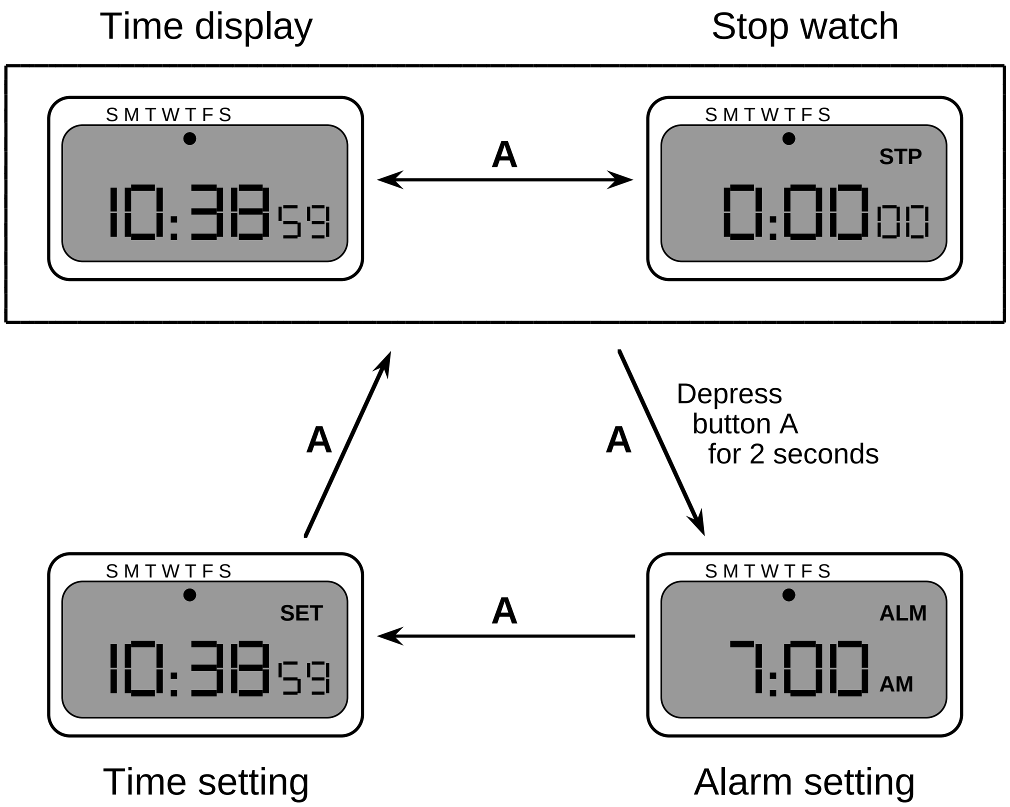
finger catches Esc

F1-Esc-F2 - **disaster!**



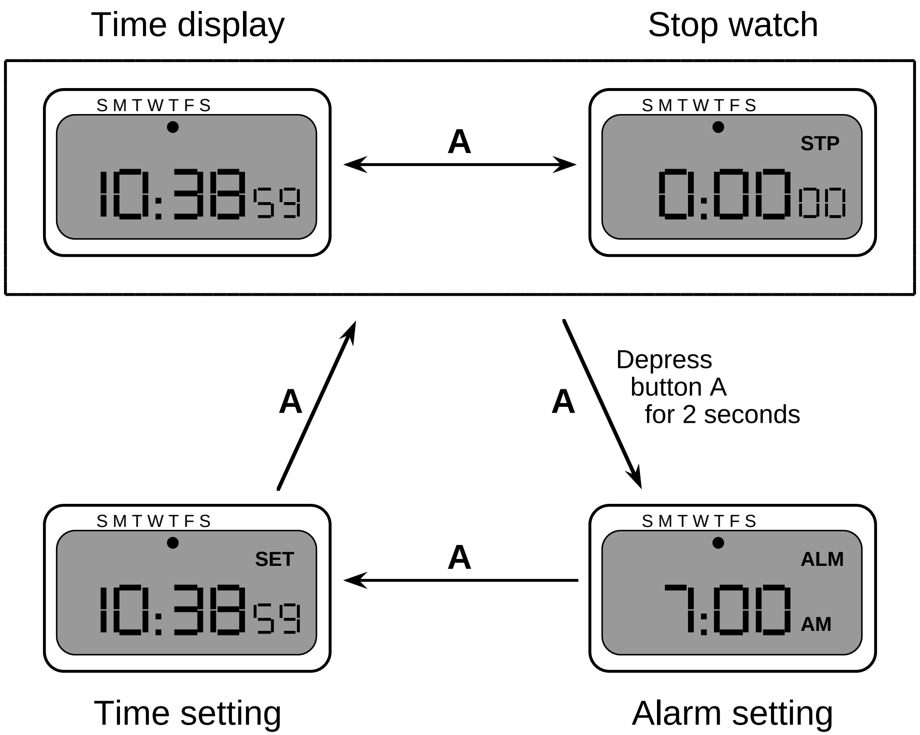
Digital watch – User Instructions

- two main modes
- limited interface
  - 3 buttons
- button A
  changes mode



Digital watch – User Instructions

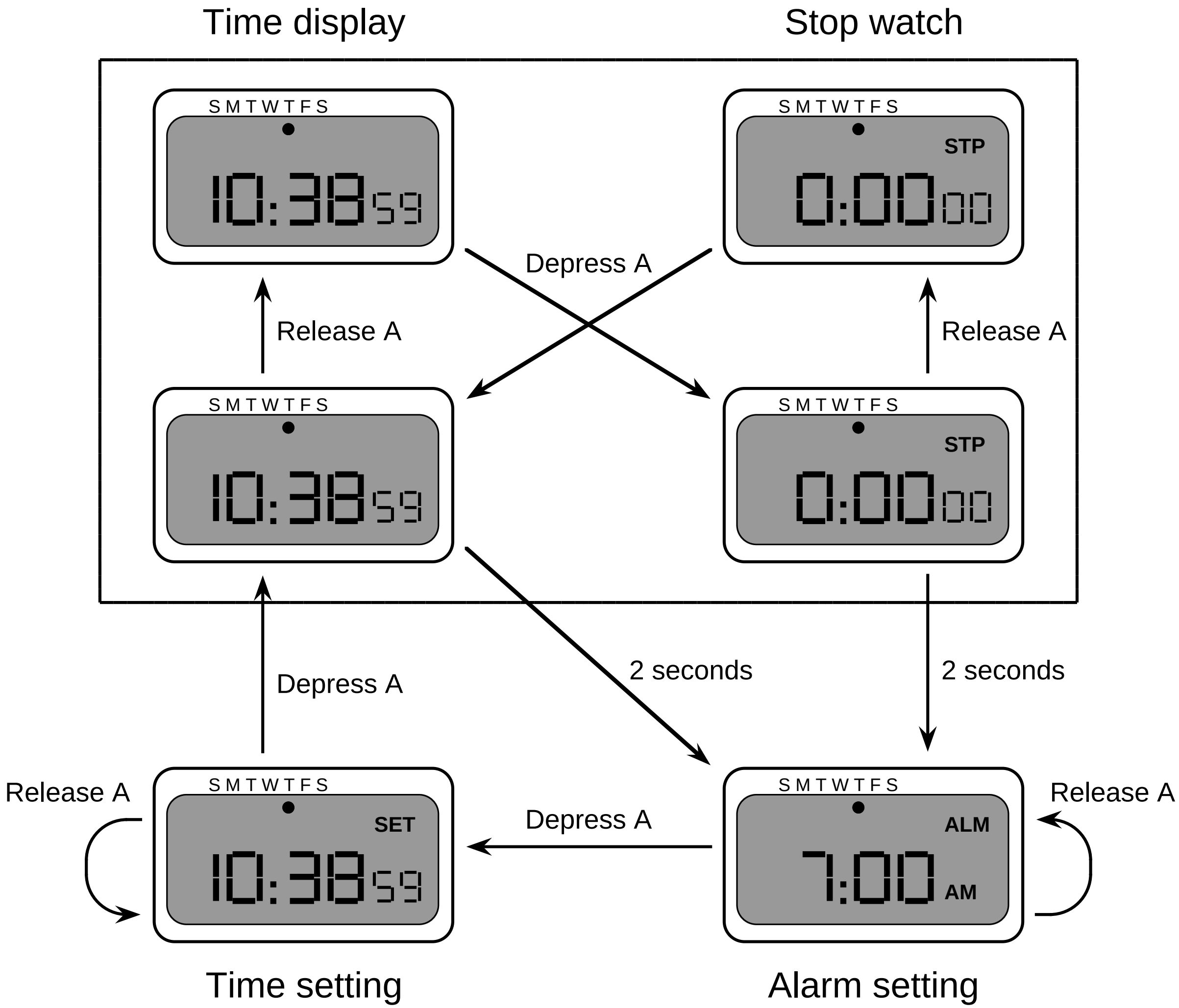
- **dangerous states**
  - *guarded*
    … by two second hold
- **completeness**
  - distinguish depress A and release A
  - what do they do in all modes?



Digital watch – Designers instructions

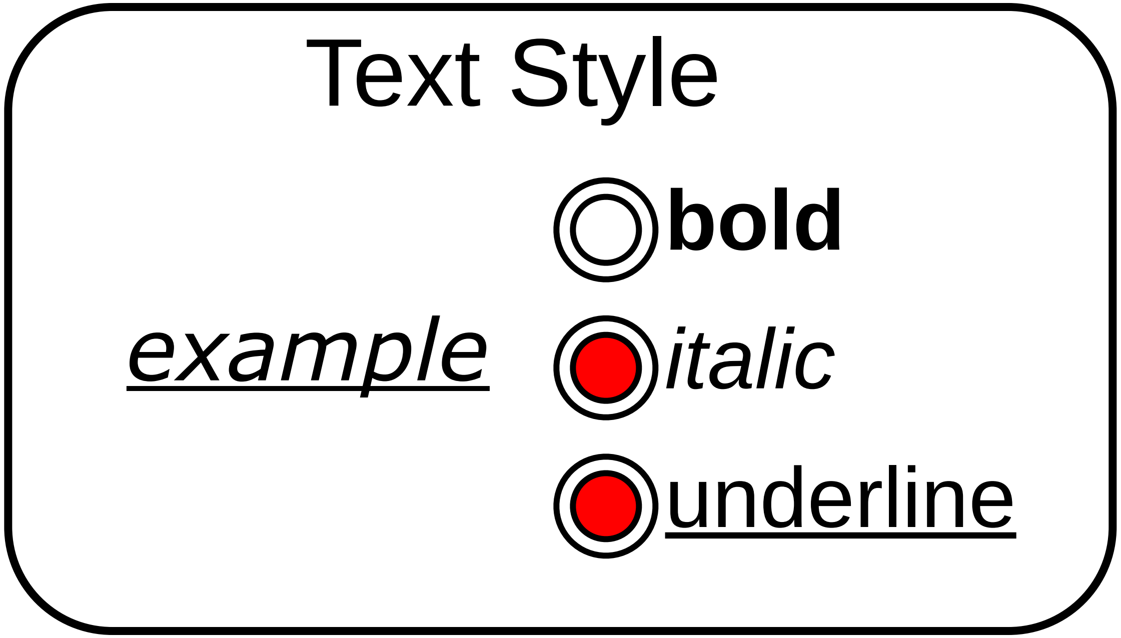
and …

that’s just
one button



Concurrent dialogues - I

Example: a simple dialogue box



Text Style

example

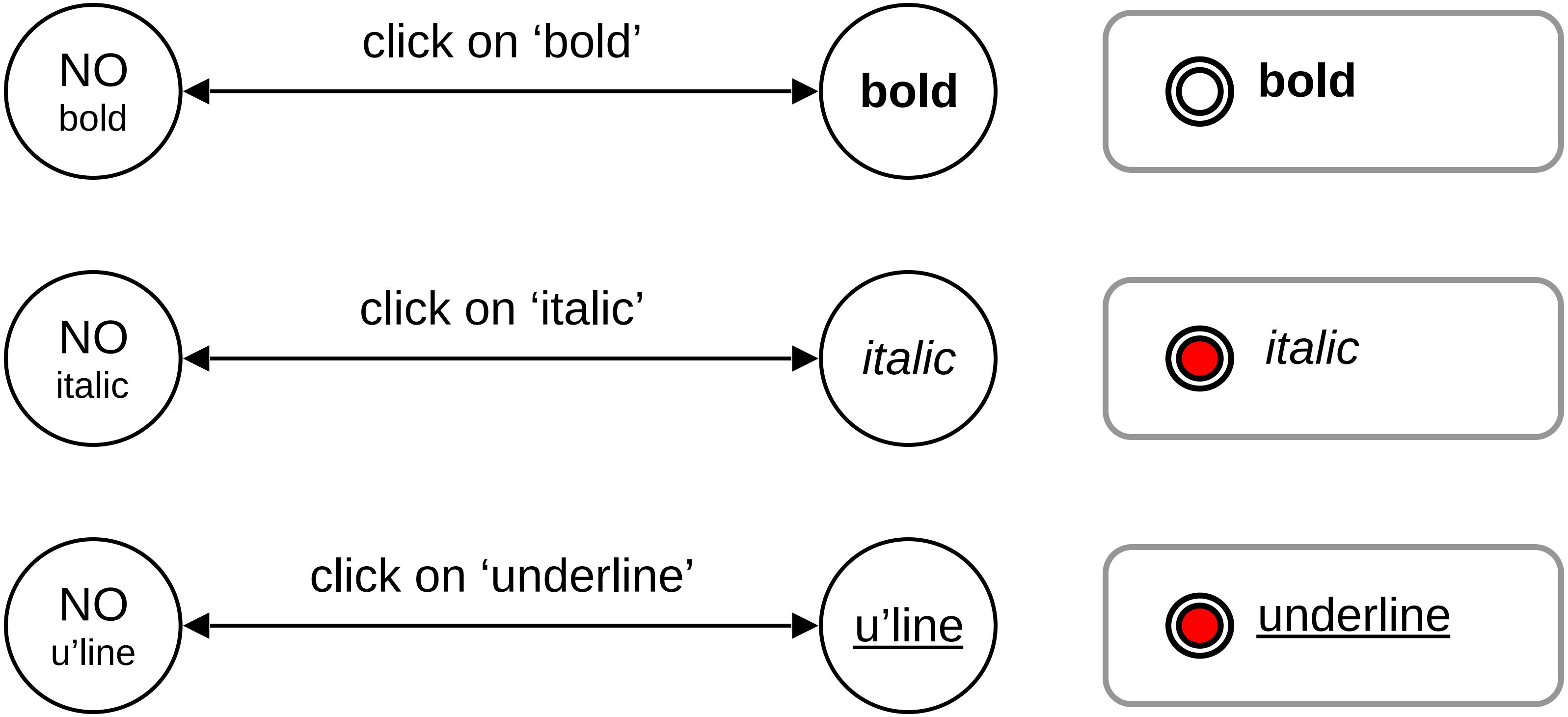
○ **bold**

● *italic*

● underline

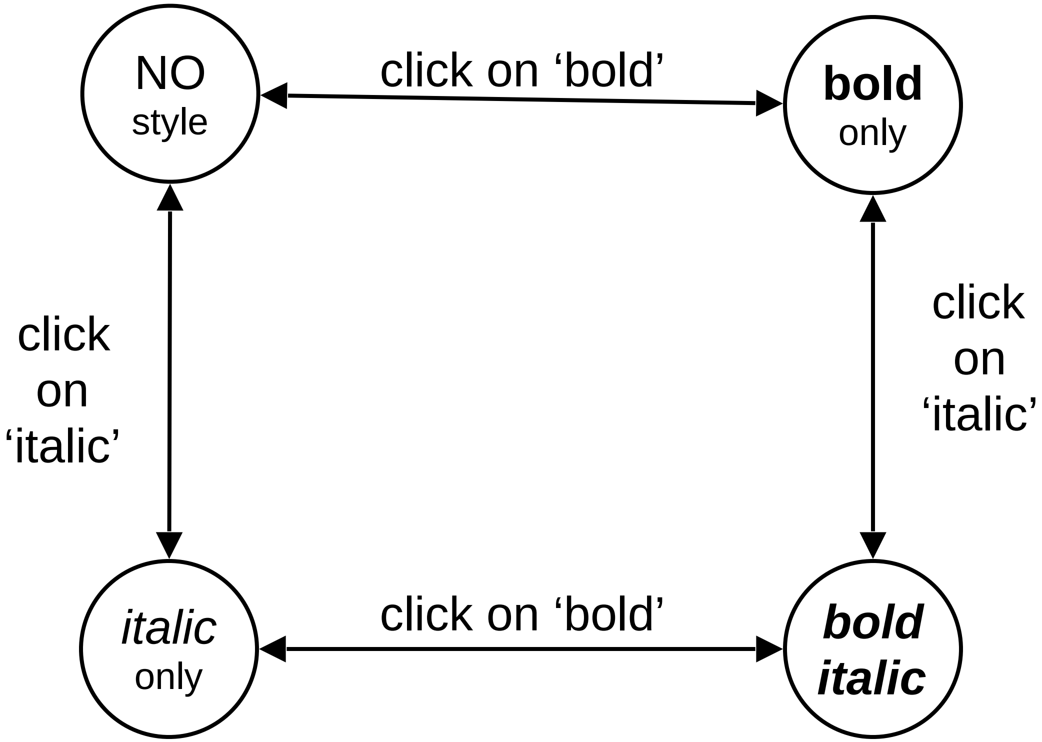
Concurrent dialogues - II

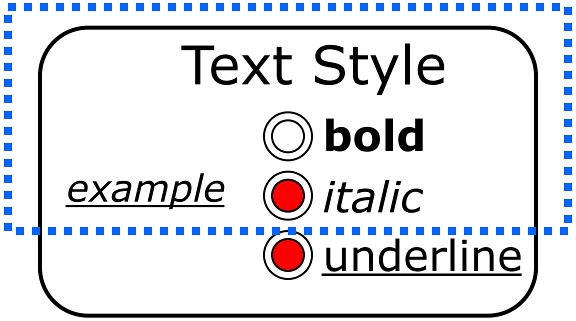
three toggles - individual STNs



Concurrent dialogues - III

bold and italic combined




Text Style

example

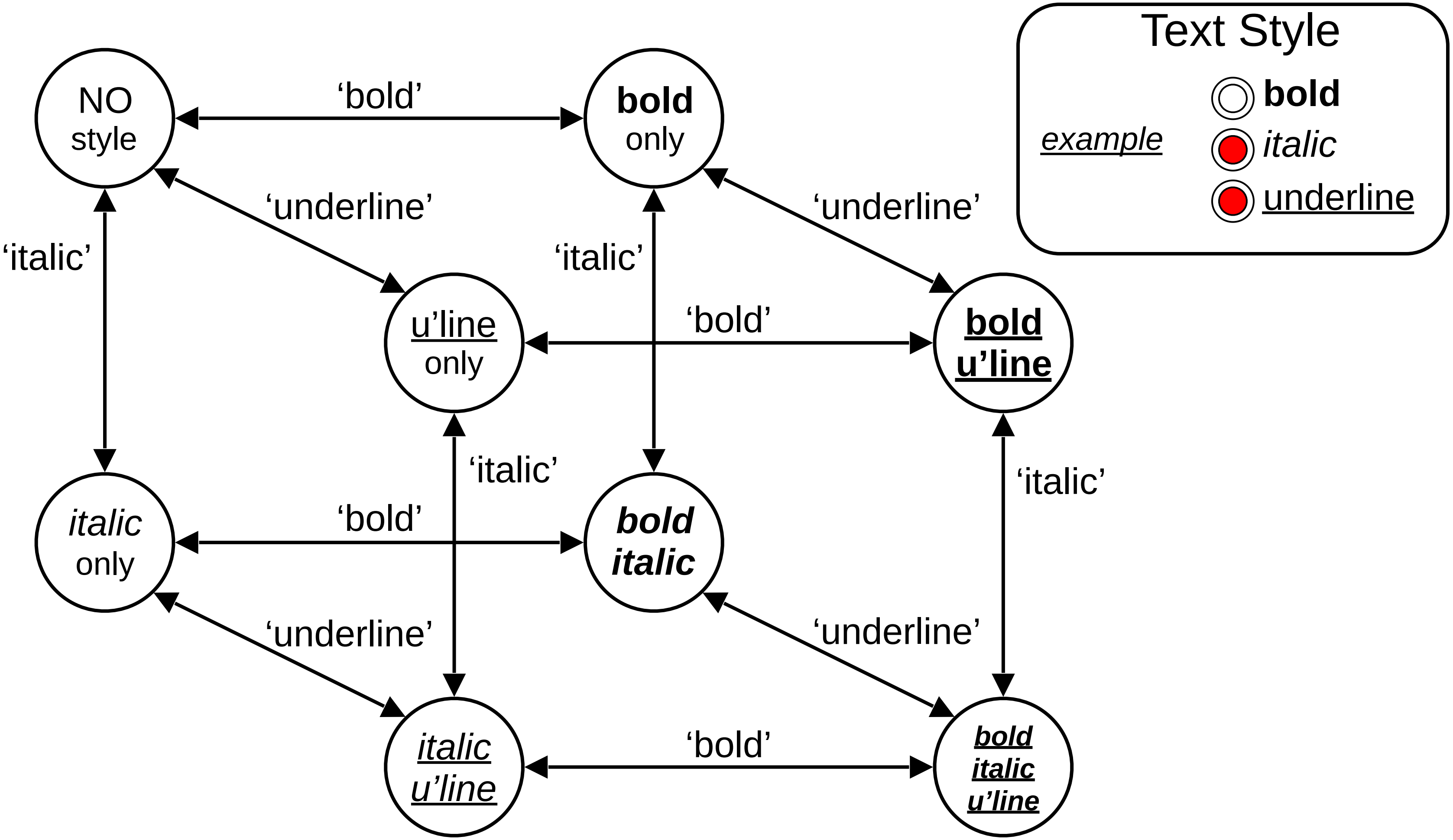
○ **bold**

● *italic*

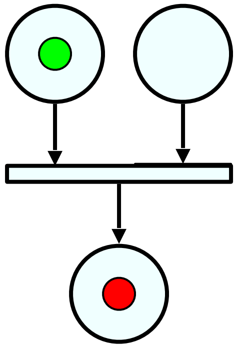
● underline

Concurrent dialogues - IV

all together - combinatorial explosion

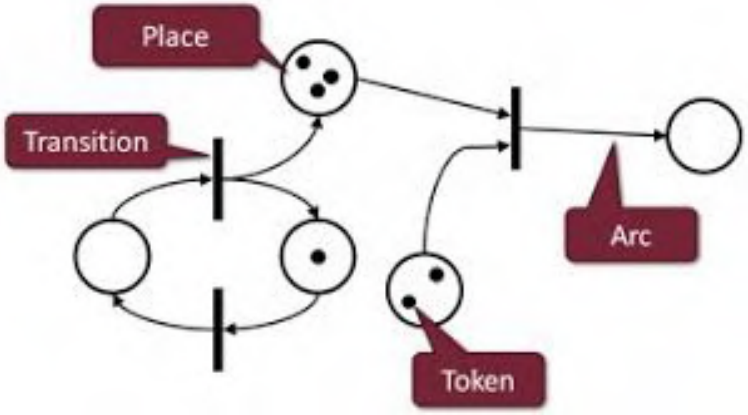


Petri Nets

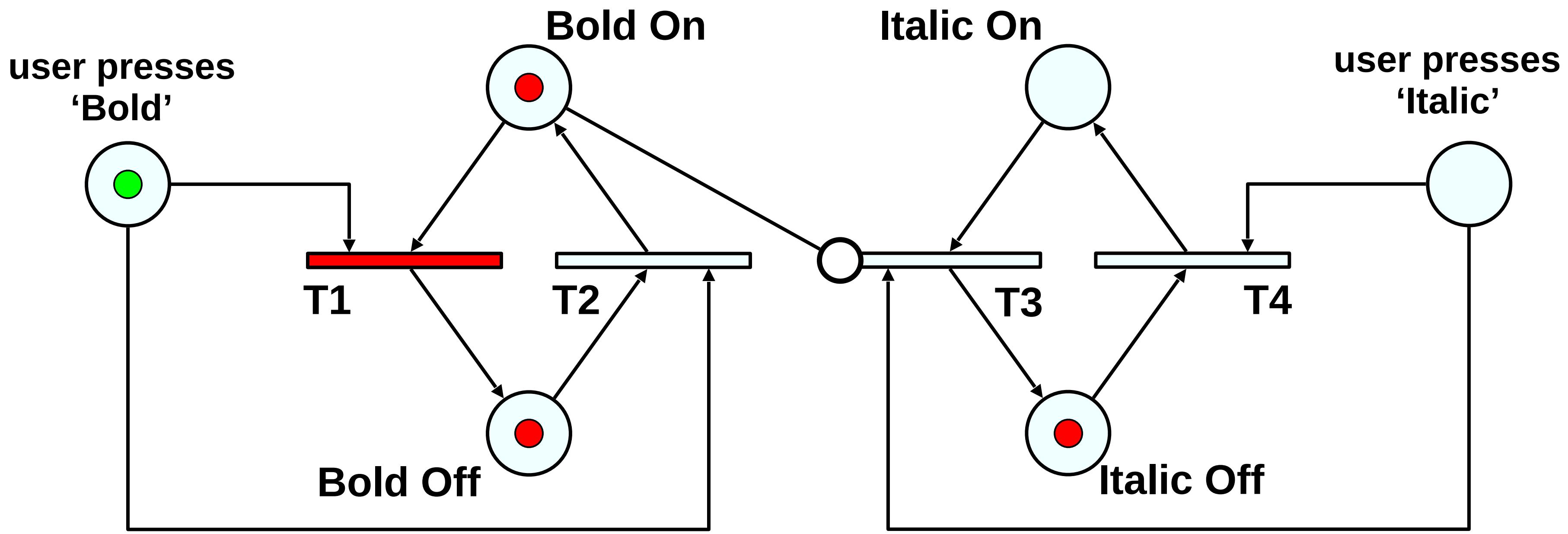


- One of the oldest notations in computing!
- **Flow graph:**
  - **places** – a bit like STN states
  - **transitions** – a bit like STN arcs
  - **counters** – sit on places (current state)
- **Several counters allowed**
  - concurrent dialogue states
- **Used for UI specification**

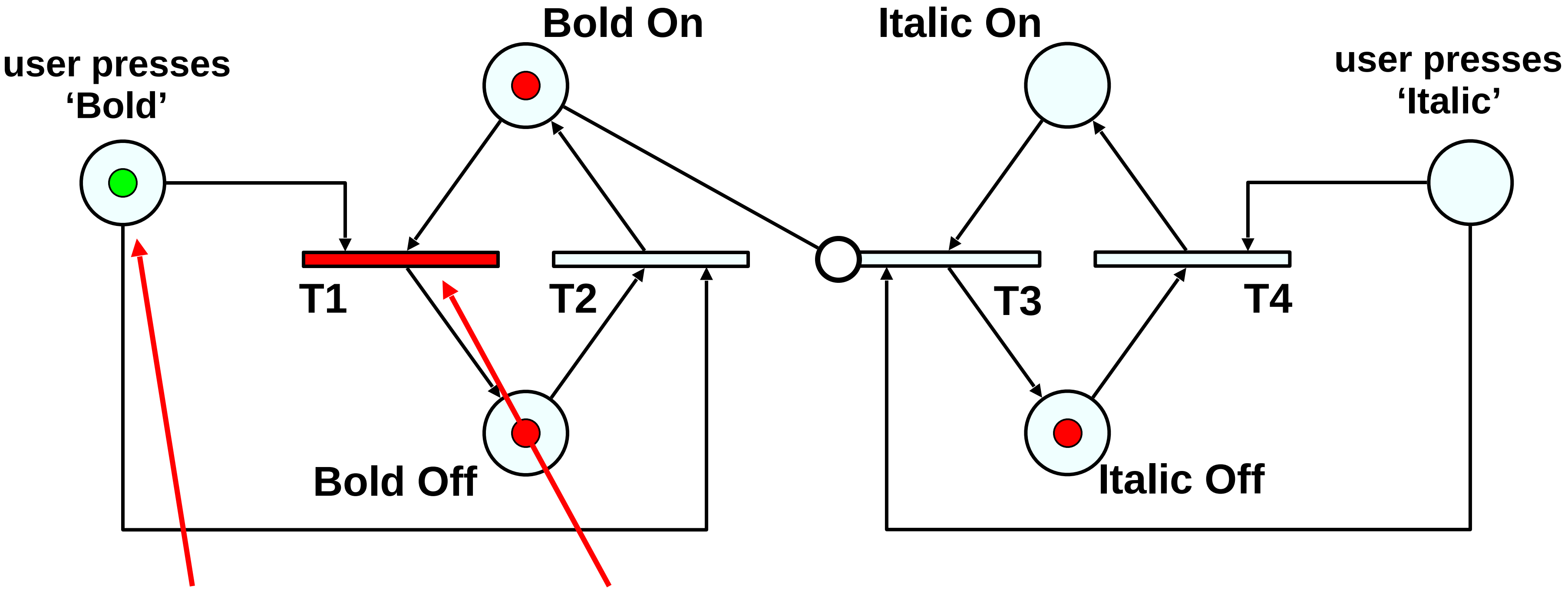
  (ICO at Toulouse)
  - tool support – *Petshop*
- Reasoning about concurrent activities.



Petri net example



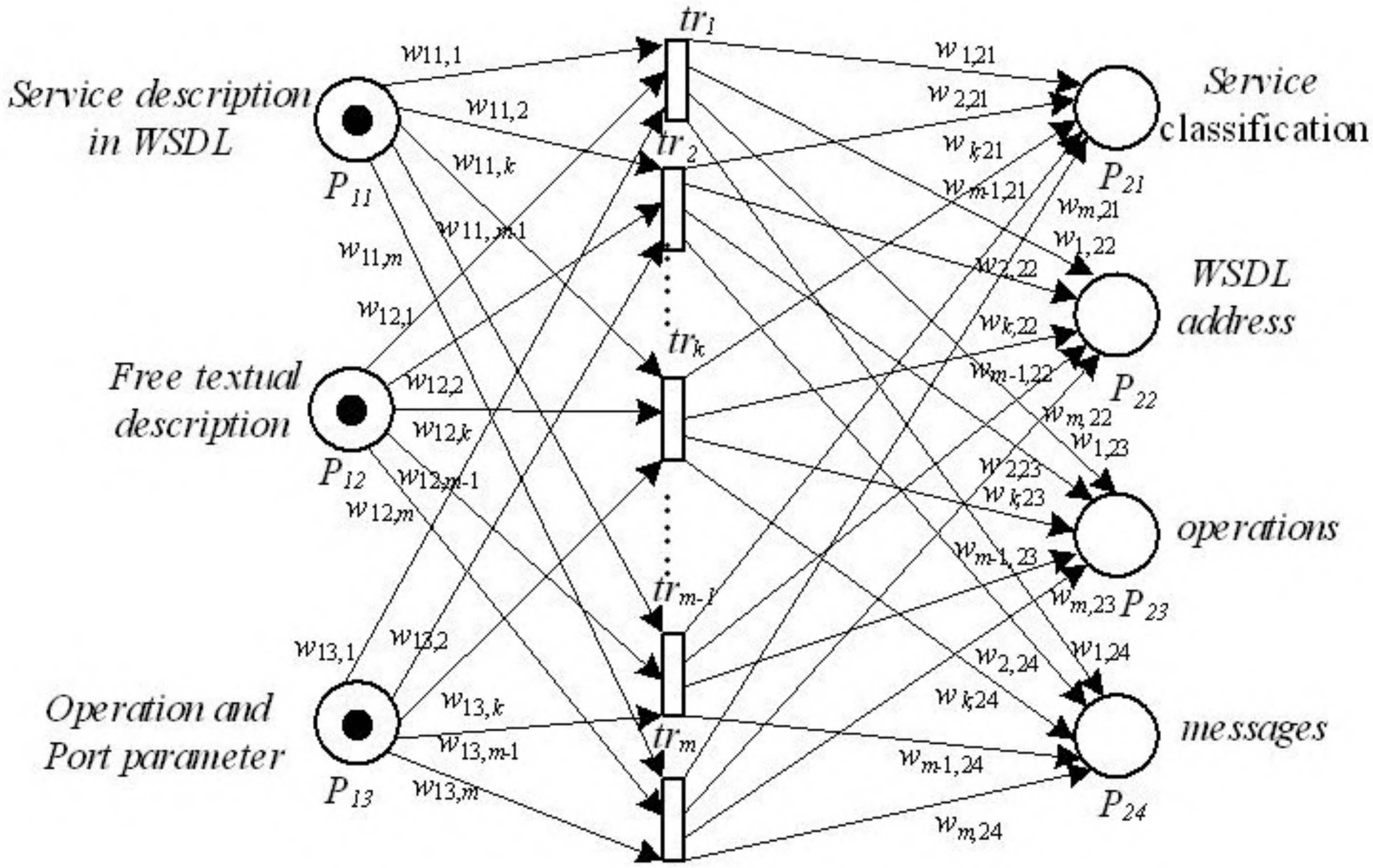
Petri net example



user actions represented as a new counter

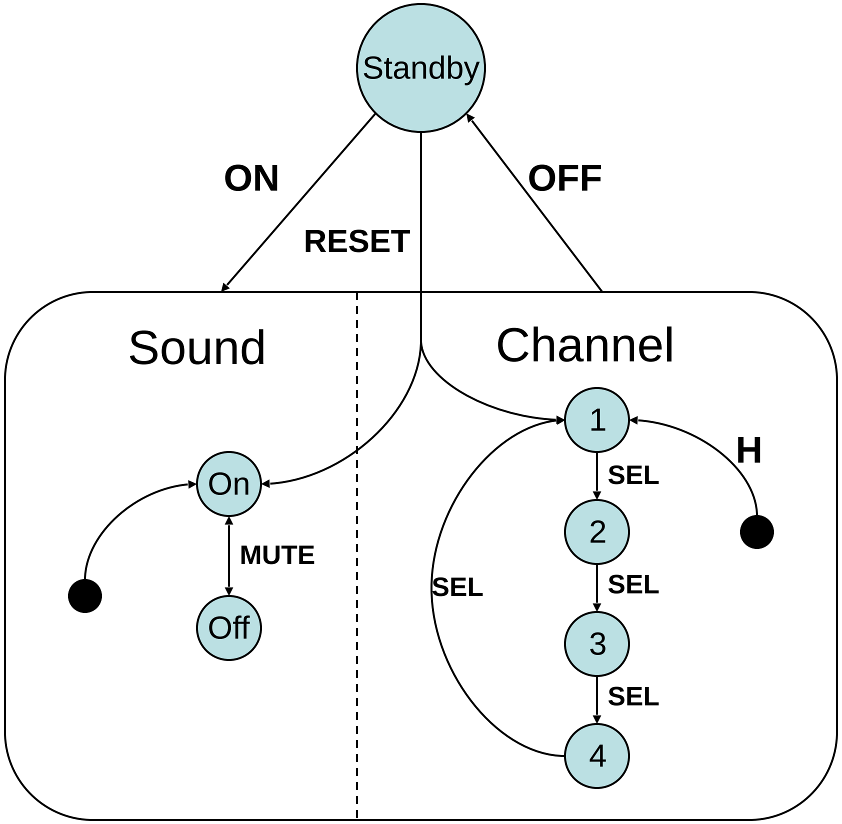
transition 'fires' when all input places have counters

Petri net example



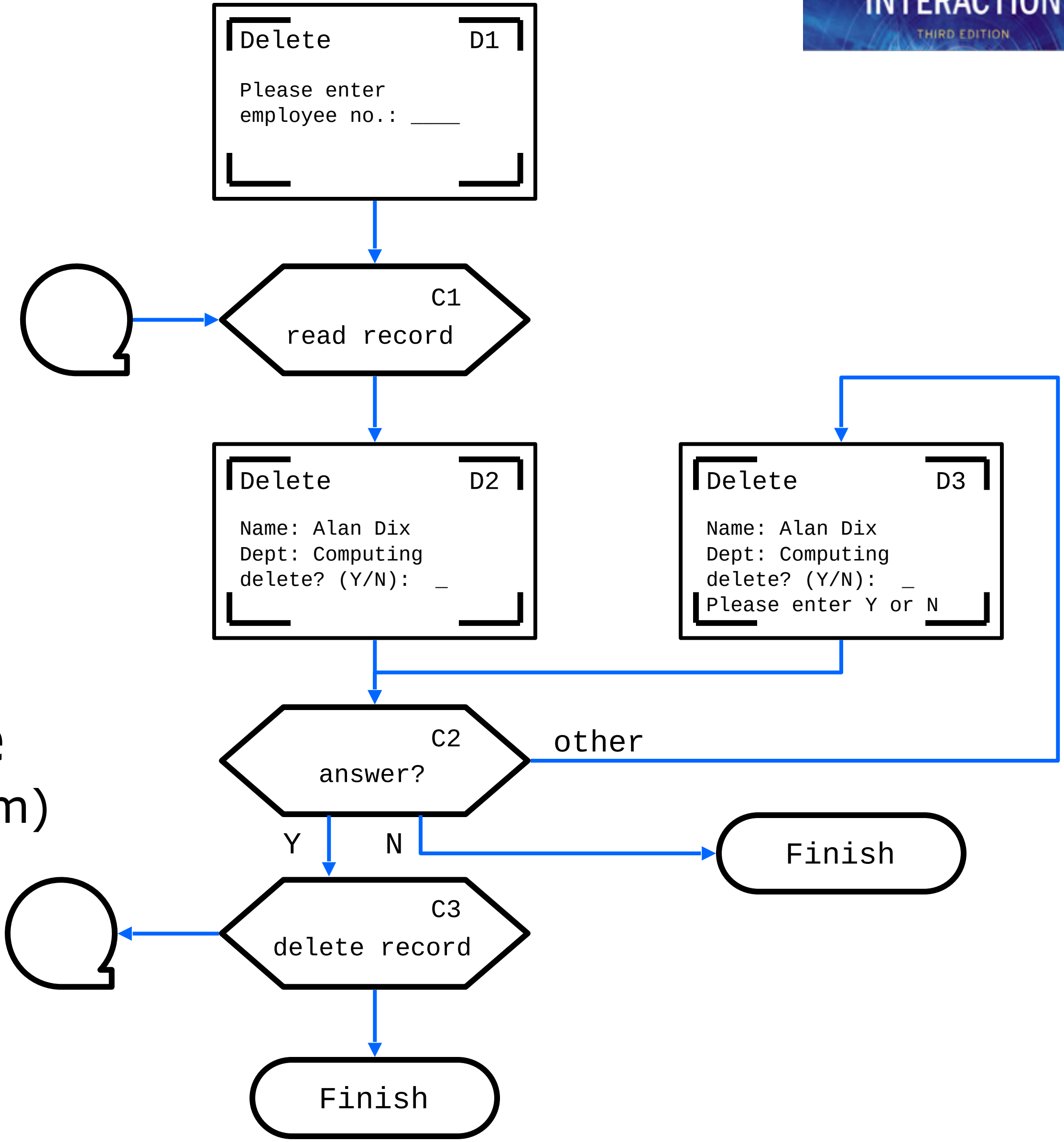
State Charts

- Used in UML
- Extension to **STN**
  - hierarchy
  - concurrent sub-nets
  - escapes
    - OFF always active
  - history
    - link marked H goes back to last state on re-entering subdialogue



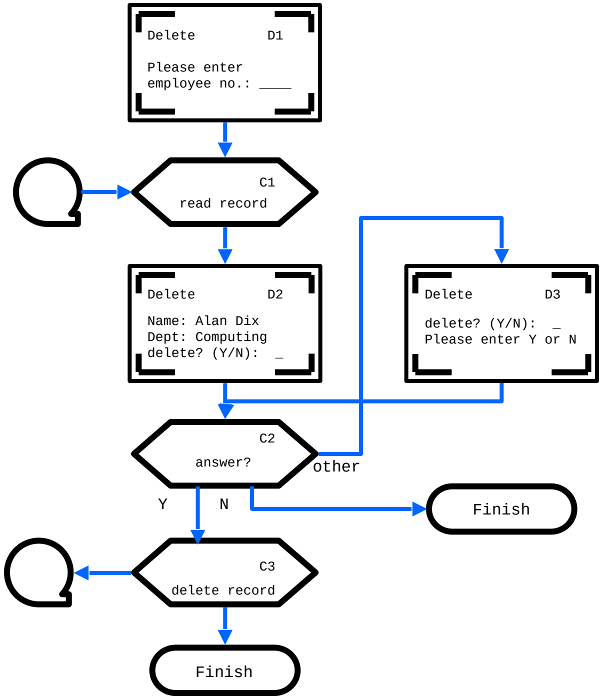
Flowcharts

- Familiar to programmers
- Boxes
  - process/event
  - not state
- Use for dialogue
  (not internal algorithm)



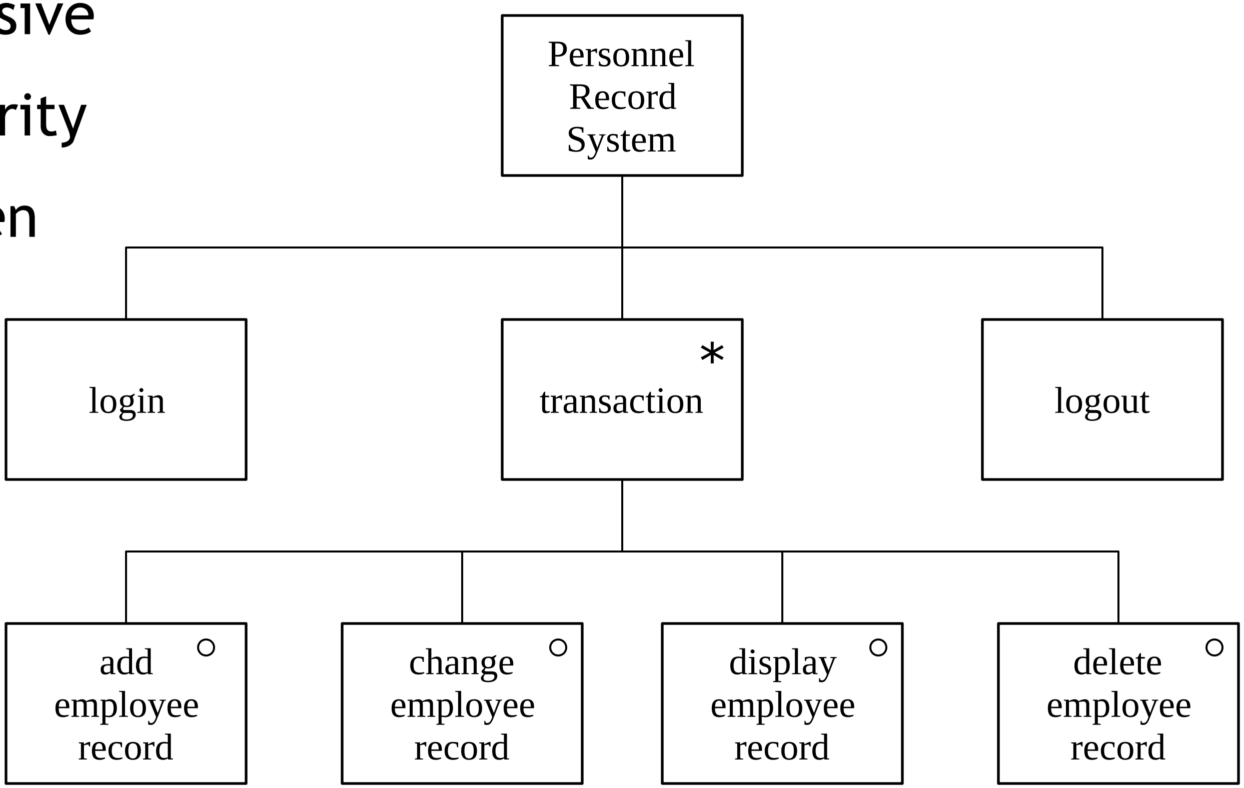
Case study - it works!

- Formal notations – too much work?
- COBOL transaction processing
  - event-driven – like web interfaces
  - programs structure
    ≠ dialogue structure
- Used dialogue flow charts
  - discuss with clients
  - transform to code
  - systematic testing
  - 1000% productivity gain
- Formalism saves time !!!



JSD – Jackson Structured Design diagrams

- For tree structured dialogues
  - less expressive
  - greater clarity
  - menu-driven



Textual Notations

grammars
production rules
CSP and event algebras

Textual - Grammars

- Regular expressions

```
sel-line click click* dble-click
```

- compare with JSD
  - same computational model
  - different notation
- Backus-Naur Form (BNF)

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

- more powerful than regular exp. or STNs
- Still NO concurrent dialogue

Backus-Naur Form (BNF)

- Very common notation from computer science
- A purely syntactic view of the dialogue
- Terminals
  - lowest level of user behavior
  - e.g. CLICK-MOUSE, MOVE-MOUSE
- Nonterminals
  - ordering of terminals
  - higher level of abstraction
  - e.g. select-menu, position-mouse

Example of BNF

- **Basic syntax:**
  - nonterminal ::= expression
- **An expression**
  - contains terminals and nonterminals
  - combined in sequence (+) or as alternatives (|)

```
draw line    ::=  select line + choose points + last point
select line ::=  pos mouse + CLICK MOUSE
choose points ::=  choose one | choose one + choose points
choose one ::=  pos mouse + CLICK MOUSE
last point ::=  pos mouse + DBL CLICK MOUSE
pos mouse   ::=  NULL   |  MOVE MOUSE+ pos mouse
```

Production rules

- Unordered list of rules:

if *condition* **then** *action*

  - condition based on state or pending events
  - every rule always potentially active

- Good for concurrency
- Bad for sequence

Dialogue Analysis - Summary

- Semantics and dialogue
  - attaching semantics
  - distributed/centralized dialogue description
  - maximizing 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