Interacção Humana com o Computador

Aula 10

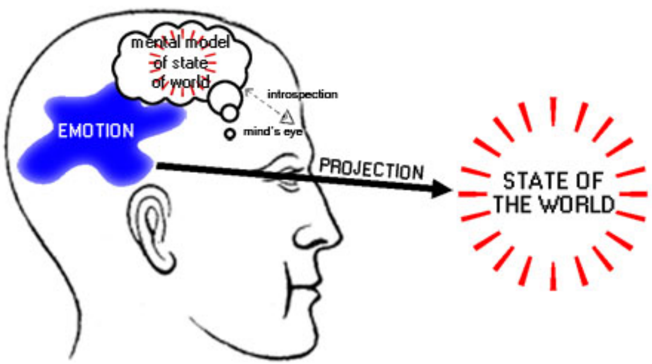
Departamento de Informática
UBI 2024/2025

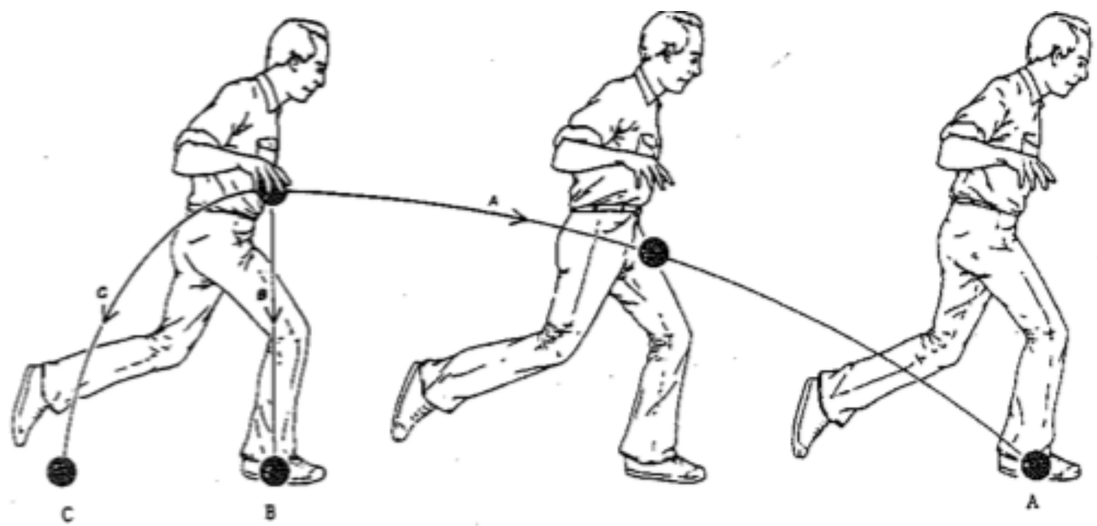
João Cordeiro
jpcc@ubi.pt

chapter 12

Cognitive Models

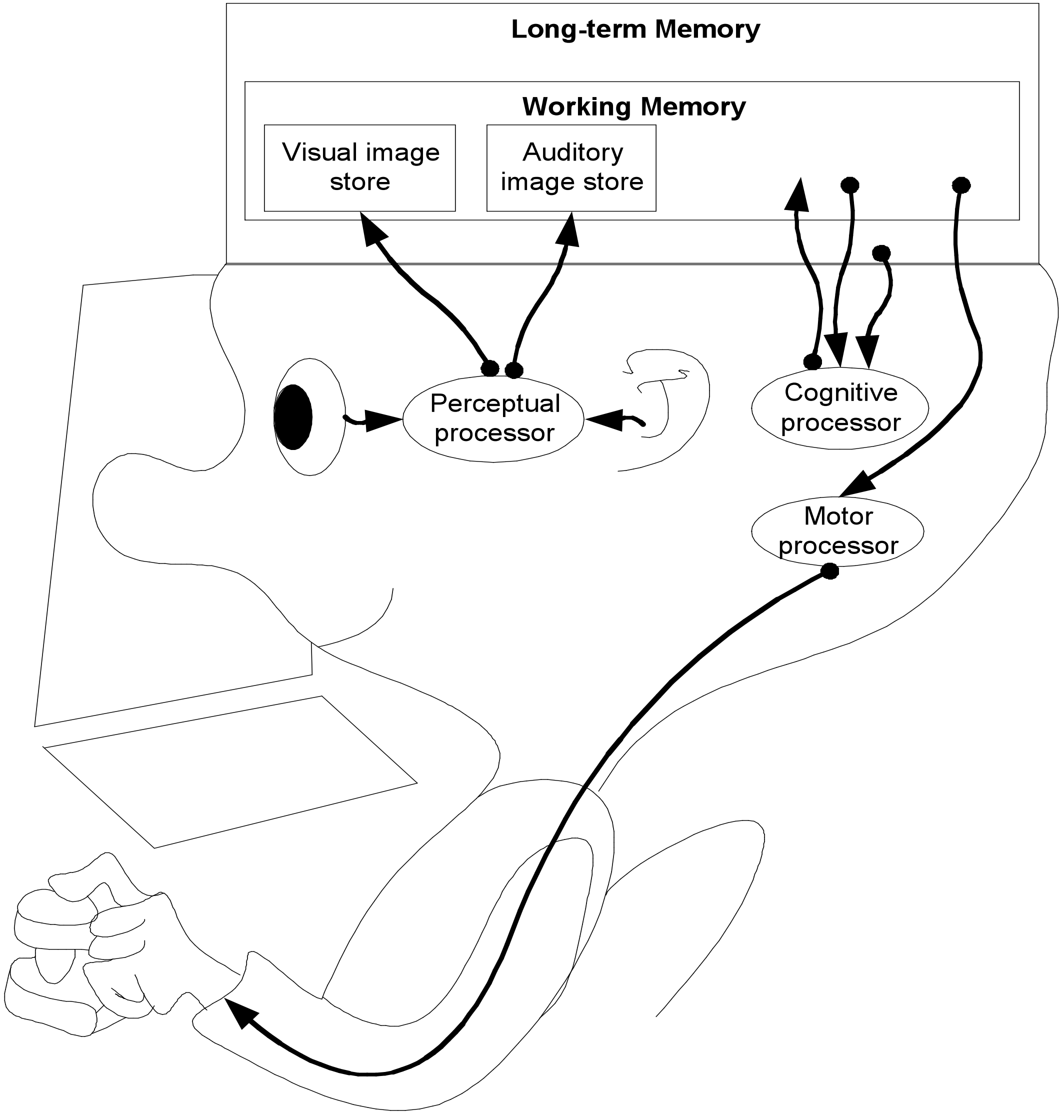
Humans create mental models to explain behavior





Cognitive Models

How do users
perceive,
think
and act



Donald Norman's Model

- Seven stages
  1. User establishes the goal
  2. Formulates intention
  3. Specifies actions at interface
  4. Executes action
  5. Perceives system state
  6. Interprets system state
  7. Evaluates system state with respect to goal

- Norman's model concentrates on user's view of the interface

Donald Norman's Model

Some systems are harder to use than others

Why?

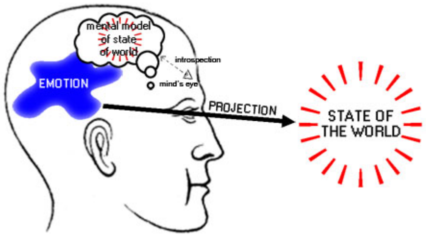
Gulf of Execution

user's formulation of actions ≠ actions allowed by the system

Gulf of Evaluation

user's expectation of changed system state ≠ actual presentation of this state

Cognitive Models



- **Hierarchical models**
  Represent a user's task and goal structure

- **Linguistic models**
  Represent the user-system grammar

- **Physical and device models**
  Represent human motor skills

- **Cognitive architecture**
  Underline all all of these cognitive models

Exercise

```
GOAL: PHOTOCOPY-PAPER
.       GOAL: LOCATE-ARTICLE
.       GOAL: PHOTOCOPY-PAGE repeat until no more pages
.       .       GOAL: ORIENT-PAGE
.       .       .    OPEN-COVER
.       .       .    SELECT-PAGE
.       .       .    POSITION-PAGE
.       .       .    CLOSE-COVER
.       .       GOAL: PRESS-COPY-BUTTON
.       .       GOAL: VERIFY-COPY
.       .       .    LOCATE-OUT-TRAY
.       .       .    EXAMINE-COPY
.       GOAL: COLLECT-COPY
.       .       LOCATE-OUT-TRAY
.       .       REMOVE-COPY
.       GOAL: RETRIEVE-JOURNAL
.       .       OPEN-COVER
.       .       REMOVE-JOURNAL
.       .       CLOSE-COVER
```

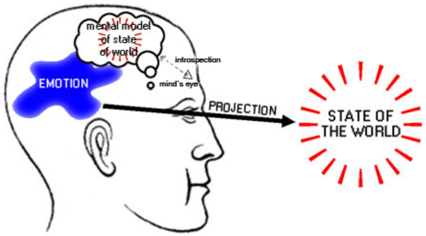
ALAN DIX, JANET FINLAY,
GREGORY D. ABOWD, RUSSELL BEALE
HUMAN-COMPUTER INTERACTION
THIRD EDITION

Exercise

```
GOAL: PHOTOCOPY-PAPER
.     GOAL: LOCATE-ARTICLE
.     GOAL: PHOTOCOPY-PAGE repeat until no more pages
.     .     GOAL: ORIENT-PAGE
.     .     .   OPEN-COVER
.     .     .   SELECT-PAGE
.     .     .   POSITION-PAGE
.     .     .   CLOSE-COVER
.     .     GOAL: PRESS-COPY-BUTTON
.     .     GOAL: VERIFY-COPY
.     .     .   LOCATE-OUT-TRAY
.     .     .   EXAMINE-COPY
.     GOAL: COLLECT-COPY
.     .     LOCATE-OUT-TRAY
.     .     REMOVE-COPY (outer goal satisfied => error)
.     GOAL: RETRIEVE-JOURNAL
.     .     OPEN-COVER
.     .     REMOVE-JOURNAL
.     .     CLOSE-COVER
```

Linguistic models

Cognitive Models



- **Hierarchical models**

  Represent a user's task and goal **structure**

- **Linguistic models**

  Represent the user-system **grammar**

- **Physical and device models**

  Represent human **motor skills**

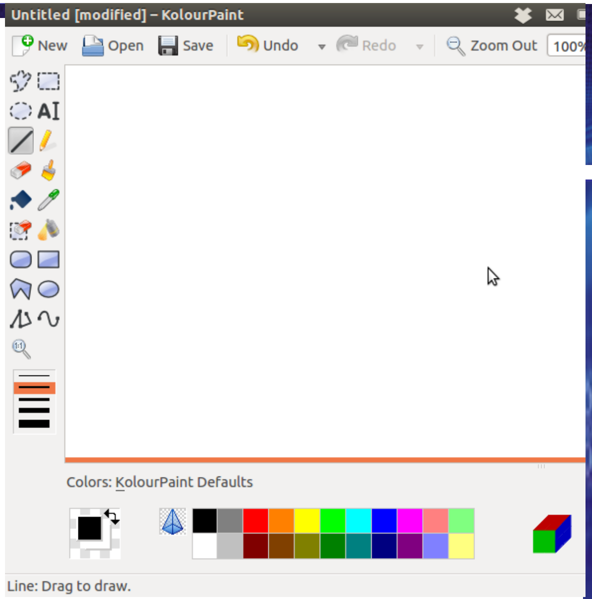
Linguistic Notations

- Understanding the **user's behavior** and **cognitive difficulty** based on language analysis between the user and system.
- Similar in emphasis to **dialogue models**
- Backus–Naur Form (**BNF**)
- Task–Action Grammar (**TAG**)

Backus-Naur Form (BNF)

- Very common notation from computer science
- A purely syntactic view of the dialogue, here from the user perspective
- **Terminals**
  - lowest level of user behaviour
  - 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
```

Difficulty Measurements with BNF

- **Number of** rules (not so good)
- **Number of** + and | operators
- **Measurement problems:**
  - Same syntax for different semantics;
  - No reflection of user's perception of system response;
  - Minimal consistency checking (up ≠ down).

Task Action Grammar (TAG)

- Making consistency more explicit;
- Encoding user's world knowledge;
- Parameterised grammar rules;
- Nonterminals are modified to include additional semantic features.

Consistency in TAG with Pre-Condamd Effects

BNF

Task → Action1 + Action2 + Action3

TAG

Action1: "Open menu"
Preconditions: window is visible
Effects: menu is opened

Action2: "Select option"
Preconditions: menu is opened
Effects: option is selected

Action3: "Confirm selection"
Preconditions: option is selected
Effects: selection is confirmed

Consistency in TAG with Pre-Condamd Effects

BNF

Edit → OpenFile + SelectText + ApplyStyle

TAG

OpenFile:
Preconditions: none
Effects: file is open

SelectText:
Preconditions: file is open
Effects: text is selected

ApplyStyle:
Preconditions: text is selected
Effects: style is applied

Consistency in TAG

- In BNF, three UNIX commands would be described as:

```
copy::= cp+filename+filename | cp+filenames+directory
move::= mv+filename+filename | mv+filenames+directory
link::= ln+filename+filename | ln+filenames+directory
```

- No BNF measure could distinguish between this and a less consistent grammar in which

```
link::= ln+filename+filename | ln+directory+filenames
```

Consistency in TAG (cont'd)

- Consistency of argument order made explicit using a parameter, or semantic feature for file operations

- Feature Possible values

```
Op = copy; move; link
```

- Rules

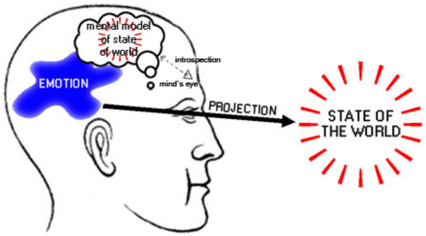
```
file-op[Op]::=   command[Op] + filename  + filename
               | command[Op] + filenames + directory

command[Op = copy] ::= cp
command[Op = move] ::= mv
command[Op = link] ::= ln
```

Other uses of TAG

- User's existing knowledge
- Congruence between features and commands
- These are modelled as derived rules

Physical models

Cognitive Models



- **Hierarchical models**

  Represent a user's task and goal **structure**

- **Linguistic models**

  Represent the user-system **grammar**

- **Physical and device models**

  Represent human **motor skills**

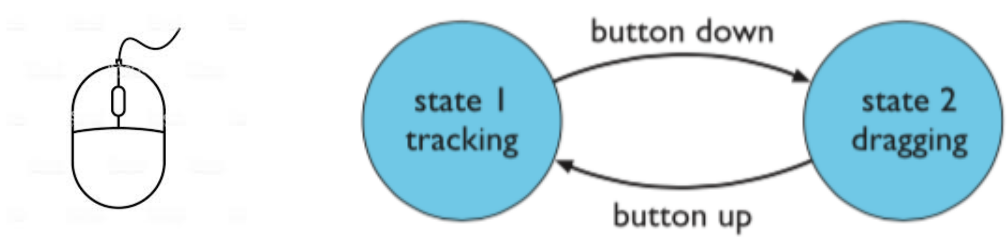
Physical and device models

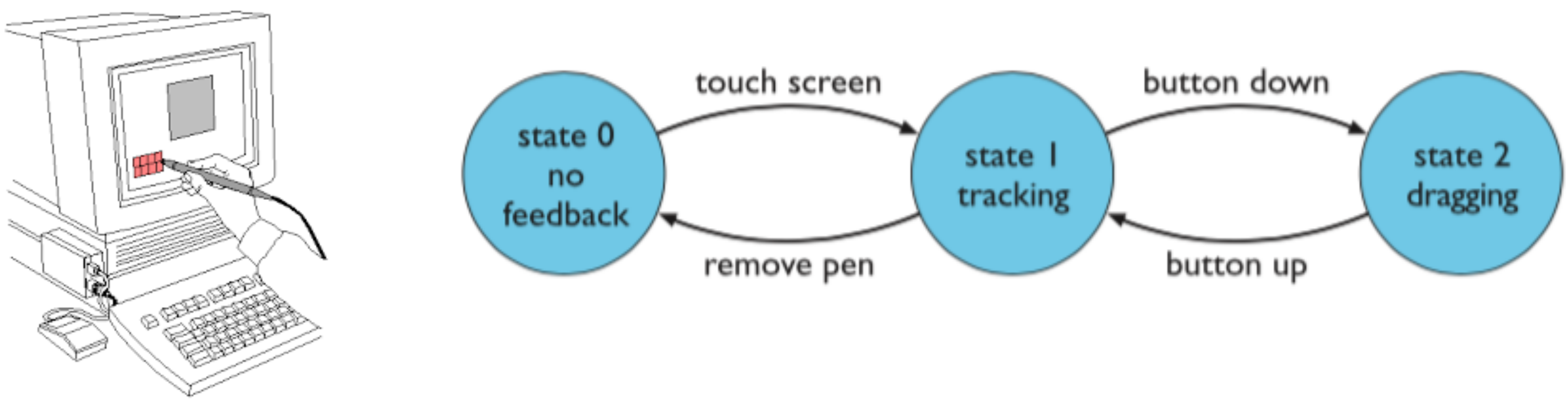
- The **K**eystroke **L**evel **M**odel (**KLM**)
- **Buxton's** 3-state model

- Based on empirical knowledge of human **motor system**;
- User's task: **acquisition** then **execution**.
  - these only address **execution**;
- **Complementary** with goal hierarchies.

Physical and device models

- Buxton's 3-state model





Physical and device models

- Buxton's 3-state model and Fitts' law: $a + b \log_2(D/S + 1)$

Table 12.2 Fitts' law coefficients (after MacKenzie, Sellen and Buxton [221], © 1991 ACM, Inc. Reprinted by permission)

	Device	*a* (ms)	*b* (ms/bit)
Pointing (state 1)			
	Mouse	–107	223
	Trackball	75	300
Dragging (state 2)			
	Mouse	135	249
	Trackball	–349	688

Mouse

$$P[\text{to menu bar}] = -107 + 223 \log_2(11) = 664 \text{ ms}$$
$$P[\text{to option}] = 135 + 249 \log_2(5) = 713 \text{ ms}$$

Trackball

$$P[\text{to menu bar}] = 75 + 300 \log_2(11) = 1113 \text{ ms}$$
$$P[\text{to option}] = -349 + 688 \log_2(5) = 1248 \text{ ms}$$

Keystroke Level Model (KLM)

- Lowest level of (original) **GOMS**
- Six execution phase operators
  - **Physical motor**: **K** - keystroking
P - pointing
H - homing
D - drawing
  - **Mental** **M** - mental preparation
  - **System** **R** - response
- Times (**T**) are empirically determined.

$$T_{execute} = T_K + T_P + T_H + T_D + T_M + T_R$$

KLM example

```
GOAL: ICONISE-WINDOW
[select
    GOAL: USE-CLOSE-METHOD
    .       MOVE-MOUSE-TO- FILE-MENU
    .       PULL-DOWN-FILE-MENU
    .       CLICK-OVER-CLOSE-OPTION
    GOAL: USE-CTRL-W-METHOD
            PRESS-CONTROL-W-KEY]
```

- **Compare alternatives:**
  - USE-CTRL-W-METHOD vs.
  - USE-CLOSE-METHOD
- **Assume hand starts on mouse**

USE-CTRL-W-METHOD		USE-CLOSE-METHOD	
H[to kbd]	0.40	P[to menu]	1.1
M	1.35	B[LEFT down]	0.1
K[ctrlW key]	0.28	M	1.35
		P[to option]	1.1
		B[LEFT up]	0.1
Total	**2.03 s**	**Total**	**3.75 s**

Table 12.1 Times for various operators in the keystroke-level model (adapted from Card, Moran and Newell [56], published and reprinted by permission of Lawrence Erlbaum Associates, Inc.)

Operator	Remarks	Time (s)
K	Press key	
	good typist (90 wpm)	0.12
	poor typist (40 wpm)	0.28
	non-typist	1.20
B	Mouse button press	
	down or up	0.10
	click	0.20
P	Point with mouse	
	Fitts' law	$0.1 \log_2(D/S + 0.5)$
	average movement	1.10
H	Home hands to and from keyboard	0.40
D	Drawing – domain dependent	–
M	Mentally prepare	1.35
R	Response from system – measure	–

wpm = words per minute

Heurísticas do Modelo KLM

- **Identificar as ações e colocá-las como uma sequência de letras K, P, B ou H**
- **Heurísticas para Colocação de Operadores Mentais (M)**
  - **Regra 0 — Inserção Inicial de Operadores Candidatos *M***
    - Inserir ***M*** antes de todos os ***K*** *ou* ***B*** *que representam entradas do utilizador*.
    - Inserir ***M*** antes de todo ***P*** que representa um comando ou inicia uma sequência de manipulação direta.
  - **Regra 1 — Remoção de *Ms* Antecipados**
    - Se um **M** está entre dois operadores que <u>variam muito de duração</u>, então este **M** deve ser eliminado. É assumido que enquanto realiza a primeira operação ele tem tempo de pensar na segunda operação
    - **Exemplo**: **PMK** torna-se **PK**, e **PMBB** torna-se **PBB** (o clique é antecipado enquanto o mouse está sendo movido)
  - **Regra 2 — Remoção de *Ms* dentro de unidades cognitivas**
    - Se uma sequência de **K** forma uma unidade cognitiva (nome de um comando ou argumento), então remover todos os Ms exceto o primeiro.
    - **Exemplo**: Se o comando "`$ dir`" é representado por **MKMKMK**, a sequência correta torna-se **MKKK**

Heurísticas do Modelo KLM

- **Heurísticas para Colocação de Operadores Mentais (continuação)**
  - **Regra 3 — Remoção de *Ms* anteriores a delimitadores consecutivos**
    - Se ***K*** é um delimitador redundante no fim de uma unidade cognitiva (comando), por exemplo um delimitador de um comando imediatamente seguido do delimitador do seu argumento, então remover o ***M***.
  - **Regra 4 — Remoção de *Ms* que são delimitadores de comandos**
    - Se ***K*** é um delimitador de um comando então apagar o *M* em frente
    - Senão:
    - Se o ***K*** é um delimitador para um argumento ou alguma sequência que pode variar manter o *M* em frente
  - **Regra 5 — Remoção de *Ms* sobrepostos**
    - Não contar os ***M*** após ***R***
    - **Exemplo**: um tempo de espera em que o utilizador aguarda uma resposta do sistema

Exemplo de Aplicação do KLM

- Mover a mão para o mouse
  H
- Apontar para o botão apropriado
  HP
- Clicar no botão de rádio
  HPBB
- Apontar para a edit box
  HPBBP
- Clicar na edit box
  HPBBPBB
- Mover a mão para o teclado
  HPBBPBBH
- Digitar a temperatura (“37.8”)
  HPBBPBBHKKKK
- Digitar Enter
  HPBBPBBHKKKKK

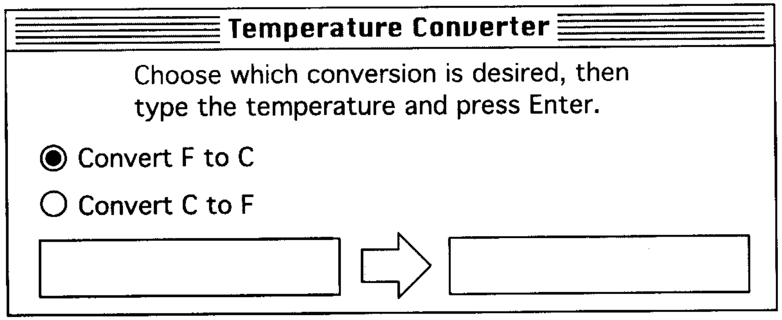
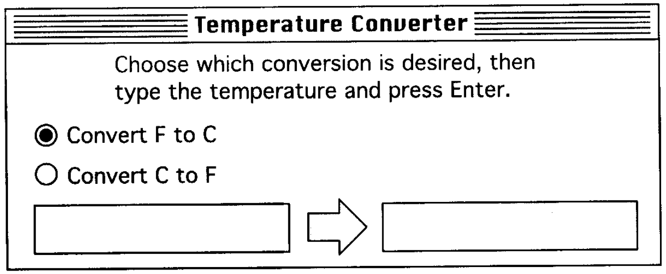


Table 12.1 Times for various operators in the keystroke-level model (adapted from Card, Moran and Newell [56], published and reprinted by permission of Lawrence Erlbaum Associates, Inc.)

Operator	Remarks	Time (s)
K	Press key	
	good typist (90 wpm)	0.12
	poor typist (40 wpm)	0.28
	non-typist	1.20
B	Mouse button press	
	down or up	0.10
	click	0.20
P	Point with mouse	
	Fitts' law	$0.1 \log_2(D/S + 0.5)$
	average movement	1.10
H	Home hands to and from keyboard	0.40
D	Drawing – domain dependent	–
M	Mentally prepare	1.35
R	Response from system – measure	–

wpm = words per minute

Exemplo de Aplicação do KLM



- **Aplicação das Heurísticas**
  - Aplicando a Regra 0:

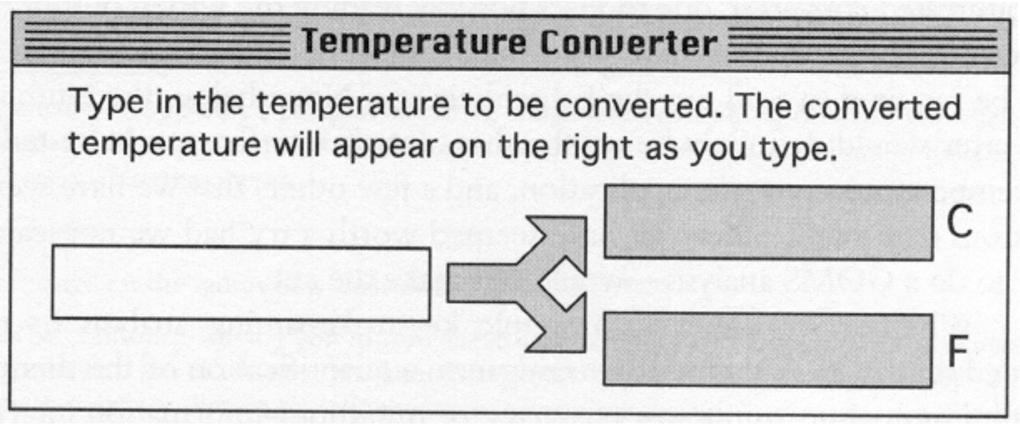
    HMPMBBPBBHMKMKMKMKMK
  - Aplicando a Regra 1: (PMK=PK, PMB = PB)

    HMPBBPBBHMKMKMKMKMK
  - Aplicando a Regra 2:

    HMPBBPBBHMKKKKMK
  - O **M** antes do último **K** tem que ser mantido pela regra 4 e as regras 3 e 5 não se aplicam neste exemplo
  - Substituindo os operadores pelos valores esperados

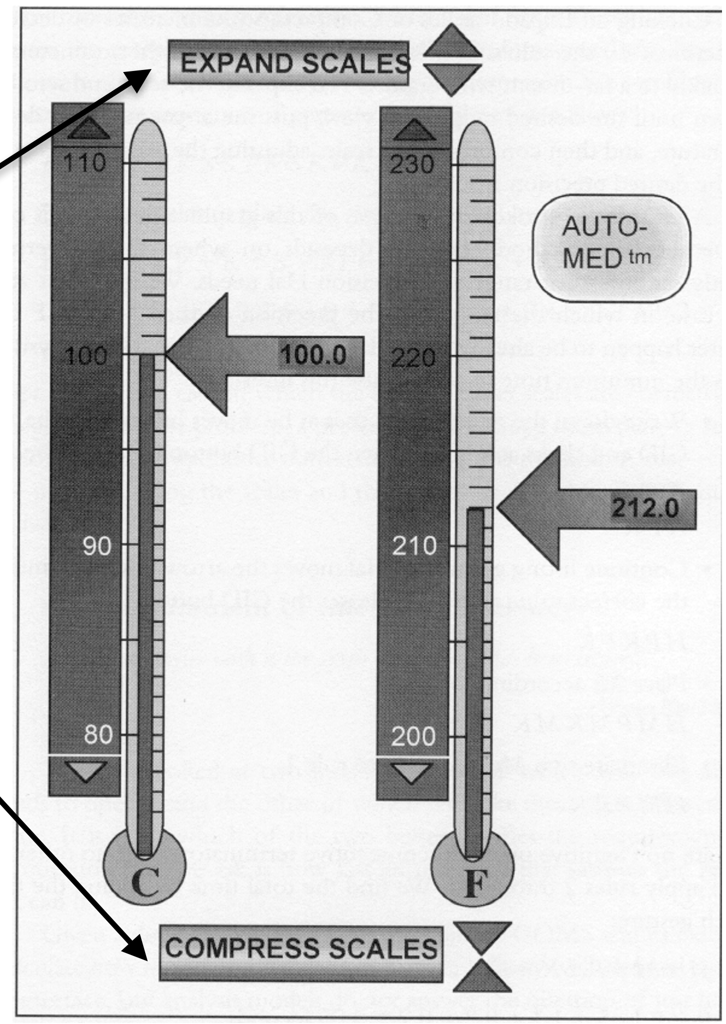
    0.4 + 1.35 + 1.1 + 0.2 + 1.1 + 0.2 + 0.4 + 1.35 + 4*(0.2) + 1.35 + 0.2 = 8.45s
  - No caso em que a conversão já está corretamente selecionada o método é:
    - MKKKKMK = 3.7s
  - Como ambas as conversões são equiprováveis:
    - (8.45+3.7)/2=6.075s

Outras Alternativas



MPKKKK 1.35 + 1.1 + 4*(0.2) = 3.25s

Outras Alternativas



Controlos

MPB 1.35 + 1.1 + 0.1 = 2.46s

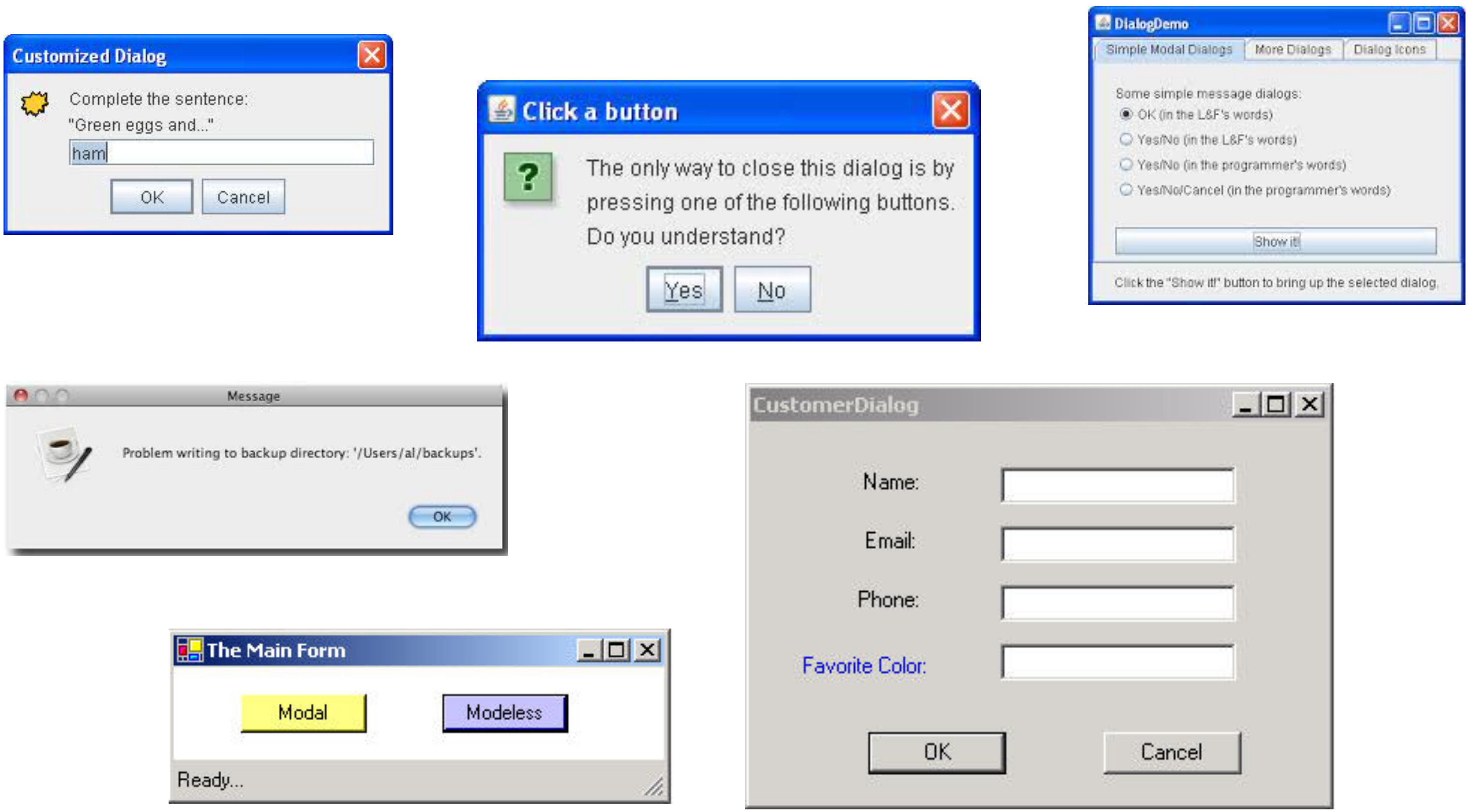
or

MHPB 1.35 + 0.4 + 1.1 + 0.1 = 2.86s

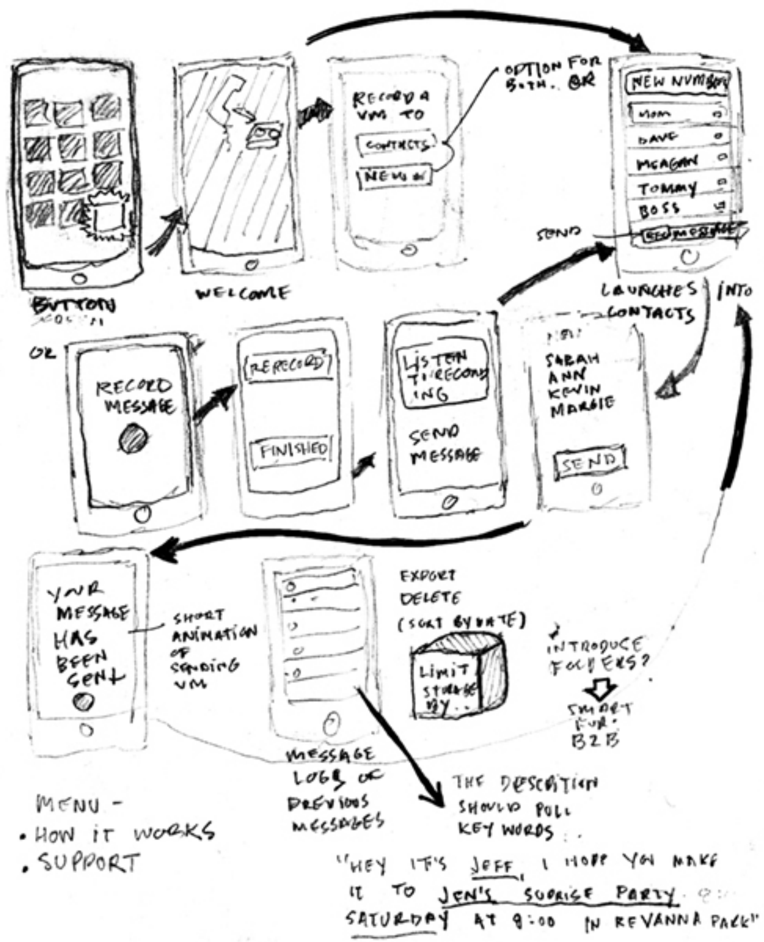
Chapter 16

Dialogue Notations and Design

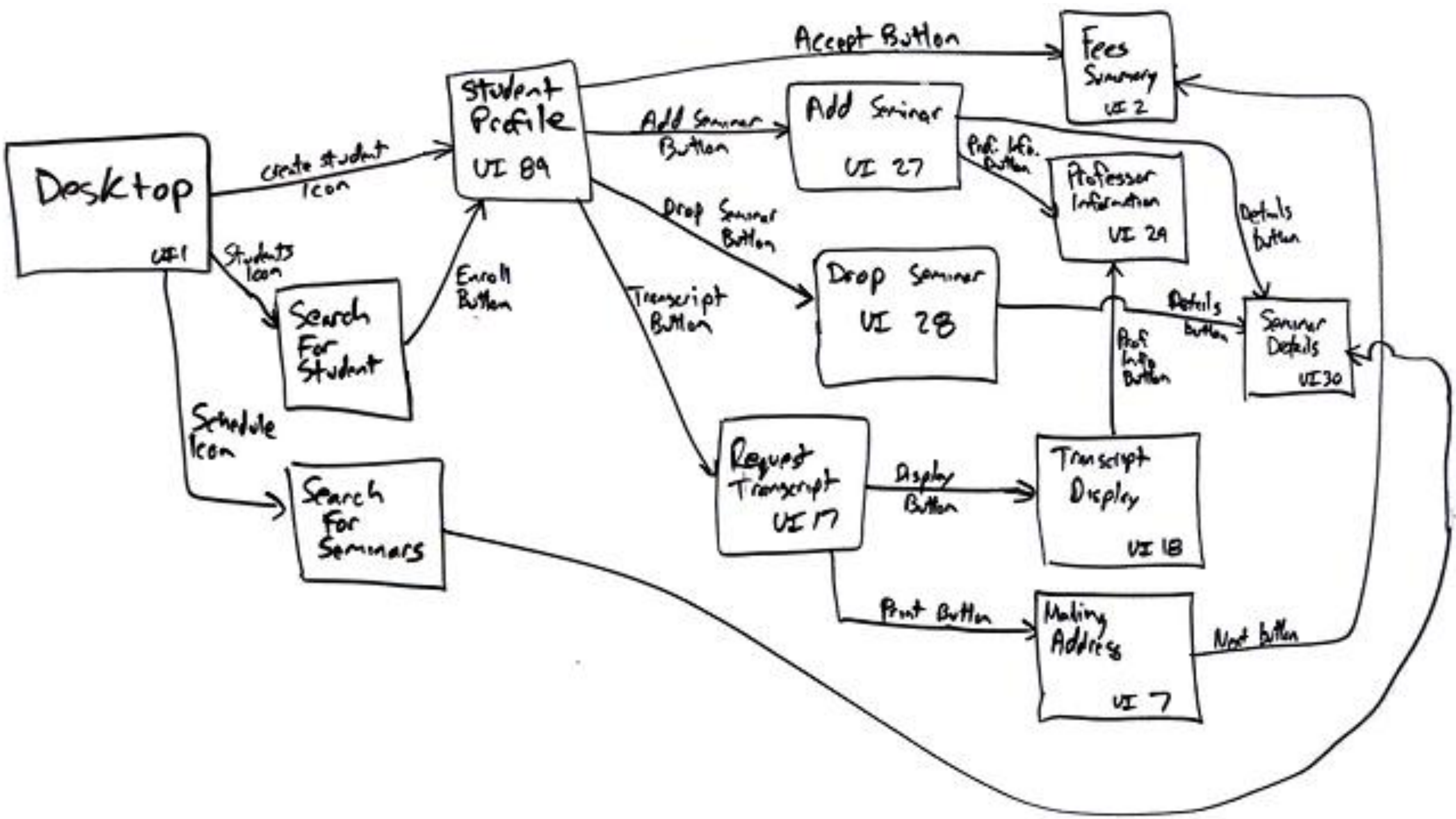
More than just dialog boxes



More than model navigation flow



More than model navigation flow



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 analysed
  - 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 are very constrained
- Some human-human dialogue are 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?

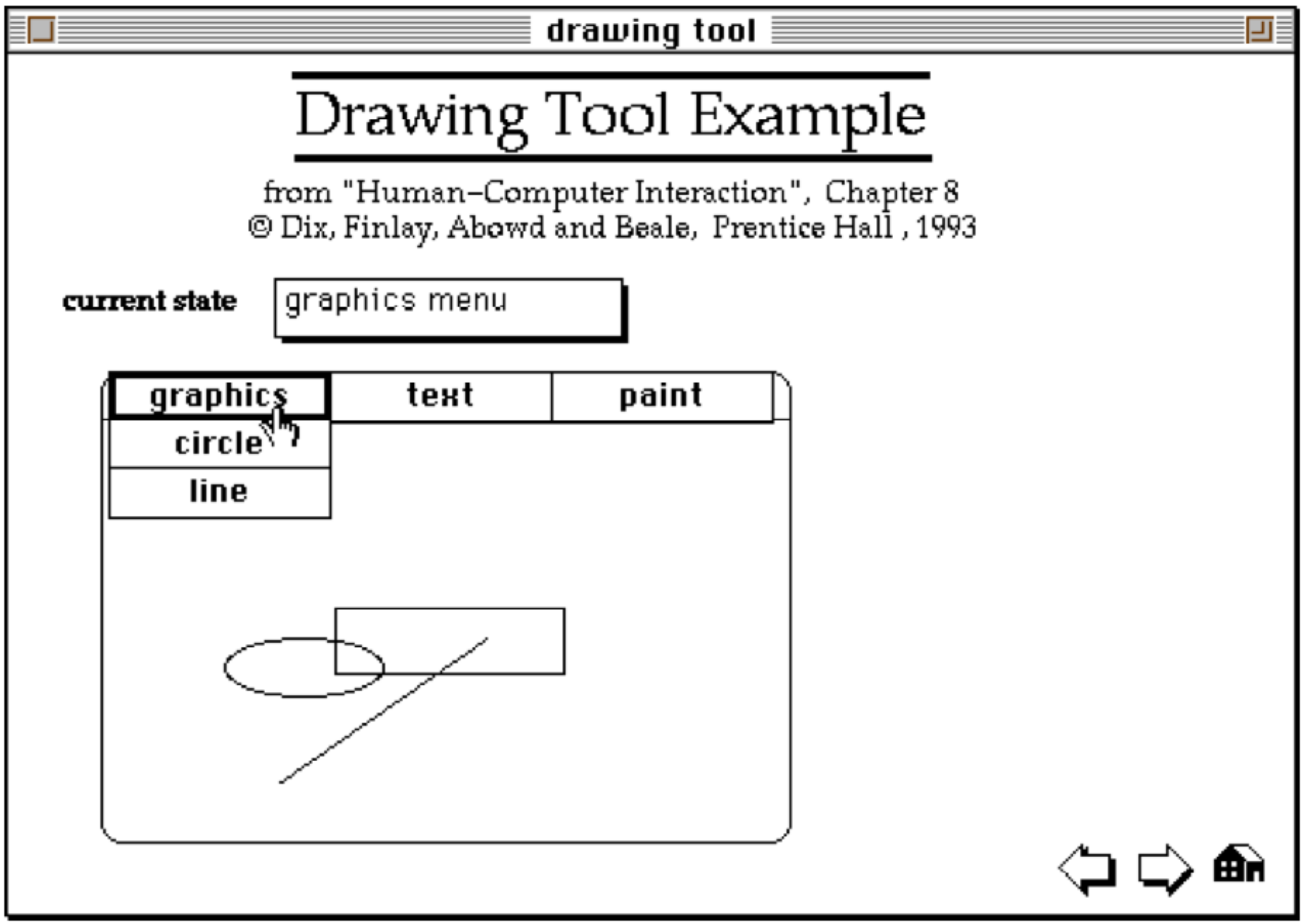
Dialogue design notations

- **Dialogue gets buried in the program**
- **In a big system can we:**
  - analyse the dialogue:
    - can the user always get to see current shopping basket
  - change platforms (e.g. Windows/Mac)
  - dialogue notations helps us to
    - analyse 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)



State Transition Networks (STN)

- circles - states
- arcs - actions/events