Designing Interaction
HCI Lecture 6

David Aspinall

Informatics, University of Edinburgh

9th October 2007

Outline

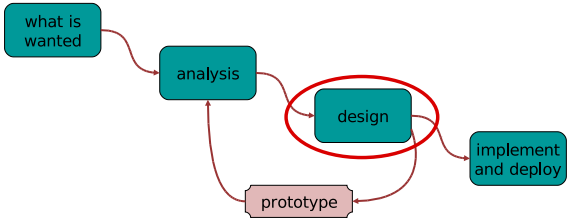
Conceptual Design

Physical Design

Interaction Modes

Exercise

Focus on Design

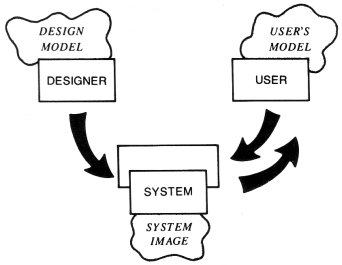


- How do we actually *do* the design?
- Temptation: start sketching windows, menus and buttons. . .
- But we can do better by starting from thinking about the *user experience* we want to provide.

Conceptual Models

A **conceptual model** is the designer's intended mental model for the user of the system: a set of ideas about how it is organized and operates.

Norman (1986) called this the *design model*:



Advantages of a Conceptual Model

- A conceptual model
  - is a starting point for interaction design
  - should help the user "figure it out"
- It helps design team:
  - Not to become narrowly focused early on
  - Ask questions about how the conceptual model will be understood by users
  - Establish a set of common terms they all understand and agree upon (a standard **lexicon** for the project)
  - Reduce the chance of misunderstandings and confusion arising later on

See Johnson and Henderson (2002) for more motivation and methodology.

Objects in the Conceptual Model

The conceptual model should specify:

- **metaphors** or **analogies** used, if any
- the (user-level) **concepts** to be created and manipulated
- the **relationships** between concepts, e.g.

  attributes has-a
  specialisations is-a
  containment contains
- the **mappings** between concepts and task domain

Actions in the Conceptual Model

The conceptual model should also specify/discuss:

- the functions performed and by whom: **task allocation**
- the relationship between functions
  - order relative position; sequential, parallel
  - importance frequency or conceptual importance
  - categorisations e.g., by action taxonomy, or object concerned
- how **data** is captured, transformed, and output

Outputs of Task Analysis can inform object and action analysis for conceptual model.

Example conceptual model (sketch)

Online library

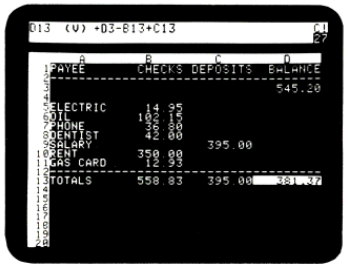
metaphor	information is organised as a physical card catalogue
concepts	*item*, *book*, `periodical`, *issue*, *DVD*, *shelf-mark*, *user account*, *librarian*, . . .
object relationships	a book is a type of item; periodicals contain issues
mappings	*item* corresponds to a physical object; *shelf-mark* to its physical location
functions	*issue item*, *return item*, *search item*
function relationships	issue before return for same item; for different items, in parallel, . . .
data	new items added by typing data

Metaphors

- Interface metaphors combine familiar knowledge with new knowledge in a way that will help the user understand the product.
- Benefits:
  - make learning easier
  - enhances understanding of conceptual model
  - introduce innovation and widen accessibility
- Three steps to consider:
  1. understand functionality
  2. identify potential problem areas
  3. generate metaphors

Classic example: Visicalc (1979)

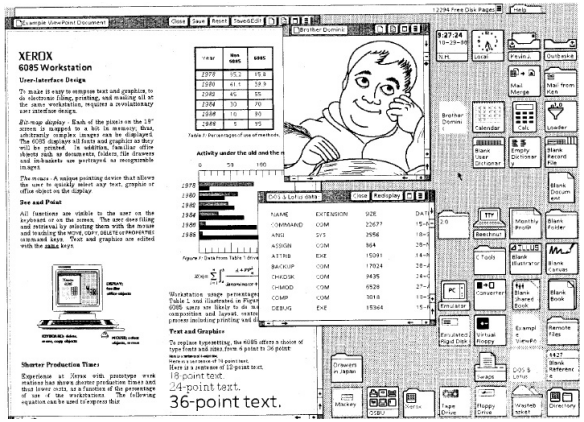
VISICALC™



- Ledger sheet analogy
- Interaction and computation

[See http://www.bricklin.com]

Classic example: The Xerox Star



[http://www.digibarn.com/friends/curbow/star/retrospect/index.html]

Issues with interface metaphors

- A metaphor can have a big impact so should be carefully considered:
  - How much structure does it provide?
  - How much is relevant to the problem?
  - Is it easy to represent?
  - How extensible is it?
- Problems:
  - Break conventional or cultural rules
  - Constrain designers in problem space
  - Conflict with design principles
  - Forces user into one mode of understanding
  - May transfer over bad design
  - May limit imagination for new conceptual model

Infamous failure: Microsoft Bob (1995)

[See http://toastytech.com/guis/bob.html]

Physical Design

- We may have lots or little choice:
  - a new special-purpose physical product, with our own choice of input/output features
  - new I/O mechanisms for existing device
  - new usage of existing mechanisms
  - standard device (e.g., PC) with standard mechanisms
- Recall cognitive and psychological design influences from earlier lectures, used to inform physical design (human motor function, affordances, natural mappings, etc).
- However physical I/O controls are realised, we will want to choose the:
  - *interaction modes* for using inputs
  - *presentation methods* for using outputs

Modes of interaction

Modes of interaction can be classified as:

instructing user tells system what to do, by typing commands, selecting menu options, pressing keys or buttons, speaking commands

conversing user has dialogue with system; typing questions and/or responses, or uses speech input/output

manipulation user interacts with physical or virtual objects, e.g., holding, moving, opening, closing; object is a *focus* of attention

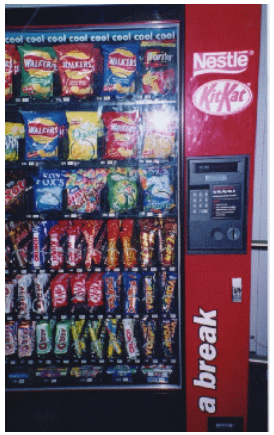
exploration user moves through physical or virtual environment

Other possibilities and higher-level classifications exist, e.g., we may interact by learning, problem solving, socializing, searching, . . .

Instructing

- Examples:
  - Shell command line interpreters for operating systems
  - Menu and key-driven GUI shells for OSes and applications
  - VCRs, hi-fis, alarm clocks, vending machines, etc.
- Advantages:
  - Quick and efficient
  - Good in case of repetition or multiple objects (especially if programmable)
- Disadvantages:
  - Hard to learn
  - Seldom standardised
  - May be overly specific

Vending machines



Conversing

- Examples:
  - Help facilities (Microsoft's Office Assistant paper clip, Bob)
  - Search engines (`http://www.ask.com`, although Jeeves has now retired)
  - Phone services (voice recognition query answering/navigation)
  - Virtual shopping or support assistants
- Advantages
  - No special knowledge required; onus on system to understand user
- Disadvantages:
  - Limited scope of understandability
  - Dialogue can become one-sided and cumbersome

Manipulation

- Shneiderman (1983) coined the term **Direct Manipulation** (DM).
- Digital objects should allow interaction analogous to how physical objects are manipulated
- Core DM principles:
  - Continuous representation of objects and actions
  - Physical **actions and button pressing** instead of issuing commands with complex syntax
  - Rapid **reversible** actions with immediate feedback on object of interest
- Examples:
  - desktop files metaphor in OSes and applications
  - also true manipulable objects: physical objects with sensors (e.g. Wii controller)

Issues around DM

- Advantages of direct manipulation include:
  - Novices can learn the basic functionality quickly
  - Intermittent users can retain operational concepts over time
  - Error messages rarely needed
  - Users can immediately see if their actions are furthering their goals and if not do something else
  - Users experience less anxiety; gain confidence and feel in control
- But there are drawbacks, e.g.:
  - Some people take the metaphor of direct manipulation too literally
  - Not all tasks can be described by objects and not all actions can be done directly
  - Some tasks are better achieved through delegating rather than manipulating e.g., spell checking

Exploring

- Examples:
  - 3D desktop **virtual worlds** where people navigate using mouse around different parts to socialize (e.g., Second Life)
  - **CAVE**s (Computer Automatic Virtual Environment) where users navigate by moving whole body, arms, and head
  - physical **context-aware environments**, embedded with sensors, that present digital information to users at appropriate places and times (e.g. cell phone tourism, smart home)
- Currently rather specialised, will be more important in future with rise of ubiquitous computing.

Exercise: Interface for Robot Cleaner

Design an interface for controlling a robot vacuum cleaner.

1. Extend and deepen the task analysis for house cleaning given in the previous lecture, to consider:
   - individual tasks that performed by the robot
   - interactions necessary to control the robot
2. Propose a suitable conceptual model
3. Consider the physical design of the system
4. . . . and interaction modes that would be appropriate for different tasks.
5. Justify your choices.

References

Jeff Johnson and Austin Henderson.
Conceptual models: begin by designing what to design.
interactions, 9(1):25–32, 2002.

D. A. Norman.
Cognitive engineering.
In *User Centered System Design*, pages 31–61. Lawrence Erlbaum Association, 1986.

Ben Shneiderman.
Direct manipulation: A step beyond programming languages.
IEEE Computer, 16(8):57–69, 1983.

Sharp, Rogers and Preece.
Interaction Design.
Wiley, second edition, 2007.

Further reading: Dix et al, Chapters 6, 7, 8, 18.