Affordances, Metaphors, and Conceptual modeling

Michel Beaudouin-Lafon - mbl@lri.fr
Laboratoire de Recherche en Informatique
In Situ - http://insitu.lri.fr

Outline

"The design of everyday things" - Don Norman

Affordances

Metaphors

Conceptual model

Examples

The design of everyday things - Norman, 1990

Everyday objects reflect the problems in user interface design
- Door handles
- Washing machines
- Telephones
- etc.

Introduces the notions of *affordance*, *metaphor*, and *conceptual model*

Provides a set of *design rules*

Example

Affordances

Direct perception of the capabilities of an object for action

The shape, size, aspect of an object suggests what it can be used for

Gibson's notion of affordance: no learning (innate)
Norman's notion of affordance (cultural affordances): require learning but are universal within a culture

A button is meant to be pushed
A handle is meant to be turned

Foundation of our knowledge of the world

«Much of our everyday knowledge resides in the world, not in the head»
Norman, 1988

Mental model

Operational representation of the world in one's head
Example : instructions to get home

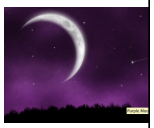
Provides a structure to link causes and effects
Supports explanations
What do I see? What does it meand?
What did just happen? Why?
What did I do that created this situation?
Supports predictions
What can I do now?
What happens if I do this?

Different types of mental models: objects-actions, state-transitions

Metaphor

Figure of speech:
establishes a link between two words, without a comparative
(while comparison includes the comparative)

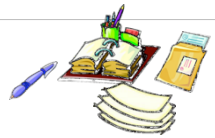
Example: *The moon is a golden sickle*
Direct metaphor *A golden sickle lights the night*



More generally:
Transfers a relationship from one set of objects to another set

To be efficient, the comparative (or the transfer)
must be immediately guessed or understood

Example: Desktop metaphor



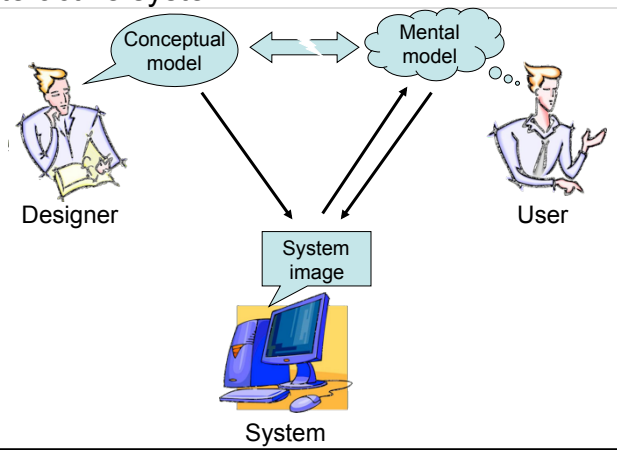
Compares objects of the virtual desk
with objects of a real desk

Transfers properties from physical to on-line world:
Move, Open, put in trashcan, …

The goal is not to simulate a real desk
but to take advantage of our knowledge of a real desk

Goal: Save learning
Capitalize on external knowledge
Takes advantages of affordances in the real world

Interactive system



Conceptual modeling

Conceptual model
- How the designer wants the user to see the system
- Must hide technical aspects
- Must refer to what the user will use the system for

System image
- What the user sees of the system (including its documentation)
- Used by users to create their mental model

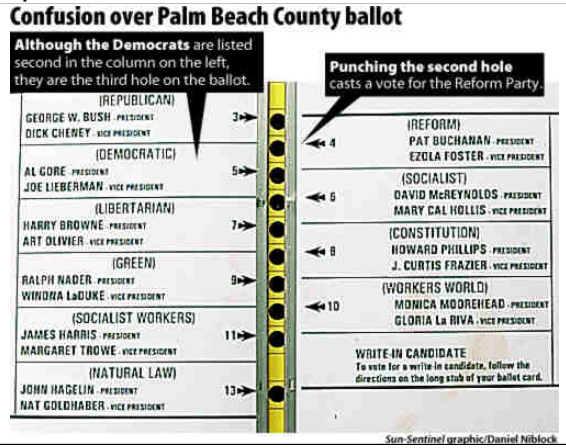
User mental model
- Created based on the users' understanding of the system image, their use of the system, what others have told them about the system, etc.

Conceptual modeling

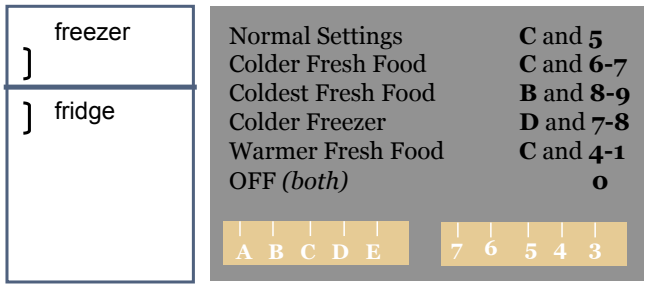
Correspondence between conceptual model and mental model:
- improved by a proper use of metaphors
- improved by taking advantage of affordances
- improved by following proper *design guidelines*

In case of poor correspondence:
- Manipulation errors
- Frustration
- Lower productivity

Example

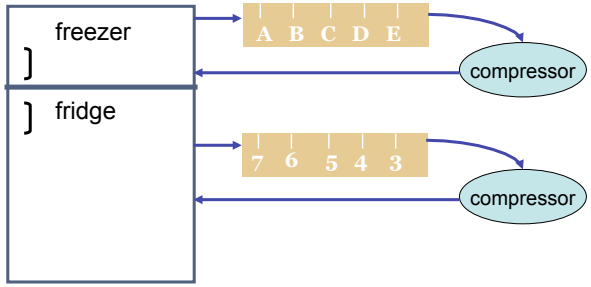


Example : Fridge

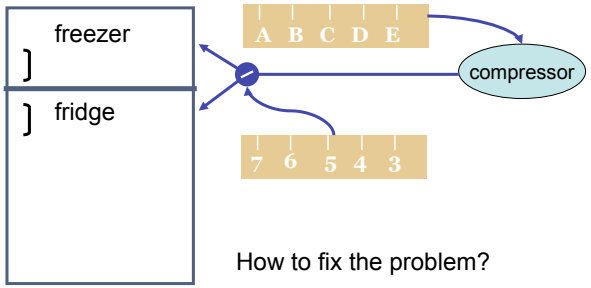


What is your conceptual model?

A likely mental model



Real conceptual model



How to fix the problem?

Change the controls so they match what users expect
Change the controls so they reflect how it really works

3 principles from Norman (1990)

Recommendations to create « good » conceptual models

1- Make things visible
The user can know the state of the system by observing the interface

2- Mapping principle
Help understand the correspondence between
- Actions and results
- Controls and their effects
- The state of the system and what is visible

3- Feedback principle
Inform the user

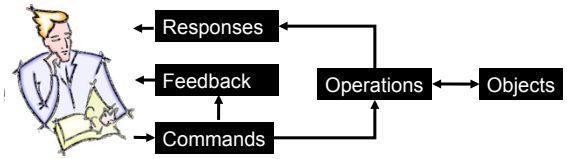
8 design rules from Shneiderman (1998)

1. Consistency
2. Short-cuts for expert users
3. Informative feedback
4. Design dialogues with closures
5. Prevent errors and help repair them
6. Provide reversible operations
7. Give control to the user
8. Reduce short-term cognitive load

There are dozens of recommendations and hundreds of rules…

For every rule, there are exceptions…

Organizing the conceptual model



Identify the objects:
- What the user wants to manipulate

Identify the operations:
- What the user wants to do with the objects

Identify the commands:
- How the user can activate the operations

Interaction tables

Organize the conceptual model into two tables:

Objects	Representations	Properties	Operations
File	Icon (according to file type) + name	Path Type, name, size, …	Delete Rename …

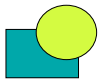
Operations	Commands	Feedback	Responses
Delete a file	Drag-and-drop the icon into the trash	The ghost of the icon follows the cursor	The icon disappears and the trash can gets bigger
	Select file and hit the Delete key	Selected icon gets highlighted	The icon moves towards the trash can and disappears

Case studies

Conceptual models of different graphical editors
- Pixel-based images (Photoshop)
- Vector-based images (Illustrator)

Other case studies (not covered here)
- Editor for images described as planar maps
- Web browser
- File browser
- Text editor
- Mail reader
- ...

Drawing tools



What is this drawing made of?
How to create this drawing?

It is a set of pixels
that can be erased

It is a rectangle and a circle
that can be moved

Two broad categories

Editing **bitmaps** – images made out of pixels
- Basic objects: set of pixels (areas)
- Basic operations:
  - Define an area
  - Apply an operation to the pixels in an area

Editing **vectors** – images made out of geometrical shapes
- Basic objects: a stack of vector-based objects
- Basic operations:
  - Modify the geometry (shape) of an object
  - Modify the graphical attributes of an object
  - Change the stacking order (2D1/2)

Editing bitmaps

Operations	Commands	Feedback	Responses
Select an area	Select rectangle tool + Click-and-drag a rectangle	Cursor change Display ghost rectangle	Area surrounded by “marching ants”
	Select lasso tool + Outline the area	Cusor change Display ghost outline	Area surrounded by “marching ants”
Paint the selected area	Select brush tool + Click-and-drag to paint	Cursor change Display ink	Apply current color to the path of the brush
	Select paint bucket tool + Click the area	Cursor change	Selected area is filled with the current color

Editing bitmaps

Operations	Commands	Feedback	Responses
Modify the selected area	Command “Invert” in the “Selection” menu		Exchanges the selected and non-selected areas
	Command “Extend” in the “Selection” menu		Extends the selection by one pixel
Transform the selected area	Select an item in the “Filters” menu	Dialog box with parameters of the filter	Apply the filter to the selected area
	etc.	...	...

Editing bitmaps

Objects	Representations	Properties	Operations
Area	“Marching ants” (blinking outline)	The set of pixels inside the area	Define Modify Fill
Brush	Cursor shape	Shape Transparency Color	Paint
Tool palette	Floating window	List of tools Selected tools	Select tool
etc.	...		

Vector-based editing

Operations	Commands	Feedback	Responses
Create an object	Select an object type in the palette + Click-and-drag	Cursor change Rubber-band the object shape	Creates new shape with current attributes on top of all other
	Select the pencil + Click-and-drag each control point	Cursor change Each click-and-drag defines a point and its tangent	Creates new shape with current attributes on top of all other shapes
Select one or more object	Click an object		Adds handles to the selected object
	Click on the background+ drag	Ghost of the selection rectangle	Adds handles to the selected objects

Vector-based editing

Operations	Commands	Feedback	Responses
Modify the geometry of an object	Select object + click-and-drag the handles	Ghost of the reshaped object	Changes the shape of the object
Modify the attributes of an object	Click object + Use the attributes inspector	Values of the attributes are displayed in inspector	Applies new values to the object
Change the stacking order	Click object + select command “bring to front” or “send to back”		Puts the object on top or below all others
	Click object + select command “Order” + slider	The stacking of the object changes according to the slider	Changes the stacking order of the object

Vector-based editing

Objects	Representations	Properties	Operations
Vector-based shapes	Graphical shape	Geometry Graphical attributes	Create Modify Change attributes
Attribute inspector	Floating window	Background color Foreground color Thickness Transparency	Change attribute value
Tool palette	Floating window	List of tools Selected tool	Select
etc.	...		

Some rules

Group commands by category
- Manage the workspace
- Global editing (layout of objects, ...)
- Local editing (individual object)
- etc.

Verify completeness
- Same operations in both tables
- Each property should be visible and editable

Verify consistency
- Similar interactions have similar effects

Some rules

Apply design principles
- Reification
  - Identify new objects
  - ex : Tool palette = object
- Polymorphism
  - Create commands that apply to different objects
  - ex : Which existing commands apply to the palette itself?
- Reuse
  - Output reuse: favor commands that reuse existing objects

Evaluating a conceptual model

Using *scenarios* and *storyboards*
- Describe realistic sequences of interaction
- Verify that they are covered by the model

Using *walkthroughs*
- Verify (and have others verify) the criteria described in the previous slides

Using *prototypes*
- Implement some of the techniques to test and refine them

Conclusion

The conceptual model is at the heart of an interactive system

Conceptual modeling is a creative activity
- One cannot simply apply rules

User-centered design
- Analyse interaction from the point of view of the user

Participatory design
- Involve users along the design process to understand their needs, validate design choices, and take advantage of their ideas and suggestions