History of Interactive Computing Systems

Outline

- History of interactive systems
- Models of Interaction

History of User Interfaces

- Batch interfaces
- Conversational interfaces
- Graphical interfaces
- UBICOMP/etc.?

Batch Interfaces

- 1945 – 1965
- Interface:
  - Program submitted via punch cards, tape (paper or magnetic)
    - Punch card deck story
  - Output on a line printer
  - No interaction during execution
  - Program ran to completion (hours or days)

Conversational Interface

- 1965 – 1985
- Interface
  - User types at a terminal
    - One command at a time (occasionally with piping!)
    - Output can be provided to user
    - Typed input can be solicited from user
    - Character based
    - Programs run to completion
  - Later evolved to include daemons
    - Run in background but can be accessed

Batch and Conversational Interfaces

- Expert users
  - Need to understand computer
  - I/O is system language not task language
- Advantages
  - Very flexible
  - Particularly with piping
- Disadvantages
  - Stressed recall (remembering command) over recognition (seeing what is wanted)
  - Why is this bad?

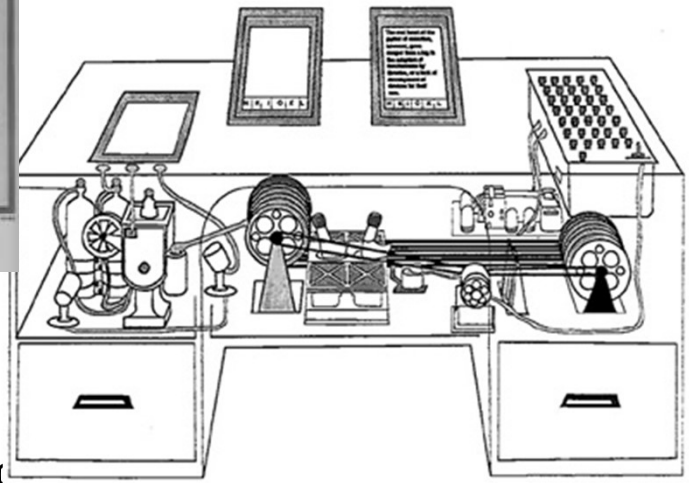
Graphical User Interface

- A type of computational interface that allows users to interact with images instead of issuing commands
- Represents the information and actions available to a user through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation.
  - Secondary notation = color, indentation, size, etc.
- Actions are usually performed through direct manipulation of the graphical elements
- Moves to recognition (as opposed to recall) centric interfaces
- Perceived of as good for novice users, but not really the case
  - Why?
  - For recognition to work, a priori knowledge was somewhere in brain
  - Good for non-expert users

Evolution of GUI

- Several visionaries lead to development
  - Vannevar Bush
    - Memex
  - Ivan Sutherland
    - Graphical input
  - Doug Englebart
    - Mouse
  - Alan Kay

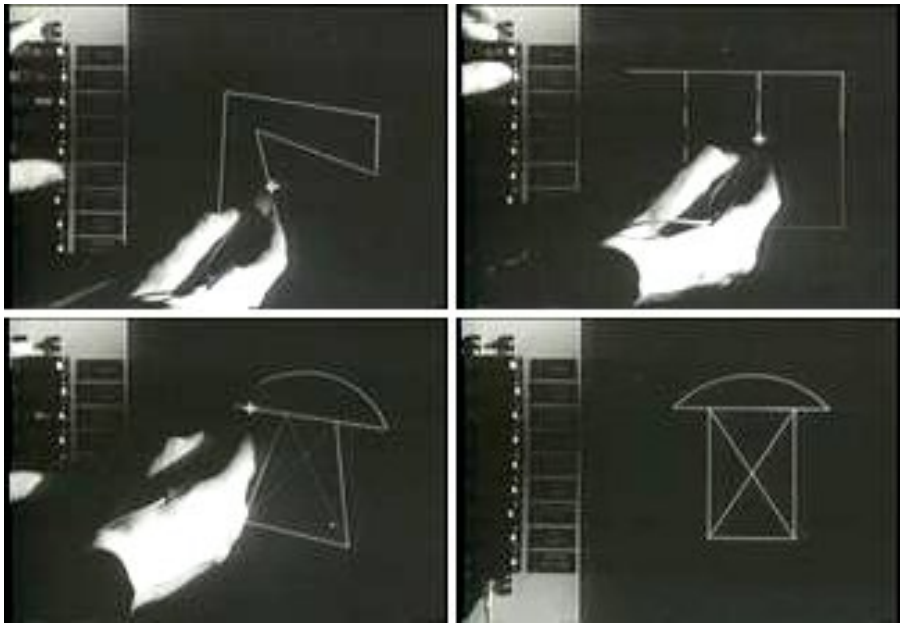
Vannevar Bush



- Memex
  - From an Article:
    - As we may think
  - Published in July, 1945
  - Predicted
    - Mass storage and retrieval
      - A library of a million volumes could be [illegible] desk.
    - Digital cameras
      - Often it would be advantageous to be able to snap the camera and to look at the picture immediately.
    - Fax machine
      - This whole apparatus constitutes a camera, with the added feature, which can be dispensed with if desired, of making its picture at a distance
    - Hyperlinking

Ivan Sutherland

- 1963
- Defined computer graphics
  - Pen-based input, CAD, etc.
  - http://www.archive.org/details/AlanKeyD1987
    - 4:35



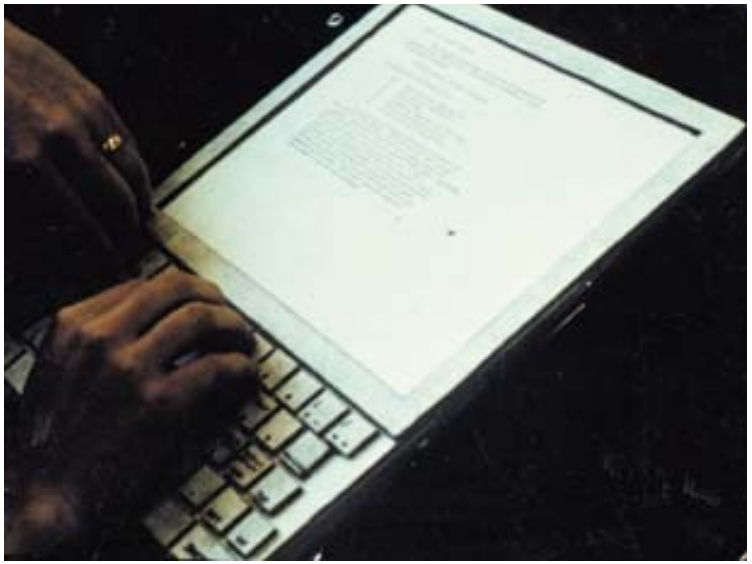
Douglas Englebart

- Inventor of mouse, chord keyboard, cut/copy/paste, drag and drop, etc.

- 1968 Live demo:
  - http://video.google.com/videoplay?docid=-8734787622017763097
  - 3:25
  - 4:30

Dynabook

- Alan Kay et al. at Xerox PARC
  - 1968 – 1970s



Graphical User Interface

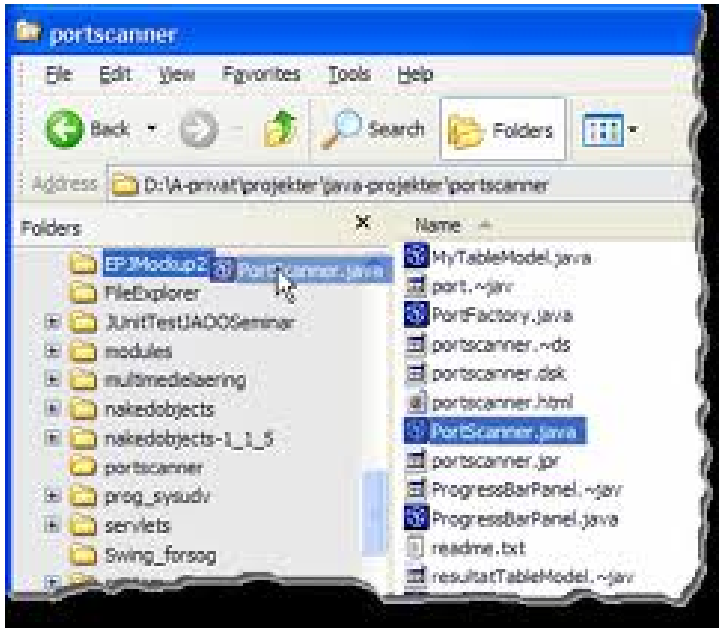
- Xerox 8010 Star
  - 1982 – 1984
- Eventually (1984)
  - Apple said “Good idea”
  - 1990 Microsoft
- Interaction
  - User in control
  - System waits for user action
  - Event driven programming
  - Sense of directly manipulating objects
  - Recognition over recall
    - Enables exploration and discovery
  - Real world metaphor
- Two common terms now
  - WIMP
  - Direct Manipulation

WIMP

- Windows, Icons, Menus, Pointer (or Mouse)
- Commands are executed by interacting with “widgets” on the display

Direct Manipulation Interfaces

- Examples – drag and drop, multi-touch
  - Touch objects of interest directly
  - Less of a point at buttons and click to invoke commands



Principles of Direct Manipulation

- Direct manipulation examples abound
  - In traditional GUI (in programs by dragging, in GUI for location-based storage)
  - In video games, CAD, etc.
- Three principles summarize goal of direct manipulation:
  - Continuous representation of objects and actions of interest with meaningful visual metaphors
  - Physical actions or presses instead of complex syntax
  - Rapid, incremental, reversible actions whose effects on objects of interest are immediately visible

DM in RW

- Many different input devices support
  - Mouse
  - Multi-touch displays
  - Tangibles
- Many different technologies
  - For WIMP, mouse, pen, puck, etc.
  - For multi-touch, FTIR, DI/DSI, capacitive touch, etc.
  - For tangibles, computer vision (e.g. OpenCV, ARToolKit), Active objects.
  - Example: Conte crayon and pen-in-hand interaction

Multi-Touch Interaction

- A special case of direct manipulation
- Original research from 1980s @ Toronto by Bill Buxton

History Redux

- All these styles of interaction
- Evolution from research to product
  - Englebart's mouse in 1964 -> Apple computer in 1984
  - Multi-touch ~1984 -> Han's FTIR displays in 2002-3
  - Bill Buxton: "The Long Nose of Innovation"
- Can we conceptualize interaction?
  - Develop models that describe how users interact with computers?

Describing User's Actions

Interaction from a Contextual Design Perspective

- All artifacts:
  - Hierarchical Task Analysis
  - UED
  - Scenarios and storyboards
  - Low fidelity prototypes
- These depict interaction, but they don't provide a conceptual framework for talking about interaction and interaction problems
  - Just breakdowns

Models of interaction (2)

- Will look at 5 models of interaction
  - High-level:
    - Instrumental Interaction
    - Execution-Evaluation Cycle
    - Interaction framework
  - Mid-level:
    - GOMS
  - Low-level
    - KLM

Instrumental Interaction

- GUIs of all variants can be described in various ways
- One common description is “Instrumental Interaction”
- Interface can be decomposed into instruments and objects
  - Instruments are the manipulable components that act on objects
  - Objects are the domain concepts, the knowledge or information, being manipulated by the user through the instruments
    - The information content of the application
    - The purpose BEHIND the interaction
  - In DM, what is the difference between instruments and objects?

Instrumental Interaction (2)

- Goals
  - Describe state of the art interaction techniques
  - Provide metrics (qualitative and quantitative for comparing techniques)
  - Present a design space to explore new ideas
  - Provide guidance for how to integrate new techniques into GUIs
- Defines Interaction Model as (paraphrased from CHI 2000 paper):
  - A set of rules, principles, and properties that guide the define of an interface. It describes how to combine techniques, the “look and feel”, and provides guidance for evaluating specific interaction designs.

Instruments

- WIMP interfaces present a set of instruments to user
  - Activated two ways: spatially and temporally
  - Examples?
- Instruments can also act on other instruments
  - Reification (becoming objects)
  - Meta-instruments (instruments designed to act on other instruments)
  - Examples?

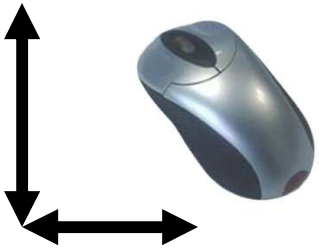
Instruments (2)

- Evaluating instruments
  - Degree of indirection
    - Spatial or temporal offset between instrument and action on object
    - Spatial
      - Close = handles on rectangle to resize
      - Far = dialog boxes
      - Is far always bad?
    - Temporal
      - Real time response versus clicking OK in dialog
  - Degree of integration
    - Ratio of degrees of freedom of instrument to degrees of freedom of input device
  - Degree of compatibility
    - Similarity of action on control device/instrument to action on object

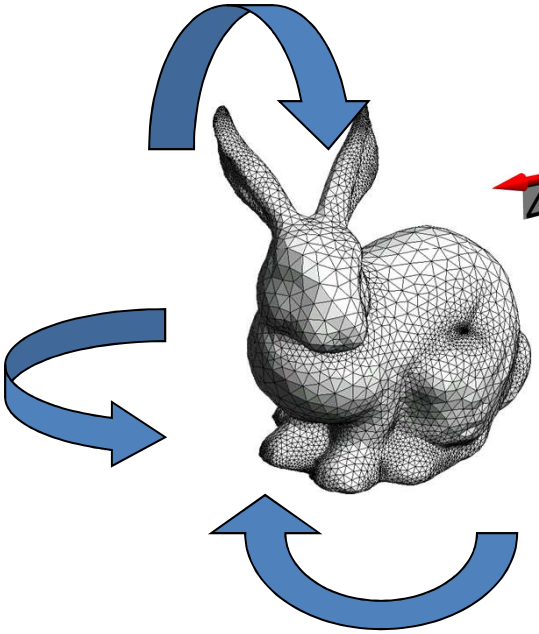
Degree of Integration

1D

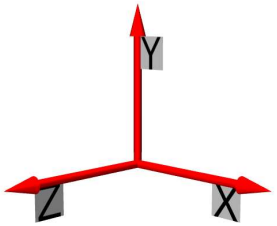
2D



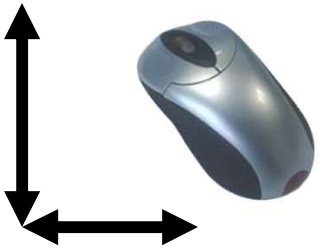
3D



6D

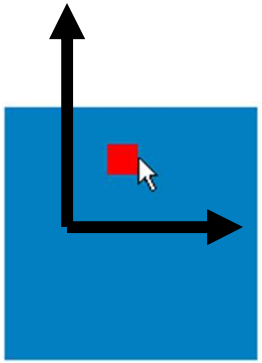


2D

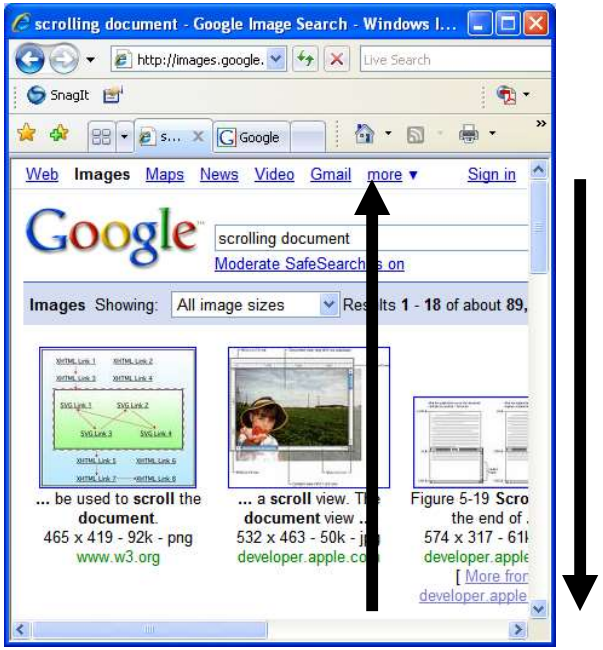


Degree of Compatibility

Dragging = high



Scrolling = medium



Dialog = low

Terms of Interaction

- Some definitions:
  1. Domain: An area of expertise and knowledge in some real-world activity
  2. Tasks: Operations that manipulate concepts in the domain
  3. Goal: Desired output from a task
  4. Core language: The computational attributes of the domain relative to the system state
  5. Task language: The psychological attributes of the domain relative to the user state.

Models of Interaction

- Can exist at a various levels of abstraction
  - High-level: How does a user manipulate a computer to accomplish real world goals
  - Low-level: What actions does a user perform in an interface
- At this stage contextual design mixes these concepts quite freely
  - UED + HTA vs scenarios/storyboards vs low-fi prototypes

Models of interaction (2)

- Will look at 5 models of interaction
  - High-level:
    - Instrumental Interaction
    - Execution-Evaluation Cycle
    - Interaction framework
  - Mid-level:
    - GOMS
  - Low-level
    - KLM

Execution-Evaluation Cycle

- 2 stages with 7 steps
- Developed by Norman (1980)
- Execution involves:
  - Establishing a goal
  - Forming the intention
  - Creating the plan (i.e. a sequence of actions)
  - Executing the plan
- Evaluation involves:
  - Perceiving system state
  - Interpreting state
  - Evaluating state wrt goal/intention

Advantages/Disadvantages

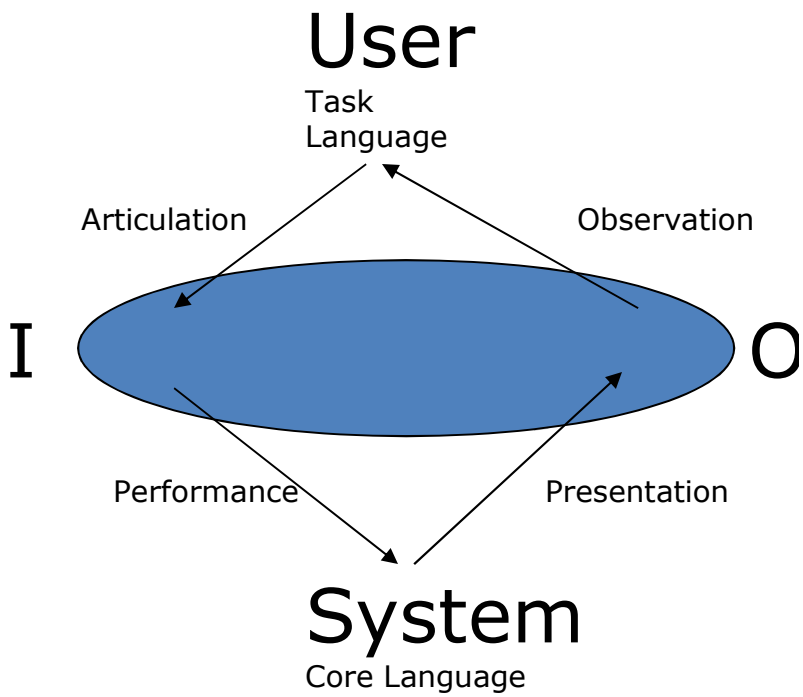
- Allows identification of gulfs of execution and gulfs of evaluation
  - Gulf of execution: Difference between the user's actions and actions allowed by system
  - Gulf of evaluation: Difference between state of program and the expectation of user
- "Problems"
  - Only considers system as far as interface
  - Focuses on user's view of interaction

What's Interesting

- Norman's interaction is very naïve
  - Execution involves:
    - Establishing a goal
    - Forming the intention
    - Creating the plan (i.e. a sequence of actions)
    - Executing the plan
  - Evaluation involves:
    - Perceiving system state
    - Interpreting state
    - Evaluating state wrt goal/intention
- Can make use of to improve much about software applications

Interaction Framework

- Extends Norman's model:
  - Includes system state explicitly
- Four nodes:
  - System, User, Input and Output
  - Each node has own language:
    - System language = core language
    - User language = task language
    - Input and Output languages form the interface
      - Translates between core and task language

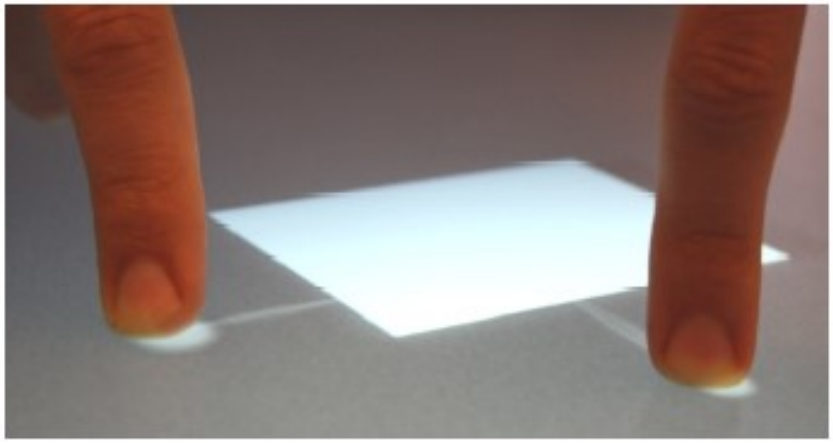


Daniel Widgor's Work on Multi-Touch

- Interaction framework can be useful when considering designs
- Widgor talked about two types of presentation language:
  - Affordance language – system perspective – encourages input
  - Output language – display of information for evaluation

Widgor: Solving Problems with Direct Input through *Affordance Language*

- Fat finger problem
- Feedback ambiguity



Commentary

- I like Execution-Evaluation framework for early stage design:
  - Focuses on user and their view of the system.
- At later stages in implementation, the Interaction Framework can be valuable
  - Forces you to think about all aspects of the interface
  - Forces you to consider what users want to accomplish and how to map
- All these frameworks are also placed with a context

Models of interaction (3)

- Will look at 5 models of interaction
  - High-level:
    - Instrumental Interaction
    - Execution-Evaluation Cycle
    - Interaction framework
  - Mid-level:
    - GOMS
  - Low-level
    - KLM

GOMS

- Goals what the user wants to achieve
- Operators basic actions user performs
- Methods decomposition of a goal into subgoals/operators
- Selection means of choosing between competing methods

GOAL: ICONISE-WINDOW
. [select GOAL: USE-CLOSE-METHOD
. MOVE-MOUSE-TO-WINDOW-HEADER
. POP-UP-MENU
. CLICK-OVER-CLOSE-OPTION
GOAL: USE-L7-METHOD
. PRESS-L7-KEY]

For a particular user:

Rule 1: Select USE-CLOSE-METHOD unless another rule applies
Rule 2: If the application is GAME, select L7-METHOD

KLM

- Short tasks, particularly commands (from the command line interface days)
- Allows calculation of performance by experienced users
- Tasks can be described as operators
  - K = keystroking = 0.35s
  - P = pointing = 1.10s
  - H = homing = 0.4s
  - D = drawing = variable with length of line
  - M = Mental operator = 1.35s
  - R = response operator by system = 1.2s
- Sum up times for each operator in a given task and you know how long it'll take

Overview of User Actions

- Describe what user does or can do within interfaces
  - Instruments to manipulate objects
  - Mechanisms to conceptualize planning and tasks
  - Models of low-level actions
- What's still needed is some information in how design should be guided
  - Goals of design and rules or guidelines to realize those goals

Designing Interfaces

Goal of Design

- Partly to design an interface people like
  - Affect
- More generally to provide an interface that is "useful, usable, and used"
  - Useful = all of Contextual Design
  - Used = binary answer
  - Usable => Ambiguous! … but still a useful concept

Usable Designs

- Ultimate goal of interface design?
  - Aligning mental models
- Attributes of good interfaces
  - Guide usage
    - What Widgor calls an *affordance language*
  - Platform Design Guidelines
    - Common “Look and Feel”
  - Desirable Metrics
    - Learnability/Flexibility/Robustness
  - Affect in Interaction

Mental Models of System

- Designer's Model
  - How the designer of system believes system should be used
- System image
  - The system itself
- User's Model
  - How the user of a system believes system should be used
- Designer and User communicate via system
  - Goal is to have both images align as closely as possible

Mental Models

- A conceptual model of how things work
  - Essentially cause and effect, or hypotheses about behaviour
    - If I do this, then system does that
- Frequently, models are inaccurate or incomplete

Thermostats for house and car

Usable Designs

- Ultimate goal of interface design?
  - Aligning mental models
- Attributes of good interfaces
  - Guide usage
    - What Widgor calls an *affordance language*
  - Platform Design Guidelines
    - Common “Look and Feel”
  - Desirable Metrics
    - Learnability/Flexibility/Robustness
  - Affect in Interaction

Affordance Language

- Purpose is guiding usage
  - Done by applying certain design principles to UI
  - Allows users to perceive what should be done, to map action onto display
- Essentially brings designer's model and user's model of system into alignment if done well.
- Set of principles to promote alignment of models

UI Design Principles (from Preece, Rogers, Sharp)

- Affordance
- Mapping
- Constraints
- Visibility/Feedback
- Consistency
- Metaphor

Affordance

- Attribute of an object that allows people to know how to use it

Push

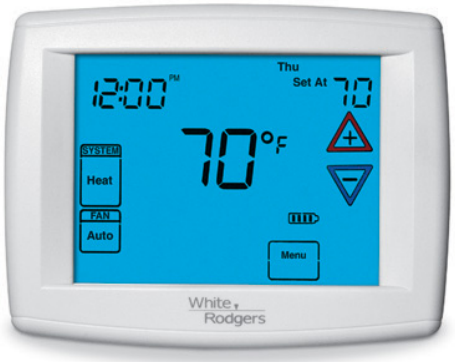
Pull

Waterloo

- Coined by Norman
  - Means “to give a clue”
  - Book: *The Design of Everyday Things*
- Norman’s current argument:
  - Should not pay too much attention during UI design
  - Objects have “real affordances”
  - Screen widgets have “perceived affordances”
  - Learned conventions

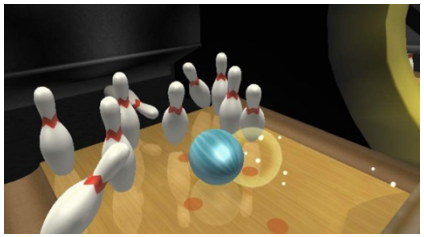
Mental Models and Affordance

- Consider thermostat

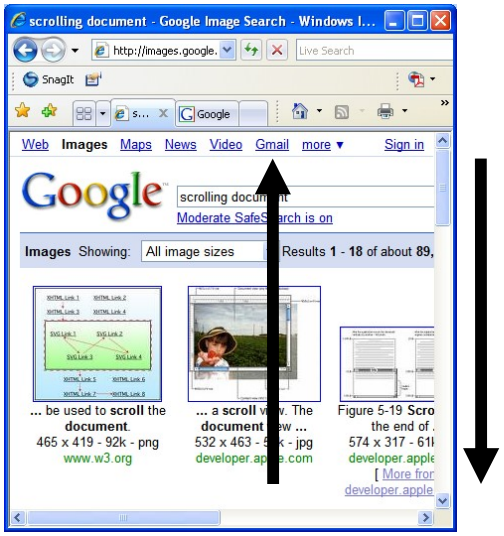


Mapping

- Physical actions performed on the device must be mapped onto on-screen effects
- Instruments operations must be mapped onto objects



- Recall instrumental interaction
  - Degree of integration
  - Degree of compatibility



Physical Versus Virtual

- Some things work well in physical, not in virtual

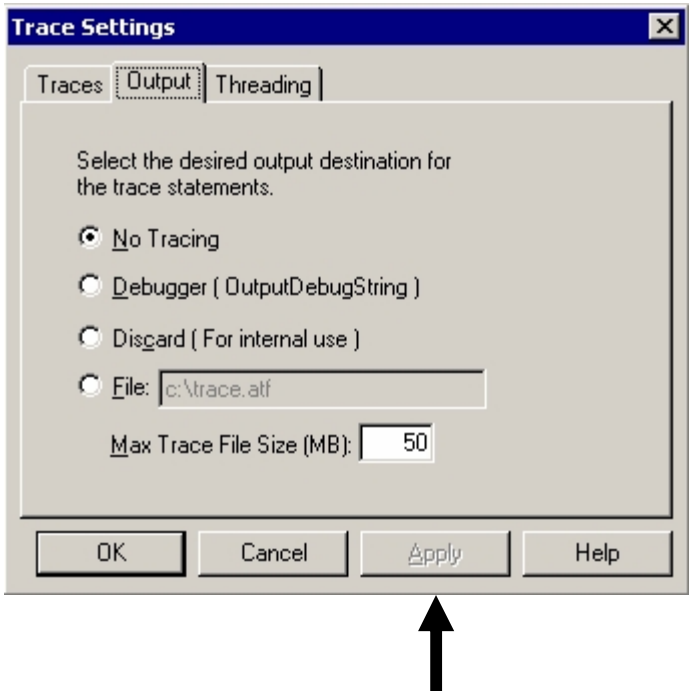
Physical

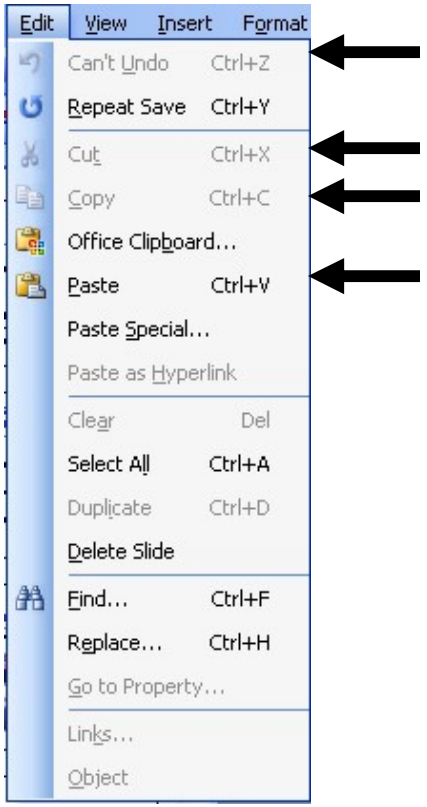
Virtual

This user interface (UI) is simply hideous. Form and function have been sacrificed to looks, in the sense that it's appearance is mimicking a physical piece of equipment. Trying to adjust the parameters with the mouse is difficult and error-prone, and I am not brave enough to try using the keyboard - I don't even know how I would do that.

Constraints

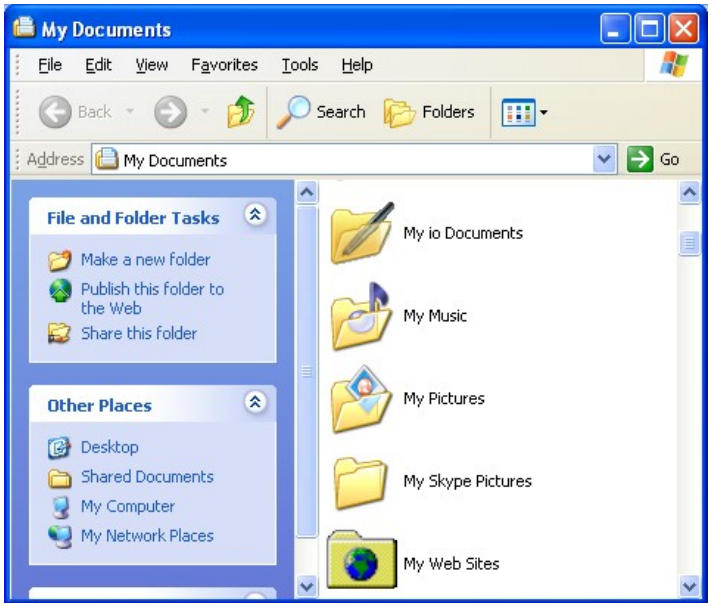
- Guide user by preventing certain actions, enabling others



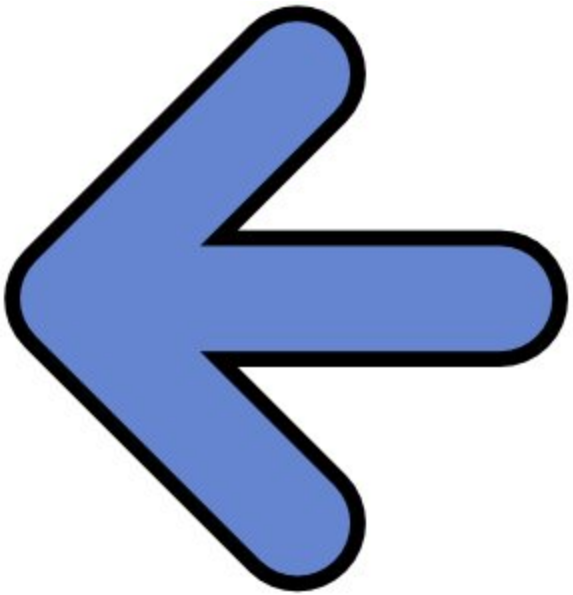


Constraints

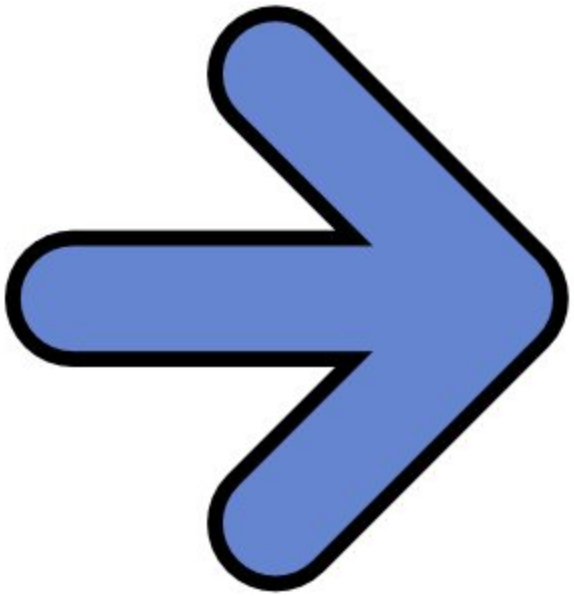
- Physical
  - Only one way to connect
  - Disabled buttons/menus
- Logical
  - My Documents, My Pictures, etc.
- Cultural
  - Examples?



Cultural Constraints



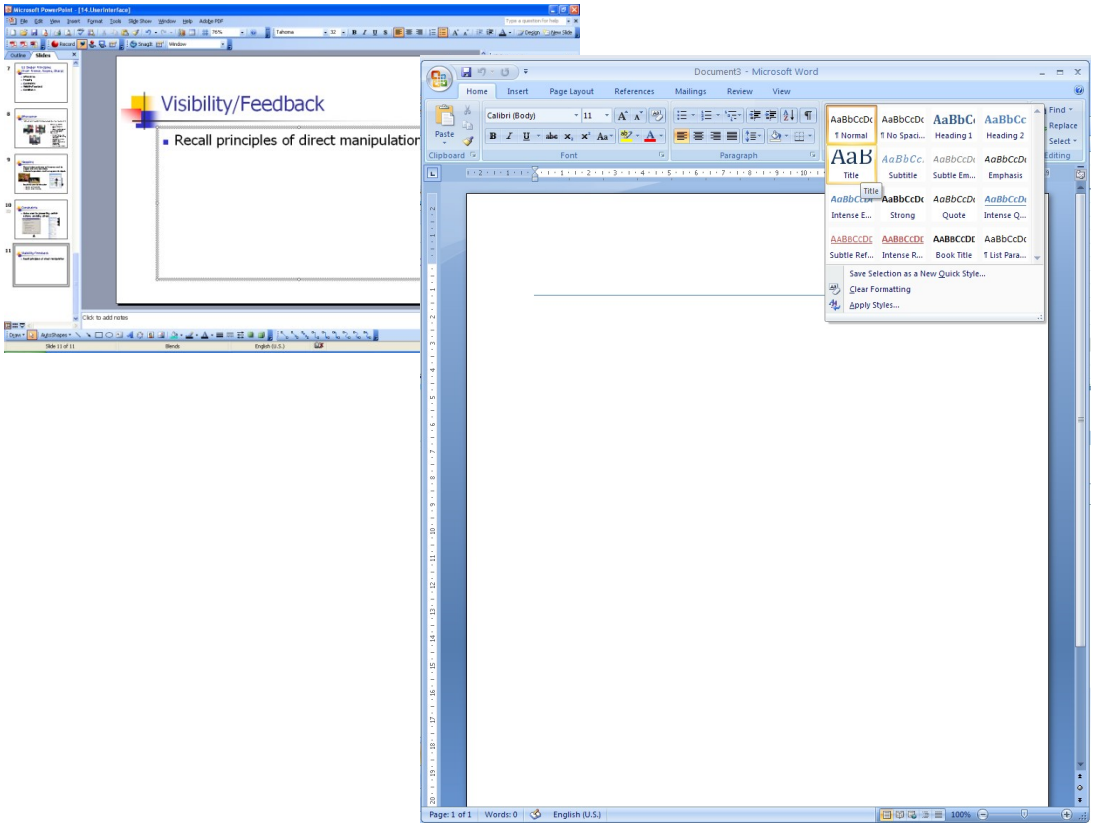
Next



Prev

Visibility/Feedback

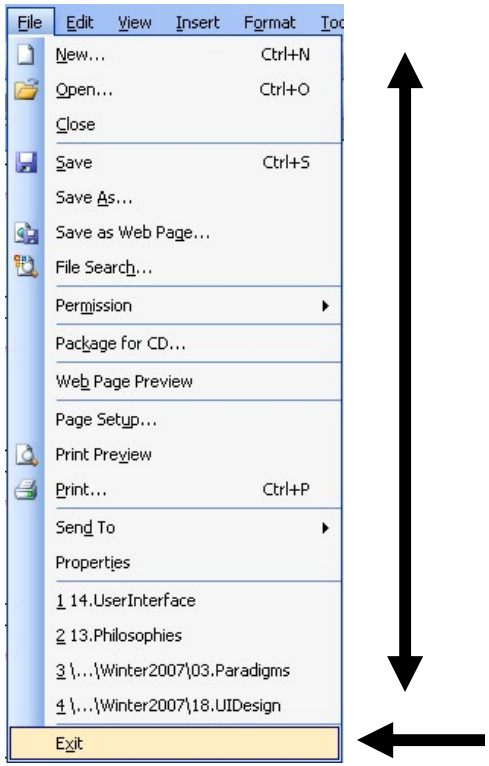
- Recall principles of direct manipulation



- Continuous representation
- Actions on objects of interest
- Fast, incremental operations with immediate feedback
- Layered learning/self-revealing

Consistency

- Allows users to leverage control from familiar onto new



Metaphors

My Documents
Skype
My Computer
Recycle Bin
Mozilla Firefox
Adobe Reader 9
Windows Media Player
iTunes
Comcast Email
Magnifier
Wind Notebook
MSI
NOTEBOOK
start
Fall 2010
Search Desktop
11:31 AM

Usable Designs

- Ultimate goal of interface design?
  - Aligning mental models
- Attributes of good interfaces
  - Guide usage
    - What Widgor calls an *affordance language*
  - Platform Design Guidelines
    - Common “Look and Feel”
  - Desirable Metrics
    - Learnability/Flexibility/Robustness
  - Affect in Interaction

Design Guidelines

- Many platform providers develop design guidelines to describe best practices in building applications for platforms
  - Apple particularly good at this

Windows Screen Design

- From MSDN Magazine, Sept. 2009.
  - Create the framework
    - A navigational scheme so that users can move from screen to screen
    - Persistent elements like headings to anchor user
  - Create wireframe templates
    - Essentially sketches of the different screens
  - Create a style guide
    - Document your design decisions for your application
  - Test the sketches
  - Thing about user model (cognitive model of software), consider layout and chunking of information, consider information flow

iPhone Design Guidelines

- Platform constraints to keep in mind
  - Screen size, memory, no WIMP paradigm, limited multi-tasking, limited on-screen help
- Application classes
  - Productivity, utility, immersive
- Platform migration issues
  - 80-20 rule is one they cite

iPhone: Human Interface Principles

- Metaphor
- Direct manipulation
- See and point
- Feedback
- User control
- Aesthetic integrity

iPhone Design Guidelines

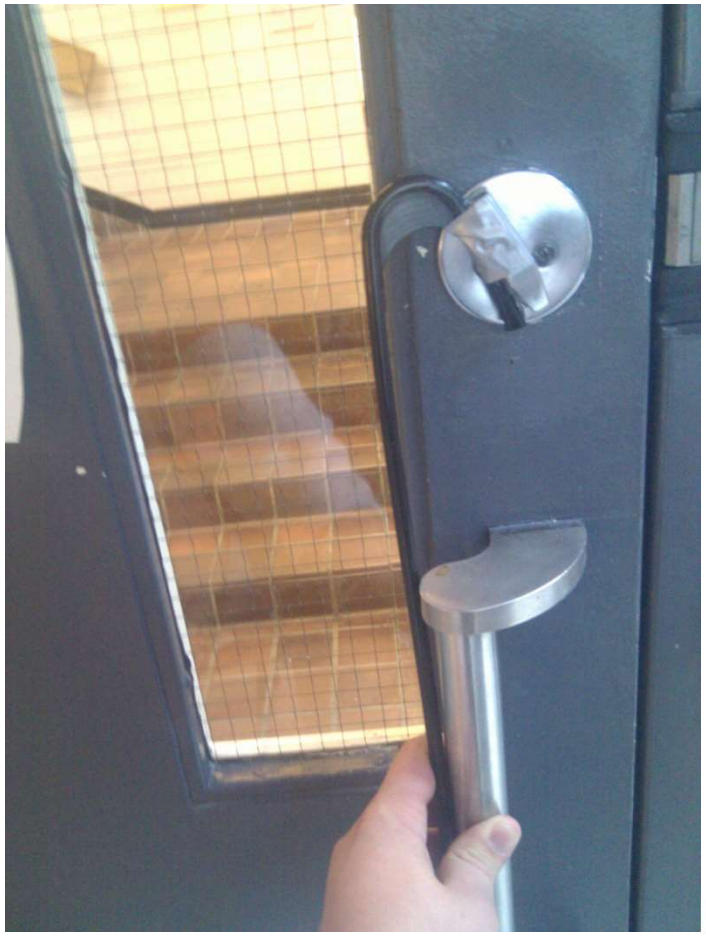
- Create a product statement
  - Understand your users and what differentiates them
- Ease of Use Constraints
  - Make it obvious
  - Think top-down
  - Minimize input
  - Keep information succinct
  - Provide finger-tip sized targets (44X44 pixels)
- Focus on the Primary Task
- Provide constant feedback
- Support gestures appropriately

Thoughts?

- Think about ease of use constraints on iPhone
  - Make it obvious
  - Think top-down
  - Minimize input
  - Keep information succinct
  - Provide finger-tip sized targets (44X44 pixels)
- What changes when you move to iPad?
- What changes when you move to the BlackBerry?

Example Design

What motivated this design?



Food for Thought: The anti-Mac interface

- A paper by Gentner and Nielsen
- Mac Interface Guidelines developed because:
  - Apple needed to sell to "naive users," that is, users without any previous computer experience.
  - Computer was targeted at a narrow range of applications
  - Computer controlled relatively weak computational resources (originally a non-networked computer with 128KB RAM, a 400KB storage device, and a dot-matrix printer).
  - Computer was supported by highly impoverished communication channels between the user and the computer (initially a small black-and-white screen with poor audio output, no audio input, and no other sensors than the keyboard and a one-button mouse).
  - Computer was a standalone machine that at most was connected to a printer
- What if we break these?

Food for Thought: The anti-Mac interface

- A paper by Gentner and Nielsen

Macintosh Guidelines	Anti-Mac Guidelines
Metaphors	Reality
Direct Manipulation	Delegation
See and point	Describe and command
Consistency	Diversity
WYSIWYG	Represent meaning
User is in control	Shared control
Feedback	System handles details
Forgiveness	Model user actions (learn)
Aesthetic integrity	Graphic variety
Modelessness	Richer cues

Usable Designs

- Ultimate goal of interface design?
  - Aligning mental models
- Attributes of good interfaces
  - Guide usage
    - What Widgor calls an *affordance language*
  - Platform Design Guidelines
    - Common “Look and Feel”
  - Desirable Metrics
    - Learnability/Flexibility/Robustness
  - Affect in Interaction

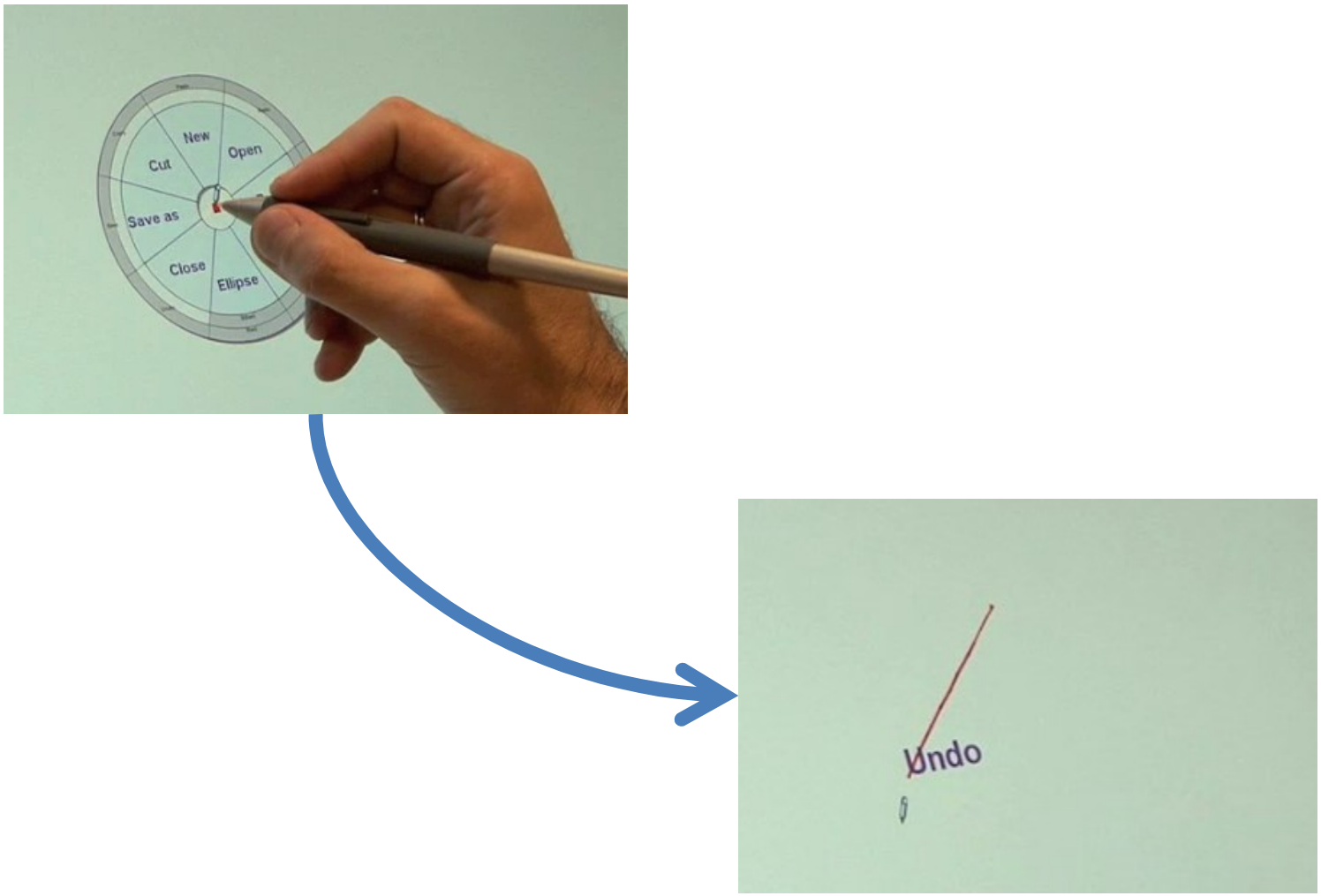
Principles to Support Usability

- Learnability:
  - How easily can a user become skilled with interface?
- Flexibility:
  - How easily can a user adapt the interface to a task?
- Robustness:
  - How rare are errors and can a user recover?

Learnability

Principle	Definition	Principles
Predictability	Can history guide future interaction?	Operation visibility
Synthesizability	Is effect of past operations visible?	Immediate/ eventual honesty
Familiarity/ Generalizability	Knowledge and experience can be leveraged? Extended within and across domain?	Guessability Affordance
Consistency	Similar behavior in similar situations	Guessability
Self-revealing	Does expert use arise “naturally”	Layered learning

Self-Revealing/Layered Learning



Flexibility

Principle	Definition	Principles
Dialog initiative	Can system dialog constraints be reduced/eliminated?	System/User Pre-emptive
Multi-threading	User can work with more than one task	Concurrent vs. interleaving Multi-modality
Task migratability	Task performed by user, by system, or shared	Guessability Affordance
Substitutivity	Values of input or output substituted	
Customizability	User or system can modify interface	!!!

Robustness

Principle	Definition	Principles
Observability	Can system state be easily evaluated from the screen?	Operation visibility, Reachability, Defaults
Recoverability	Ability to correct errors once recognized	Reachability, Commensurate effort
Responsiveness	User perception of rate of communication	Stability
Task conformance	Does system support all user tasks in a way that user understands	Task completeness and adequacy

Interface Design

- Conceptualize artifacts in interface
  - Instrumental Interaction
    - Michel Beaudouin-Lafon
- Models of interaction
- Ultimate goal of interface design
  - Mental Models
- **Attributes of good interfaces**
  - Guide usage
    - What Widgor calls an *affordance language*
  - Desirable Metrics
    - Learnability/Flexibility/Robustness
  - **Affect in Interaction**