Guidelines & Principles

(plus an example of theory)

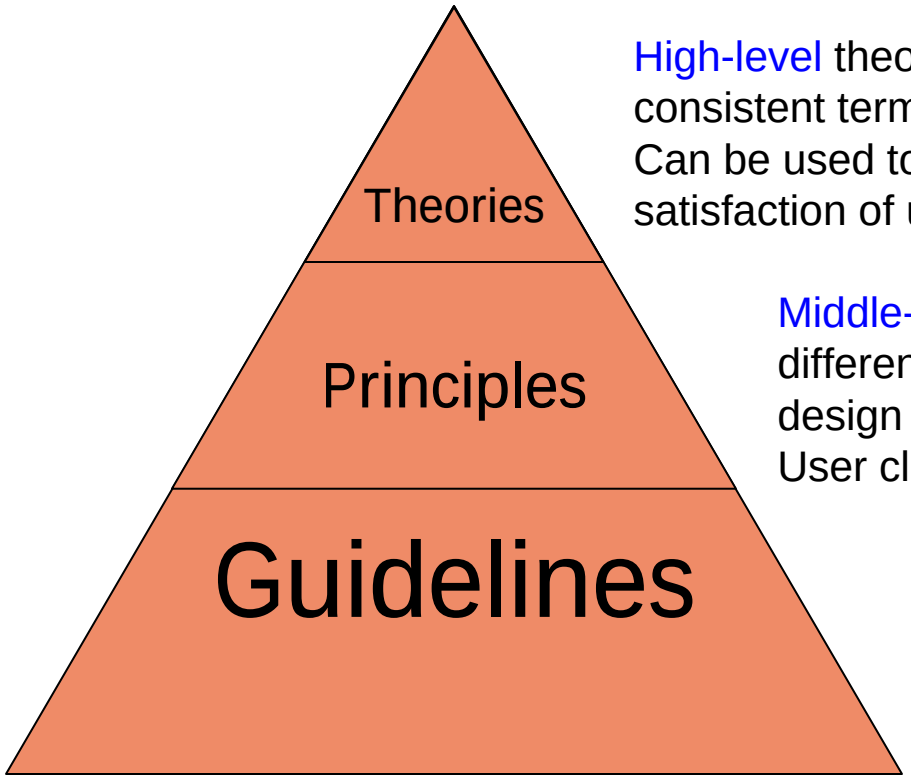
CZ2004 HCI

Nanyang Technological University

“Guidelines, Principles, and Theories”

- Reading - Textbook
  - Chapter 2.1, 2.2, 2.3, 2.4.2
  - If you are still interested, self-learn other parts in 2.4
- Goal:
  - Introduce you to the study and practice of HCI
  - Overview
    - Guidelines – narrowly focused rules
    - Principles – widely applicable and enduring
    - Theories – tested, proven, broadly useful

Introduction



High-level theories that describes objects and actions with consistent terminology to support communication & teaching. Can be used to ***predict*** performance, errors, understanding, satisfaction of user.

Middle-level practices that can be applied to different guidelines, analyzing and comparing design alternative.
User classification, “8 golden rules of UI design”, etc.

Design-level practices and rules that make for good and consistent design (some based on theory).
Examples: Apples guidelines on for UIs

Why guidelines, principles and theories?

- **Why have guidelines, principles, and theories?**
  - Help keep our UI designs focused and consistent.
  - Help avoid and remedy mistakes (e.g., cluttered display, tedious procedures, inadequate functionalities, etc.)
  - Provide theories and high-level description of interaction and design
- The role of *theory* in this course
  - We will *not* spend a lot of time on any specific theory
  - However, it is important to understand the role of theory, and its relationship between guidelines and principles
    - We will only carefully study one example of the theories – the *stage of actions*

Guidelines

- Definition
- Components
- Important (example) guidelines
  - Navigation
  - Helping people with disability
  - Display organization
  - Data entry

What are “guidelines”?

- Guidelines were developed in the “early days”
- Best Practices
  - For example, Windows and Apple UI
- From experience
  - For example, of Microsoft or Apple interface designers
- Good starting point for all projects involving a UI
- Developed “Shared language”
  - Widget Names, Functionality name, etc
  - Gives all developers involved a language to discuss the UI

Components that form guidelines

- **Rules** (Specific and practical)
  - Provides cures for design problems
  - Provides cautions for potential danger
  - Reminders based on experience
- **Examples**
  - Give details on how a design must be performed
  - Style, color usage, window appearance, etc
  - Interaction usage (when to use check-boxes, when to user buttons, etc)
    - These guidelines could be based on experience
    - Ex: developers found that users preferred a list+slider over a pull-down menu for a long list of choices
  - Require all developers to follow the guidelines

Components (cont.)

- **Document**
  - Any serious large-scale UI design should have a "Guideline Document"
  - Provides a "Shared language" that developers and ***customers*** can use
  - Allows consistency within a design team
    - Especially a large-team working on a large project
  - Guidelines document is not trivial
    - Think of the effort needed to specify "everything" pertaining to the UI, but it is *necessary*

(Guideline) Document

- **Can be used to specify many aspects of an interface:**
  - Input and output formats
  - Action sequences
  - Terminology
  - Hardware devices/platforms
  - Provide Examples & counterexamples
- **Pros:**
  - Builds upon (good previous) experience
  - Continued improvements
- **Cons:**
  - Too specific
  - Hard to innovate
  - Not applicable/realistic to the situation
  - Hard to apply
  - What do you do when having an exception?

Guideline Document: Example

- Think about templates
  - Word
  - PowerPoint
  - LaTeX
- Any other from you?

A case study: iOS 7 guideline

Apple's "Design Principles"

http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Principles/Principles.html#//apple_ref/doc/uid/TP40006556-CH5-SW1

Apple's "Human Interface Guidleine for iOS 7"

https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html#//apple_ref/doc/uid/TP40006556

iOS 7: Navigation Bar (from Apple library)

- Definition
  - A **navigation bar** enables navigation through an information hierarchy and, optionally, management of screen contents
- Example

- Rules
  - A navigation bar is translucent.
  - Generally appears at the top of an app screen, just below the status bar. On iPad, a navigation bar can also display within a view that doesn't extend across the screen, such as one pane of a split view controller.
  - Can automatically change its height when an iPhone changes orientation. Maintains the same height in all orientations on iPad

From: iOS UI Element Usage Guidelines
https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/Bars.html#//apple_ref/doc/uid/TP40006556-CH12-SW1

iOS 7: Navigation Bar (cont.)

- Selected guidelines
  (Only portions of those listed by Apple)
  - When it adds value, use the title of the current view as the title of the navigation bar
  - Consider putting a segmented control in a navigation bar at the top level of an app
  - Avoid crowding a navigation bar with additional controls, even if it looks like there's enough space
  - Make sure text-titled buttons have enough space between them
  - Don't create a multisegment back button

- **Counterexample**
  - 

  - Why is bad?

Navigating the Interface

- Example from National Cancer Institute (NCI)
  - Developers developed 388 guidelines backed by research for designing their web-pages
- Some guidelines established by the NCI website developers
  - Standardize task sequences
  - Ensure that embedded links are descriptive
  - Use unique and descriptive headings (related to the content they describe)
  - Use check boxes for binary choices
  - Develop pages that will print properly
    - All pages/information on the web should be printable
  - Use thumbnail images to preview larger images
- Go to their website, http://www.nci.nih.gov/ and evaluate:
  - Pretend to be different types of User: Novice, Intermediate, Expert
  - Test Different Tasks: Education, Search, Research
  - Are the pages consistent? Are the guidelines followed?

Guidelines for Disabled

- WWW Consortium adopted these guidelines for designing web pages for disabled
  - Text equivalent for every non-text element (images, image map, animations, applets, ascii art, frames, scripts, bullets, sounds, audio, video, etc.)
  - Any time-based multimedia, provide equivalent synchronized alternatives (captions, descriptions)
  - All color info can be captured by users without color – from context or markup
  - Title each frame, facilitating frame identification and navigation
- Enables screen readers or other technologies to have multiple methods to obtain the webpage info
- How does this end up helping everyone?

Display organization guidelines

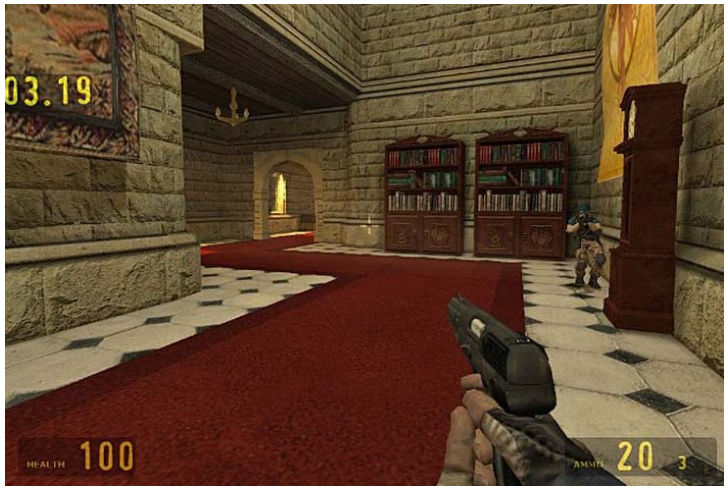
- **Consistency of data display**
  - Terminology, abbrev., formats, colors, grammar, capitalization should be consistent!
- **Efficient information assimilation by the user**
  - Familiar format
  - Related to tasks at hand
  - e.g. spacing, formatting, labels, units/measurements, numbers of decimal points
- **Minimal memory load on the user**
  - Minimal carry information over from on screen to another
  - Require fewer actions
    - TAB key to move to next entry field vs. having to use the mouse
  - Labels and common formats should be provided for novice (Ex. SSN/phone #)

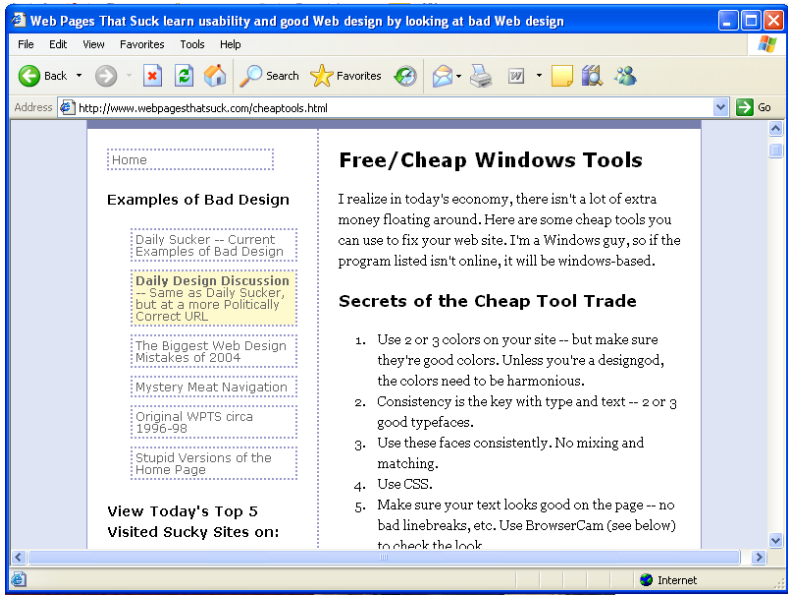
Display organization guidelines (cont.)

- **Compatibility of data display with data entry**
  - Entering data should look similar to the eventual viewing of the data
- **Flexibility for user control of data display**
  - User control for information display (e.g., sorting, ordering of columns and rows)
- Only a starting point
  - Has many special cases
  - Application specific, hardware independent (e.g., ATMs)

Displays vs. User's attention!

- User sees lots of data in front of them
- Urgent, exceptional, and time-dependent conditions need to be brought forward
- Ex. games and damage (visual, audio)
- Intensity – two levels only, limited use of high intensity
- Marking – underlines, enclose it in a box, arrows, asterisk, bullet, dash
- Size – Up to 4 sizes, with larger sizes attracting more attention
- Fonts – three fonts





Displays vs. User's attention! (cont.)

- Inverse video – inverse coloring
- Blinking Colors
  - Should blink at 2-4 blinks per second (Hz), Color – *no more than 4 on a screen*
- Audio
  - soft tones – positive
  - harsh – emergency
  - multiple levels are difficult to distinguish, do we like human voices?
- Danger in overusing the above
  - Animation should provide needed information (e.g. progress indicator)
  - Similarly highlighted items imply relationships
  - Novices need simple, logically organize, well labeled displays
  - Experts want shorter labels, more flexibility, subtle highlight of changed values

Example of a Bad Webpage

This part blinks.

Compare with the current version: http://www.notrag.org.uk/
(still available in early 2014, but just before our class, it is not accessible -> check web archive)

From: http://www.webpagesthatsuck.com

A case study: good vs. bad attention

Data Entry guidelines

- Data entry can occupy a substantial portion of user's time and be the source of frustrating and potentially dangerous errors
- Consistency of data-entry transactions – similar sequence of actions, delimiters, abbrev.
- Minimal input actions by user
  - fewer actions = greater productivity and less error
  - E.g., single key-stroke vs. mouse selection, vs. typing is typically better
  - E.g., Command line vs. GUI
  - Too much hand movement is not good. Ex. Experts prefer to type 6-8 characters instead of moving a mouse, joystick, etc.
  - Avoid redundant data entry (waste of time, perceived effort, increased error). System should aid but allow overriding

Data Entry guidelines (cont.)

- Minimal memory load
  - Don't use codes, complex syntactic strings
    - E.g., Don't user codes for a country on a web form
  - Provide "selection" from a list
    - don't need to memorize choices
- Compatibility of data entry with data display
  - should matches display capability
- Flexibility for user control – Experienced vs. novice
    - Experienced may want "hot-keys", novice doesn't
    - All you Ctrl-F file people are happy!
    - Should be used cautiously, since it goes against consistency

A case study: booking flights

Flight
Hotel
Car

Build your trip, find a great deal!

Flight + Hotel
Hotel + Car

From City name or airport code
RDU

To City name or airport code

Leave
Sep 22
Anytime

Return
Sep 29
Anytime

OPTIONAL (U.S. & CANADA ONLY)

Search one day before and after

Find low fares for weekends and flexible trips

Travelers (up to 9)

Adult (18-64)	Senior (65+)	Youth (12-17)	Child (2-11)	Infant in lap (under 2 yrs)	Infant in seat (under 2 yrs)
1	0	0	0	0	0

Expand search options
(One-way, multi-city, non-stops, cabins, nearby airports)

Find

Principles

- More fundamental, widely applicable, and enduring than guidelines
- Fundamental principles for all UI
  - Determine user's skill levels / **Spiral design strategy**
  - Identify the tasks of the application
- Five primary interaction styles
- Eight golden rules of interface design
- Prevention of errors

Determine user's skill levels

- **"Know thy user" Hansen (1971)**
  - First principle in Hansen's classic list of user-engineering principles.
  - Simple idea, but a difficult and unfortunately often undervalued goal.
- Start with population profile:
  - Age, gender, physical and cognitive abilities, education, cultural or ethnic background, training, motivation, goals and personality
- Design goals based on skill level
  - Novice or first-time users
  - Knowledgeable intermittent users
  - Expert frequent users

Novice/First-Time Users

- What would you need to consider for:
  - Grand-parents sending first email
  - Airport check-in kiosks
- Inexperience with interface (e.g., first time professionals)
- Anxiety
- Solutions
  - Restrict vocabulary
  - Providing help: Instructions, dialog boxes, know who to turn to for help, multiple languages, consistent terms
  - Small number of actions
  - Feedback
    - Good Error messages
  - Documents: Video demonstrations, online tutorials, good manuals

Knowledge-able Intermittent Users

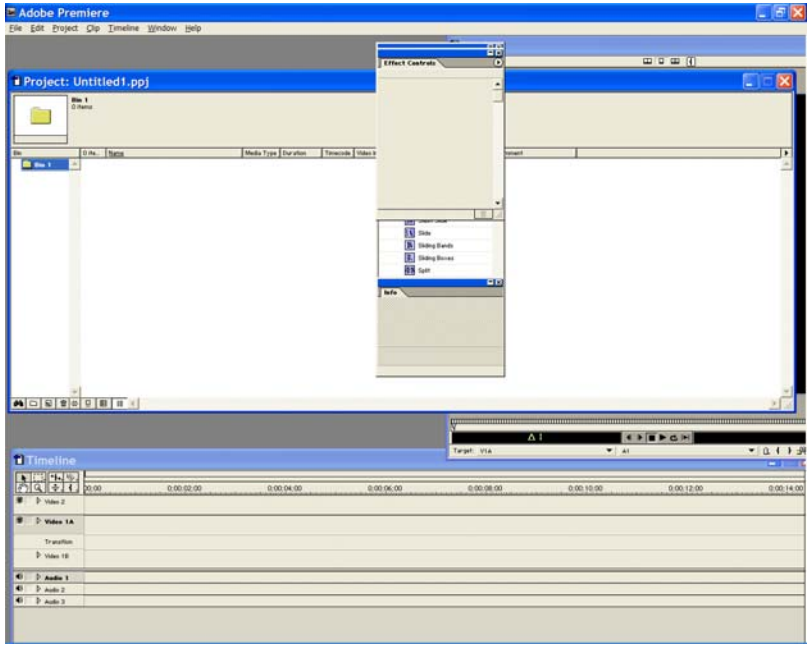
- E.g., Frequent travelers, managers and code/word processors.
- These users understand task concepts, and interface basics
- May have difficulty retaining the structure of menus, or the location of features
- Solutions:
  - Consistent sequences of actions
  - Meaningful feedback
  - Guides to frequent patterns of usage
  - Protection from danger (encourage exploration), e.g., undo
  - Context dependent help

Expert/Frequent Users

- Thoroughly familiar with task and interface
- Goal is efficiency (high speed, low error)
- Solutions:
  - Rapid response time
  - Brief feedback
  - Shortcuts
  - Macros, abbreviations and other accelerator

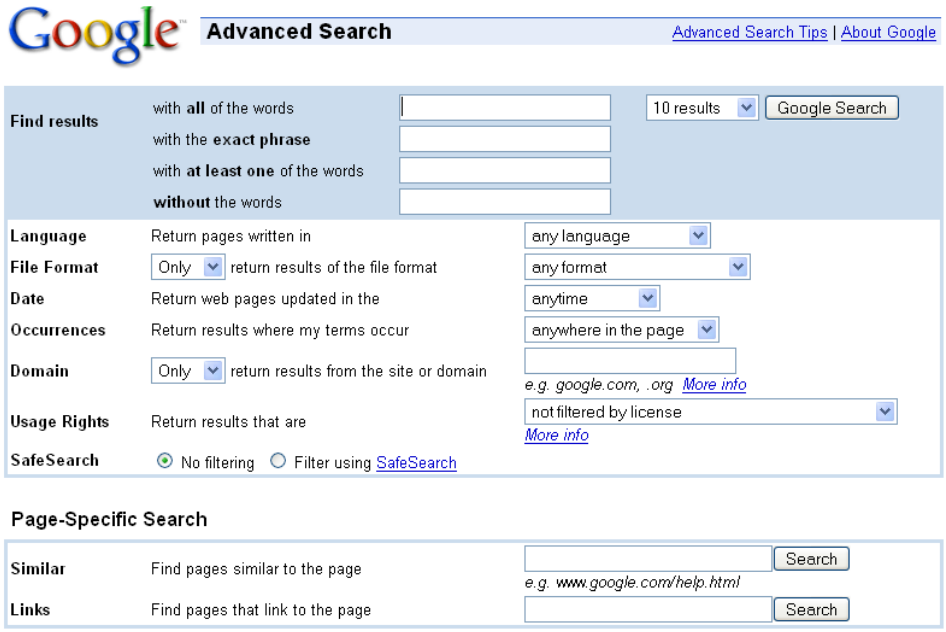
Multi-layer strategy

- You might be designing for more than one of these classes
- Approach is typically a *multi-layer* (a.k.a. *level-structured* or *spiral*)
  - Novices use a subset of commands, actions, and objects
  - Can move up when they feel comfortable
- Ex. Cellphones
  - Novices: phone calls easy to make
  - Experts: store #s, web, game, address book
- Also involves manuals, help screens, errors messages, tutorials, feedback
  - Different for multi-layer users



A case study: Google search & options

- Expand control to accommodate different users



Easy control **Advance control**

Identify the tasks

- After carefully drawing the user profile, the developers must identify the tasks to be carried out.
  - Every designer would agree that the set of tasks must be determined before design can proceed, but too often the task analysis is done informally or implicitly.
- Task Analysis usually involve long hours observing and interviewing users
- Decomposition of high level tasks
- Relative task frequencies

Identify the Tasks

- How?
  - Brainstorm
  - Observe and interview users (esp. newer versions)
- Example: *Palm Pilot*
  - Limited functionality = universal usability
  - Successful because of ruthlessly limiting functionality (calendar, to-do list, contacts and notes) to guarantee simplicity
- "Atomicity" of tasks is important to consider
  - Too small = too many steps (inefficient, frustrating)
  - Too many = need special cases, inflexible, frustrating
- Task frequency
  - High frequency = simple, quick, even if it slows other tasks down
- Task vs. Job Frequency Matrix (see next slide)
- Task analysis and task objects and objects defined

A case study: Hospital information system

- User-needs assessment clarifies what tasks are essential for the design and which ones could be left out to preserve system simplicity and ease of learning
- Should be starting point for any good UI designer

	TASK				
Job title	Query by Patient	Update Data	Query across Patients	Add Relations	Evaluate System
Nurse	0.14	0.11			
Physician	0.06	0.04			
Supervisor	0.01	0.01	0.04		
Appointment personnel	0.26				
Medical-record maintainer	0.07	0.04	0.04	0.01	
Clinical researcher			0.08		
Database programmer			0.02	0.02	0.05

Choose Interaction Style

Five main types of interaction style

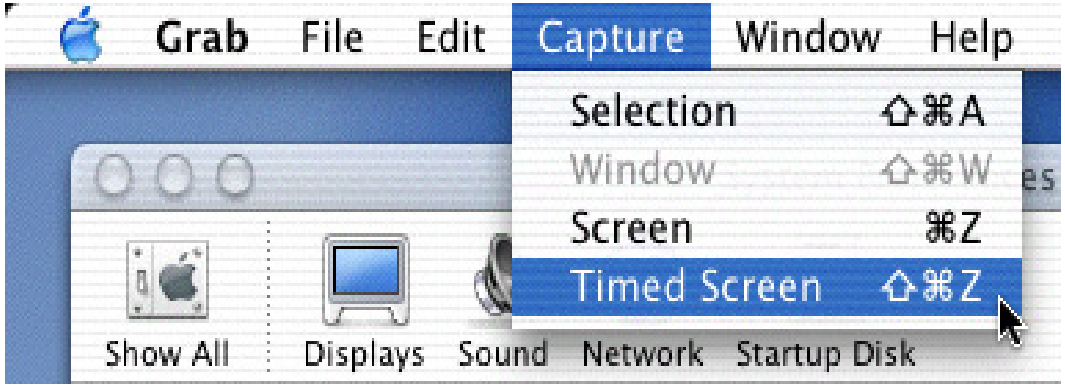
1. Direct Manipulation
2. Menu Selection
3. Form-fill in
4. Command Language
5. Natural Language

1. Direct manipulation

- Manipulate visual representations, e.g. Desktop metaphor, CAD, games
- Appealing to novices and easy to remember for intermittent users.
- **Pros:** fast, feedback, easy to understand and retain (ex. icons on your desktop), exploration encouraged, good for novices, and can be good for other classes, visual data
- **Cons:** hard to program, interaction devices are harder to design or modify

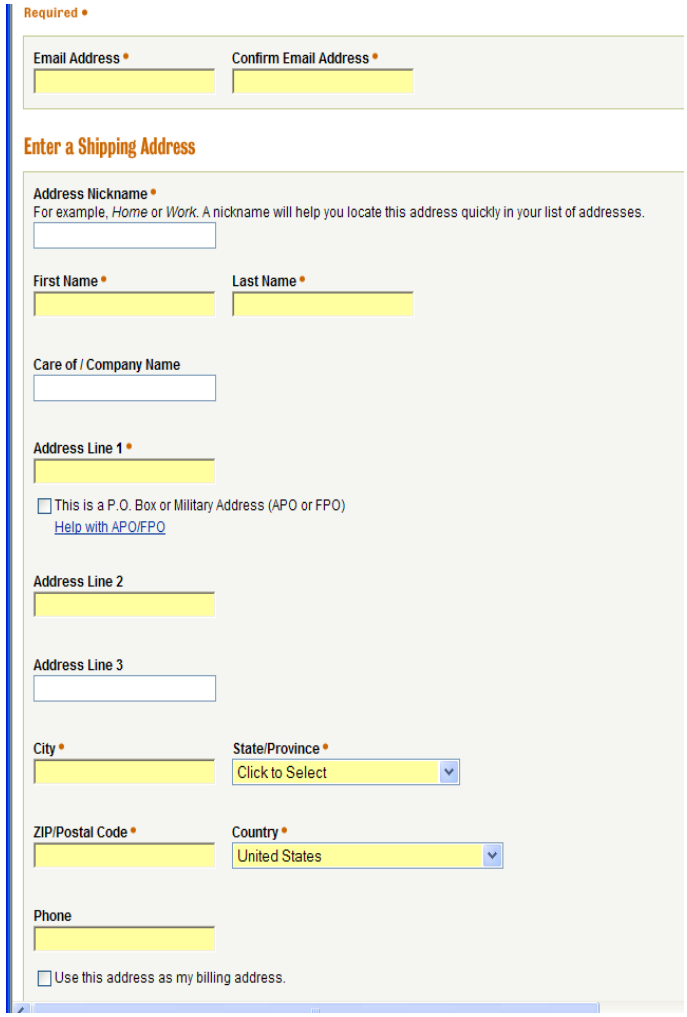
2. (Menu) Selection

- User reads a list of items, and selects one
- Appropriate for novice and intermittent users and can be appealing to frequent users if the display and selection mechanisms are rapid.
- **Pros:** no memorization, few actions, clear structure, tools for validity and consistency exist
- **Cons:** Make actions understandable not easy, careful task analysis



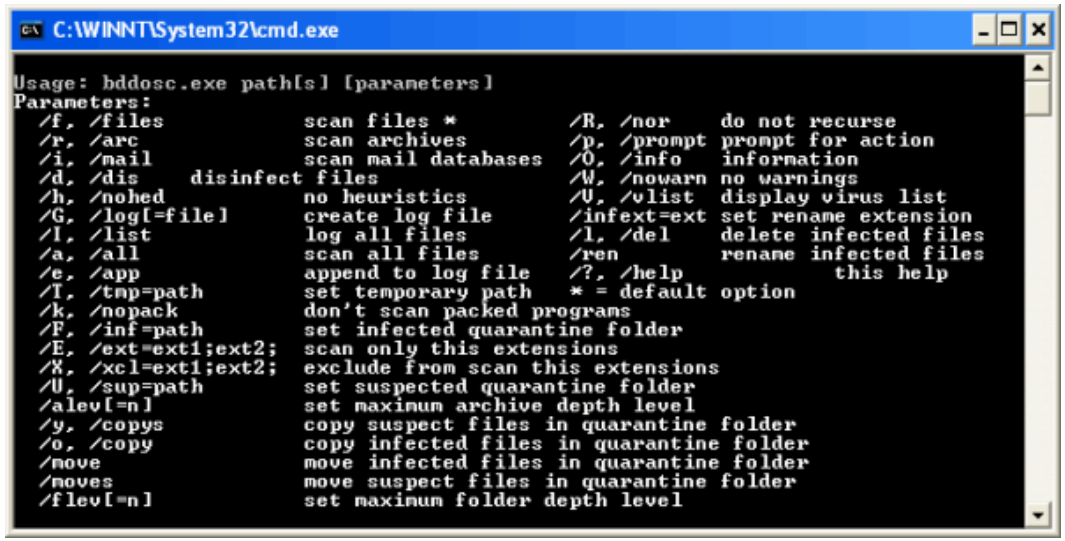
3. Form fill-in

- Data entry into fields
- Most appropriate for knowledgeable intermittent users or frequent users.
- **Pros:** rapid, for more advanced users, tools available for forms
- **Cons:** must understand labels and request format, be able to respond to errors, training required



4. Command language

- Suitable for expert frequent users who derive great satisfaction from mastering a complex set of semantics and syntax.
- **Pros:** feeling of control, most advanced users like it, rapid, histories and macros are easy, flexibility
- **Cons:** high error rates, training required, poor retention rate, hard to create error messages



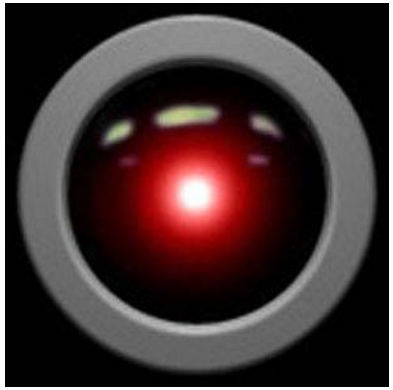
5. Natural language

- Computers will respond properly to arbitrary natural language sentences or phrases.
- **Pros:** easy to learn
- **Cons:** unpredictable, requires clarification dialog, technology is not fully developed . . Still in research stage.

\

- *NL is the Ultimate Goal*
  - Was "science fiction", Example: HAL9000
  - Is Siri or other similar natural language service ready now?

http://www.kubrick2001.com/



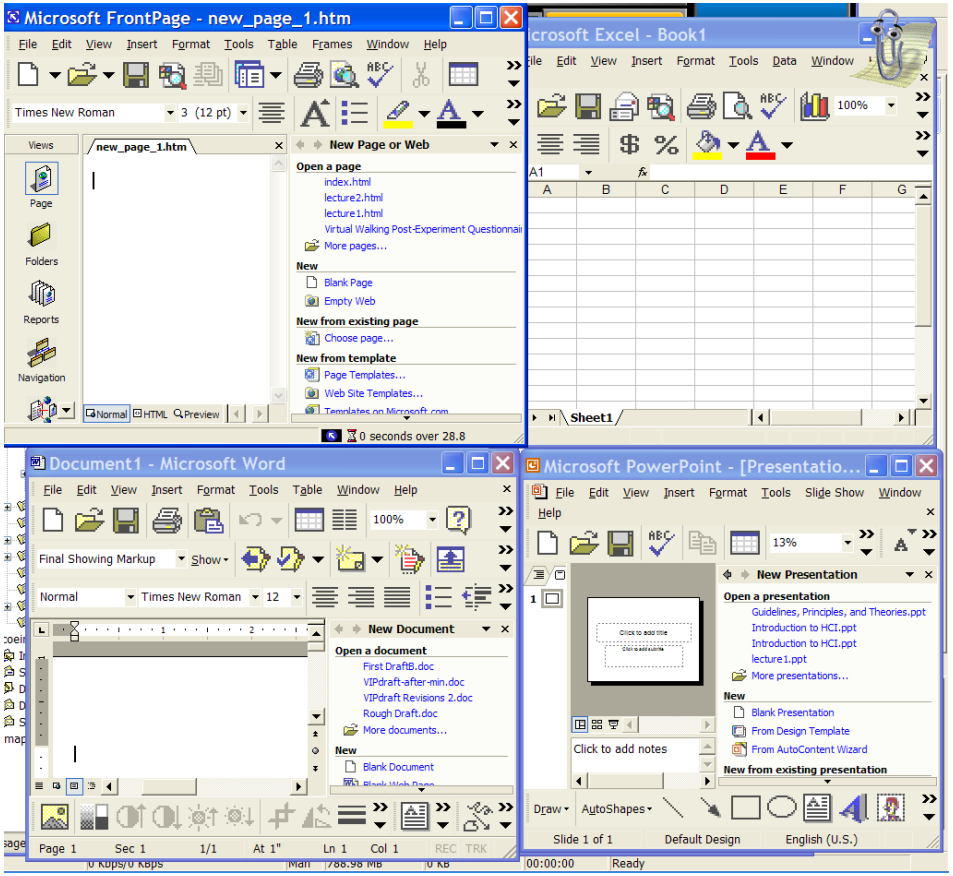
HAL9000 Computer (From 2001)

8 golden rules of interface design

1. Strive for consistency
2. Cater to universal usability
3. Offer informative feedback
4. Design dialogs to yield closure
5. Permit easy reversal of actions
6. Support internal locus of control
7. Reduce short term memory
8. Prevent errors

1. Strive for Consistency

- Consistent sequence of actions for similar situations
- Identical Terminology (prompts, menus, help)
- Consistent visual layout (fonts, color, etc.)
- Exceptions:
  - Confirmation of deletion
  - No password echoing



2. Cater to Universal Usability

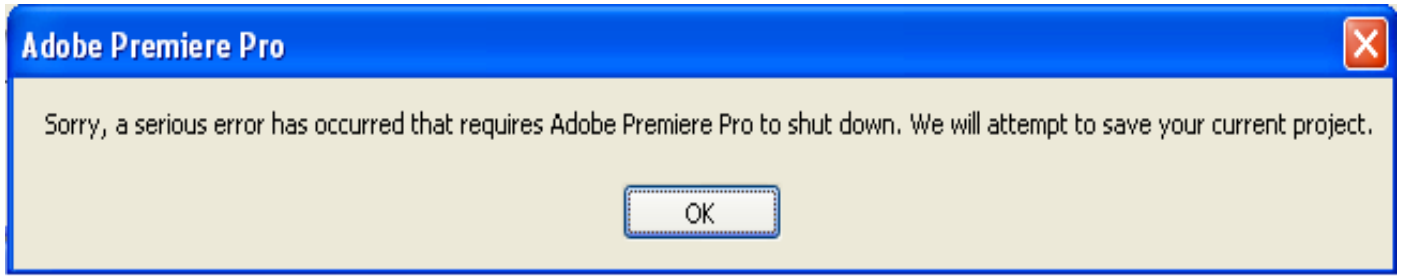
- Recognize the needs of a diverse user group
- Design for *plasticity* (transformation of content)
  - Plasticity means content can be used on any type of display
- Interface supports Novice -> Expert
- Usable by Disabled

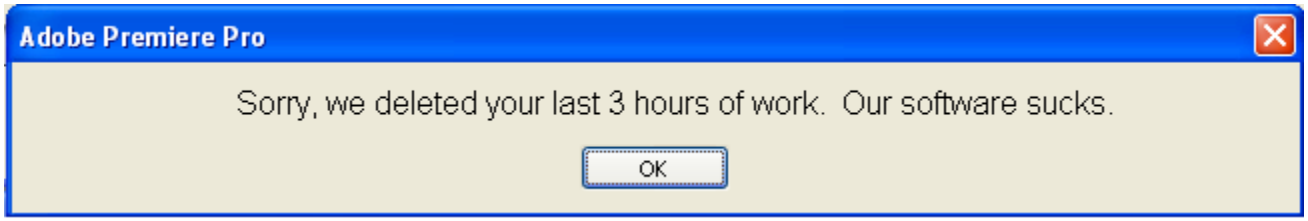
http://www.cnn.com (web and mobile version)

3. Offer Informative Feedback

- For every user action, system should provide feedback
- Frequency of task affects feedback type
  - Common tasks – modest feedback
  - Errors/uncommon tasks – substantial feedback
- Visual approaches make feedback easy

Adobe Premiere Pro 1.5 (USD$500-700)

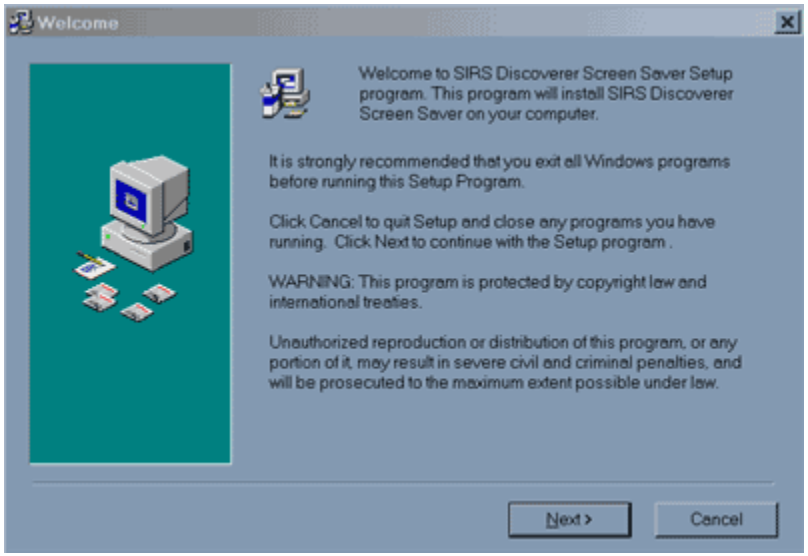




(not more useful, but more honest)

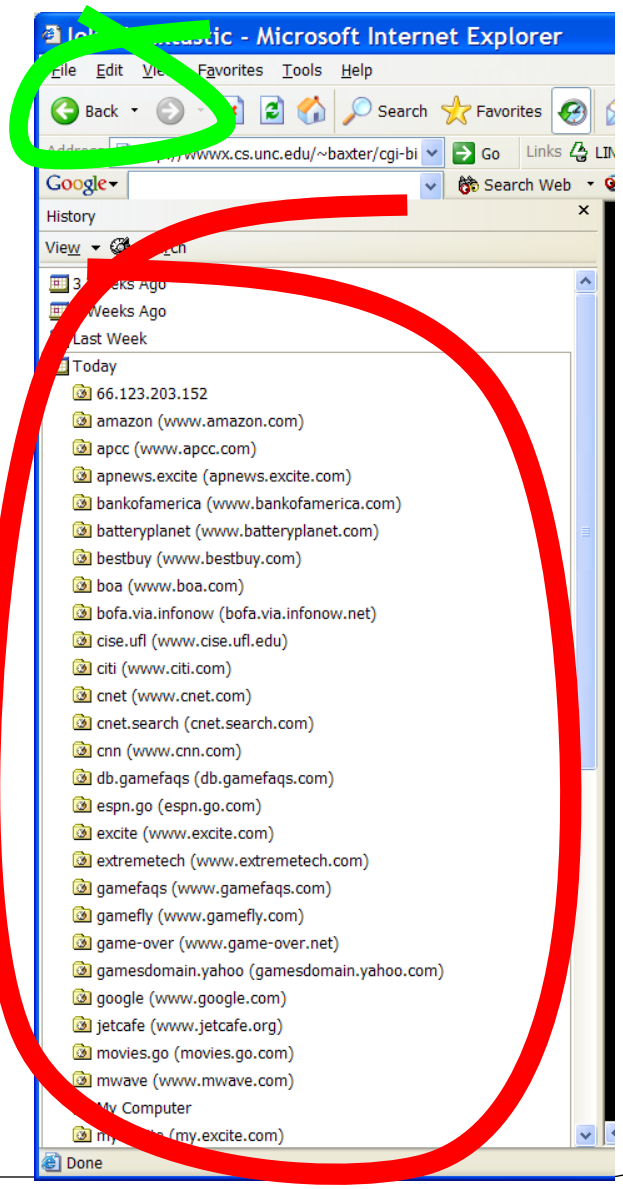
4. Design Dialogs to Yield Closure

- Action sequences should have a beginning, middle, and end.
- Feedback provides sense of accomplishment
- E.g., Purchasing items via internet has a clearly defined step-by-step process



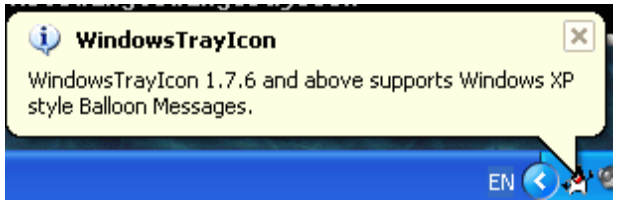
5. Permit easy reversal of actions

- As much as possible, actions should be reversible
  - Trash can
  - Relieves anxiety
- Design decision should include
  - History size
  - What does it mean to undo something?
- Let the user ***know*** they can reverse an action



6. Support Internal Locus of Control

- Experiences operators want to feel in control
  - User is in charge of the interface
  - Interface rapidly responds to the user



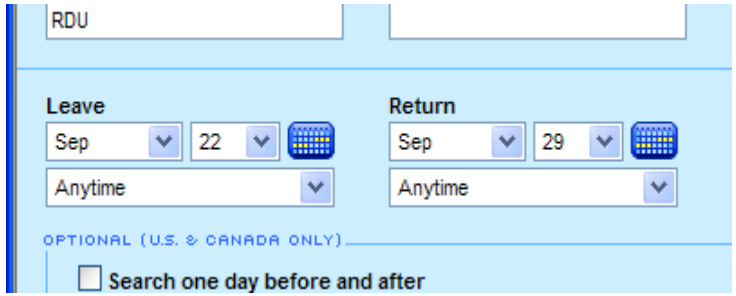
- Builds anxiety and dissatisfaction
  - Surprising interface actions
  - Tedious actions
  - Difficulty in obtaining necessary ability
  - Difficulty in producing action
  - Ex. Long lag when using UI
- **Good rules:** Avoid a*causality*, make users initiators rather than responders
  - Acausality (BAD)
    - Ex. some sound happens to get your attention (user not involved), little paper-clip appears when not expected
  - Causality (user in control) (GOOD)
    - Ex. Sound when clicking on a link

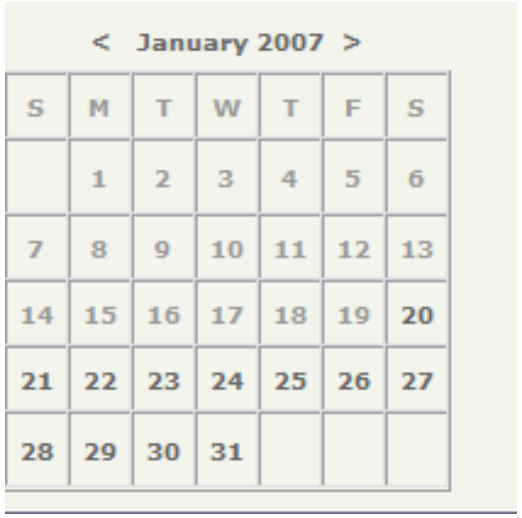
7. Reduce Short-term Memory Load

- Rule of thumb: Humans can remember 7 +/- 2 chunks of information
- Keep display simple
- Multiple page displays should be consolidated
- Training will be required if using codes, mnemonics, long sequence of actions
- You should provide online access to command-syntax, abbreviations, codes, etc. (i.e., provide help)

8. Prevent Errors

- Limit errors a user can make
  - Gray out menu items that don't apply
  - No characters in a numeric field
- In case of errors
  - Detect error
  - Simple, constructive, and specific instructions
  - Do not change system state





8. Prevent Errors (cont.)

- Error rate is typically higher than expected
  - What are common errors for us?
    - Coding, typing, dialing, grammar
- How can we design software to reduce them?
  - Better error messages
    - Helps fix current error
    - Helps reduce similar errors
    - Increases satisfaction
    - Specific, positive, and constructive
      - "Printer is off, please turn on" instead of "Illegal Operation"
  - Reduce chance for error
    - Organizing info, screens, menus
    - Commands and menu choices should be distinctive
    - State of the interface should be known (change cursor when busy)
    - Consistency of actions (Yes/No order of buttons)

8. Prevent Errors (cont.)

- Correct actions
  - Elevator – can't open doors until not moving
  - Aircraft engines – can't go in reverse unless landing gear is down
  - Choose a date from a visual calendar instead of having them type it in
  - Cell phones let you choose from recently dialed #s or received calls
  - Automatic command completion
  - Spell checker

8. Prevent Errors (cont.)

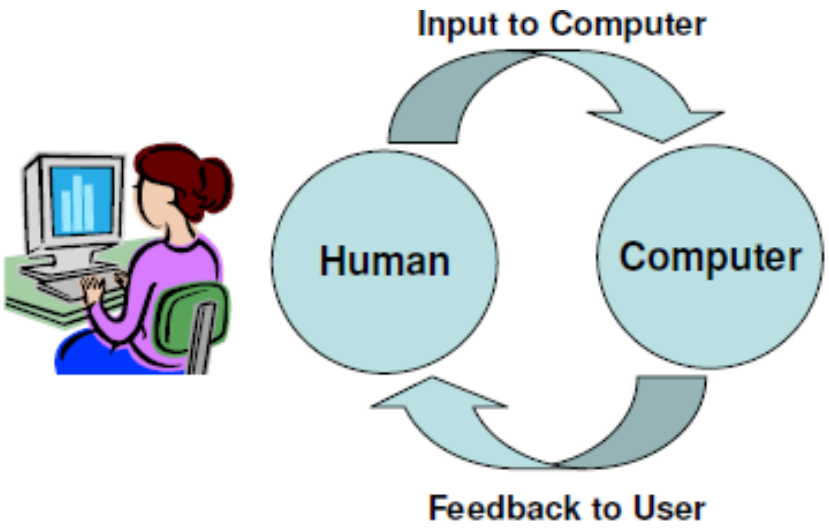
- Complete Sequences
  - One action can perform a sequence of events
    - E.g., Left turn signal (front and rear light flashing)
  - Study usage, error patterns, and user preferences via user groups, studies
  - Log errors
- Universal Usability can help lower errors
  - Large buttons helps with readability, and reduces error

Theories

- Beyond the specifics of guidelines
- Principles are used to develop theories
- A case study: Stages of Action

“Stages of action models”

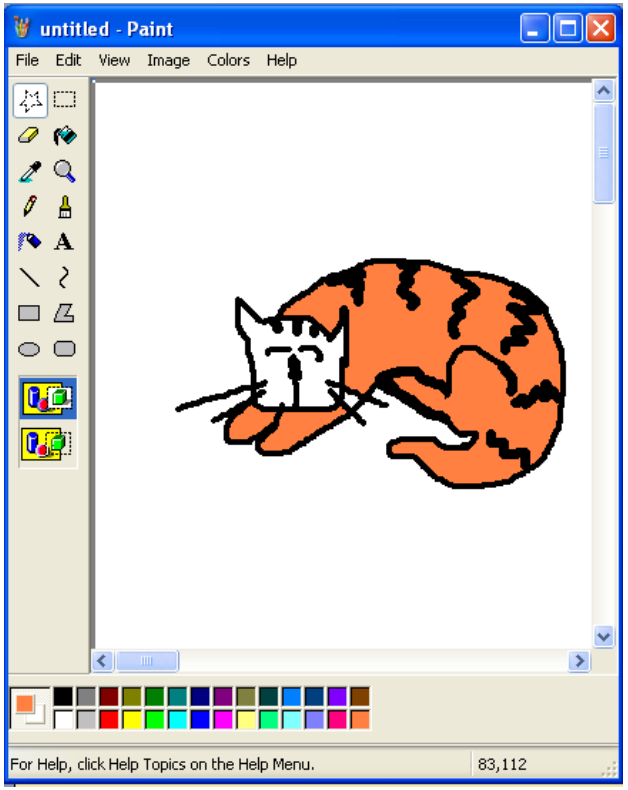
- Seven stages of action theory by Donald Norman:
  1. Forming the goal
  2. Forming the intention
  3. Specifying the action
  4. Executing the action
  5. Perceiving the system state
  6. Interpreting the system state
  7. Evaluating the outcome



- Norman's contributions
  - Context of cycles of action and evaluation.
  - *Gulf of execution*: Mismatch between the user's intentions and the allowable actions
  - *Gulf of evaluation*: Mismatch between the system's representation and the users' expectations

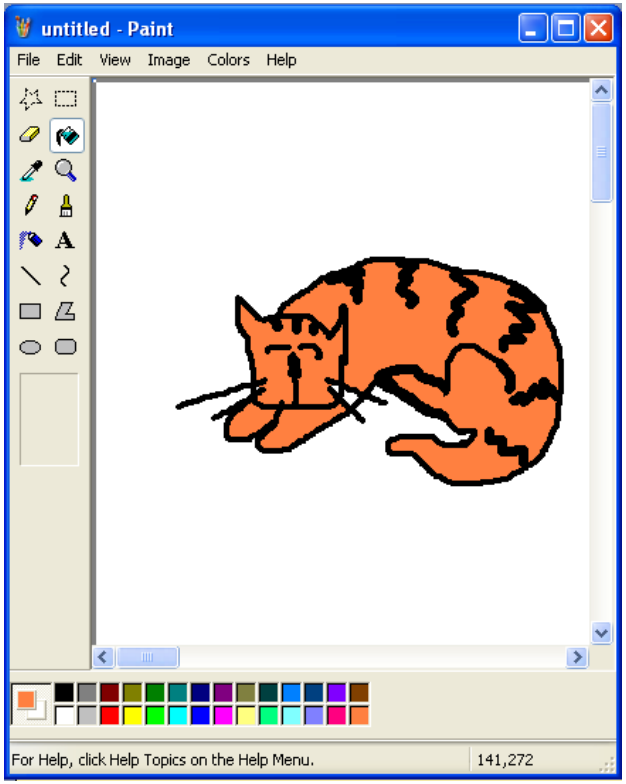
Norman's Example

1. Forming the goal
   * I want to paint the cat's head
2. Forming the intention
   * I will use the paintbucket (instead of brush)
3. Specifying the action
   * To do this, I need to click on the paint-bucket icon then the cat's head region
4. Executing the action
   * Physically doing the action with mouse and clicks.



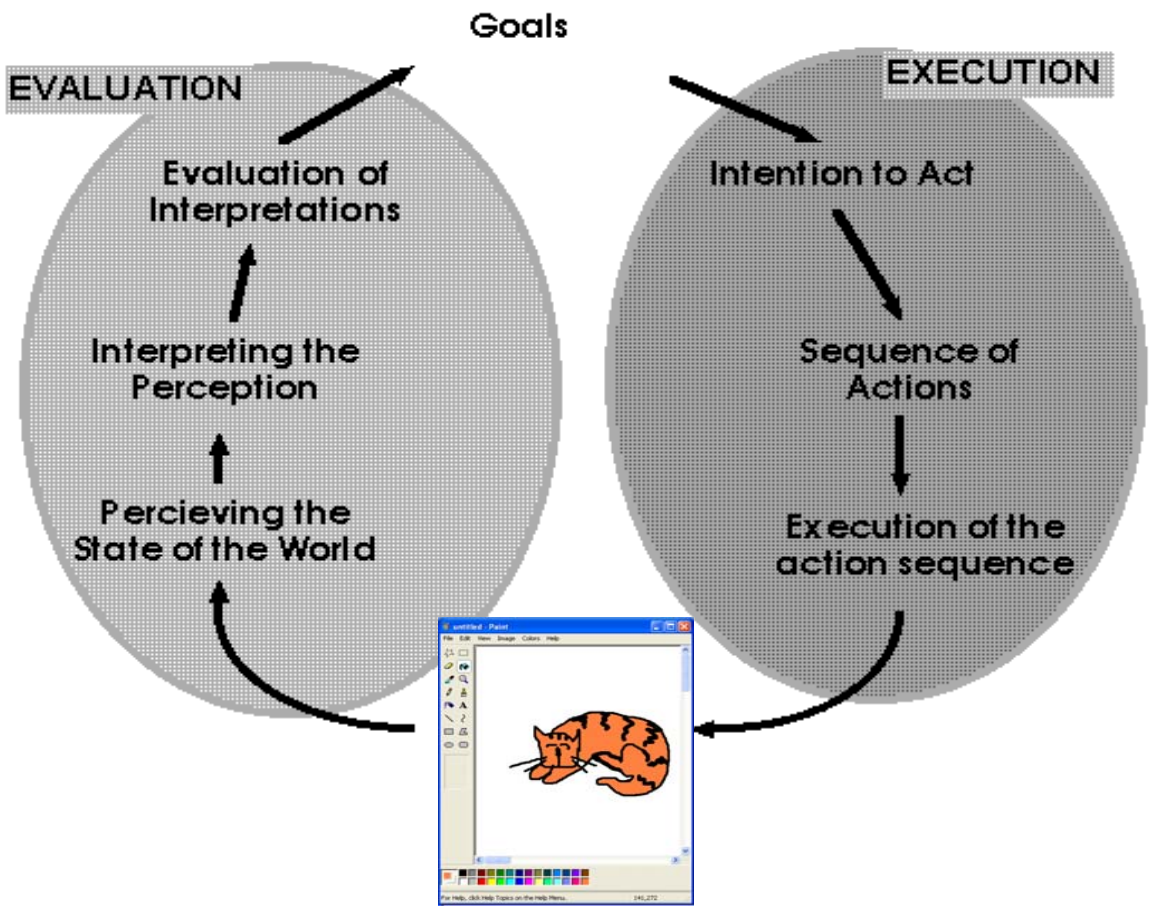
Norman's Example

5. Perceiving the system state
   * Display has changed
6. Interpreting the system state
   * Cat head is orange
7. Evaluating the outcome
   * Outcome is good, I'm a happy user.



Norman's Example

- Provides a "cycle" theory of usage



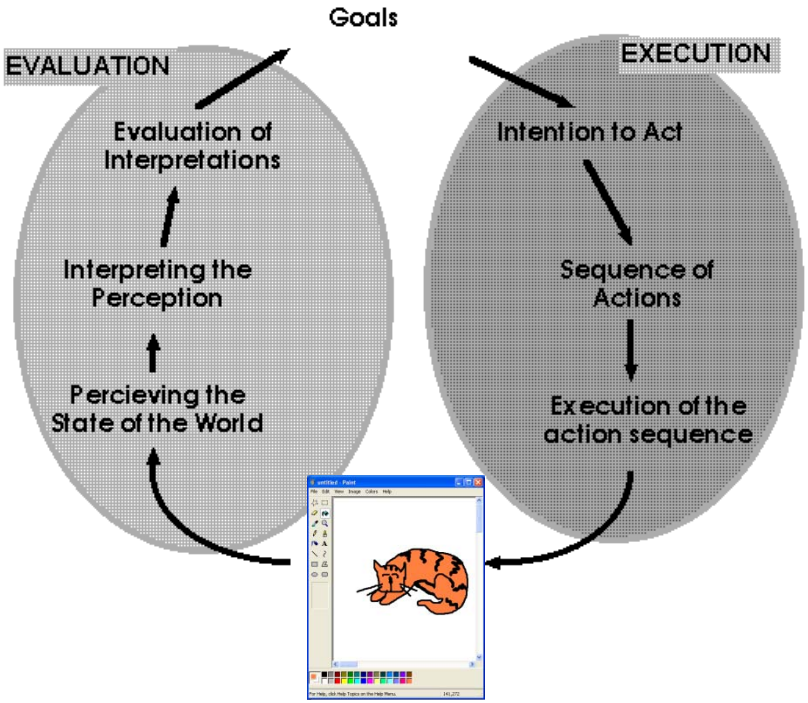
Issues in Execution and evaluation

- **Gulf of execution**
  - **Mismatch between the user's intentions and the allowable actions**
- **Gulf of evaluation**
  - **Mismatch between the system's representation and the users' expectations**

Gulf of evaluation error happens here.

Could be a real error in the system, or mismatch between what the user expected.

For example, imagine you want to paint the cat's head orange, but the program paints the entire image orange!



Gulf of execution error happens on this side.

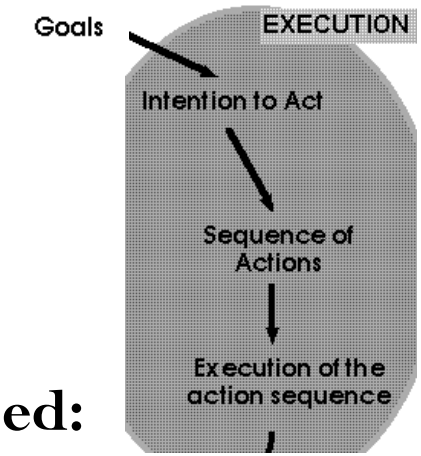
User wants to paint the cat's head stripped.

There is no corresponding action to perform this.

The user is lost, confused, and makes errors.

Suggestions from the Theory

- **Four critical points where user failures can occur**
  - Users can form an inadequate goal
  - Might not find the correct interface object because of an incomprehensible label or icon
  - May not know how to specify or execute a desired action
  - May receive inappropriate or misleading feedback



- **To avoid "gulf" errors, the following is proposed: Four principles of good design**
  - Have a good conceptual model with a consistent system image
  - State and the action alternatives should be visible
  - Interface should include good mappings that reveal the relationships between stages
  - User should receive continuous feedback

Summary

- Guidelines
  - Narrow rules established to guide basic UI design
- Principles
  - Widely accepted procedures and rules for UI design
- Theories
  - High-level analysis of users to help explain reasons for designs and predict usability