

## *HCI for Recommender Systems: An Introduction*

Joseph A. Konstan  
konstan@umn.edu



UNIVERSITY OF MINNESOTA

## *Do Interfaces Matter?*

*(Adapted from Norman, Design of Everyday Things)*

- A Two-Person Game
  - Start with the numbers 1, 2, 3, ..., 9
  - Alternate turns, taking one number at a time
  - A player wins with *any 3 numbers* that sum to 15
    - 1, 3, 9, 5 wins because  $1+9+5$  equals 15
    - 9, 6, 8, 7 does not win, because no 3 sum to 15
  - Tie if numbers used up without a winner



Konstan: Introduction to User Interfaces

## *Magic Square*

2	7	6
9	5	1
4	3	8



Konstan: Introduction to User Interfaces

## *What Makes a Bad UI?*

- Hard to Learn?
- Hard to Use?
  - Automated teller machines
  - Grocery checkout machines
  - Piano
  - Violin
  - Hammer
- Does not Fit the Task?



Konstan: Introduction to User Interfaces

## *Foundations of User Interfaces*

- Field of Human-Computer Interaction (HCI)
  - Psychology
  - Computer Science
  - Ergonomics
  - other disciplines
- Focus: Design Computer Systems for Humans



Konstan: Introduction to User Interfaces

## *Human Capabilities*

- Humans are very good at:
  - recognizing (images, voices, etc.)
  - associative memory
  - explaining phenomena
- Humans are very limited in:
  - short-term memory
  - complex, multi-layered tasks
  - perfection



Konstan: Introduction to User Interfaces

## *Brain Hemisphere Research*

- “Left Brain”
  - methodical, logical, step-by-step
  - symbolic, works with components
  - generally dominant
- “Right Brain”
  - wholistic, intuitive, rapid
  - handles missing values
  - works with gestalts



Konstan: Introduction to User Interfaces

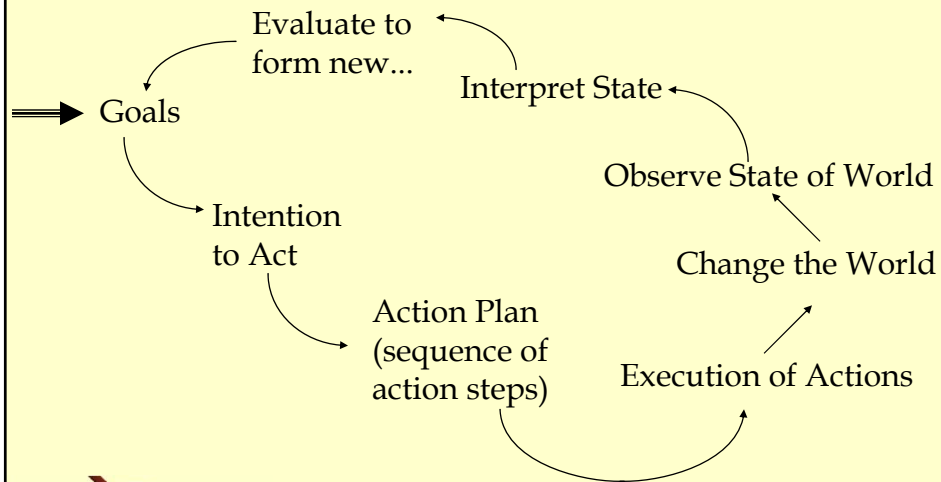
## *Limits of Human Memory*

- Short-Term Memory
  - instant recall
  - limited capacity
  - fragile
- Long-Term Memory
  - slower recall, depends on organization
  - rote memory vs. relationships vs. explanation
  - “muscle memory”



Konstan: Introduction to User Interfaces

## *Norman's Model of User Action*



Konstan: Introduction to User Interfaces

## *Humans Err*

- Humans are not perfect!
- Slips -- errors in automatic actions
  - tied to skilled behaviors
  - easy to detect
- Mistakes -- errors in intention or logic
  - e.g., false generalizations
  - may be hard to detect



Konstan: Introduction to User Interfaces

## *Where Does This Put Us?*

- The Problem
  - humans are imperfect!!
- Possible Solutions
  - yank them out of the process
    - lose benefits of human strengths
  - design for imperfect users



Konstan: Introduction to User Interfaces

## *Put Support into the Interface*

- Affordances
- Visibility of Controls
- Feedback
- Conceptual Models
- Mappings
- Information in the World
- Constraints
- Error Avoidance and Handling
- Standardization



Konstan: Introduction to User Interfaces

## *Affordances*

- What something can be used for
  - a button (or plate) affords pushing
  - a knob affords turning
- Cultural (and learned) affordances
  - a scrollbar affords scrolling
  - various cursors afford operations
- Key: helps the user discover possibilities



Konstan: Introduction to User Interfaces

## *Visibility of Controls and Information*

- Don't hide the controls!
  - telephone systems: hold, transfer, etc.
  - VCR programming
- Make status available
  - well-designed display (34% complete)
  - use sound if needed (click/beep/etc.)



Konstan: Introduction to User Interfaces

## *Feedback*

- Don't hide the results!
- Make feedback immediate
  - did I hit the button? (visual or audio)
  - did I have an effect? (cursor change?)
- Each action should have an effect
  - promote exploration



## *Conceptual Models*

- Rote memorization prevents inference and adaptation
  - users will develop conceptual models
    - but they will likely be wrong!
- Models should help people adapt to new situations
  - gulf of execution -- not knowing how
  - gulf of evaluation -- not knowing whether it worked



## *Mappings*

- Humans infer from mappings
  - layout of light switches in a room
  - controls on a range
- Natural mappings are easiest, but ...
  - avoid mappings that don't generalize



## *Information in the World*

- Avoid relying on memory alone
  - menus and toolbars
- Support memory aids
  - never require remembering information between screens
- Great precision is not required



## *Constraints*

- Narrow the task search space
- Physical Constraints
- Semantic Constraints
- Cultural Constraints
- Logical Constraints



Konstan: Introduction to User Interfaces

## *Error Avoidance/Handling*

- Design to prevent slips
  - different things should look different
  - consistent confirmation is useless
  - immediate confirmation can be nearly useless
- Simplify tasks
  - make decision trees narrow or shallow



Konstan: Introduction to User Interfaces

## *Error Avoidance/Handling*

- Support recovery from errors
  - undo operations and back-up versions
  - support exploration towards a goal
- Prevent errors with forcing functions
  - don't make illegal operations available
  - disable buttons or menus
  - turn illegal operations into legal ones



## *Standardization*

- If all else fails, ...
  - fewer things to memorize
  - shorter learning time
  - clocks should run clockwise



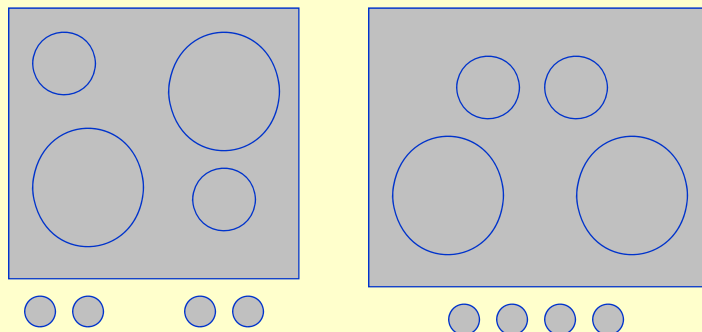
## *Examples*

- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones



Konstan: Introduction to User Interfaces

## *Stove Control Design*



Konstan: Introduction to User Interfaces

## Examples

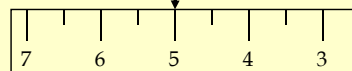
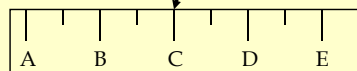
- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones



Konstan: Introduction to User Interfaces

## Refrigerator Controls

NORMAL SETTINGS C AND 5  
COLDER FRESH FOOD C AND 6 - 7  
COLDEST FRESH FOOD B AND 8 - 9  
COLDER FREEZER D AND 7 - 8  
WARMER FRESH FOOD C AND 4 - 1  
OFF (FRESH FD & FRZ) 0



Konstan: Introduction to User Interfaces

# Examples

- Stove Control Design
- Refrigerator controls
- Light Switches
- One-button slide projectors
- Doors
- Phones



The screenshot shows the MovieLens website interface. At the top, there is a browser window with the URL <http://www.movielens.org/main>. Below the browser window, there is a navigation bar with links for Home, Find Movies, Q&A (new), Preferences, and Help. The main content area is divided into several sections:

- Shortcuts**: A sidebar menu with links for Rate and Find Movies (Top Picks For You, Newest Additions, Most Often Rated, Rate Random Movies, Browse Movies by Tags), Your Movies (Your Ratings, About Your Ratings, Your Wishlist, Your Tags), Your Account (Your Profile (edit), Preferences, Manage Buddies, Manage RSS Feeds), and Help MovieLens (Volunteer Center).
- Search**: A search bar with the text "Search".
- New Movies**: A list of new movies with their ratings and titles, including Inception (2010), Town, The (2010), Kids Are All Right, The (2010), Scott Pilgrim vs. the World (2010), Get Low (2009), Four Lions (2010), Soul Kitchen (2009), American, The (2010), Machete (2010), and Disappearance of Alice Creed, The (2009).
- New DVDs**: A list of new DVDs with their ratings and titles, including Secret in Their Eyes, The (El secreto de sus ojos) (2009), Girl with the Dragon Tattoo, The (Män som hatar kvinnor) (2009), Reindeerspotting - Peko joulumaasta (2010), Mother (Madeo) (2009), Ghost Writer, The (2010), Kick-Ass (2010), Town Called Panic, A (Panique au village) (2009), Docks of New York, The (1928), Underworld (1927), and Batman: Under the Red Hood (2010).
- Movie Tuner**: A section titled "Movie Tuner New!" with a slider control for "What I want" (less, ok, more) and a text input field. Below the slider, there are three rows of controls for "action", "classic", and "surreal" genres, each with a slider and a radio button. The text input field contains the text: "Want a movie like Pulp Fiction but less 'violent'? Or a movie like Mission: Impossible but more 'realistic?'"

## *Task-Centered UI Design*

*(adapted from Lewis and Rieman, Task-Centered User Interface Design)*

- The interface should match the users and what the users are trying to accomplish
- The development process should use the users' tasks throughout design and evaluation



Konstan: Introduction to User Interfaces

## *Usage Stories*

- Stories to motivate and capture design
  - sometimes called "scenarios"
- For our purposes:
  - capture the essence, or the quintessence of use
  - include application and context
  - one to two paragraphs



Konstan: Introduction to User Interfaces

## *Example*

- For several years, Juan has wanted to exercise more and lose weight, but he's found it difficult. Now that he's purchased an exer-minder wristwatch, he's finding it easier. At any time, Juan can check the watch and with the press of a single button see how much exercise he's had, measured in either steps or calories.
- He's also programmed the watch to alert him at key times in the day if he's falling behind. In the morning, the watch beeps if he didn't meet his goal the previous day, offering encouragement to get back to exercise today. At lunchtime he gets a prompt reporting on how much he's exercised so far during the day, and reminding him to eat well at lunch. Again, he gets a reminder after work, encouraging him to walk (if needed) or stop at the gym. In the two weeks since he started wearing the watch, Juan has been exercising more regularly and has lost 1.5 kilograms.



## *Key Components of TCUID*

- Users
- Tasks
- Scenarios



## *Users*

- Who is going to use the system?
  - if you can't find a user -- you're in trouble
  - "everyone" is not a user
  - "the designer" is not a good user
- Go talk with the user
  - too busy?
    - how will they have time to evaluate/use it?
    - are there good surrogate users?



Konstan: Introduction to User Interfaces

## *Talking with the Users*

- What do they know?
  - systems, skills, etc.
- What do they do?
  - tasks
- How do they do it now?
  - scenarios
- What do they want to do?
  - new tasks



Konstan: Introduction to User Interfaces

## *Users Sometimes Bite!*

- Users aren't all-knowing
  - they may not understand the possibilities
  - they may have a very narrow view
- They aren't designers
  - learn about the tasks from the users
  - use your design skills to create a design
  - get user feedback on the design/prototype



Konstan: Introduction to User Interfaces

## *Tasks*

- What is a task?
  - a specific description of a complete job that specific users want to accomplish
  - not tied to how they would do the job
- Detailed
  - some typical details are important
- Complete job
  - covers transitions between sub-tasks



Konstan: Introduction to User Interfaces

## Example Task

- (e.g., for Netflix)
  - Joe recently saw an episode of *The West Wing* on TV while in a hotel and remembered that he really enjoyed it. He decided it might be a fun program to watch more regularly at home, but he'd really prefer to see the whole series in order. He remembers that his TiVo can get content from Netflix – he set it up a while ago but had never found anything to watch. He checks whether Netflix can deliver *The West Wing* directly to his TV set, and if so sets it up so he can watch the first episode now.



## Another Task

- (e.g., for MovieLens)
  - Pearl just saw the movie *Return of the Jedi* again, and remembered how much she liked it. She goes into MovieLens to see whether she'd ever rated it, and if not, to give it a rating of 5. She also wants to see her predictions for the other Star Wars movies.



## *Why Tasks*

- Tasks are fundamental to TCUID
  - determine who actually uses the system
  - sets goals for system functionality
  - basis for system design
  - basis for comparative evaluation
  - basis for user testing



## *How Many Tasks?*

- Depends on nature of problem
  - 3-5 general-purpose tasks for a simple system
  - separate tasks for special-purpose cases (maintenance, installation)
  - 10+ tasks for complex systems
  - depth/quality more important than number of tasks



## *From Task to Design*

- Write-up tasks, circulate among users
  - clarify missing details
- Rough out an interface, using existing systems or designs where possible
- Sketch out how each task would be accomplished in the interface: develop *scenarios*



## *Scenarios*

- Specific instance of system use
  - for a particular task
  - for a particular interface
  - what would the user do, in detail
- Example
  - click on "search" tab; type "Return of the Jedi" into the text field; hit return; look under "your Rating" column to see number of stars; ...



## *Properties of Scenarios*

- Interface-dependent
- Detail appropriate to user, task, interface
- Brings forward issues
  - how components work together
  - design arguments
  - tricky parts of the interface
- Guideline to create prototype



## *User Environment*

- Physical
- Social
- Cultural
  
- Independence
- Management
- Accountability
- Risk Behaviors



## *Studying Users and Tasks*

- Interviews / Focus Groups
- Site Visits / Observation
  - Contextual Inquiry
- Logs of Current Behavior
- Participatory design
  
- A plethora of methodologies



Konstan: Introduction to User Interfaces

## *Contextual Inquiry*

- Tools and techniques
  - Ethnographic observation
  - Artifacts as prompts
  - Goals and focus
    - Knowledge to set goals and focus



Konstan: Introduction to User Interfaces

## *Personas*

- Popular technique
  - Research-driven
  - Narrative
  - Descriptions of “model” users
- Basis
  - Cluster users by relevant attributes
  - Identify clusters
  - Create “realistic” representatives
  - Force you to consider whether your design is appropriate



## *Example Persona*

- Sara is a graduate student living in Minneapolis. She travels by plane about three times a year – about half of that time for conference trips for her University research group. When she travels on her own dollar, she is very price-conscious, and wants to be sure to get the lowest price, even if that involves obscure routes or indirect trips. When she travels for the University, she is happy to let their staff make the arrangements.

As a computer scientist, Sara knows all about search engines and other computer systems. Sometimes this makes her skeptical that the system may be hiding the best fares. ...



## *Another Persona*

- Patricia is a 31 year old accountant for a technical publisher who has used Windows for six years at the office. She is fairly competent and technical. She installs her own software; she reads PC Magazine; she has programmed some Word macros. She has a cable modem for her home PC. She's never used a Macintosh. "They're too expensive", she'll say, "you can get a quad-core PC with four gig of RAM for the price of..."



Konstan: Introduction to User Interfaces

## *Yet Another Persona*

- Nelson has been an English professor at Carleton College since 1975. He's written several books of poetry and has been using computer word processors since 1980, but has only used two programs, WordPerfect and Microsoft Word. He doesn't care **how** computers work; he stores all his documents in whatever directory they get put in if you don't know about directories.



Konstan: Introduction to User Interfaces

## *Design: Broad Spectrum*

- System Design
  - Requirements
  - Architecture
- Information Design
- Graphic Design
- Interaction Design



Konstan: Introduction to User Interfaces

## *Design as a Discipline*

- Balance between creation and evaluation
  - What *could* be
  - What *should* be
- Form vs. function
  
- Design as “good taste”
  - How do you get it?



Konstan: Introduction to User Interfaces

## *Interface Evaluation*

- Goals of interface evaluation
  - find problems
  - find opportunity for improvement
  - determine if interface is “good enough”



## *With or Without Users*

- Users are expensive and inconsistent
  - usability studies require several users
  - some users provide great information, others little
- Users are users
  - cannot be simulated perfectly
- Best choice--Both



## *Evaluation Without Users*

- Quantitative Methods
  - GOMS/keystroke analysis
  - back-of-the-envelope action analysis
- Qualitative Methods
  - expert evaluation
  - cognitive walkthrough
  - heuristic evaluation



Konstan: Introduction to User Interfaces

## *GOMS/Keystroke Analysis*

- Formal action analysis
  - accurately predict task completion time for skilled users
- Break task into tiny steps
  - keystroke, mouse movement, refocus gaze
  - retrieve item from long-term memory
- Look up average step times
  - tables from large experiments



Konstan: Introduction to User Interfaces

## *GOMS/Keystroke Analysis*

- Primary utility: repetitive tasks
  - e.g., telephone operators
  - benefit: can be very accurate (within 20%)
  - may identify bottlenecks
- Difficulties
  - challenging to decompose accurately
  - long/laborious process
  - not useful with non-experts



Konstan: Introduction to User Interfaces

## *Back-of-the-Envelope Action Analysis*

- Coarse-grain
  - list basic actions (select menu item)
  - each action is at least 2-3 seconds
  - what must be learned/remembered?
  - what can be done easily?
  - documentation/training?
- Goal is to find major problems
  - Example: 1950's 35mm camera (example from Lewis and Rieman's TCUID)



Konstan: Introduction to User Interfaces

## 1950's 35-mm camera

- - take light meter out of pocket
- - make sure film speed is set correctly on light meter
  - -- remember: what speed of film is in the camera
- - aim light meter at scene
- - read light value from needle on light meter
- - adjust calculator dial on light meter to light value
- - the calculator shows several possible combinations of f-stop and shutter speed, all of which give the correct exposure. So...
- -- remember: big f-stop number means small lens opening
- -- remember: small lens opening means more depth of field
- -- remember: big shutter speed number means short exposure
- -- remember: short exposure means moving things aren't blurred
- -- decide: which combination of f-stop and speed to use
- - set the f-stop on the camera lens
- - set the shutter speed on the camera speed dial
- - remove the lens cap
- - cock the shutter
- - aim the camera
- - focus the camera
  - -- remember: if lens aperture is small, depth of field is shallow
- - press the shutter release
- - advance the film



## Expert Evaluation

- Usability specialists are very valuable
  - double-specialists are even better
- An inexpensive way to get a lot of feedback
- Be sure the expert is qualified in your area



## *Cognitive Walkthrough*

- Goals
  - imagine user's experience
  - evaluate choice-points in the interface
  - detect confusing labels or options
  - detect likely user navigation errors
- Start with a complete walkthrough scenario
  - never try to "wing it" on a walkthrough



## *Tell a Believable Story*

- How does the user accomplish the task
- Action-by-action
- Based on user knowledge and system interface



## *Best Approach*

- Work as a group
  - don't partition the task
- Be highly skeptical
  - remember the goal!
- Every gap is an interface problem



## *Who Should Do the Walkthrough*

- Designers, as an early check
- Team of designers & users
  - remember: goal is to find problems
  - avoid making it a show
- Skilled UI people may be valuable team members



## *How Far Along*

- Basic requirements
  - description or prototype of interface
  - know who users are (and their experience)
  - a task description
  - a list of actions to complete the task (walkthrough scenario)
    - DO NOT try to create the action list on the fly!
- Viable once the scenario and interface sketch are completed



Konstan: Introduction to User Interfaces

## *How to Proceed*

- For each action in the sequence
  - tell the story of why the user will do it
  - ask critical questions
    - will the user be trying to produce the effect?
    - will the user see the correct control?
    - will the user see that the control produces the desired effect?
    - will the user select a different control instead?
    - will the user understand the feedback to proceed correctly?



Konstan: Introduction to User Interfaces

## *More on Questions*

- Some extras can be helpful:
  - What happens if the user is wrong? Feedback to self-correct?
  - How would a former user of <alternative product> react here?
- Remember, the question is a way to help you see problems – it is a focus, not a blindfold.



## *Walkthroughs are not Perfect*

- They won't find every problem
  - limited by nature
    - new users who know what task they need to accomplish
    - biased towards correct action sequence
  - limited in implementation
    - hard to shed the expertise of evaluators
- A useful tool in conjunction with others



## Heuristic Evaluation

- Usability heuristics are broad “rules of thumb” that describe features of usable systems
  - Derived by evaluating common design problems across a wide range of systems
  - Not task-oriented
- Heuristic evaluation is a procedure for applying heuristics to evaluate a design – an “expert evaluation”
- See <http://www.useit.com/papers/heuristic/>



Konstan: Introduction to User Interfaces

## Jakob Nielsen's heuristics

1.0 – circa 1990	2.0 – circa 1994
Simple and natural dialog	Aesthetic and minimalist design
Speak the user's language	Match between system and real world
Minimize user memory load	Recognition rather than recall
Be consistent	Consistency and standards
Provide feedback	Visibility of system status
Provide clearly marked exits	User control and freedom
Provide shortcuts	Flexibility and efficiency of use
Provide good error messages	Help users recognize, diagnose, and recover from errors
Prevent errors	Error prevention
Help and documentation	Help and documentation



Konstan: Introduction to User Interfaces

## *Pros / Cons*

- + Cheap (no special lab or equipment)
- + Easy
- + Fast (about 1 day)
- + Cost-effective
- + Detects many problems without users
- + Complementary to task-centered approaches
- + Coverage
- + Catches cross-task interactions
- - Requires subjective interpretation / application
- - Does not specify how to fix problems
- - Performance improves as evaluator knowledge increases



Konstan: Introduction to User Interfaces

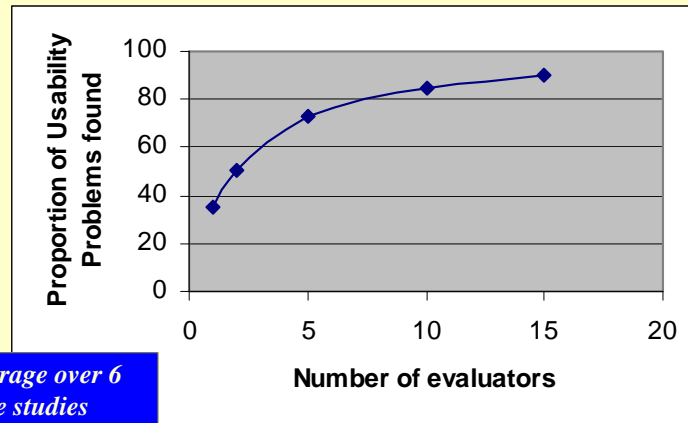
## *Procedure*

- A set of evaluators (3-5 is about optimal) evaluate a UI (some training may be needed)
- Each one independently checks for compliance with the heuristics
  - Different evaluators find different problems
- Evaluators then get together and merge their findings
- Collectively rate severity of the problems
- Debriefing/brainstorming → how to fix the problems (and point out what's really good)



Konstan: Introduction to User Interfaces

## Why multiple evaluators?



Average over 6 case studies



Konstan: Introduction to User Interfaces

## So how many evaluators?

- One evaluator does very poorly – only 35% of problems detected
- 5 evaluators find about 75% of problems
- So more is better, right?
  - Well...
  - More evaluators costs more
  - And don't find many more problems
  - So there are diminishing returns



Konstan: Introduction to User Interfaces

## *What an individual evaluator does*

- Each evaluator goes through the UI at least twice
- First, get an overall feel for the system
- Second, inspect the various interface elements and consider them in terms of the heuristics
  - May use a supplementary-list of domain-specific guidelines



## *Preparing the evaluators*

- If system is intended to be “walk up and use” or the evaluators are domain experts, no particular training is needed
- Otherwise, evaluators may need some knowledge about the domain and scenarios



## *Output of an individual Heuristic Evaluation*

- List of problems
- For each problem, what heuristics were violated



## *Severity ratings*

- Used to allocate resources to fix problems
- Based on
  - Frequency the problem will occur
  - Impact of problem (hard or easy to overcome)
  - Persistence (will users learn a work around or will they be bothered every time?)
- 1 - cosmetic problem
- 2 - minor usability problem
- 3 - major usability problem; important to fix
- 4 - usability catastrophe - must fix



## *Debriefing*

- Conduct with evaluators, observers, and development team members
- Discuss general properties of UI, including good points
- Brainstorm potential improvements to fix major usability problems
- Development team rates how much effort each fix would require



Konstan: Introduction to User Interfaces

## *The individual heuristics*

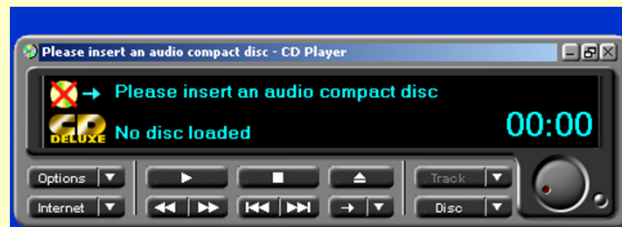
- 1990 - we'll discuss/illustrate
- 1994 - highlights FYI



Konstan: Introduction to User Interfaces

## 1. Simple and natural dialog

- Exploit the user's conceptual model
- Match user tasks as naturally as possible
  - Maximize mapping between interface and task semantics



## Simple and natural dialog

- Info should appear in natural order (for the task)
- Remove or hide irrelevant or rarely needed info
  - It competes for users' cognitive attention
- Less is more – easier to learn, fewer errors, less distraction...

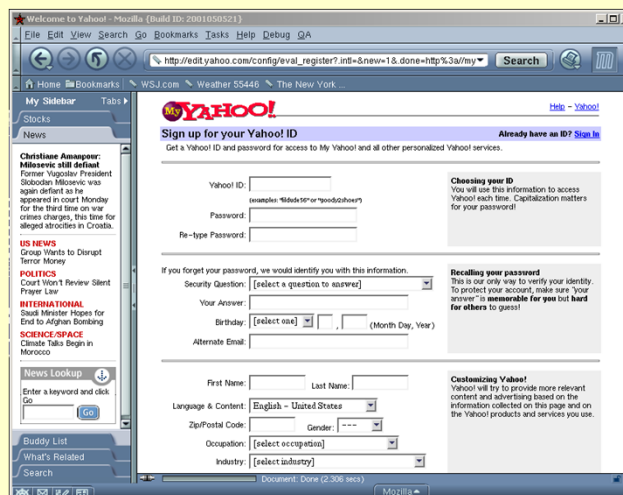


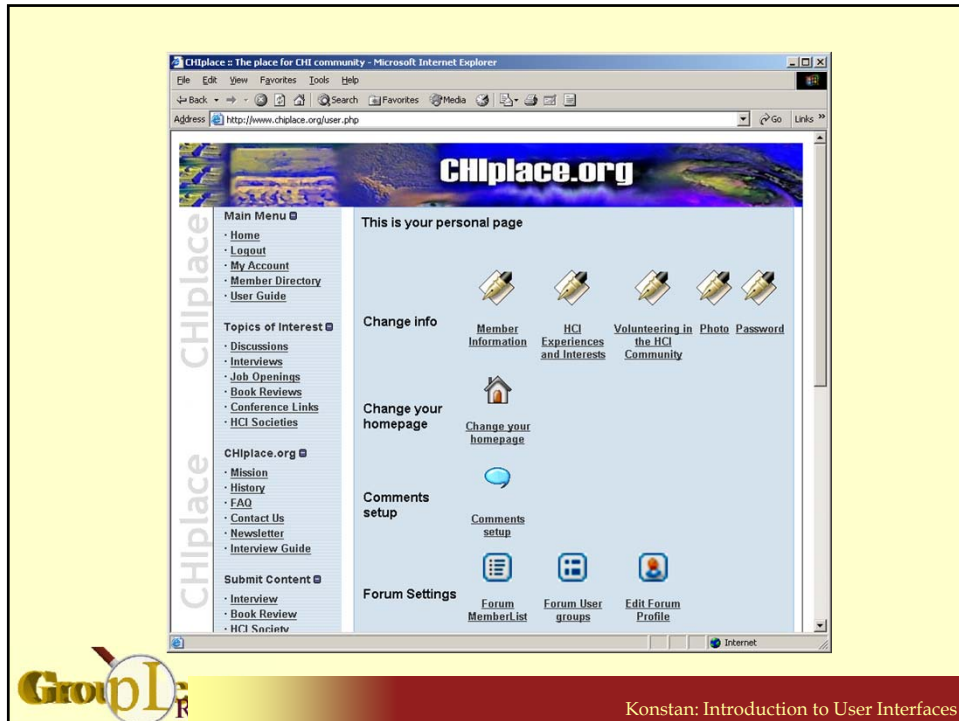
# Graphic Design Principles

- Color
  - Don't over-do it
  - Make sure interface works without it
  - Use color to categorize, differentiate, highlight
    - not to give information



# Grouping

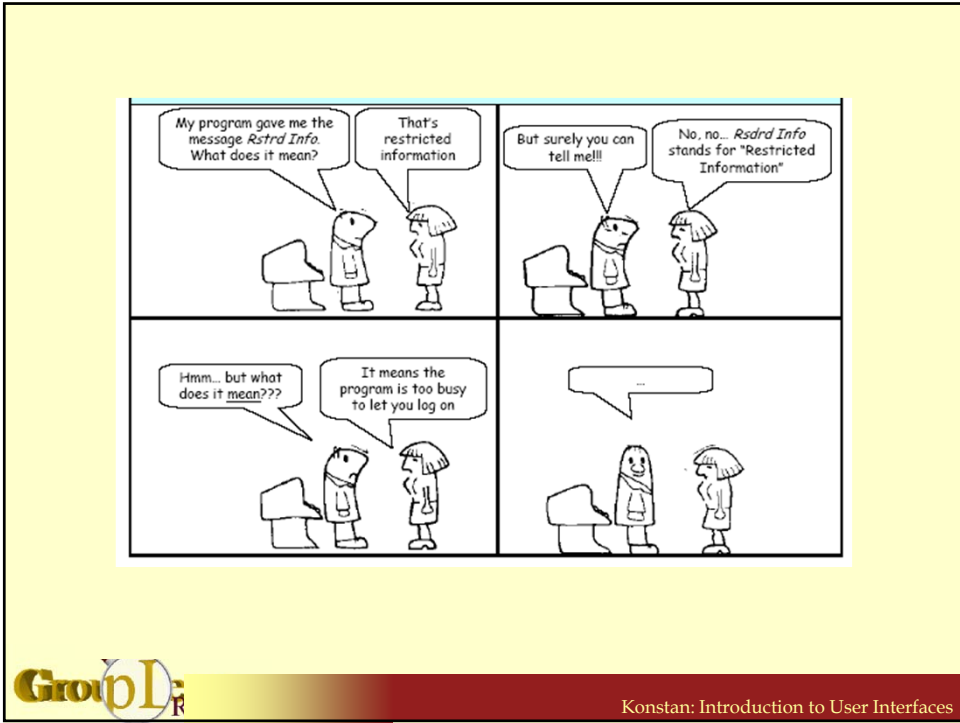




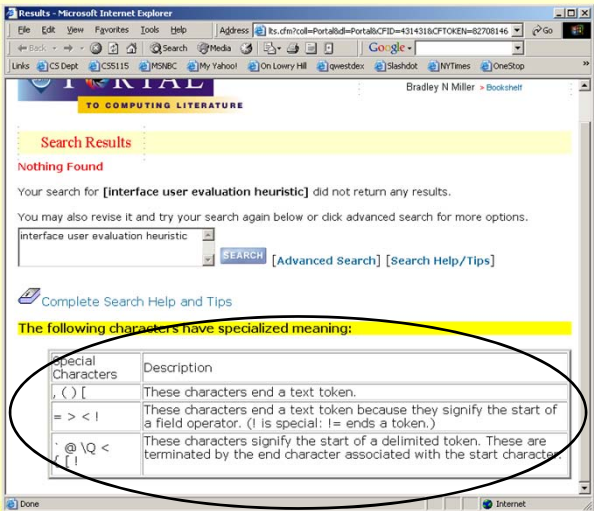
## 2. Speak the User's Language

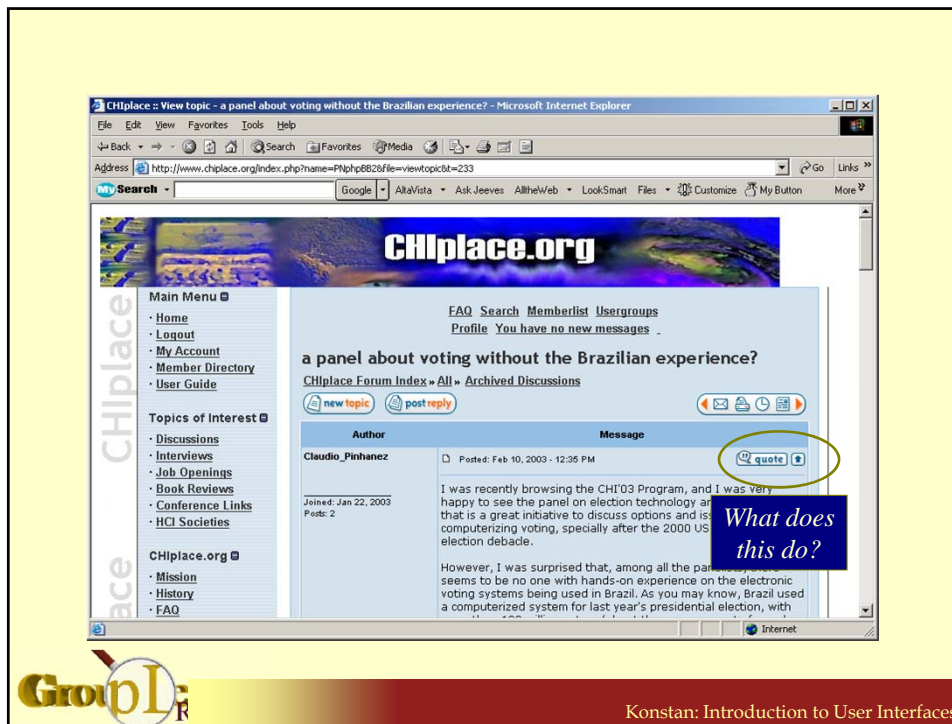
- Use terminology based on user's language for the task
- Avoid engineering jargon
- Use the user's native language
- Use conventional meanings
- View the interaction from the user's perspective
- Do not force naming conventions
- Exploit natural mappings and metaphors





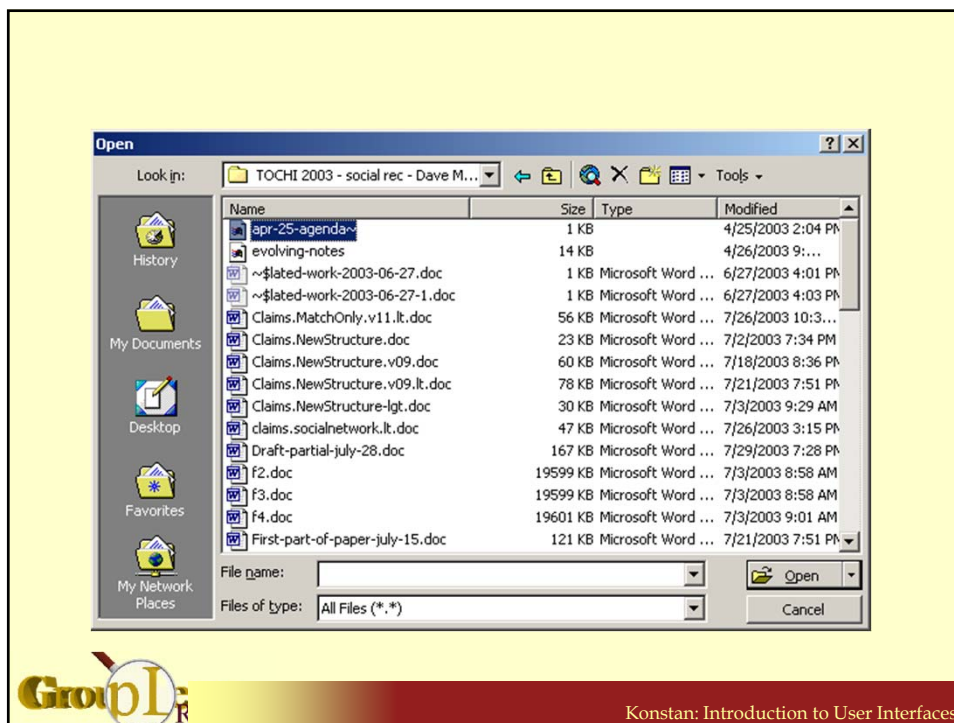
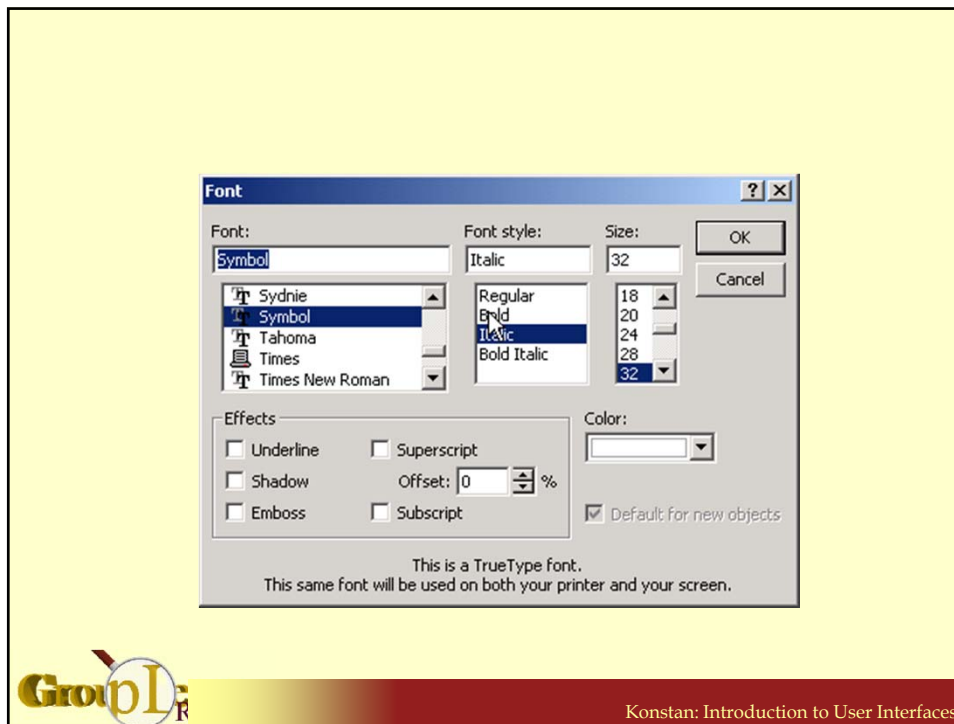
# Poor use of language

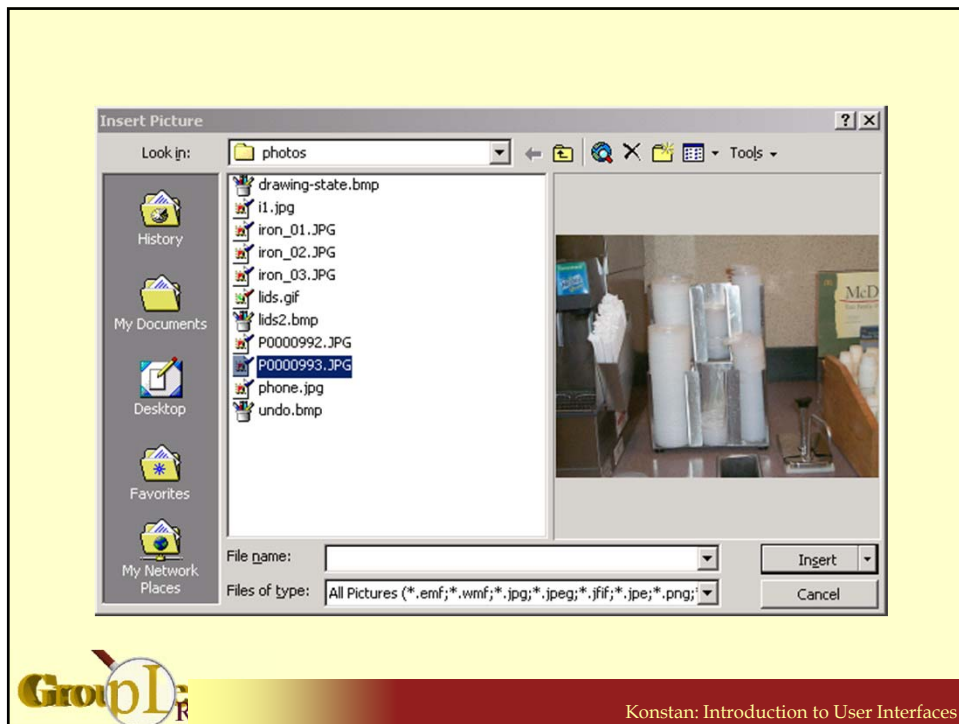




### 3. Reduce User Memory Load

- Promote recognition over recall
- Menus, icons, constrained input vs. command lines and free text fields
- Use examples, including sample input
- Don't make users remember information between actions or screens
- Leave information on the screen as long as it's needed



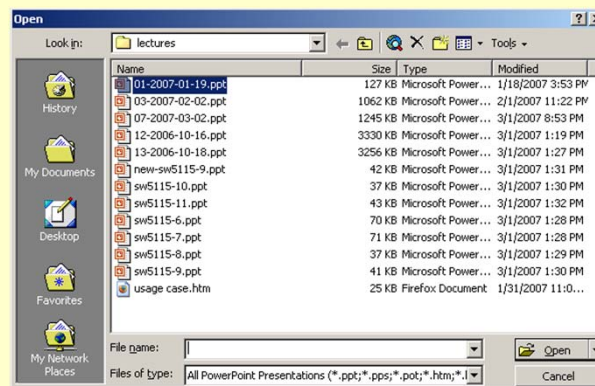


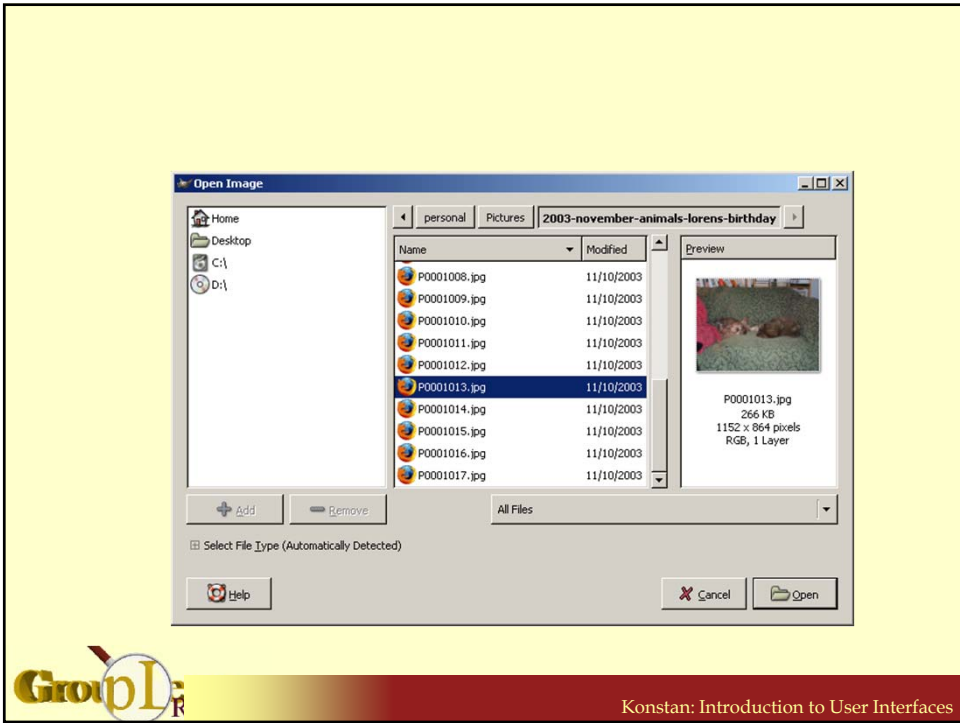
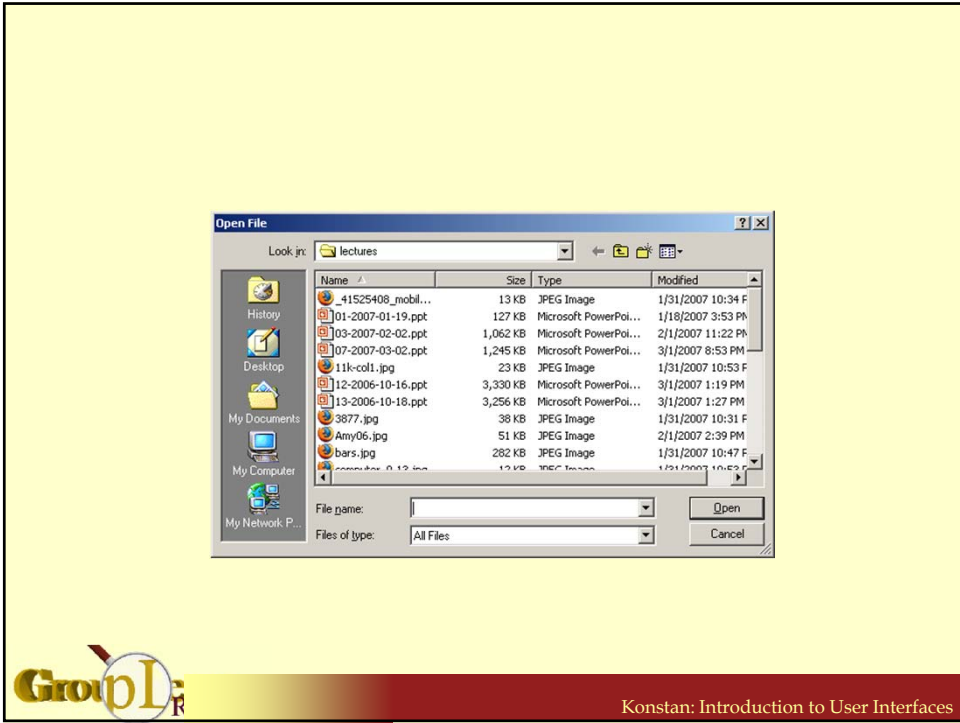
## 4. *Be consistent*

- Consistent with:
  - Task
  - User model
  - User experience (e.g., other interfaces)
- Consistent within an application

## *What should be consistent?*

- Location of information (e.g., on menu bar)
- Language / graphics
- Layout
- Input syntax
  - Use boilerplate forms
- Effects – same commands should have the same effect in the same situation (predictability)



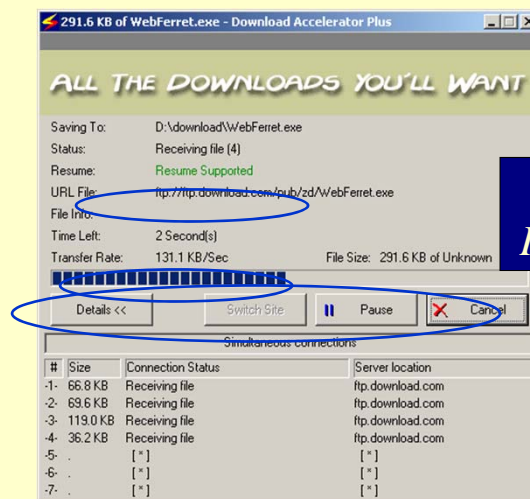


## 5. Provide Feedback

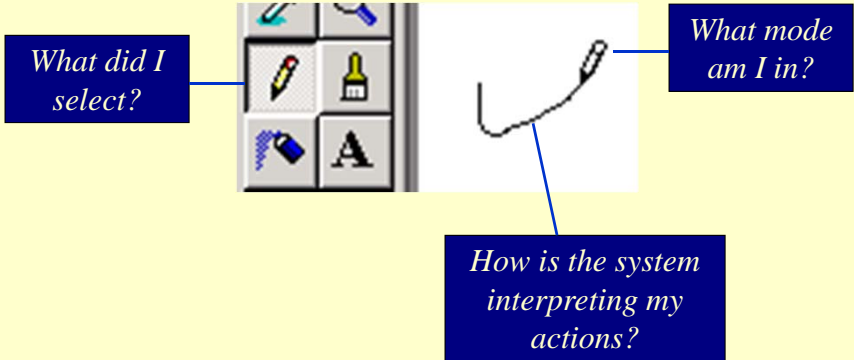
- System should continuously inform users what it is doing, how it is interpreting user actions
- Response Time
  - < 0.1 sec - seems instantaneous
  - 0.1 - 1.0 sec - noticeable, breaks “DM illusion”, but doesn’t disrupt user’s train of thought
  - > 10 sec - user’s will want to know this so they can work on other tasks
- Unpredictable amounts of time → progress bars
- Too fast can be bad → may need to animate changes



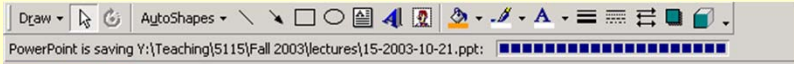
## Good Feedback



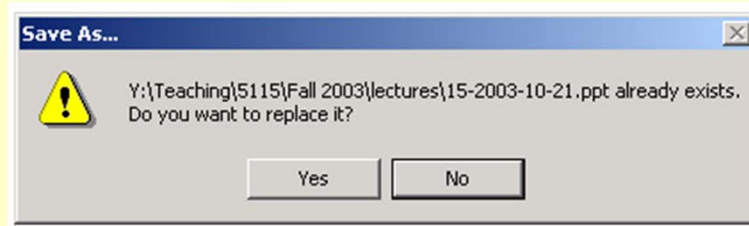
# Good (subtle) feedback



# Feedback in context of action



## *Feedback to prevent an error*

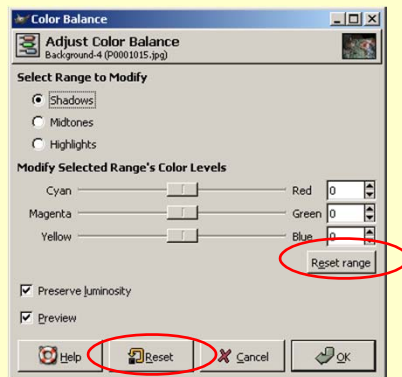


## *(Better) feedback*



## 6. Provide clearly marked exits

- Users do not like to feel trapped – if you're somewhere you don't like, you should be able to "get back" quickly and easily
- Support exploration
- Universal undo
- Let users interrupt long-running operations
- Restore defaults
- Cancel, Esc – get out of dialogs
- Quit – leave the program



## 7. Provide shortcuts

- Experienced users should be able to do frequent, familiar operations fast
- Keyboard and mouse accelerators
  - Function keys
  - Menu shortcuts
  - Command completion
  - Double-clicking to invoke default action
- Type-ahead
- Macros and scripting
- Reusable command/edit history
  - ~60% of page visits are revisits



The screenshot shows the Microsoft PowerPoint 2003 interface with several green callout boxes pointing to specific features:

- Keyboard accelerators for menus:** Points to the menu bar and the File menu.
- Customizable toolbars and palettes for frequent actions:** Points to the standard toolbar and the AutoShapes palette.
- Recently used files:** Points to the 'Recently used files' list in the File menu.
- Right-click brings up menu:** Points to the right-click context menu on the slide.
- Scrolling within a page or by whole pages:** Points to the navigation arrows at the bottom of the slide.

The slide content includes the title '7. Provide shortcuts' and a list of user experience goals and shortcuts:

- Experience users should be able to do frequent, familiar operations fast
- Keyboard and mouse accelerators
  - Function keys
  - Menu shortcuts
  - Command completion
  - Double-clicking to invoke default action
- Type-ahead
- Macros and scripting
- Reusable command/edit history
  - ~60% of page visits are revisits

Footer: GroupL logo and Konstan: Introduction to User Interfaces

## 8. *Provide good error messages*

- Speak the user's language!
- Use clear, simple, and polite language
- Be specific about the problem and offer possible solutions
- Turn it into a learning experience (encourage exploration)
  
- *But preferably...*



## 9. *Prevent Errors*

- Look at error messages – could I prevent this instead?
- Confirm risky operations
- Minimize modes



## *Remember: slips and mistakes*

- Slips
  - Capture errors
    - Frequent activity “takes over”
  - Description errors
    - Intended action is similar to other possible actions
  - Loss of activation
    - Forgot your goal in the middle of doing something
  - Mode errors
    - You think you’re in the wrong mode



Konstan: Introduction to User Interfaces

## *Preventing slips*

- Capture errors
  - Undo
  - Reverse actions (e.g., take files out of recycle bin)
- Description errors
  - Distinguish icons
  - Check for reasonable input (“Did you really want to order 5000 pencils?”)
- Loss of activation
  - Make goals and histories visible
- Mode errors
  - Minimize use of modes; make modes visible



Konstan: Introduction to User Interfaces

## *10. Help and Documentation*

- Most users do not read the manual
- When users are reading the manual, they probably are in a panic!
  - Make documentation task oriented
- Online help can be context sensitive



Konstan: Introduction to User Interfaces

## *Heuristics 2.0 – circa 1994*



UNIVERSITY OF MINNESOTA

## *Visibility of system status*

- The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.



## *Match between system and the real world*

- The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.



## *User control and freedom*

- Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.



## *Consistency and standards*

- Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.



## *Error prevention*

- Even better than good error messages is a careful design which prevents a problem from occurring in the first place.



## *Recognition rather than recall*

- Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.



## *Flexibility and efficiency of use*

- Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.



Konstan: Introduction to User Interfaces

## *Aesthetic and minimalist design*

- Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.



Konstan: Introduction to User Interfaces

## *Help users recognize, diagnose, and recover from errors*

- Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.



## *Help and documentation*

- Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.



## *Evaluation with Users*

- Big investment -- big potential
- Many issues
  - dealing with human subjects
  - which users? which tasks?
  - when in the process?
  - what to measure?
  - how to measure?



Konstan: Introduction to User Interfaces

## *Dealing with Human Subjects*

- Your responsibility to protect subjects
  - distress, embarrassment
  - remind them that you are not testing them
- Informed, voluntary consent
  - understand that they can quit at any time
  - explain test in lay terms
  - if necessary, there is "equipment failure"
- Privacy: anonymity, use of image/voice



Konstan: Introduction to User Interfaces

## *Which Users? Which Tasks*

- As close to real users as possible
  - if real users are scarce, try surrogates
- Keep close to the real tasks
  - may need to shorten some for time reasons
  - may need to provide users with background information



## *When in the Process?*

- Early is important
  - low investment
  - time to change
- Mock-ups and Drawings are OK
  - issues in how to handle user choice
- Partial prototypes when necessary
- Summary: as early as possible



## *What to Measure*

- Process data
  - problems, questions, reactions
  - what users are thinking
- Bottom-line data
  - mostly later for usability measurement
  - not very useful early in design
- Asking users questions?
  - problematic -- users will answer



## *Thinking-Aloud Method*

- User asked to think-aloud
  - ask questions (though not answered)
  - explain decisions, identify confusion
- Tester records session
  - avoids interfering as much as possible
    - only when test would end otherwise
  - explain to subject that you won't answer



## *Eye-Tracking Testing*

- Technology to support monitoring where subjects are looking and for how long
- Challenge: easy to direct results
  - avoid thinking out loud
  - careful presentation of tasks
  - careful design to avoid distractions



## *Alternative Methods*

- Natural testing conditions
  - gather performance data
  - video-prompted review
- Two-person tests
  - forces "thinking" aloud
- Field studies instead of user tests
  - consider deployment, logging



## *Gathering Field Data*

How to get it?

- Log high-level actions
- Log low-level actions
- Log problems
- Work products

What to get?

- Detailed and statistical usage data
- Example cases



## *Examples*

- Consider a Word Processor
  - many alternative solutions for commands
    - toolbars, menus, keyboard shortcuts
  - relative frequencies of commands
  - co-occurrence of commands with undo
  - document statistics



## *Examples*

- Consider a Website
  - maps of link traversal rates
    - traffic maps
  - hidden co-occurrence
    - web usage mining
  - errors
  - apparent rates of “back” from destinations
  - time on page



## *Different Goals, Different Approaches*

- Overall “what’s happening”
  - general data
  - possibly lots of data
- Test specific questions
  - targeted data
  
- Always consider issues of user consent



## *General Guidelines for User Testing*

- Plan ahead of time
  - what data to record
  - what instructions to deliver
  - what to do if user “falls off prototype”
  - when to provide help, and what help
- Know your objectives
  - but never lose sight of the user



Konstan: Introduction to User Interfaces

## *General Guidelines*

- Always have a pilot study
- Get professional help for big studies
- In general, it is better if you aren't there
  - too much bias
  - subtle clues
  - stay behind one-way glass



Konstan: Introduction to User Interfaces

## *When You Need Bottom-Line Data*

- Need specific measurements
  - median time for task
  - comparison of alternatives
- Work out the statistics involved
  - statistics cookbooks
- Focus on one type of test at a time
  - can't time and use think-aloud



## *Putting it Together*

- Critique this test plan
  - background
    - outdoor information Kiosk for 2020 Olympics in Minneapolis (we can hope!)
    - used by English-speaking Americans
    - will provide information on events and schedules, locations, transportation, results, maps, and other useful information



## *Kiosk Test Plan*

- use prototype kiosk
- recruit test users: offer \$15 for up to an hour
  - post at Twin Cities employment and social service agencies
- users shown kiosk and 20 minute demonstration video
- each user has 3 tasks and the system times them on each task
- finally, groups of 5-10 users will be asked
  - which tasks they could not accomplish
  - what problems they had with the system



Konstan: Introduction to User Interfaces

## *Specific Challenges for Recommender Systems*

- Having a coherent UI model
  - Tool vs. Agent (level of initiative, predictability)
- Making recommender natural, or at least understandable
- User sees entire application, not just recommender algorithm



Konstan: Introduction to User Interfaces