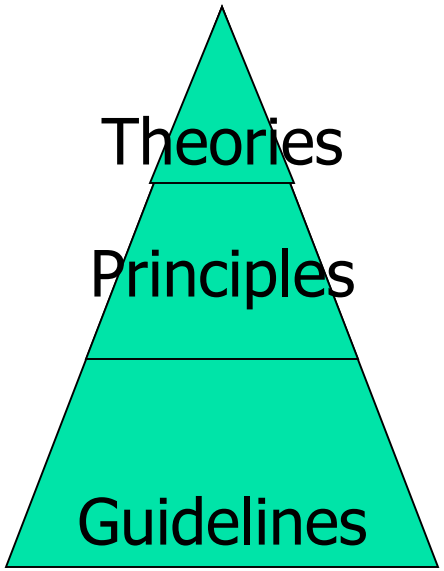
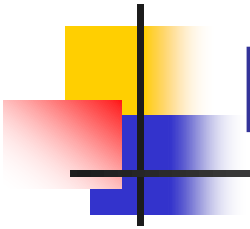
Guidances for UI Designers

- High-level theories
- Middle-level principles
- Specific and practical guidelines

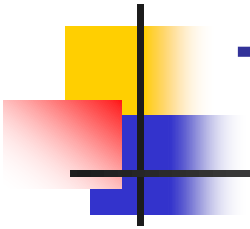




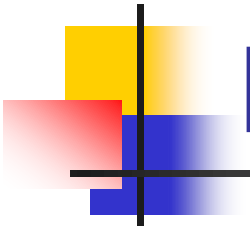
High-Level Theories

- Descriptive and explanatory theories
  - Help develop consistent terminology, actions, and objects
- Predictive theories
  - Useful in predicting the behavior of users

Descriptive and Explanatory Theories



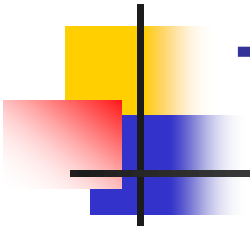
- Taxonomy
  - A classification scheme to put order on a complex set of phenomena.
- Examples
  - User experience levels: novice, knowledgeable, expert.
  - User interface styles: menus, form fillin, commands.



Predictive Theories

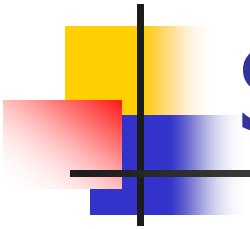
- Motor-task predictions
  - Fitts' Law for predicting keystrokes or pointing times.
- Perceptual or cognitive subtask prediction
  - Predict the time to find an item on a display.
  - Predict reading times for free text, lists and formatted displays.

Examples of High-level Theories



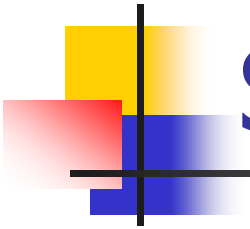
- Conceptual, semantic, syntactic, and lexical model.
- GOMS and the keystroke-level model.
- Stages of action models.
- Consistency through grammars.
- Widget-level theories.
- Object-Action Interface Model (OAI)

Conceptual, Semantic, Syntactic, and Lexical Model

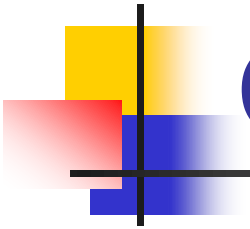


- Conceptual level
  - User's mental model of the system.
- Semantic level
  - Meanings conveyed by user input and computer output .
- Syntactic level
  - The way of assembling the units into a complete command.
- Lexical level
  - The precise mechanisms by which a user specifies the syntax.

Conceptual, Semantic, Syntactic, and Lexical Model

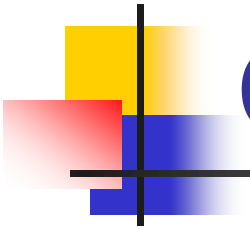


- Conceptual level
  - In UNIX, users have their own concepts of what they can do with the system: e.g., copying, listing, reading and writing files.
- Semantic level
  - ls -l means listing all the files in detailed format.
- Syntactic level
  - ls -l vs ls l-
- Lexical level
  - The detailed sequence of actions for command entry.



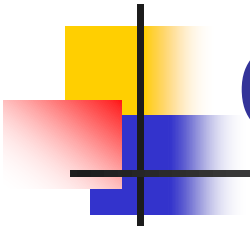
GOMS

- GOMS= goals, operators, methods and selection rules.
  - Users formulate goals.
  - Use methods to achieve goals.
  - Use elementary perceptual, cognitive or motor acts (operators) to perform methods.
  - Choose among several methods to achieve the goal (selection rules).



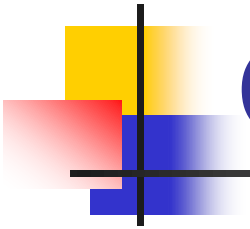
GOMS

- Goals
  - A particular state the user wants to achieve.
- Operators
  - The cognitive processes and physical actions that need to be performed to attain these goals.



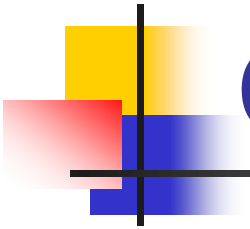
GOMS

- Methods
  - Learned procedures for accomplishing the goals.
- Selection rules
  - Used to determine which method to select when there is more than one available.



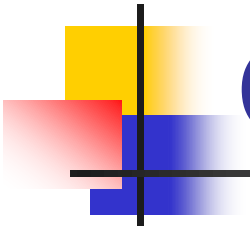
GOMS Example

- Goal
  - Delete a word in a sentence
- Two possible methods
  - Deleting a word using menu option
  - Deleting a word using delete key



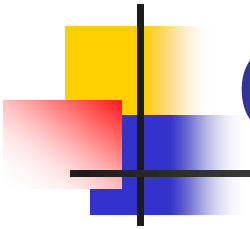
GOMS Example

- Using menu option
  1. Recall that word to be deleted has to be highlighted.
  2. Recall that the command is “cut”.
  3. Recall that “cut” is in the edit menu.
  4. Select and execute the “cut” command.
  5. Return with goal accomplished.



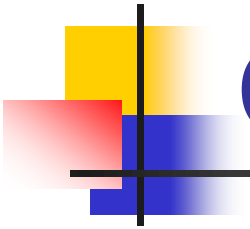
GOMS Example

- Operators used (menu option)
  - Click mouse
  - Drag cursor over text
  - Select menu
  - Move cursor to command



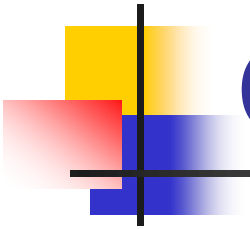
GOMS Example

- Using delete key
  1. Recall where to position cursor in relation to word to be deleted.
  2. Recall which key is delete key.
  3. Press “delete” key to delete each letter.
  4. Return with goal accomplished.



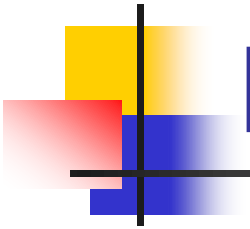
GOMS Example

- Operators used (delete key option)
  - Click mouse
  - Press keyboard key



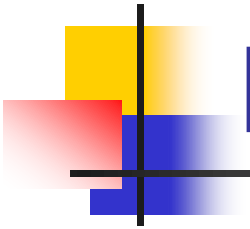
GOMS Example

- Selection rules
  - If large amount of text is to be deleted, use the menu option.
  - If small amount of text is to be deleted, use the delete key option.



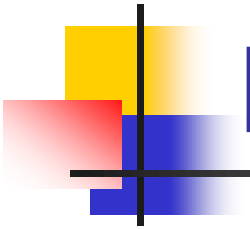
Keystroke-level Model

- Keystroke-level model predict users' performance by summing up the time for
  - Thinking
  - Keystroking, pointing, homing, drawing
  - Waiting for the system to respond.
- Concentrate on expert users and error-free performance.



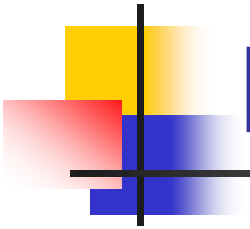
Main Operators

- Operator K
  - Pressing a single key or button
    - 0.35s
  - Skilled typist (55wpm)
    - 0.22s
  - Average typist (40wpm)
    - 0.28s
  - User unfamiliar with the keyboard
    - 1.20s
  - Pressing shift or control key
    - 0.08s



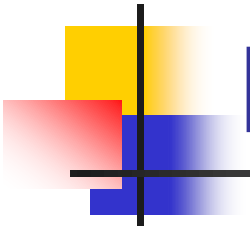
Main Operators

- Operator P
  - Pointing with a mouse or other device to a target on a display
    - 1.10s
- Operator P_1
  - Clicking the mouse or similar device
    - 0.20s



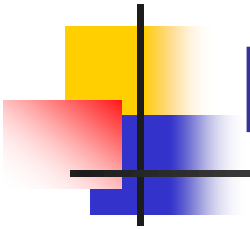
Main Operators

- Operator H
  - Homing hands on the keyboard or other device
    - 0.40s
- Operator D
  - Draw a line using a mouse
    - Variable depending on the line length.



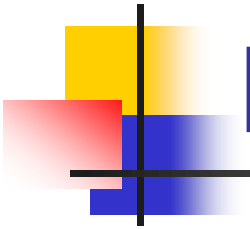
Main Operators

- Operator M
  - Mentally prepare to do something, e.g., make a decision
    - 1.35s
- System response time R
  - Variable depending on system.



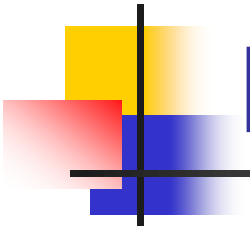
Example

- The predicted time for a given task is predicted as follows:
  - $T_{execute}=T_K+T_P+T_H+T_D+T_M+T_R$
- Example task: word insertion
  - The story is very interesting.
  - The story is not very interesting.



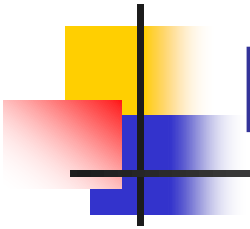
Example

- Mentally Prepare (M) 1.35s
- Reach for the mouse (H) 0.40s
- Position mouse before “very” (P) 1.10s
- Click mouse (P_1) 0.20s
- Move hands to home position on keys (H) 0.40s
- Mentally prepare (M) 1.35s



Example

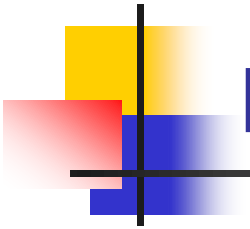
- Type “n” (good typist)(K) 0.22s
- Type “o” (K) 0.22s
- Type “t” (K) 0.22s
- Type “space” (K) 0.22s



Example

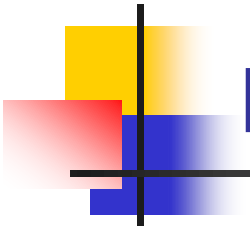
- Total predicted time:
  - $2(M)+2(H)+1(P)+1(P_1)+4(K)=5.68s$
- Issues to consider
  - When to insert mental preparation time.
  - How to model the variation of mental preparation between individuals.

GOMS and keystroke level model

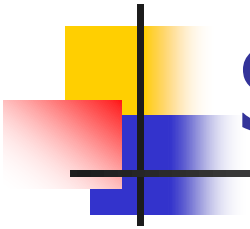


- Benefits
  - Enable the prediction of user performance without the need for actual user participation.
  - Allows comparative analyses to be performed for different interfaces or computer systems.

GOMS and keystroke level model



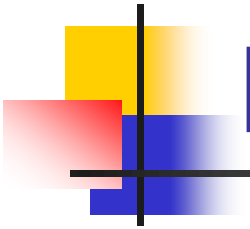
- Limitations
  - Modeling is limited to a small set of highly routine data-entry type tasks.
  - Can be used only to predict expert performance, and does not allow for errors to be modeled.



Stages of Action Models

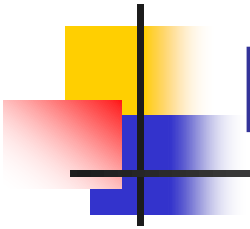
- Forming the goal
- Forming the intention
- Specifying the action
- Executing the action
- Perceiving the system state
- Interpreting the system state
- Evaluating the outcome

Stages of Action Models: Example



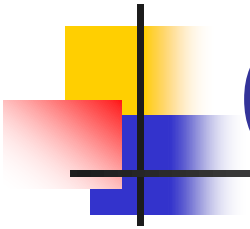
- Forming the goal
  - Increase the amount of free space on the hard disk
- Forming the intention
  - Delete some files
- Specifying the action
  - Select a set of files, and press the delete key.
- Executing the action

Stages of Action Models: Example

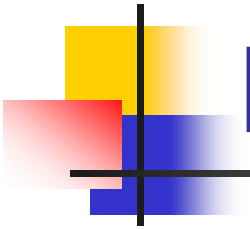


- Perceiving the system state
  - Some file icons have disappeared
- Interpreting the system state
  - There are less files and the amount of free space has increased
- Evaluating the outcome
  - Is the amount of free space enough ?

Stages of Action Models (cont'd)



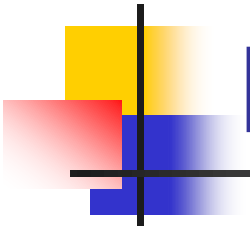
- Main contribution: cycles of action and evaluation.
- Leads naturally to identifying
  - Gulf of execution: mismatch between users' intentions and allowable actions.
    - e.g. when a user needs to import a graphics file but this file format is not supported.
  - Gulf of evaluation: mismatch between users' expectation and system's representation.
    - e.g. when a user tries to print a document and the printout appears different from that on the screen.



Four principles

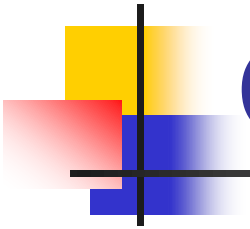
- Four principles for good design
  - state and action alternatives should be visible
  - there should be a good conceptual model with a consistent system image
  - the interface should include good mappings that reveal the relationships between stages
  - the user should receive continuous feedback
- Errors often occur when moving from goals/intentions to actions/executions.

Possible Sources of User Failures



- Users form inadequate goal.
- Users cannot find the correct interface object.
- Users do not know how to specify or execute a desired action.
- Users receive inappropriate or misleading feedback.

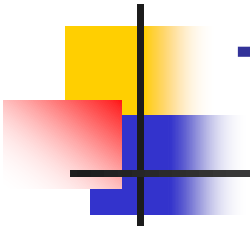
Consistency Through Grammars



- Useful for designing consistent user interface which is
  - Orderly
  - Predictable
  - Describable by a few rules
  - Easy to learn and retain
- Requires the performance of task-action mapping

Consistencies-Inconsistencies

Consistent +	Inconsistent A	Inconsistent B
delete/insert character	delete/insert character	delete/insert character
delete/insert word	remove/bring word	remove/insert word
delete/insert line	destroy/create line	delete/insert line
delete/insert paragraph	kill/birth paragraph	delete/insert paragraph



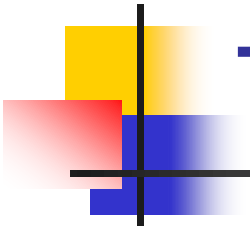
Task-Action Mapping

- Tasks to be performed
  - Move cursor one character forward [Direction=forward, Unit=char]
  - Move cursor one character backward [Direction=backward, Unit=char]
  - Move cursor one word forward [Direction=forward, Unit=word]
  - Move cursor one word backward [Direction=backward, Unit=word]

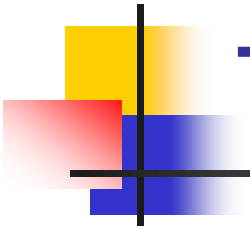
Task-Action Mapping (cont'd)

- Sets of symbols
  - {"CTRL","ESC"}
- Sets of letters
  - {"W","C"}
- Objective: to associate a specific task with a unique sequence of symbols and letters.

Task-Action Mapping (cont'd)



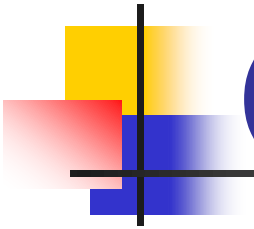
- task→symbol+letter
- forward→"CTRL"
- backward→"ESC"
- word→"W"
- char→"C"



Task-Action Mapping (cont'd)

- Move cursor one character forward
  - CTRL-C
- Move cursor one character backward
  - ?
- Move cursor one word forward
  - ?
- Move cursor one word backward
  - ?

Object-Action Interface Model (OAI)



- Icons as metaphoric representations of real-world objects.
- A set of actions is defined on the icons to represent real-world actions.
- Mapping between task and interface
  - Task object→Interface object
  - Task action→Interface action

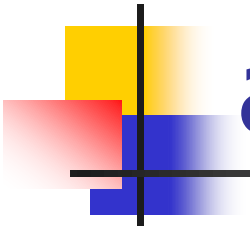
Task Hierarchies of Objects and Actions

- Task objects can be described at different levels
  - High level objects

  - Intermediate-level components

  - Atomic units

Task Hierarchies of Objects and Actions (cont'd)

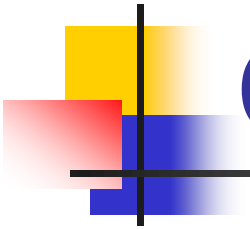


- Task actions can also be characterized at different levels.
  - High-level goal.
  - Intermediate sub-goals.
  - Individual steps.

Interface Hierarchies of Objects and Actions

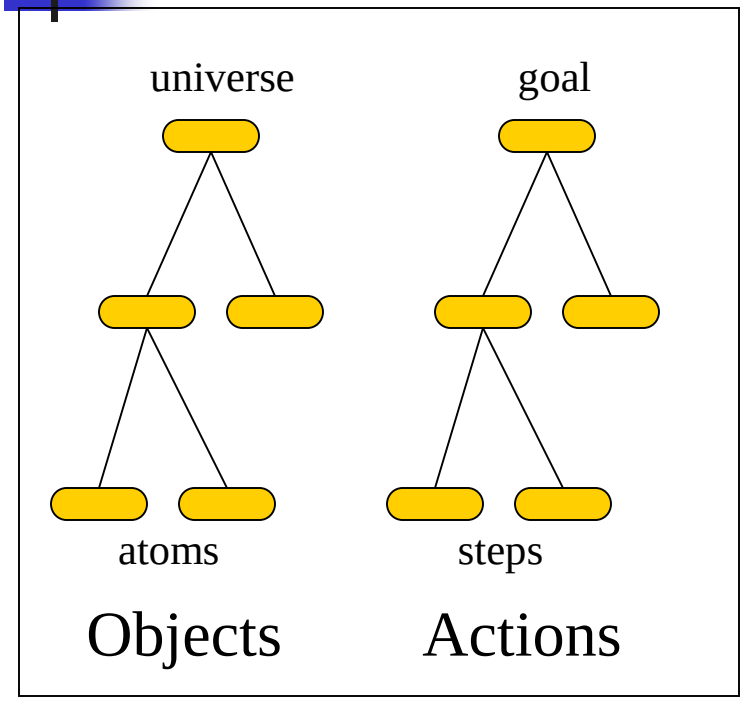
- Interface objects can be described at different levels
  - Icons representing high level objects 
  - Icons representing intermediate-level components. 
  - Pixels within a single icon

Interface Hierarchies of Objects and Actions (cont'd)

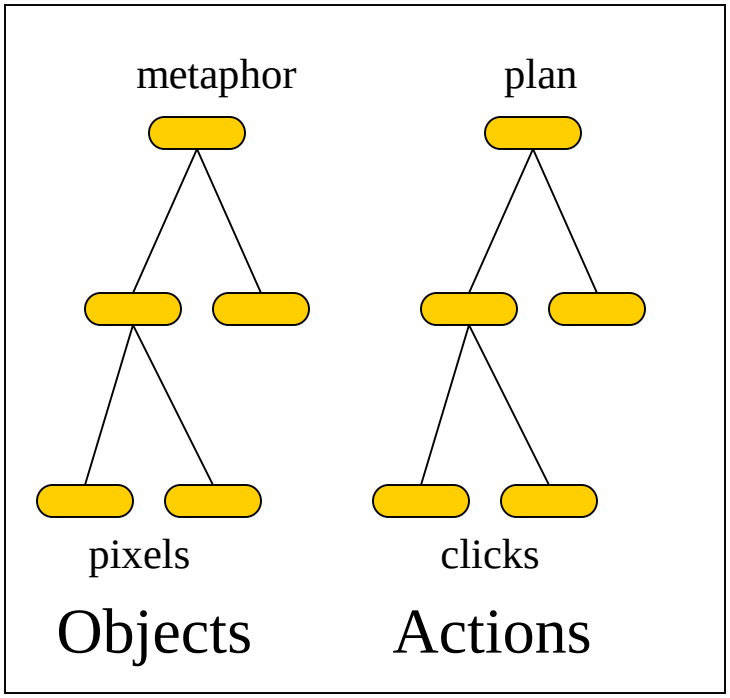


- Interface actions can be described at different levels
  - Overall plan
  - Intermediate actions
  - Detailed keystrokes and clicks

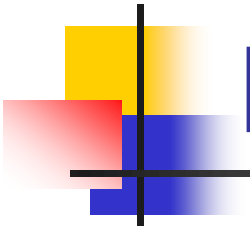
Relationships between the two hierarchies



Task

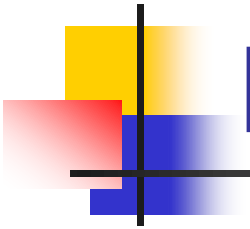


Interface



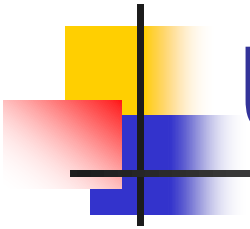
Principles of Design

- Recognize the diversity
- Apply the 8 golden rules of interface design.



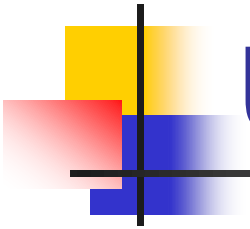
Recognize Diversity

- Great diversity in
  - Usage profiles
  - Task profiles
  - Interaction styles



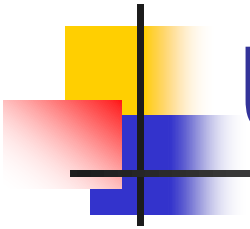
Usage Profiles

- Novice or first-time users
  - Restrict vocabulary to a small number of familiar terms.
  - Restrict the set of possible actions.
  - Informative feedback and constructive, specific error messages.
  - Carefully designed paper manuals and online tutorials.



Usage Profiles (cont'd)

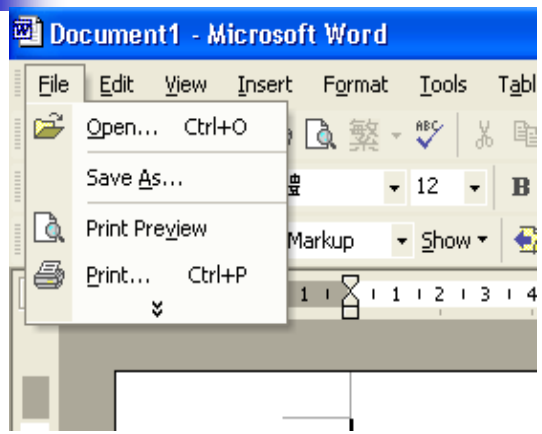
- Knowledgeable intermittent users
  - Emphasizes recognition rather than recall
  - Orderly menu structure
  - Meaningful messages
  - Consistent sequence of actions
  - Online help screens



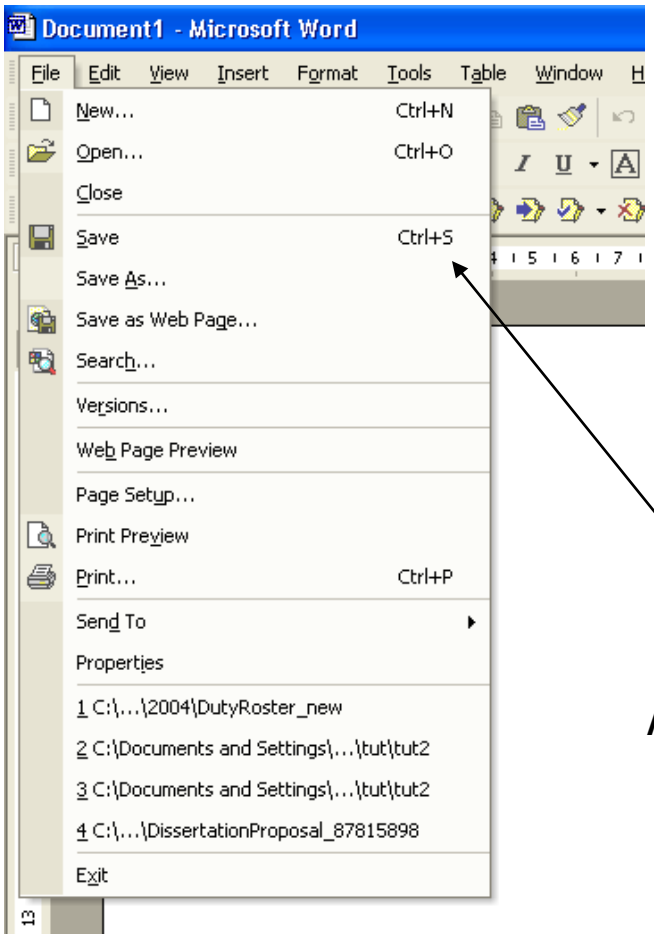
Usage Profiles (cont'd)

- Expert frequent users
  - Rapid response times
  - Brief and non-distracting feedback
  - Capacity to carry out actions with a few keystrokes/selection.
  - Create macro to reduce the number of steps.

Usage Profiles (cont'd)

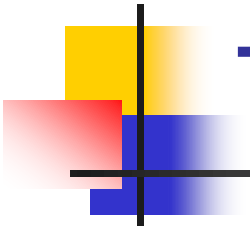


Novice users



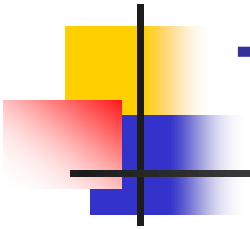
Advanced users

Knowledgeable intermittent users



Task Profiles

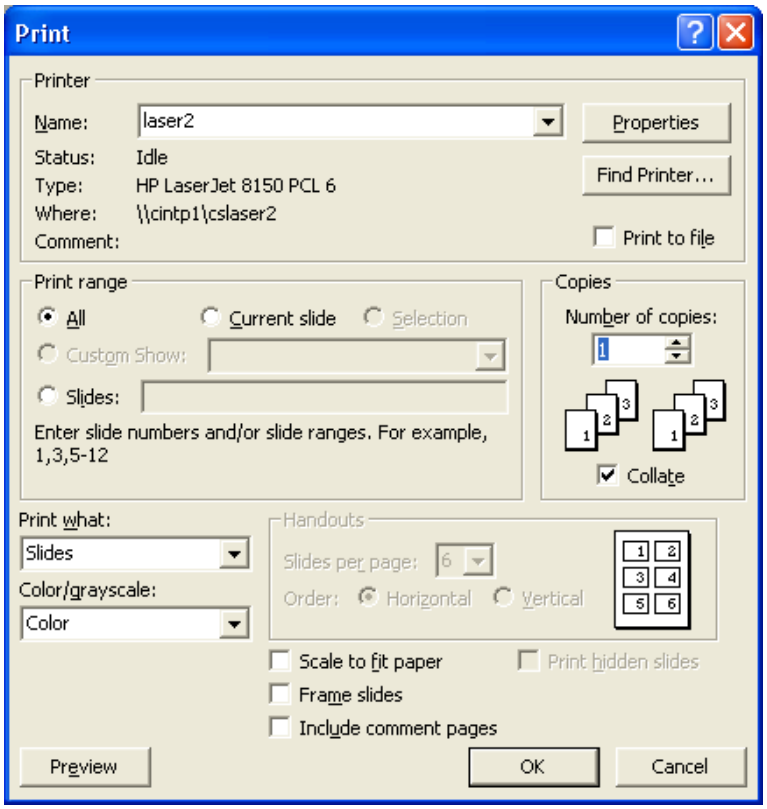
- Carry out task analysis
  - Determine a set of tasks before design.
  - System functionality should not be dictated by design or implementation convenience.
- Choose the appropriate set of atomic actions
  - Atomic actions too small $\rightarrow$ large number of actions required.
  - Atomic actions too large $\rightarrow$ less flexibility for users.



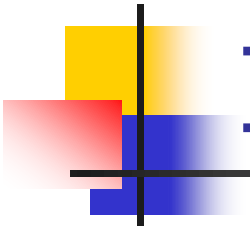
Task Profiles (cont'd)

- Design based on task frequencies.
  - Frequent actions to be performed by single special keys.
  - Intermediately frequent actions to be performed by a small number of keys/menu selection.
  - Infrequent actions to be performed by a sequence of menu selections/form fillin.

Task Profiles (cont'd)



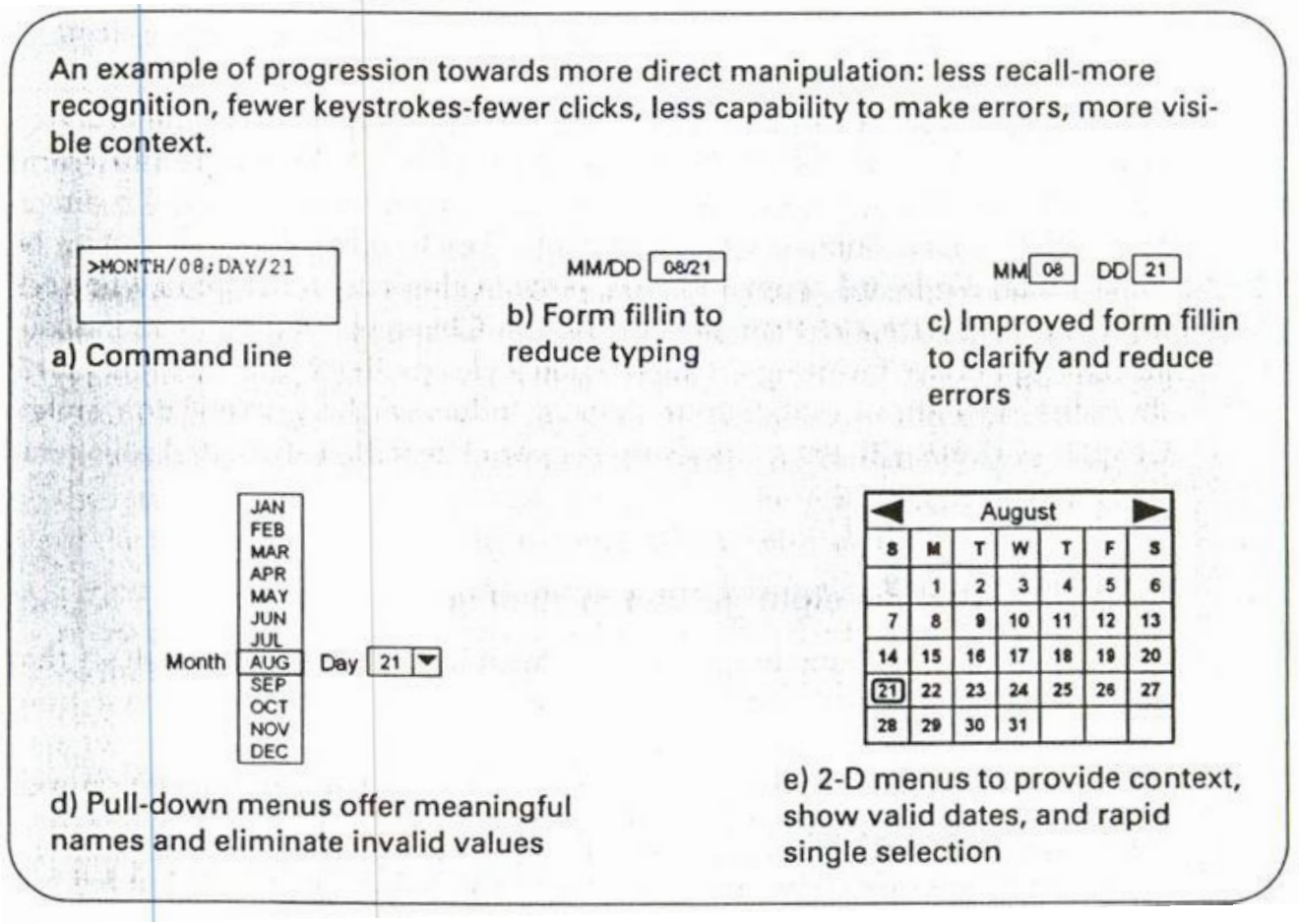
Different ways of printing in Powerpoint



Interaction Styles

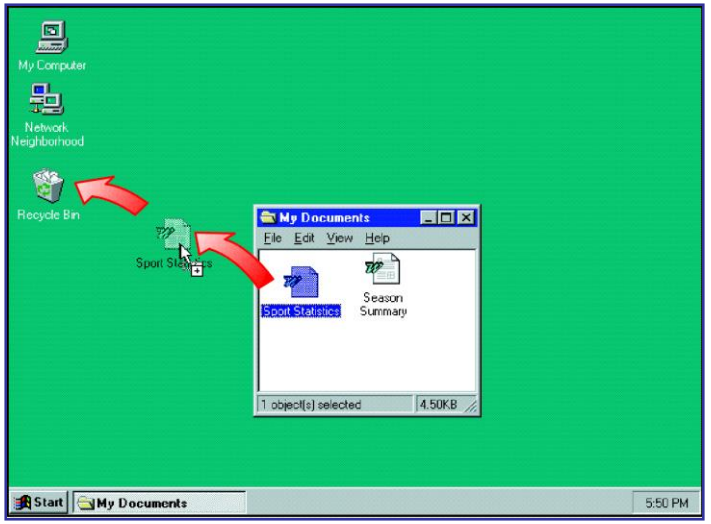
- Direct manipulation
- Menu selection
- Form fillin
- Command language
- Natural language

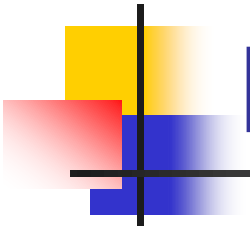
Spectrum of directness



Direct Manipulation

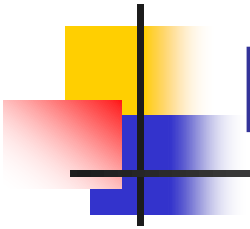
- Create a visual representation of the world of objects and actions.
- Examples include
  - Windows desktop
  - Video games
  - Drawing tools





Direct Manipulation (cont'd)

- Advantages
  - Easy learning and retention.
  - Reduce errors
  - Encourages exploration
  - Affords high subjective satisfaction

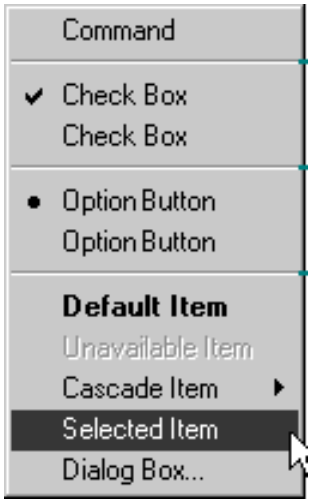


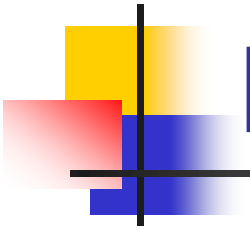
Direct Manipulation (cont'd)

- Disadvantages
  - Hard to program
  - Not suitable for visually impaired users

Menu Selection

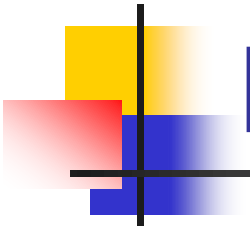
- Present a list of items to users.
- Users select the one most appropriate to their task.





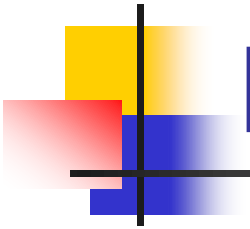
Menu Selection (cont'd)

- Advantages
  - Requires little learning or memorization
  - Clear structure to decision making
  - Reduces the number of keystrokes
  - Reduces the possibility of errors



Menu Selection (cont'd)

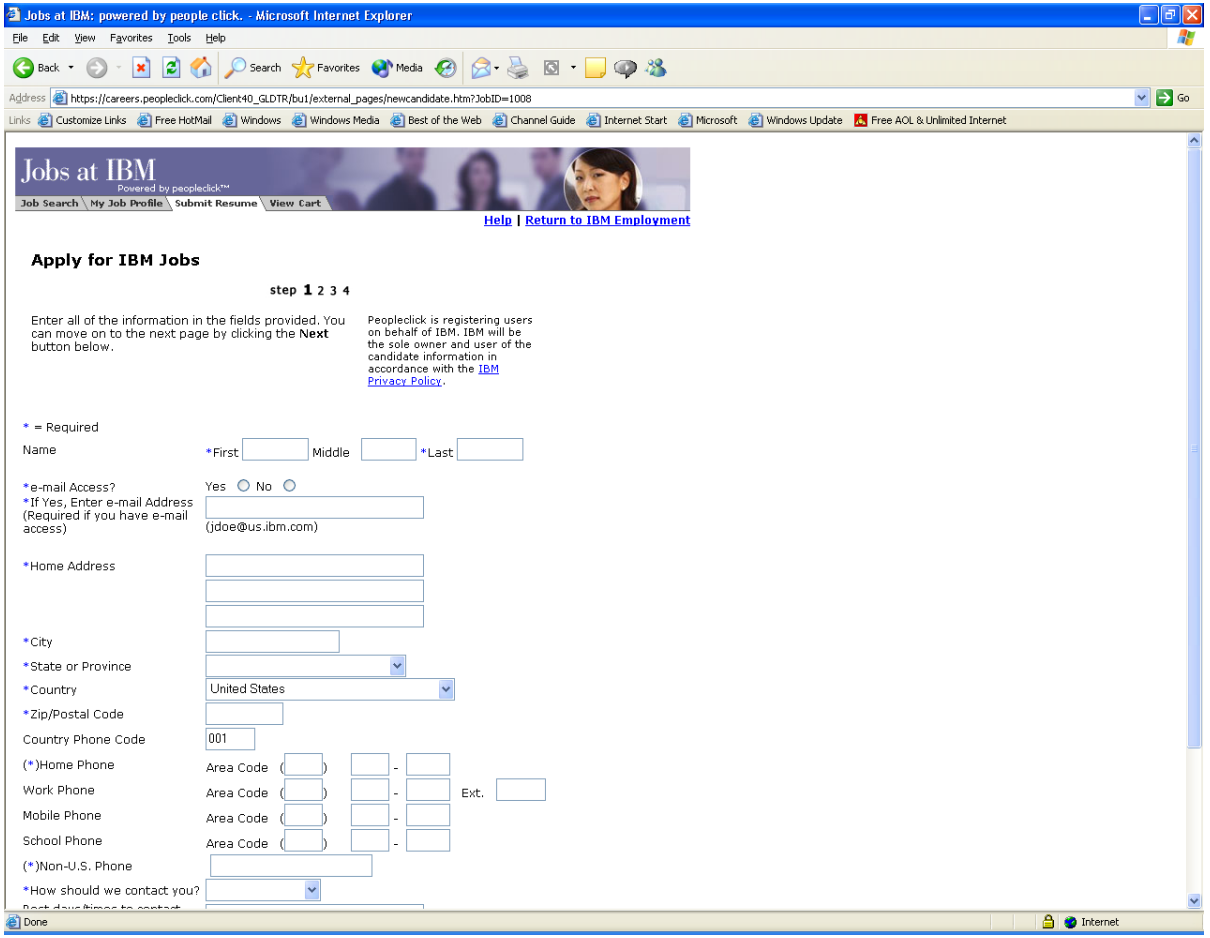
- Disadvantages
  - Frequent users may find it slow
  - Consumes screen space
  - Too many items on menus may distract users

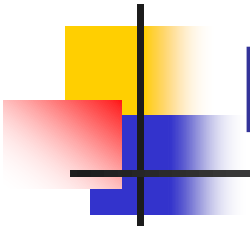


Form Fillin

- Users see a display of related fields.
- Move a cursor among the fields.
- Enter data at the desired field.

Form Fillin (cont'd)



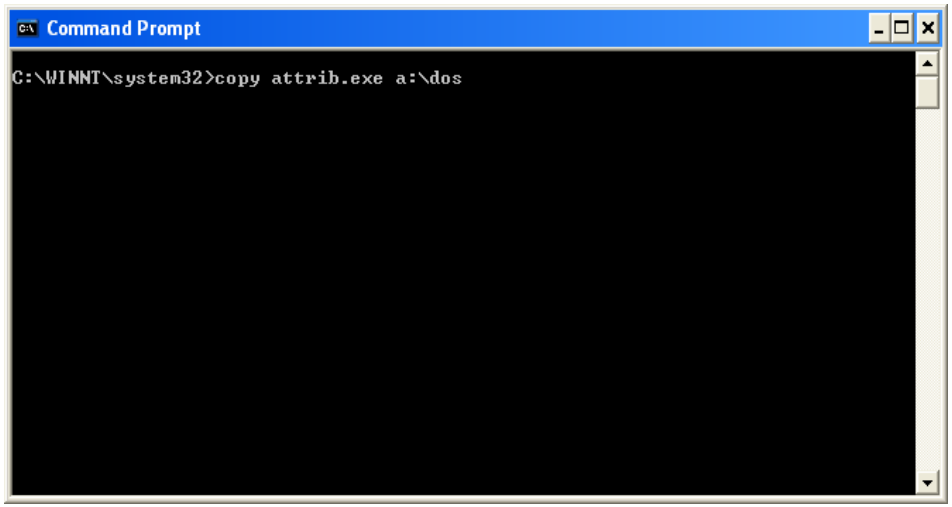


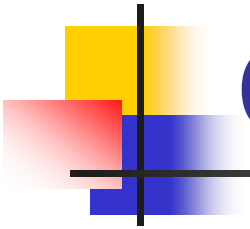
Form Fillin (cont'd)

- Advantages
  - Simplifies data entry
  - Requires only modest training
  - Give convenient assistance
- Disadvantages
  - Consumes screen space

Command Language

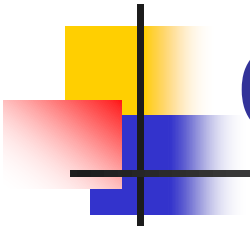
- Action is expressed in terms of a sequence of typed commands.
- Users learn the syntax to express their requirements to the system.





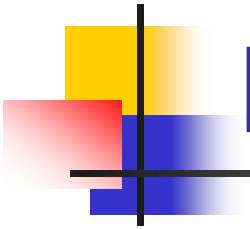
Command Language (cont'd)

- Advantages
  - More flexible
  - Expert users can rapidly express their requirements without distracting prompts.
  - Allows convenient specification of user-defined macros.
  - Expert users derives great satisfaction from mastering the complex syntax.



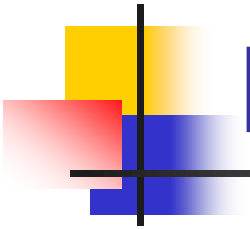
Command Language (cont'd)

- Disadvantages
  - High error rates
  - Long period of training is necessary.
  - Poor retention
  - Error messages and online assistance are hard to provide.



Natural Language

- Computers respond directly to spoken sentences and phrases.
- Belongs to the class of recognition-based interfaces.



Natural Language (cont'd)

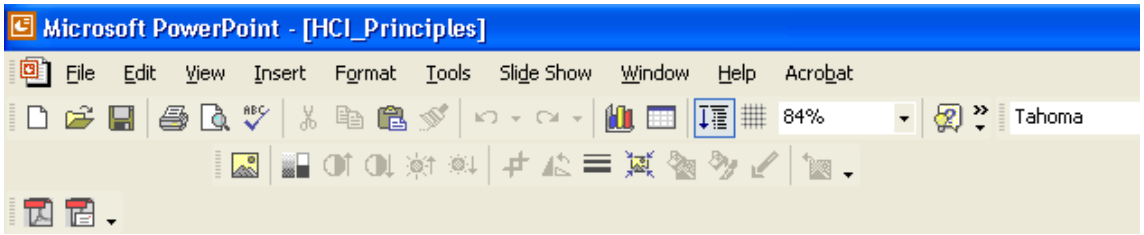
- Advantages
  - Requires little learning.
  - Suitable for mobile devices where keyboard and mouse input is not convenient.
  - Allows the mobility impaired users to have access to computers.

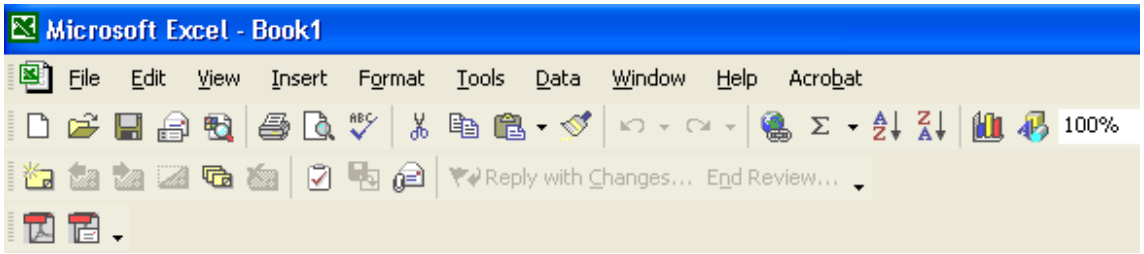
Natural Language (cont'd)

- Disadvantages
  - Speech recognition software requires a period of training to recognize a user's voice.
  - Recognition error
  - Requires clarification dialog.

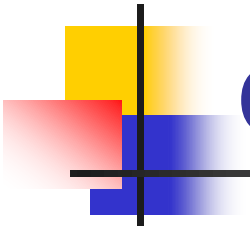
Shneiderman's 8 Golden Rules of Interface Design

1. Aim for consistency
   - Similar command sequences for similar actions.
   - Consistent terminology and layout.





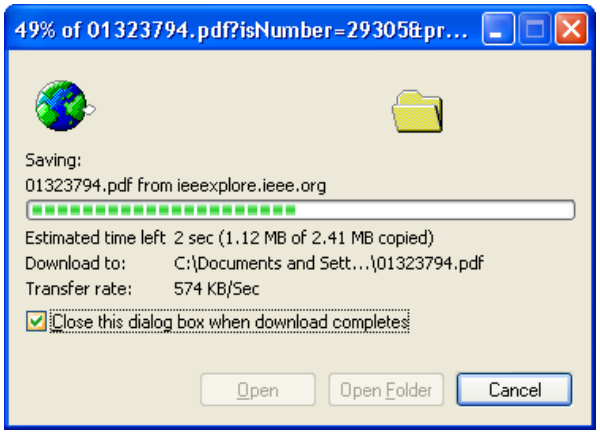
Shneiderman's 8 Golden Rules of Interface Design



2. Cater to universal usability
   - Recognize the needs of diverse users.
   - Design for plasticity.
   - Add features for novices and experts.

The 8 Golden Rules of Interface Design (cont'd)

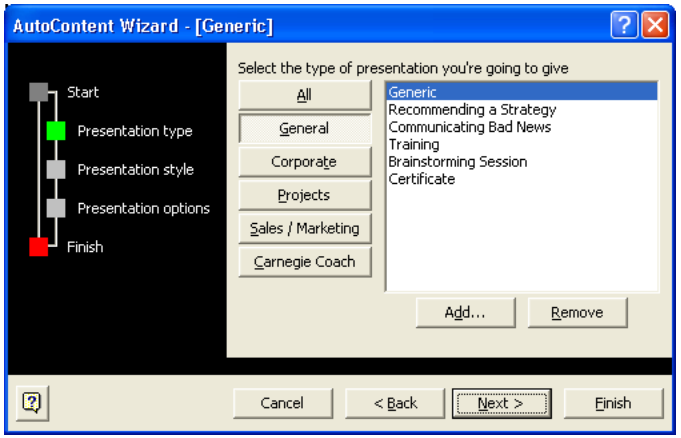
3. Offer informative feedback
   - System feedback for every user action.
   - Modest feedback for minor actions and substantial feedback for major actions.



Feedback during the file download process

The 8 Golden Rules of Interface Design (cont'd)

4. Organize actions into groups
   - Organize sequences of actions into groups with a beginning, middle and end.
   - Completion of each action group gives users the satisfaction of accomplishment.



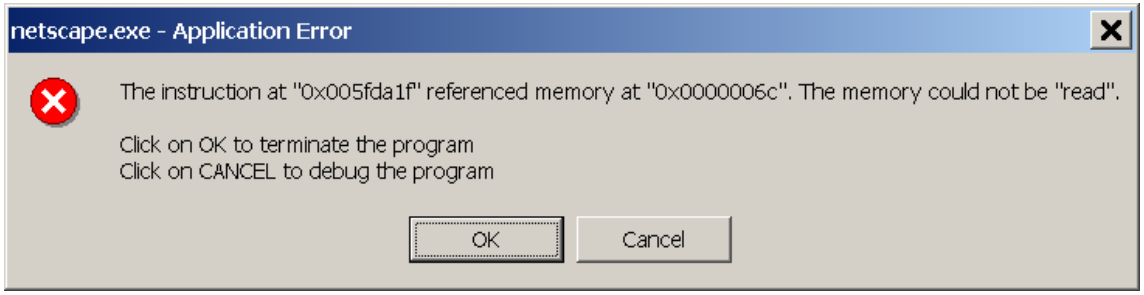
AutoContent Wizard in Powerpoint

The 8 Golden Rules of Interface Design (cont'd)

5. Prevent errors
   - Design the system to prevent serious errors.
     - Gray out menu items that are not appropriate.
     - Do not allow alphabetic characters in numeric entry fields.
     - Provide automatic command completion.
     - Apply direct manipulation strategies.
   - Offer simple, constructive and specific instructions for recovery.

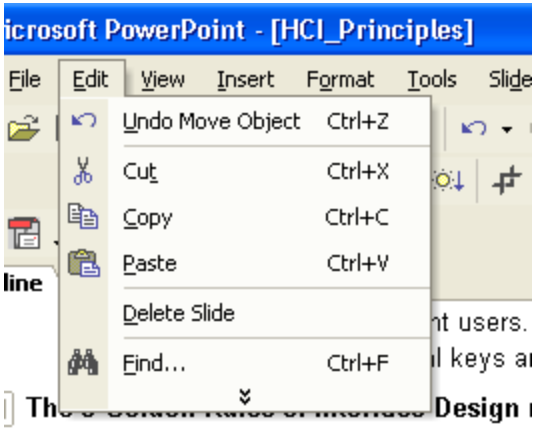
The 8 Golden Rules of Interface Design (cont'd)

- How will you respond when you encounter this error message ?



The 8 Golden Rules of Interface Design (cont'd)

6. Permit easy reversal of actions
   - Errors can be undone.
   - Encourage exploration of unfamiliar options.



The 8 Golden Rules of Interface Design (cont'd)

7. Allow users to be in control
   - Avoid surprising system actions.
   - Make users the initiators rather than responders to actions.

The 8 Golden Rules of Interface Design (cont'd)

8. Reduce short-term memory load
   - Limitations of human short-term memory require simple displays.
   - Provide online help to users.