

# SE 705 Human Computer Interaction

Spring 2008-2009

Week #5

user interface design  
paradigms

# why study paradigms

## Concerns

- how can an interactive system be developed to ensure its usability?
- how can the usability of an interactive system be demonstrated or measured?

History of interactive system design  
provides paradigms for usable designs

# What are Paradigms

- Predominant theoretical frameworks or scientific world views
  - e.g., Aristotelian, Newtonian, Einsteinian (relativistic) paradigms in physics
- Understanding HCI history is largely about understanding a series of paradigm shifts
  - Not all listed here are necessarily “paradigm” shifts, but are at least candidates
  - History will judge which are true shifts

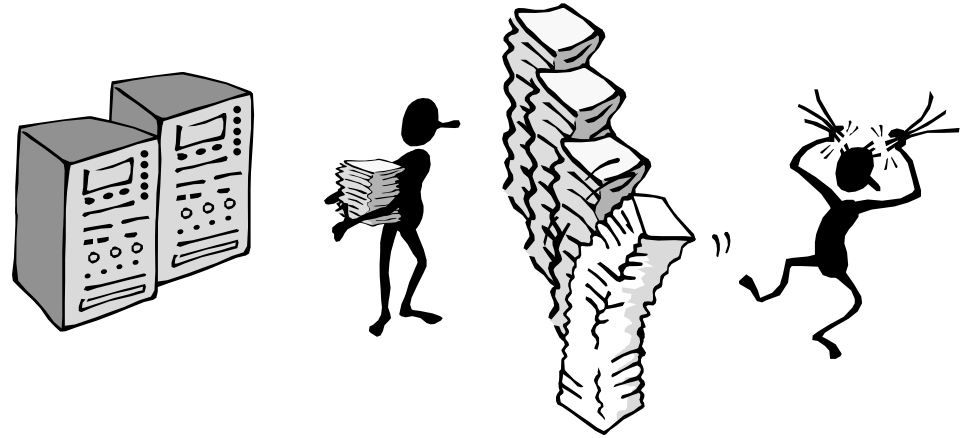
# Paradigms of interaction

New computing technologies arrive, creating a new perception of the human—computer relationship.

We can trace some of these shifts in the history of interactive technologies.

# The initial paradigm

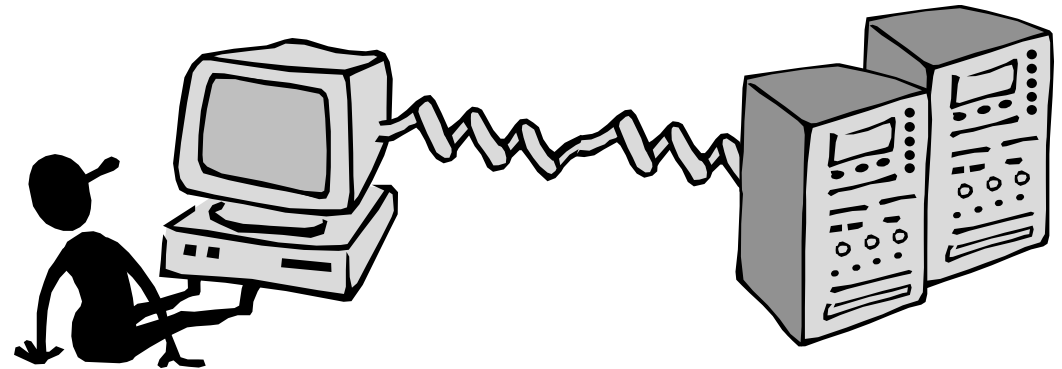
- Batch processing



*Impersonal computing*

# Example Paradigm Shifts

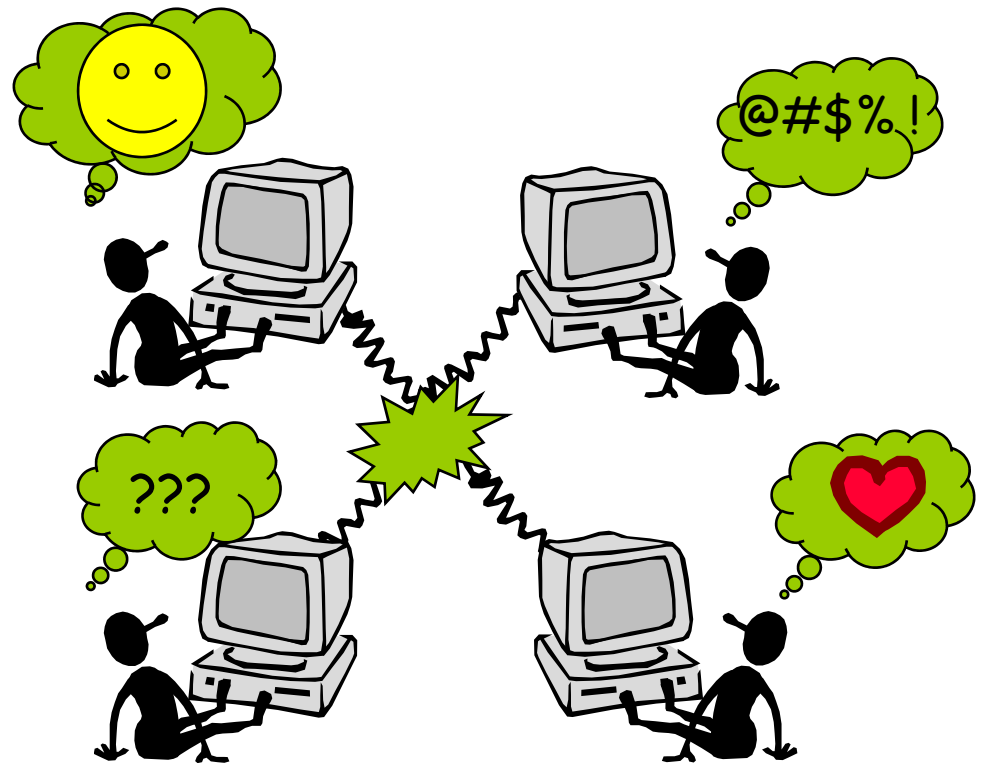
- Batch processing
- Time-sharing



*Interactive computing*

# Example Paradigm Shifts

- Batch processing
- Timesharing
- **Networking**

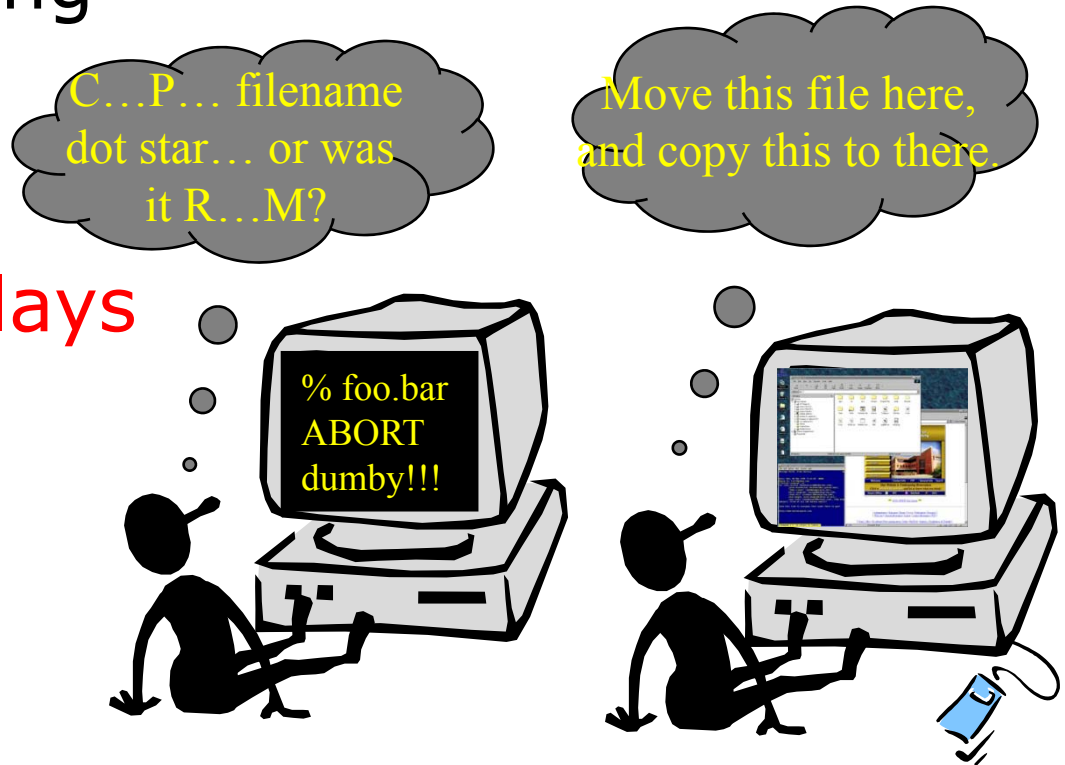


*Community computing*



# Example Paradigm Shifts

- Batch processing
- Timesharing
- Networking
- **Graphical displays**



*Direct manipulation*

# Example Paradigm Shifts

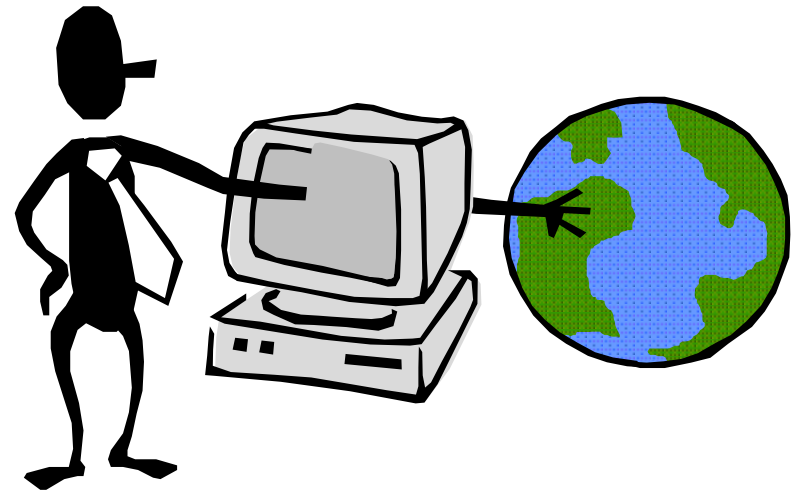
- Batch processing
- Timesharing
- Networking
- Graphical display



*Personal computing*

# Example Paradigm Shifts

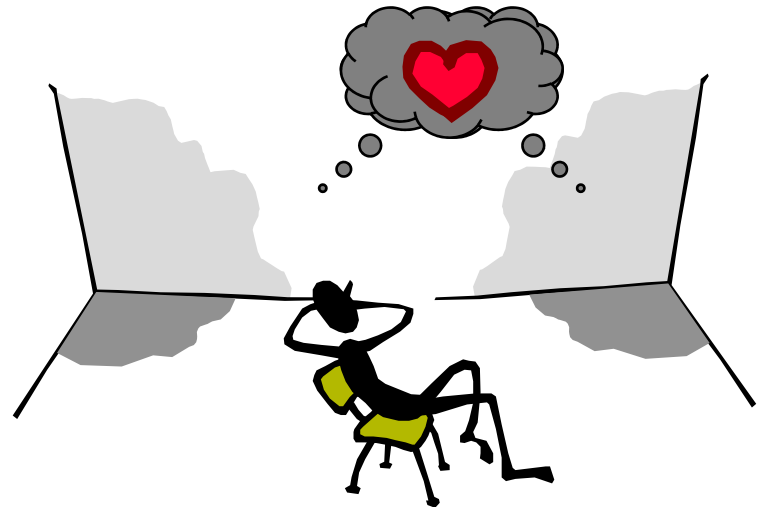
- Batch processing
- Timesharing
- Networking
- Graphical display
- WWW



*Global information*

# Example Paradigm Shifts

- Batch processing
  - Timesharing
  - Networking
  - Graphical display
  - WWW
  - Ubiquitous Computing
- A symbiosis of physical and electronic worlds in service of everyday activities.



# Time-sharing

- 1940s and 1950s – explosive technological growth
- 1960s – need to channel the power
- J.C.R. Licklider at ARPA
- single computer supporting multiple users

# Video Display Units

- more suitable medium than paper
- 1962 – Sutherland's Sketchpad
- computers for visualizing and manipulating data
- one person's contribution could drastically change the history of computing

# Programming toolkits

- Engelbart at Stanford Research Institute
- 1963 – augmenting man's intellect
- 1968 NLS/Augment system demonstration
- the right programming toolkit provides building blocks to producing complex interactive systems

# Personal computing

- 1970s – Papert's LOGO language for simple graphics programming by children
- A system is more powerful as it becomes easier to user
- Future of computing in small, powerful machines dedicated to the individual
- Kay at Xerox PARC – the Dynabook as the ultimate personal computer



# Window systems and the WIMP interface

- humans can pursue more than one task at a time
- windows used for dialogue partitioning, to “change the topic”
- 1981 – Xerox Star first commercial windowing system
- windows, icons, menus and pointers now familiar interaction mechanisms

# Metaphor

- relating computing to other real-world activity is effective teaching technique
  - LOGO's turtle dragging its tail
  - file management on an office desktop
  - word processing as typing
  - financial analysis on spreadsheets
  - virtual reality – user inside the metaphor
- Problems
  - some tasks do not fit into a given metaphor
  - cultural bias

# Direct manipulation

- 1982 – Shneiderman describes appeal of graphically-based interaction
  - visibility of objects
  - incremental action and rapid feedback
  - reversibility encourages exploration
  - syntactic correctness of all actions
  - replace language with action
- 1984 – Apple Macintosh
- the model-world metaphor
- What You See Is What You Get (WYSIWYG)

# Language versus Action

- actions do not always speak louder than words!
- DM – interface replaces underlying system
- language paradigm
- interface as mediator
- interface acts as intelligent agent
- programming by example is both action and language

# Hypertext

- 1945 – Vannevar Bush and the memex
- key to success in managing explosion of information
- mid 1960s – Nelson describes hypertext as non-linear browsing structure
- hypermedia and multimedia
- Nelson's Xanadu project still a dream today

# Multimodality

- a mode is a human communication channel
- emphasis on simultaneous use of multiple channels for input and output

# Computer Supported Cooperative Work (CSCW)

- CSCW removes bias of single user / single computer system
- Can no longer neglect the social aspects
- Electronic mail is most prominent success

# The World Wide Web

- Hypertext, as originally realized, was a closed system
- Simple, universal protocols (e.g. HTTP) and mark-up languages (e.g. HTML) made publishing and accessing easy
- Critical mass of users lead to a complete transformation of our information economy.



# Agent-based Interfaces

- Original interfaces
  - Commands given to computer
  - Language-based
- Direct Manipulation/WIMP
  - Commands performed on “world” representation
  - Action based
- Agents - return to language by instilling proactivity and “intelligence” in command processor
  - Avatars, natural language processing

# Ubiquitous Computing

*"The most profound technologies are those that disappear."*

Mark Weiser, 1991

Late 1980's: computer was very apparent

How to make it disappear?

- Shrink and embed/distribute it in the physical world
- Design interactions that don't demand our intention

# Sensor-based and Context-aware Interaction

- Humans are good at recognizing the “context” of a situation and reacting appropriately
- Automatically sensing physical phenomena (e.g., light, temp, location, identity) becoming easier
- How can we go from sensed physical measures to interactions that behave as if made “aware” of the surroundings?

design rules

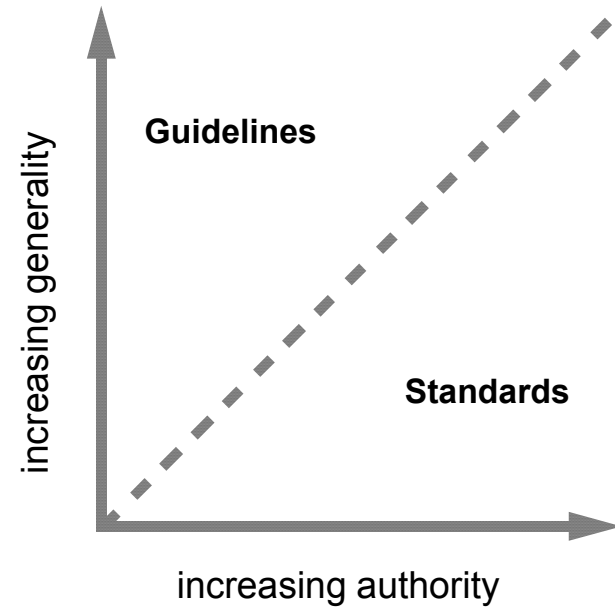
# design rules

Designing for maximum usability  
– the goal of interaction design

- Principles of usability
  - general understanding
- Standards and guidelines
  - direction for design
- Design patterns
  - capture and reuse design knowledge

# types of design rules

- principles
  - abstract design rules
  - low authority
  - high generality
- standards
  - specific design rules
  - high authority
  - limited application
- guidelines
  - lower authority
  - more general application



# Principles to support usability

## Learnability

the ease with which new users can begin effective interaction and achieve maximal performance

## Flexibility

the multiplicity of ways the user and system exchange information

## Robustness

the level of support provided the user in determining successful achievement and assessment of goal-directed behaviour

# Principles of learnability

## Predictability

- determining effect of future actions based on past interaction history
- operation visibility

## Synthesizability

- assessing the effect of past actions
- immediate vs. eventual honesty



# Principles of learnability (ctd)

## Familiarity

- how prior knowledge applies to new system
- guessability; affordance

## Generalizability

- extending specific interaction knowledge to new situations

## Consistency

- likeness in input/output behaviour arising from similar situations or task objectives

# Principles of flexibility

## Dialogue initiative

- freedom from system imposed constraints on input dialogue
- system vs. user pre-emptiveness

## Multithreading

- ability of system to support user interaction for more than one task at a time
- concurrent vs. interleaving; multimodality

## Task migratability

- passing responsibility for task execution between user and system

# Principles of flexibility (ctd)

## Substitutivity

- allowing equivalent values of input and output to be substituted for each other
- representation multiplicity; equal opportunity

## Customizability

- modifiability of the user interface by user (adaptability) or system (adaptivity)

# Principles of robustness

## Observability

- ability of user to evaluate the internal state of the system from its perceivable representation
- browsability; defaults; reachability; persistence; operation visibility

## Recoverability

- ability of user to take corrective action once an error has been recognized
- reachability; forward/backward recovery; commensurate effort

# Principles of robustness (ctd)

## Responsiveness

- how the user perceives the rate of communication with the system
- Stability

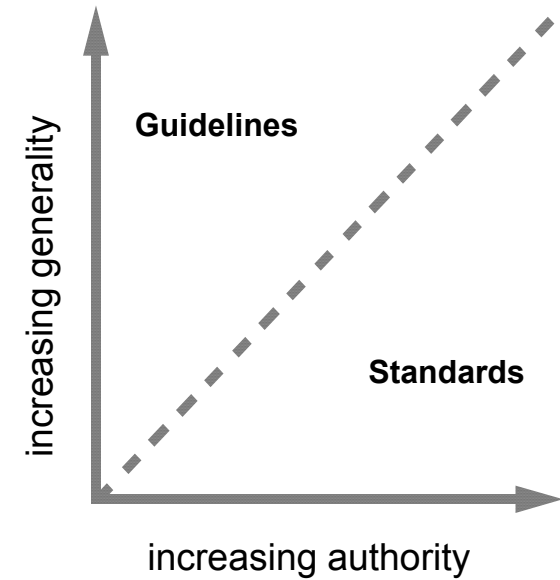
## Task conformance

- degree to which system services support all of the user's tasks
- task completeness; task adequacy

# Using design rules

## Design rules

- suggest how to increase usability
- differ in generality and authority



# Standards

- set by national or international bodies to ensure compliance by a large community of designers standards require sound underlying theory and slowly changing technology
- hardware standards more common than software high authority and low level of detail
- ISO 9241 defines usability as effectiveness, efficiency and satisfaction with which users accomplish tasks

# Guidelines

- more suggestive and general
- many textbooks and reports full of guidelines
- abstract guidelines (principles) applicable during early life cycle activities
- detailed guidelines (style guides) applicable during later life cycle activities
- understanding justification for guidelines aids in resolving conflicts



# Golden rules and heuristics

- “Broad brush” design rules
- Useful check list for good design
- Better design using these than using nothing!
- Different collections e.g.
  - Nielsen’s 10 Heuristics (see Chapter 9)
  - Shneiderman’s 8 Golden Rules
  - Norman’s 7 Principles

# Shneiderman's 8 Golden Rules

- 1. Strive for consistency*
- 2. Enable frequent users to use shortcuts*
- 3. Offer informative feedback*
- 4. Design dialogs to yield closure*
- 5. Offer error prevention and simple error handling*
- 6. Permit easy reversal of actions*
- 7. Support internal locus of control*
- 8. Reduce short-term memory load*

# Norman's 7 Principles

- 1. Use both knowledge in the world and knowledge in the head.*
- 2. Simplify the structure of tasks.*
- 3. Make things visible: bridge the gulfs of Execution and Evaluation.*
- 4. Get the mappings right.*
- 5. Exploit the power of constraints, both natural and artificial.*
- 6. Design for error.*
- 7. When all else fails, standardize.*

# Heuristics (by Nielsen)

- use simple and natural dialogue sequences
- speak the users' language
- minimize user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

# Windows Interface Guidelines

- Set of general principles for interface design in Microsoft's software development documentation
  - directness
  - user in control
  - consistency
  - forgiveness
  - feedback
  - aesthetics
  - simplicity

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimize user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

- strive for consistency
  - enable frequent users to use shortcuts
  - offer informative feedback
  - design dialogues to yield closure
  - offer simple error handling
  - permit easy reversal of actions
- reduce short term memory load

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

Be consistent

# Consistency.....

- important to enable user to build a reliable model of how the interface works
- makes the interface familiar and predictable by providing a sense of stability
- allows users to transfer existing knowledge to new tasks and focus more on tasks because they need not spend time trying to remember the differences in interaction.
- important through all aspects of the interface, names of commands, layout of information, and operational behaviour.

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimize user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

### n

- strive for consistency
- enable frequent users to use shortcuts
- offer informative feedback
- design dialogues to yield closure
- offer simple error handling
- permit easy reversal of actions
- support internal locus of control
- reduce short

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

Reduce memory load



# Minimize user memory load

- **Basic rule:** don't expect the user to remember what has already been done. Make this visible at the interface
- If a command is made up of a number of pieces of data entered by the user in sequence, display these rather than expecting the user to remember the data already entered
- Help the user remember where they are in a transaction sequence – Menu 2/5 Step 1 - 4

# Example: American Airlines site

AA® Travel Planning AAdvantage® Net SAver™ & Special Offers Programs & Services About AA Contact Us

Home Gates Times & Status Book Flight Flight Status Notification View My Miles

## TRAVEL PLANNING

YOU ARE HERE

FIND FLIGHTS PRICE ITINERARY PASSENGER DETAILS PURCHASE FINISH

Place in transaction sequence

## Find Flights: Choose by Schedule

[Show American Airlines flights only](#)

Birmingham, England (BHX) to Chicago, IL (CHI)  
Friday, 02 November 2001

Data previously entered

Select	Flight	Departing		Arriving		Seats Available			Aircraft
		City	Date & Time	City	Date & Time	First	Bus	Coach	
<input type="radio"/>	KLM 2052 *	BHX	02Nov 07:10pm	AMS	02Nov 09:25pm			●	100
	Singapore Airlines 36\$@	AMS	03Nov 08:10am	ORD	03Nov 10:20am		●	●	777
<input type="radio"/>	British Airways 1764	BHX	02Nov 07:20pm	FRA	02Nov 09:55pm		●	●	ER4
	United Airlines 3501 *\$@	FRA	03Nov 10:40am	ORD	03Nov 01:00pm		●	●	742

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimize user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

- strive for consistency
- enable frequent users to use shortcuts
- offer informative feedback
- design dialogues to yield closure
- offer simple error handling
- permit easy reversal of actions
- support internal locus of control
- reduce short term memory load

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

Feedback to user

# Feedback from the system

- Every action the user makes should produce a perceptible response.
- The intention is to reduce user uncertainty that the system has:
  - received the last input,
  - is currently doing something about it,
  - or is waiting for the next input.
- Commands should result in some visible change to the interface
  - E.g 'mail has been sent' in response to a 'Send' command
  - Presentation of objects on screen updated to reflect their current state
- Task analysis should enable appropriate information to be identified as feedback for a specific task

# Feedback: Response Time

- Response time for feedback should be appropriate to the type of user action:
  - e.g. response to keystroke - instantaneous;  
response to command input - may take longer
- Provide 'system busy' feedback if time will exceed a few seconds or is unpredictable
- Provide indication of how many transactions remain, for example as a bar chart or as a percentage.
- This largely disappeared as a problem with fast single user PCs and has re-appeared with distributed web-based applications

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimise user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

- **n**
- strive for consistency
- enable frequent users to use shortcuts
- offer informative feedback
- design dialogues to yield closure
- offer simple error handling
- permit easy reversal of actions
- support internal locus of control
- reduce short term memory

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

Appropriate user support

# Appropriate user support

- HELP messages
  - important to recognise different types of help;
  - should be available when required and context-specific;
  - can the user get help about what responses are possible at a given point in a dialogue.
- ERROR messages
  - should explain what is wrong and what corrective action is required;
  - should use 'jargon' familiar to the user;
  - often this support is poorly designed in terms of what information is given to the user.

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimise user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

- strive for consistency
- enable frequent users to use shortcuts
- offer informative feedback
- design dialogues to yield closure
- offer simple error handling
- permit easy reversal of actions
- support internal locus of control
- reduce short term memory load

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

Flexibility



# Flexibility

- Measure of how well a dialogue can cater for different levels of user skill.
- Provide alternative means of achieving the same goal which match different models of how the interface works.
  - e.g. word selection: cursor to start of word and double click, click and drag, click and shift-click.
  - e.g. word deletion: word highlighted and Control +X key, select 'Cut' menu option, backspace.

# Flexibility

- Adapt to the skill level of the user by:
  - providing accelerators:
    - allow user to answer ahead,
    - provide key bindings for menu options;
  - providing macro facility;
  - accepting abbreviations for command words;
  - accepting synonyms (alternative names);
  - allowing user to choose level of instructions or help.

# Many common elements...

## Nielsen

- use simple and natural dialogue sequences
- speak the users language
- minimise user memory load
- be consistent
- provide feedback
- provide clearly marked exits
- provide shortcuts
- provide good error messages
- prevent errors

## Shneiderman

- strive for consistency
- enable frequent users to use shortcuts
- offer informative feedback
- design dialogues to yield closure
- offer simple error handling
- permit easy reversal of actions
- support internal locus of control
- reduce short term memory load

## Microsoft

- directness
- user in control
- consistency
- forgiveness
- feedback
- aesthetics
- simplicity

User in control

# User in control

- user initiates actions, not the computer or software
- use techniques to automate tasks, but implement them in a way that allows the user to chose or control the automation.
- users must be able to personalize aspects of the interface, such as colour, fonts, or other options

# Minimal user input

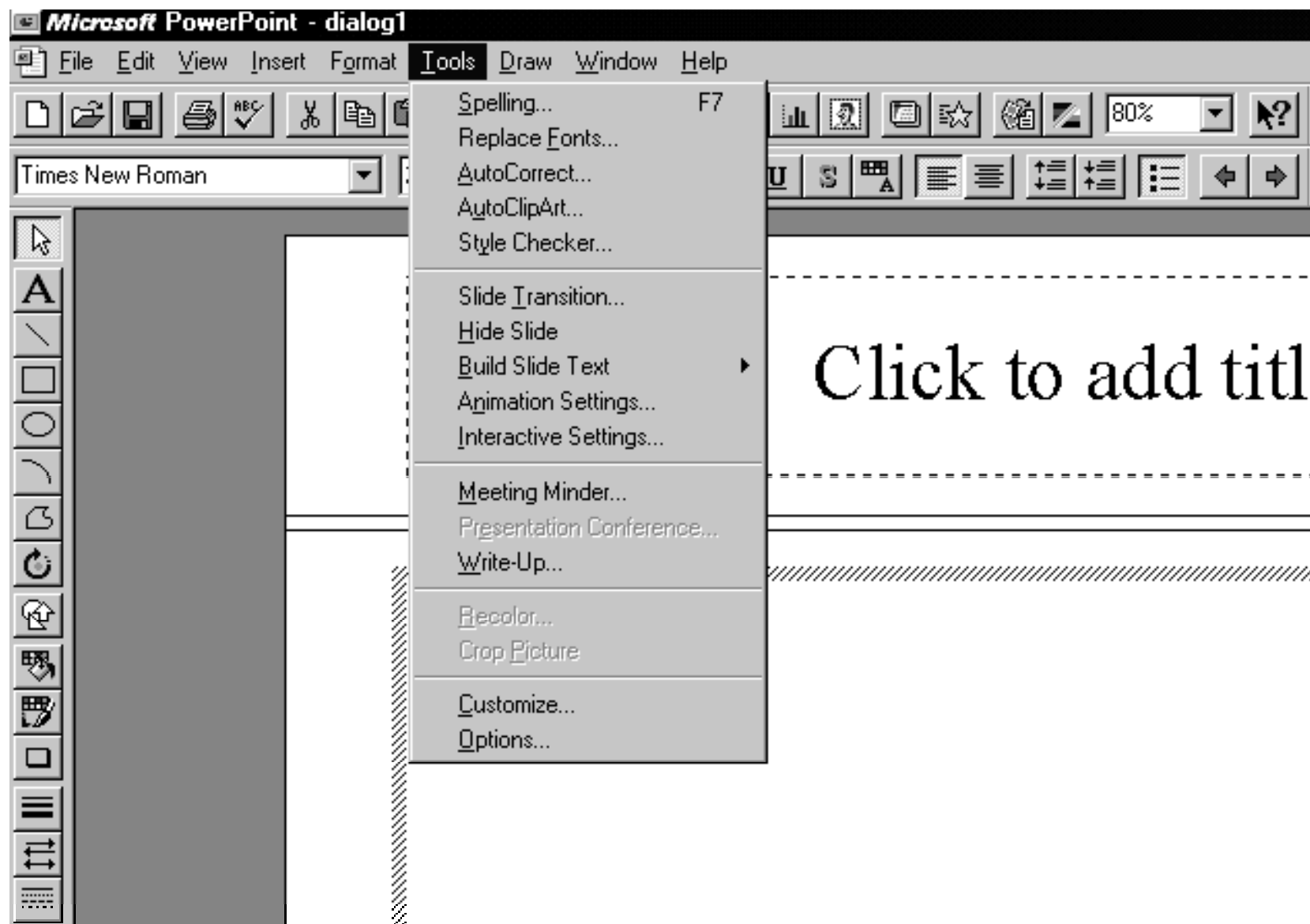
- Balance between number of keystrokes or mouse movements/clicks and memory load.
- Reducing keying errors increases speed of data entry.
- Allow selection from a list rather than typing in a value (recognise rather than recall).
- Edit a command that has produced an error rather than retyping the command.
- Do not request input of information which can be derived automatically or which has been entered previously.
- Use default values.

# Menus

- Usually a collection of actions, structured into a list from which a user chooses
- Actions applied to objects
  - Explicitly selected by user – `format + font...` [selected text]
  - Implicitly assumed by system – `print` [current file]
  - Pop-up menu over selected object shows common actions on that object
- Actions may be represented
  - by text (e.g pull-down menu)
  - by icons (e.g toolbar)
- Actions completed
  - Immediately by selecting menu item
  - Following collection of more data from user (via a dialogue box)

# Overloading menus

- Most common Windows applications use an 'anything, anytime' approach – i.e., all commands are available to the user at all times
- Leads to large, cumbersome menu structures where the user can forget how to find a particular command
- Toolbars attempt to provide shortcuts to frequently used items
  - order of icons in toolbars different from items in pull-down menus representing same actions
- Many CAD systems use an alternative, "mode" approach where a general type of operation, or task is selected
  - Only a restricted set of menus relevant to that operation are displayed
- This approach is now used in some MS applications





# Menu Structure

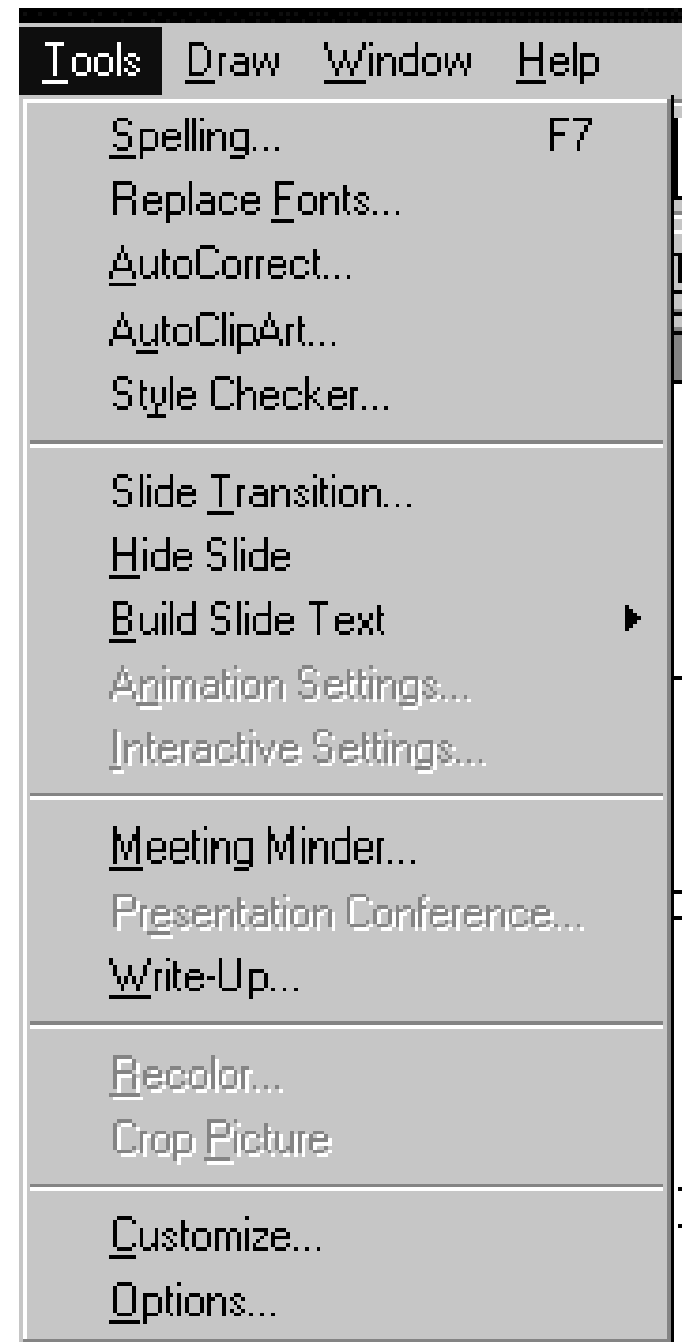
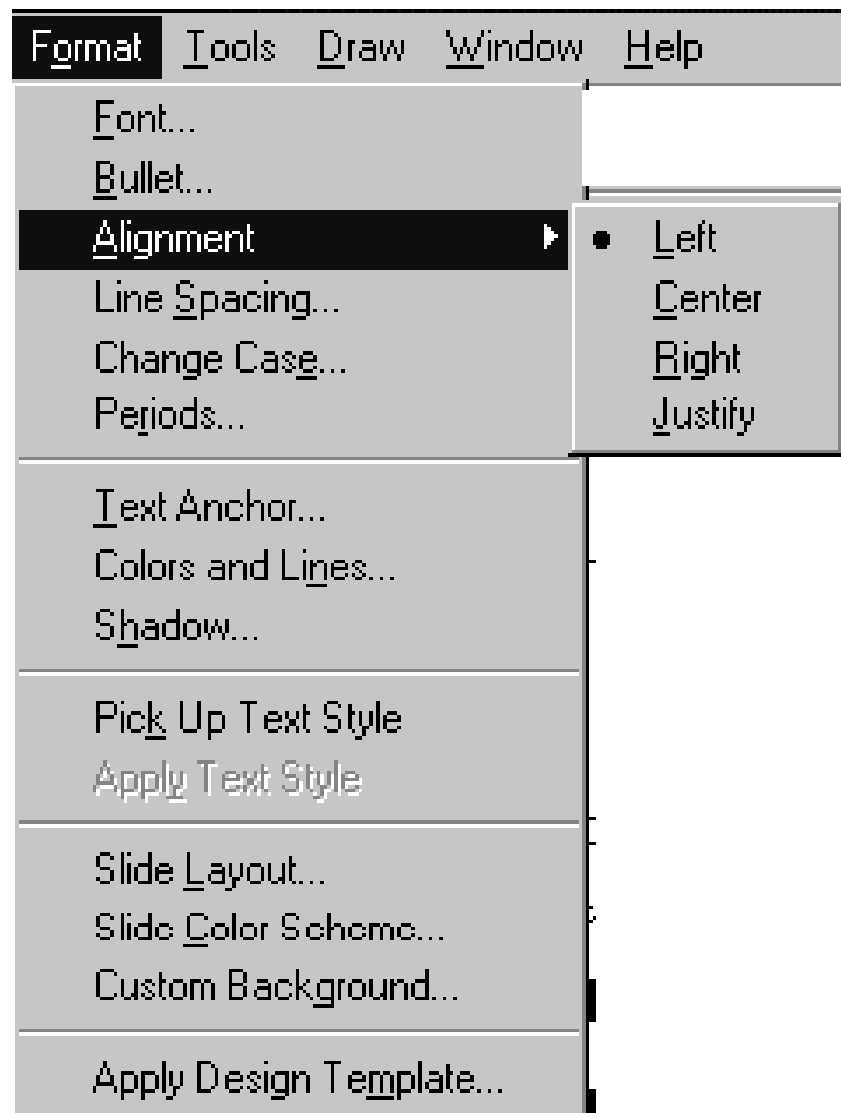
'Structures should reflect users expectations.. and support users flow of work' (ISO 9241/14)

## **Priorities**

- Conventional categories (file, edit,...)
- Use of dividers to break menus into groups
- Logical groups of related actions (cut,copy,paste)
- Arbitrary groups
  - consistently ordered, numerically or alphabetically

# Sequencing options within groups

- **consistency** - use the same relative order of items where the group is presented again
- **importance** - place important items first in the group
- **conventional** order e.g days of the week
- **order of use** - e.g 'copy' precedes 'paste'
- **frequency** of use
  - if frequency of option is known, place frequent items first
- **alphabetical** order
- What ordering rules have been applied in the next slide?



# Functional Objectives with Screen Layout

- arrange items on screen to give highest probability of eliciting an acceptable level of human performance
- the user will be able
  - extract information she is seeking
  - identify related groups of information
  - distinguish exceptional items (warnings and error messages)
  - determine what action is necessary

# Formatting recommendations

- split strings more than 6 alphanumeric into smaller groups

(bad)

ABBA347686A2

ABBA456388A3

(good)

ABBA 347686 A2

ABBA 456388 A3

- identical data should be presented in the same way even if variations in input format are tolerated

30 11 95

30 Nov 1995      -> 30/11/95 (for example)

30 11 1995

30th nov 95

# Formatting recommendations

- data should be presented in full version even if abbreviated input allowed, provide feedback to user

Party:[            ]

Party:[ ch,cai] Chemical Bank, Cairo

# Formatting recommendations

- numeric codes displayed with right justification

47321

47321

539

539

67

67

482645

482645

- lists of numeric with decimal points should be aligned around the point

34.723

43.908

2341.5

# Labeling in screen design

- descriptive title or phrase adjacent to a group of related items or information
- ensure labels are meaningful to the user
- labeling should be visually distinct from the data
- data labeling should not be able to be confused with help messages or command descriptions



# Labeling in screen design

- use consistent relationship between labels and data being described

e.g. above and left justified

Name:

$$[ \quad ]$$

- include units in label to reduce ambiguity

e.g. Weight( Kg):

$$[ \quad ]$$

# Aesthetic issues

- Design is valued for its fitness to a particular user and task
- Design aesthetics is intended to make the product or system appear attractive & appealing
- Nielsen advocates Simplicity – particularly for Website design
- However careful use of color, graphics and formatting can make the design more aesthetically pleasing
  - Need to get the right balance

# Style guides and sources of design guidance

- Plenty of these....
- Manufacturers
- Web-based style guides e.g., Yale Style Manual
  - <http://info.med.yale.edu/caim/manual/index.html>

# Optional readings

- **The Promise of Pattern Languages for Interaction Design**

- <http://www.it.bton.ac.uk/staff/lp22/HF2000.html>

- **Interaction Design and Agile Methods**

- <http://webservices.xml.com/pub/a/ws/2002/09/03/udell.html>

# Summary

## Principles for usability

- repeatable design for usability relies on maximizing benefit of one good design by abstracting out the general properties which can direct purposeful design
- The success of designing for usability requires both creative insight (new paradigms) and purposeful principled practice

## Using design rules

- standards and guidelines to direct design activity

# Reading Assignment #3

- **Dynamic Positioning Systems  
Usability and Interaction Styles**

**by Bjorneseth et al.**

**ACM NordiCHI 2008**