Course Overview

- Introduction
- Understanding Users and Their Tasks
- Principles and Guidelines
- Interacting with Devices
- **Interaction Styles**
- UI Design Elements
- Visual Design Guidelines
- UI Development Tools
- Iterative Design and Usability Testing
- Project Presentations & Selected Topics
- Case Studies
- Recent Developments in HCID
- Conclusions

Chapter Overview Interaction Styles

- Motivation
- Objectives
- Batch Systems
- Command-Line Interfaces
- Full-Screen Interfaces
- Menus
- Forms
- Direct Manipulation Interfaces
- WIMPs
- Natural Language
- Intelligent Agents
- Important Concepts and Terms
- Chapter Summary

Motivation

- the way interaction with computers is performed depends on available technology on one hand, but also on different methods and usage styles for a particular technology
- the change in interaction style from command-line to graphical user interface has contributed considerably to the popularity of personal computers
- for a given task and user population, different interaction styles may make the difference between success, acceptance, or failure of a product

Objectives

- to know the advantages and drawbacks of the most often used interaction styles, in particular command-based vs. graphical user interfaces
- to be familiar with evaluation criteria for a comparison of different interaction styles
- to be able to select an appropriate interaction style for a specific task, environment, and user population
- to be exposed to emerging interaction styles like natural language, gestures, or intelligent agents

Evaluation Criteria

- name and briefly describe at least three interaction styles
  - e.g. batch, command line, GUI
- apply three criteria to the evaluation and comparison of interaction styles
  - representation, operation, navigation
- discussion of potential advantages and drawbacks of emerging interaction styles
  - e.g. usability, user control, computational overhead, prevention of errors

Interaction

- exchange of information between user and system
- actions of the user that change the status of the system
- feedback to the user concerning actions of the system
- requires translation between the intentions of the user and the actions of the system

Terminology

- task
  - actions to be performed in order to solve a problem in an application domain
- goal
  - desired output from a completed task
- task analysis
  - identification of the problems space
    - in particular domain, goals, intentions, specific tasks
- user language (task language)
  - describes the problem to be solved in terms familiar to the user
- system language (core language)
  - describes the functionality of the system in terms familiar to the designer or developer

Interaction Model

- execution
  - establishing the goal
  - forming the intention
  - specifying the action sequence
  - executing the action
- evaluation
  - perceiving the system state
  - interpreting the system state
  - evaluation of the system state
    - with respect to goals and intentions

Interaction Styles

- term covers all of the ways that users interact with a computer system
  - also referred to as communication styles or dialog styles
- represent alternative design strategies for the UI
  - each style offers its own way of organizing the system's functionality, of managing the user's inputs, and of presenting information
    - e.g. display-based interfaces -> menus, mice, windows, widgets, icons, buttons, function keys, etc.
- provide a behavioral view of how the user communicates with the system
  - "look and feel"

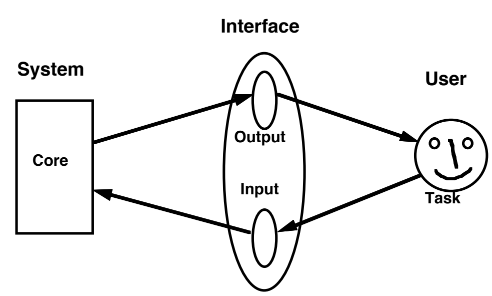
Evaluation Criteria for Interaction Styles

- representational
  - how a system "looks and feels" (graphical), or "sounds and feels" (speech)
- operational
  - how a system links sequences of operations together, and maps the representations used onto its functions
- navigational
  - awareness of what users can do at each stage, and how they can move through the system

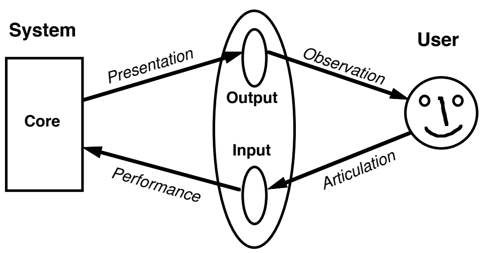
Interaction Framework

- components
  - system user, input, output
  - input and output together form the interface
  - each of the components may have its own language to describe the objects and actions it is concerned with

Interaction Framework



Interactive Cycle



Translations Between Components

- articulation
  - user translates task intentions into the input language
- performance
  - input language is translated into stimuli for the system
- presentation
  - system activities are translated into output language
- observation
  - output language is translated into the user's task model

Example: Light in a Room

- controlling the lighting in a room
  - task language: turn lights on/off
  - input language: flip switch
  - system language: close/open circuit for light bulbs
  - output language: lights on/off

Example (cont)

- translations
  - articulation
    - user decides to turn on the light, and flips a switch
  - performance
    - flipped switch closes the circuit
  - presentation
    - light bulb emits light
  - observation
    - user notices that the light is on
- frequent problem
  - multiple switches in large rooms

Practical Exercise

- Describe the languages and translations for the interactions required to open a document containing the third homework for this class.

Practical Exercise: Open a Document

- languages
  - task language: activities described in natural language
  - input language: user commands, file names
  - system language: system calls, OS objects
  - output language: commands for an output device
    - e.g. screen, or PostScript for a printer

Practical Exercise (cont.)

- translations
  - articulation
    - user translates “document with the third homework” into the corresponding file name
      - double-click on file “HW-3.html”
  - performance
    - system translates user-level commands and parameters into system-level commands and objects
      - system calls to invoke an application program compatible with HW-3.html’s type, and to load the file
  - presentation
    - system translates internal representation into a form suitable for output devices
      - display of the file on the screen
  - observation

Practical Exercise: Digital vs. Analog Clocks

- Describe and compare the languages and translations used for setting digital and analog clocks. What are frequent problems with the two device types and their languages or translations?
- languages
  - task language: set time to a certain value
  - input language: wheel vs. buttons
  - system language: mechanical movement vs. ICs
  - output language: minute and hour hands vs. LCD display

Practical Exercise (cont.)

- translations
  - articulation
    - turn wheel / press buttons
  - performance
    - clock translates input actions into modified display
  - presentation
    - new time setting is displayed
  - observation
    - user translates hand positions or numbers into time

Practical Exercise (cont.)

- problems
  - which way do you turn the wheel
  - which button do you press, how often, in which combination
  - turning the time “back” on a digital clock is frequently impossible

Batch Systems

- first generation of user interfaces
- interaction restricted to a single point in time
  - submission of a batch job as a single unit
- user commands have to be specified before the result of any of them is made known to the user
- work well for repetitive jobs (e.g., payroll processing, billing, etc.), and are still used today
  - Examples: Revenue Canada, Canada Post, Bell Canada.
- drawbacks
  - not interactive
  - inflexible

Command-Line Interfaces

- also referred to as *Command-Language* or *Line-Oriented* Interfaces
- A new user’s first view of almost all command-line interfaces: “>“
- one-dimensional
  - user interacts with a system on a single line that serves as the command line
  - once the user hits the return key, the input can no longer be modified
- typical of many early computer systems, but remain common even today
  - e.g., MS-DOS, UNIX

Command-Line Interfaces (cont.)

- responsibility for navigation is on the user
  - user has to know what the allowable commands are
  - needs to have a clear idea of the function to be performed)
- difficult for novice users to learn, but once mastered, *command languages* represent the quickest form of communication (i.e., with abbreviations, use of function keys, etc.)
- *query languages* represent a special case of command languages; allow users to request information

Example 1 Command-Line Interface

```
C > A:

A > dir

Not ready error reading drive A
Abort, Retry, Ignore? _
```

A typical dialog using an MS-DOS system

Example 2 Command-Line Interface

```
HP-UX bmtlh730 A.09.05 A 9000/755 (ttyw0)

login: pmust
Password:
Please wait...checking for disk quotas

mtlh730:~> cokol
Enter EDMX password:

mtlh730:~> exit
```

A typical dialog using a UNIX system (Telnet session)

Advantages Command-Line Interfaces

- powerful
  - commands can be stringed together, automated scripts can be run, macros can be used, commands can be executed repetitively)
- flexible, user controlled
  - users can do anything in any order
- fast, efficient
  - requires more typing than other interaction styles, but is probably the fastest, most efficient of all interaction styles
  - very fast and efficient for expert, high-frequency users)
- use minimal screen space
  - only one line at a time

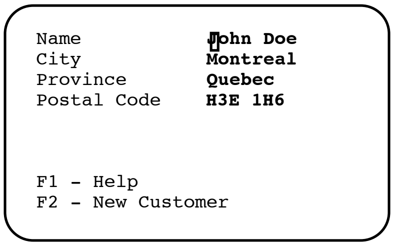
Drawbacks Command-Line Interfaces

- operating system or program exposed to the user
- difficult to learn
  - very cryptic, use of arbitrary syntactic delimiters, require rote memorization
- difficult to remember
  - especially for infrequent users
  - rely totally on recall
- inflexible
  - commands need to be expressed using a precise syntax
  - any variation represents an error
- susceptible to errors, poor error handling

Full-Screen Interfaces

- two-dimensional
  - user is no longer limited to a single command line, but can move around the screen along two dimensions
    - e.g. form-filling application, in which the user is presented with a number of labeled fields that can be edited in any sequence.
- many full-screen interfaces use function keys, enumerated choices, or the first characters of a menu item as the primary interaction style

Example Full-Screen Interfaces



Example of a full-screen interface

Menus

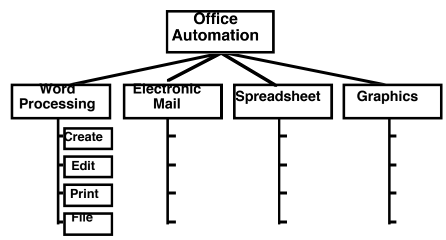
- mechanism that allows the user to make a selection from a limited set of options
- one of the most popular interaction styles
- the user does not have to remember the name or abbreviation of a command
  - recognize it from a list of available options
- names of options, contents of the icons, descriptions of buttons should be self-explanatory

Menu Building Blocks

- menus can consist of
  - textual descriptions of the available functions
  - icons
    - small graphic images that represent different aspects of an interface metaphor
  - buttons
    - horizontal or vertical
  - boxes

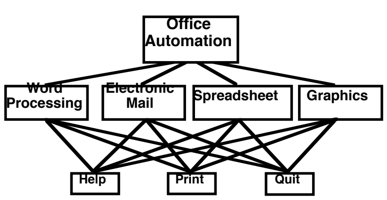
Hierarchical Menus

- most frequently used form of menus



Networked Menus

- makes nodes accessible from different locations in a menu
  - "menu bypass"
  - often realized through keyboard shortcuts



Menu Types

- fixed menus
  - remain in place until the option is selected
- pull-down menus (drop-down menus)
  - dragged down from a menu bar at the top of the screen, an item is selected, and the menu automatically returns back to its original title
- pop-up menus
  - appear when a user clicks on a particular area of the screen, which may be designated by an icon
  - menu remains in place until the user instructs it to disappear, usually by clicking on a "close" box in the border of the menu's window
- cascading menus
  - display all options chosen one after the other in a cascading fashion

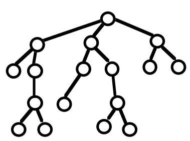
Design Considerations for Menus

- order of items
  - alphabetically, by category, or by frequency
  - some functions should be kept apart (e.g., ‘create’ option should not be placed next to the ‘delete’ option)
- selection of items
  - number or letter corresponding to the required option
  - pointing at the option using a pointing device
  - highlighting the item through cursor control keys
- navigation through a series of menus
  - hierarchically-structured menus
    - main menu with a series of sub-menus to make further selections

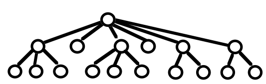
Tradeoffs with Menus

- depth vs. breadth
  - deep: few alternatives, many selection steps
  - broad: many alternatives, few selection steps
- deep and narrow is often difficult for users
- broad and shallow is usually easier
  - user does not have to go through as many levels, thereby reducing the need for user navigation
  - each node in the menu hierarchy becomes more complex, making the user choose between more options at each level

Depth and Breadth in Menus



deep menu hierarchy



broad menu hierarchy

Advantages of Menus

- self-explanatory
  - require little training or learning
- require little human memory
  - easy to remember, recognition rather than recall
- indicates when parts are not relevant
- interaction under user control
- fast, efficient interface for experienced users
  - shortcuts to mouse actions
- few keystrokes, which reduces errors
- easy error handling
- enhancements and changes are visible

Problems with Menus

- not appropriate or efficient for some users and tasks
  - may slow down frequent or expert users
- users may get lost in menu hierarchies
- menu names may not be meaningful to users
- inflexible
  - unless highly networked, the user is forced through set sequences of steps
- impractical for numerous choices
  - good for a limited number of valid inputs at any given time
- use lots of screen space
- most effective with a pointing device
  - mouse, trackball, touch screen

Fill-in Forms

- similar to a paper fill-in form
  - presented on a computer screen instead of paper
- formatted structure containing fields, in which the user inputs data
  - each field has a label or caption that indicates the type of data to be entered in that field
- organization and layout are important aspects

Example Fill-in Forms

```
Type in the information below.
Press TAB to move the cursor to the next field.
Press ENTER when done.

Name:          ______________________ Phone: (___) ___-___
Address:       ______________________
               ______________________
City: ________________ Province: ____ Postal Code: _______

Catalog No.  Quantity           Catalog No.  Quantity
__________   ___________        __________   ___________
__________   ___________        __________   ___________
__________   ___________        __________   ___________
__________   ___________        __________   ___________
__________   ___________        __________   ___________
```

A fill-in form interface design for a department store

Advantages Fill-in Forms

- self-explanatory
  - little training or learning required
- require little human memory
  - rely on recognition rather than recall
- efficient use of screen space
  - multiple fields can be presented on one screen
- parameters with many possible input values
  - e.g., names, street names, city names, postal codes, etc.
- provide context
  - context information from other fields
- enhancements and changes are visible

Problems Fill-in Forms

- assume knowledge of valid inputs
  - e.g., catalog numbers, part numbers, etc.
- assume typing skills
- users need to know how to use TAB to move to the next field, BACKSPACE to correct
- resent opportunities for making errors
- inflexible
  - input order cannot be changed

Questions-and-Answer Interfaces

- combine some of the features of both menus and fill-in forms in a unique dialog style
  - the user is faced with a single question at a time
  - the user has to input an answer, rather than select one from a list
- system driven
  - the user does not have to worry about navigation.
  - suitable for novice or untrained users
  - can be frustrating to more experienced users
- often the best choice for data-entry tasks performed by unskilled users

Example Questions-and-Answers

Canada Census 1998

Enter your name:
(Last, First, Initial)
> Kurfess, Franz J.

Enter your marital status:
(M - married, S - single, D - divorced, W - widowed)
> M

Citizenship:
(C - Canadian, O - Other))
> O

For instructions	**Press HELP**
To accept responses	**Press RETURN**
To backup	**Press BACKSPACE**
To skip a question	**Press RETURN**
To cancel and quit	**Press ESC**
To save and quit	**Press EXECUTE**

A question-and-answer interface design of a Canada census form

Advantages Questions-and-Answers

- self-explanatory
  - little training or learning required
- require little human memory
  - recognition rather than recall
- simple, non-intimidating
- parameters with many possible input values
  - e.g., names, street names, city names, postal codes, etc.
- accommodate tasks with a hierarchical structure
  - skip questions that do not pertain
- accommodates complete and clear prompting
  - only one question at a time on the screen
- enhancements are visible

Drawbacks Questions-and-Answers

- inefficient
- assume typing skills
- opportunities for making errors
- inflexible
  - input order cannot be changed
  - except in wizards, which may have a “Previous” button
- no forward context
  - serial nature of questions may restrict context information

Direct Manipulation

- describes the interaction between user and object
  - directly connects an action to an observable response from an object
- follows an object-action paradigm
  - user performs tasks by selecting an object (e.g., icon, window, or text), and then selects an action (e.g., move, close, underline) to perform on that object
- permits users to control their environment
  - by directly manipulating graphical objects and controls similar to those that they encounter in real life
  - push button starts an action
  - slider is used to select an analog setting, etc.
  - mapped onto affordances

Principles of Direct Manipulation

- visible objects of interest
  - representing an object with an icon
- continuous representation of objects and actions of interest in a meaningful metaphor
- objects and actions are shown
- rapid, reversible, incremental actions
- immediate visibility of results of actions
- direct manipulation of the object of interest instead of a complex command language syntax
  - typing is replaced with pointing and selecting

Direct Manipulation Examples

- name three applications or domains where direct manipulation
  - is appropriate
    -
    -
    -
  - is not appropriate
    -
    -
    -

Indirect Manipulation

- in many industrial applications, objects on the computer screen stand for devices in the real world
- manipulation of the screen objects may not directly constitute an identical manipulation of the real-world objects
- feedback is required on two levels
  - directly from the user interface
  - indirectly, but more relevant from the real-world devices

Direct Manipulation Interfaces

- powerful
  - expert users can work quickly, wide range of tasks
- novice users can learn basic functionality quickly
  - either through self exploration or through a demonstration by a more experienced user.
  - no need to learn and remember complex commands
- goal-oriented
  - users can see immediately if their actions are helping them realize their goals
  - if not, they can simply change the direction of their activity
- error messages are rarely needed

WIMP Interfaces

- direct manipulation interfaces are characterized by
  - **windows**
    - to divide the screen into areas
  - **icons** representing objects
    - can be moved around the screen
  - **mouse** (or another pointing device)
    - used to manipulate objects on the screen
  - pop-up or **pull-down menus**
    - display the available options

WIMP Advantages

- system as a whole is easily visible
  - objects are represented by icons
  - available menu options can be inspected by pulling down menus
- basic actions are consistent across systems
  - opening, closing, copying, deleting, scrolling, etc.
  - makes learning easier
- users are free to explore different aspects of the system
  - most actions can be 'undone' (reversed) if it does not have the desired or required effect

GUIs and Direct Manipulation

- GUIs rely heavily on direct manipulation
  - have become synonymous with the WIMP interface
- in addition to having all of the characteristics of DM and WIMP interfaces, GUIs are also strongly graphical
- the primary interaction style is direct manipulation

Advantages of Direct Manipulation

- easy to learn and remember
  - rely on recognition, not recall
- comprehensible, predictable, and controllable
  - users experience less anxiety
  - users feel in control
- visual: WYSIWYG
- flexible, reversible actions
  - undo -> you can always choose to de-select a menu item
- provide context, and instant visual feedback
- exploit human use of visual-spatial cues
  - "a picture is worth a thousand words!"
- less prone to errors
  - low typing requirement, visual feedback
  - less need for error messages

Problems with Direct Manipulation

- not self-explanatory
  - not necessarily intuitive or obvious to first-time users
  - users must learn meaning of the visual representation
- can be inefficient for high-frequency users
  - same tasks much faster/ more efficient with command languages
- moving a mouse and pointing slower than pressing keys
- repetitive tasks
  - users need a macro or scripting mechanism to handle repeated manipulations
- limited accuracy
  - pointer system may not be accurate enough or sufficiently controllable for some tasks

Natural Language Interaction

- natural language (NL) to communicate and interact with a computer system
  - Such a system would have to be able to handle the vagueness, ambiguity, and ungrammatical constructions associated with natural language
- ideally, a NL dialog should be based on speech input
  - speech recognition technology is still far from being perfect
  - users may have to rely on typing, which has its own problems
    - spelling errors, slow, verbose
- field is still in its infancy

Non-command Interfaces

- no explicit commands
  - computer interprets observations of the user
- ubiquitous computing, immersive computing
- gesture recognition, user observation
- intelligent agents, user preferences

Selection of Interaction Styles

- considerations
  - who has control
  - ease of use for novice users
  - learning time to become proficient
  - speed of use (efficiency) once users become proficient
  - generality, flexibility, power
  - task requirements
  - user background

Task Requirements

- high degree of data entry is required
  - fill-in forms are recommended, but a command-line interface may also be used
- familiar notation exists
  - use a command-line interface
- visual exploration is required
  - use a direct manipulation interface
- users have poor keyboard skills
  - rely on menu selection and direct manipulation

User Background

- novice users
  - use menu selection and direct manipulation
- modest domain knowledge, some computer skills
  - use menu selection, direct manipulation, or fill-in forms
- users work with the application on an intermittent basis
  - any interaction style may be used, but some form of help (user guide, on-line) may also be required
- frequent users
  - use a command-line interface with macros, menu selection with type-ahead capability, direct manipulation with shortcuts, or a fill-in form with a dense display

Overview Interaction Styles

Interaction Style	Mainly used in	Main Characteristics
Batch	Batch processing, e-mail servers	Does not require user intervention. Works even when user and computer are in a different time or place.
Command language	Command-line interfaces	Easy to edit and reuse command history. Powerful language can support very complex operations.
Fill-in Form	Full-screen interfaces	Many fields can be seen and edited at once.
Question-and-Answer	Command-line interfaces	Computer controls the user. Suited for casual use.
Menu	Full-screen, WIMP, telephone-based interfaces	Frees user from remembering options, at cost of potentially being slow or having confusing hierarchy.
Direct Manipulation	WIMP, virtual reality interfaces	User is in control. Enables metaphors from real world. Good for graphics.
Non-Command	Future interfaces, virtual reality interfaces	User is freed to concentrate on the domain, and need to control the computer. Computer monitors users and interprets their actions.
Natural Language	Future interfaces	Ideally, allows unconstrained input to handle frequently challenging problems.

Conclusions Interaction Styles

- many interaction styles
- not always possible to draw clear, unambiguous distinctions between different interaction styles
- the selection of an appropriate interaction style depends on many factors
  - application, users, characteristics
  - tasks
    - simple, walk-up-and-use vs. browsing and data entry
  - time to complete the tasks
  - accuracy to complete the tasks
  - levels of experience

Important Concepts and Terms

- batch system
- command-line interface
- contextual task analysis
- desktop
- direct manipulation
- forms
- full-screen interface
- goal
- graphical user interface (GUI)
- heuristic evaluation
- hierarchical menu
- human-machine interface
- intelligent agent
- interaction style
- menu
- mouse
- natural language
- networked menu
- system language
- task
- task analysis
- usability
- user-centered design
- user interface design
- user language
- user requirements
- What You See Is What You Get" (WYSIWYG)
- WIMP
- window

Chapter Summary

- practically all current interaction styles are variations of command-based interfaces
- the currently predominant interaction style is the WIMP interface
  - windows, icons, mouse, pull-down menus
- the selection among various possible interaction styles depends on user background, task requirements, technology, and economical factors
- emerging technologies like natural language processing or intelligent agents may cause a shift towards non-command interfaces