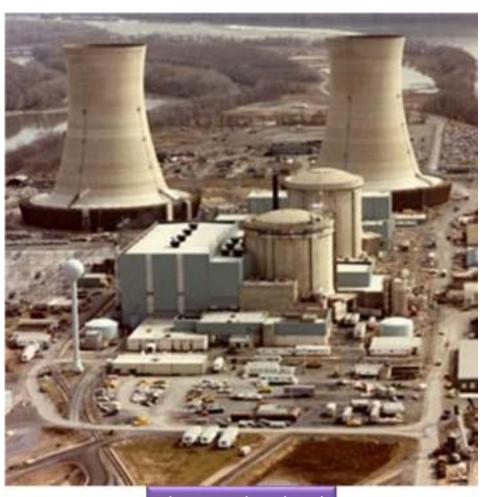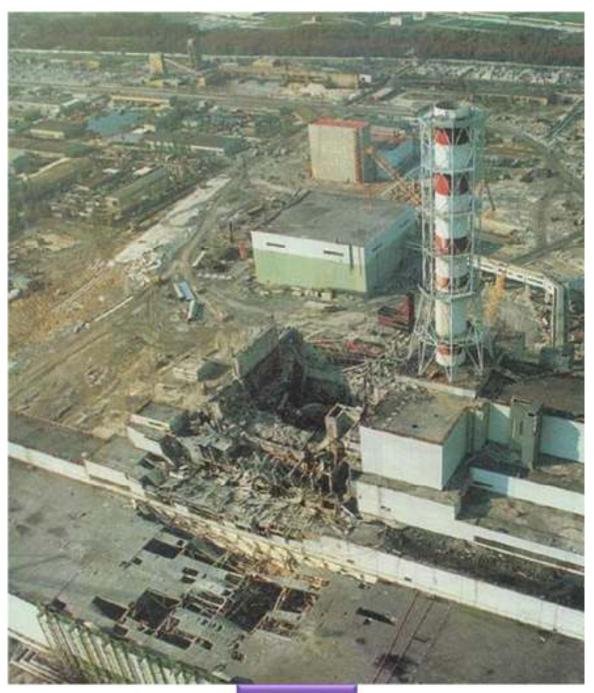# User Interface
CSE 403

Three Mile Island

Chernobyl

How do we avoid bad UI?

- Learn from past mistakes

- Build prototypes

Big questions

- What's the point of prototyping? Should I do it?
    - If so, when in the overall process or "lifecycle" should I?

- Should I make my prototype on paper or digitally?

- How do I know whether my UI is good or bad?
    - What are the ways in which a UI's "quality" can be quantified?
    - What are some examples of software you use that have especially good/bad UIs? What do you think makes them good/bad?

Usability and software design

- **usability**: the effectiveness of users achieving tasks
  - Human-Computer Interaction (HCI).
  - Usability and good UI design are closely related.
  - A bad UI can have serious results...

Achieving usability

- User testing and field studies
  - having users use the product and gathering data
  - card sorting: ask users to group/design menus
- Evaluations and reviews by UI experts
- Prototyping
  - Paper prototyping
  - Code prototyping

- Good UI design focuses on the *user*
   not on the developer, not on the system environment

Prototyping

- **prototyping**: Creating a scaled-down or incomplete version of a system to demonstrate or test its aspects.

- Reasons to do prototyping:
  - aids UI design
  - help discover requirements
  - help discover test cases and provide a basis for testing
  - allows interaction with user and customer to ensure satisfaction
  - team-building

Some prototyping methods

1. UI builders (Visual Studio, ...)

   draw a GUI visually by dragging/dropping UI
   controls on screen

2. implementation by hand

   writing a "quick" version of your code

3. **paper prototyping**: a paper version of a UI

Why do paper prototypes?

- much faster to create than code
- can change faster than code
- more visual bandwidth (can see more at once)
- more conducive to working in teams
- can be done by non-technical people
- feels less permanent or final

Where does paper prototyping fit?

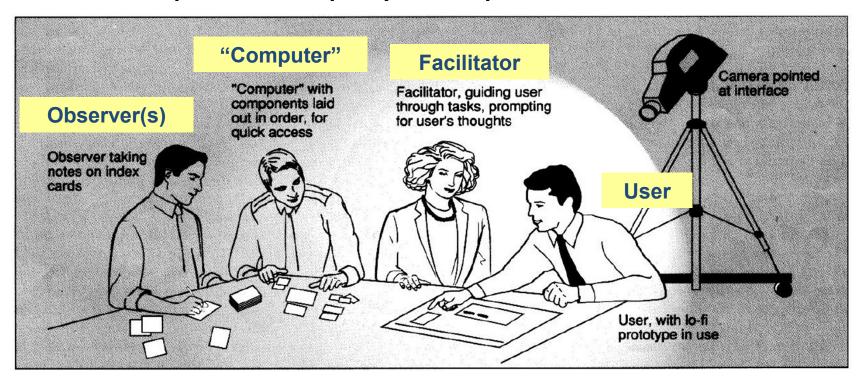When in the software lifecycle is it
most useful to do (paper) prototyping?

- Requirements are the what and design is the how. Which is paper prototyping?

- Prototyping
  - helps uncover requirements and upcoming design issues
  - during or after requirements but before design
  - shows us what is in the UI, but also shows us details of how the user can achieve goals in the UI
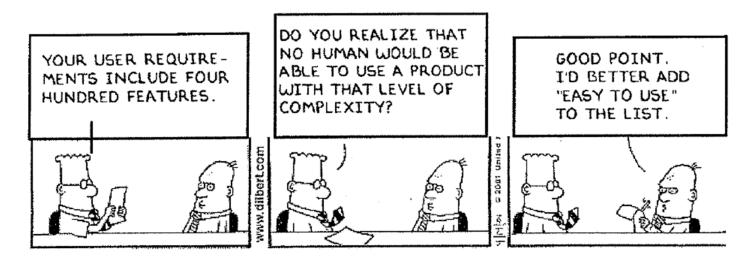
Paper prototyping usability session

- user gets tasks to perform on a paper prototype
  - use real-world terminology, not that used by your GUI
- observed by people and/or recorded
- a developer can "play computer"

Observer(s)

Observer taking notes on index cards

"Computer"

"Computer" with components laid out in order, for quick access

Facilitator

Facilitator, guiding user through tasks, prompting for user's thoughts

Camera pointed at interface

User

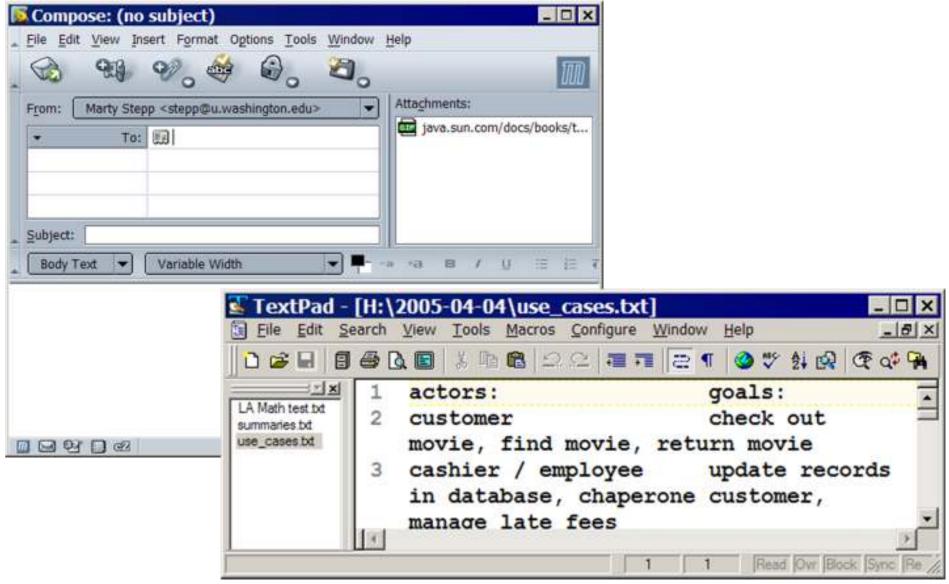User, with lo-fi prototype in use

Schneiderman's 8 Golden Rules

1. Strive for consistency.
2. Give shortcuts to the user.
3. Offer informative feedback.
4. Make each interaction with the user yield a result.

5. Offer simple error handling.
6. Permit easy undo of actions.
7. Let the user be in control.
8. Reduce short-term memory load on the user.

YOUR USER REQUIRE-MENTS INCLUDE FOUR HUNDRED FEATURES.

DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?

GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.

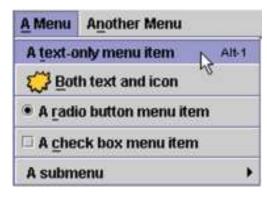*(from Designing the User Interface, by Ben Schneiderman of UMD, noted HCI and UI design expert)*

UI design examples

UI design, components

- When should we use:
  - A button?
  - A check box?
  - A radio button?
  - A text field?
  - A list?
  - A combo box?
  - A menu?
  - A dialog box?
  - Other..?

UI Hall of Shame

Feedback | Server Info | Notes | Completion

Radio Buttons | List Boxes | Graphics | Links | Table | Trailing

General | Leading Text | Text Inserts | Text Fields | Comment Boxes | CheckBox

HOME **SEARCH** OPEN HOUSES FIND REP

FARM | COMM | MFAM | RENT

RESIDENTIAL | CONDO | LOT

-Low Price- | -Hi Price-

-# Beds- | -# Baths-

Neighbourhood:

All

WinSock API Breakpoints

Enabled | Breakpoint Description
BREAK ON API CALL: connect
BREAK ON API CALL: send
BREAK ON API CALL: recv
BREAK ON API CALL: closesocket
BREAK ON API CALL: socket
BREAK ON API CALL: shutdown
BREAK ON API CALL: accept
BREAK ON API CALL: bind
BREAK ON API CALL: select

Modify

Update | Cancel

eZip Wizard - Evaluation Copy

What would you like to do?

eZip

○ Unzip an existing ZIP file
○ Create a new ZIP file
○ Update an existing ZIP file

About... | Register... | < Back | Next > | Cancel

Transfer Disk

Trash

Layout and color

Bad error messages

AK-Mail

? Do you really want to delete the selected folder ?

Please enter 'YES' to start the operation

[   ]

[OK] [Cancel]

Eye Candy

? Are you sure you want to delete 'Ridges'?

[OK]

Microsoft Access

⚠

[Ja] [Nee]

---

www.wvfiremarshal.org - [JavaScript Application]

? Welcome to the West Virginia State Fire Marshal On-line information center. This site is best viewed using Explorer or Navigator versions 4.0 or later and a display setting of 800x600.

[OK] [Cancel]

Microsoft Access

⚠ **Wrong button!**
This button doesn't work.

Solution
Try another.

[OK]

Dialog

⚠ CuteFTP is currently working. If you press Disconnect, the session will be interrupted. Do you want to disconnect?

☐ Don't show this dialog again

[OK] [Help]

Document Wizard Result

HTML Conversion complete!

Press View Result to view resulting documenation.

[View Result]

UI design – buttons, menus

- Use **buttons** for single independent actions that are relevant to the current screen.
  - Try to use button text with verb phrases such as "Save" or "Cancel", not generic: "OK", "Yes", "No"
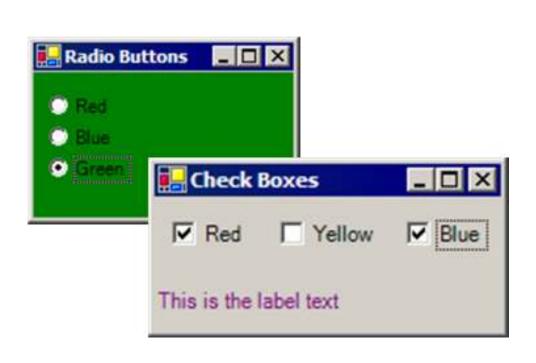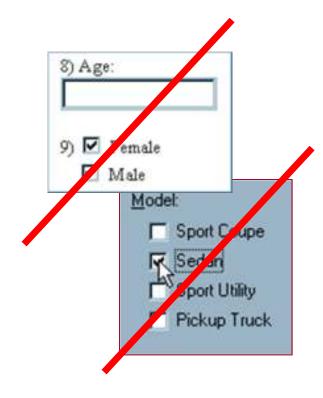 - use <u>M</u>nemonics or Accelerators (Ctrl-S)

- Use **toolbars** for common actions.

- Use **menus** for infrequent actions that may be applicable to many or all screens.
  - *Users hate menus!* Try not to rely too much on menus. Provide another way to access the same functionality (toolbar, hotkey, etc.)

UI design – checkboxes, radio buttons

- Use **check boxes** for independent on/off switches
- Use **radio buttons** for related choices,
  when only one choice can be activated at a time

UI design – lists, combo boxes

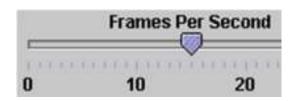- use **text fields** (usually with a label) when the user may type in anything they want

- use **lists** when there are many fixed choices (too many for radio buttons); *all* choices visible on screen at once
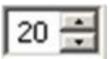
- use **combo boxes** when there are many fixed choices; don't take up screen real estate by showing them all at once
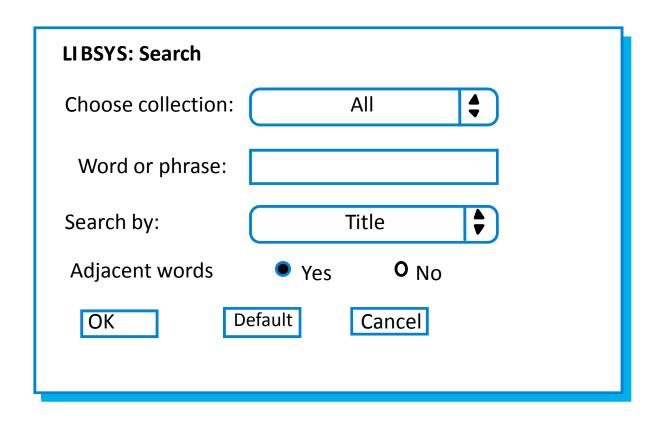
- use a **slider** or **spinner** for a numeric value

An example UI

- Good UI dialog?
  Did the designer choose the right components?
  assume there are 20 collections and 3 ways to search

LIBSYS: Search

Choose collection: [All ▲▼]

Word or phrase: [   ]

Search by: [Title ▲▼]

Adjacent words   ● Yes   ○ No

[OK]   [Default]   [Cancel]

UI design – multiple screens

- use a **tabbed pane** when there are many screens that the user may want to switch between at any moment

- use **dialog boxes** or **option panes** to present temporary screens or options
  - "modal" dialog box prevents any other action
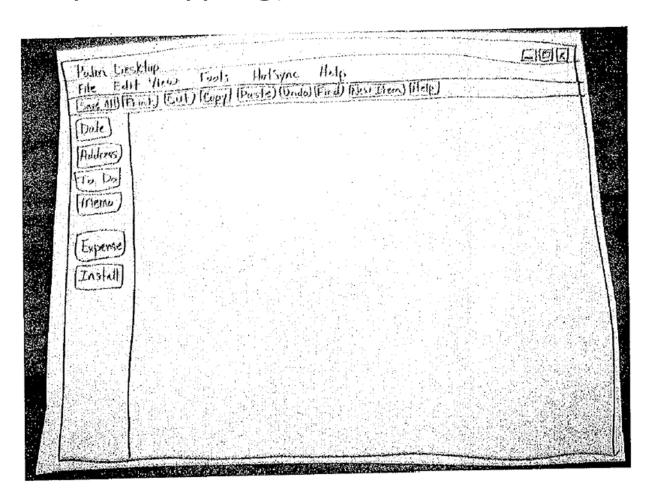
Creating a paper prototype

- Gather materials
  – paper, pencils/pens
  – tape, scissors
  – highlighters, transparencies

- Identify the screens in your UI
  – consider use cases, inputs and outputs to user

- Think about how to get from one screen to next
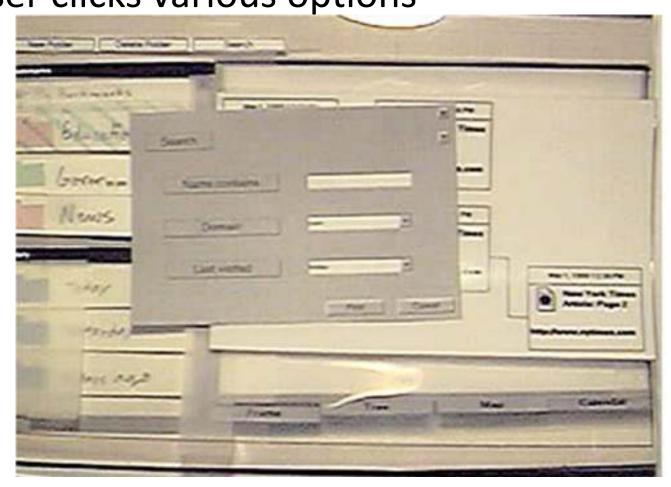 – this will help choose between tabs, dialogs, etc.

Application backgrounds

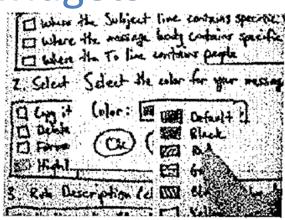- Draw the app background (parts that matter for the prototyping) on its own

Representing a changing UI

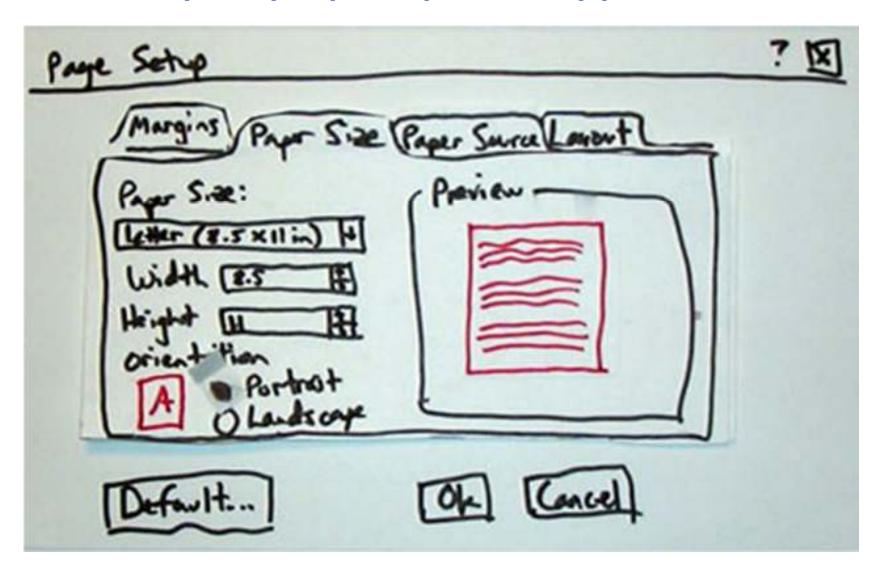- Place layers of UI on top of background as user clicks various options
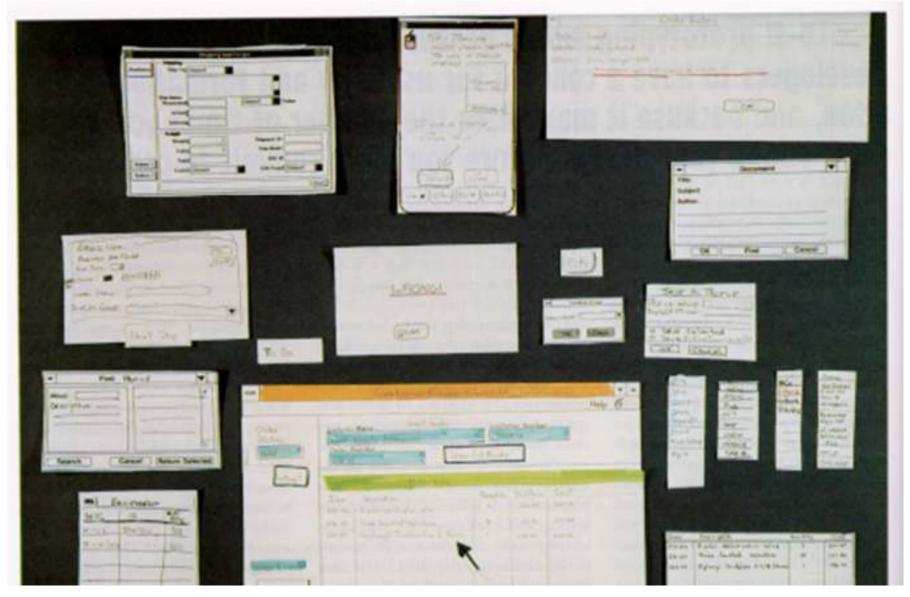
Representing interactive widgets

- buttons / check boxes: tape
- tabs, dialog boxes: index cards
- text fields: removable tape
- combo boxes: put the choices on a separate piece of paper that pops up when they click
- selections: a highlighted piece of tape or transparency
- disabled widgets: make a gray version that can sit on top of the normal enabled version

- computer beeps: say "beep"

Example paper prototype screen

Example full paper prototype

How to Watch Users

- Brief the user first (being a test user is stressful)
  - "I'm testing the system, not testing you"
  - "If you have trouble, it's the system's fault"
  - "Feel free to quit at any time"
  - Ethical issues: informed consent
- Ask user to think aloud
- Be quiet!
  - Don't help, don't explain, don't point out mistakes
  - Sit on your hands if it helps
  - Two exceptions: prod user to think aloud ("what are you thinking now?"), and move on to next task when stuck
- Take lots of notes

Prototyping exercise

- In your project groups, draw a rough prototype for a music player (e.g., WinAmp or iTunes).

  – Assume that the program lets you store, organize, and play songs and music videos.

  – Draw the main player UI and whatever widgets are required to do a **search for a song or video**.

  – After the prototypes are done, we'll try walking through each UI together.

- Things to think about:

  – How many clicks are needed? What controls to use?

  – Could your parents figure it out without guidance?