CS 520

Theory and Practice of Software Engineering
Fall 2026

User Interfaces

1

User Interface



Three Mile Island

2

At Three Mile Island



The light reported the command

It said the valve was told to close. It was stuck open.

Over 100 alarms in two minutes

Nothing told operators which alarm mattered most.

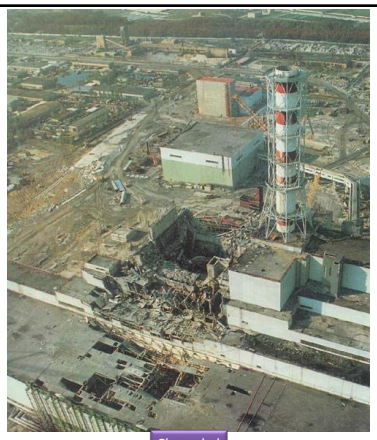
The alarm printer fell behind

Key messages reached people far too late.

Show the real state. Prioritize alarms.

4

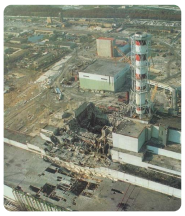
3



Chernobyl

4

Chernobyl hid its danger from operators



A hidden danger

The reactor became unstable at low power. Operators had not been told.

The safety control backfired

A design flaw let inserting the control rods briefly add power.

Same pattern as Three Mile Island

Operators could not see the true state.

Tell users the risks. A safety control must never make things worse.

5

5

How do we avoid bad UI?

- Learn from past mistakes
- Build prototypes

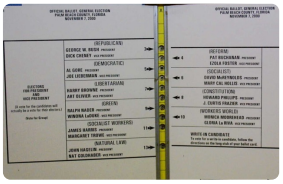
6

Big questions

- What's the point of prototyping? Should I do it?
  - If so, when 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 quality can be quantified?
  - What are some examples of software you use that have an especially good/bad UI?
    What do you think makes them good/bad?

7

Florida, 2020: The butterfly ballot



Palm Beach County, 2000

Two columns, one row of holes

Names sat on both sides. The holes ran down the middle.

Holes did not match names

The 2nd hole is not the 2nd name on the left.

Easy to mark the wrong person

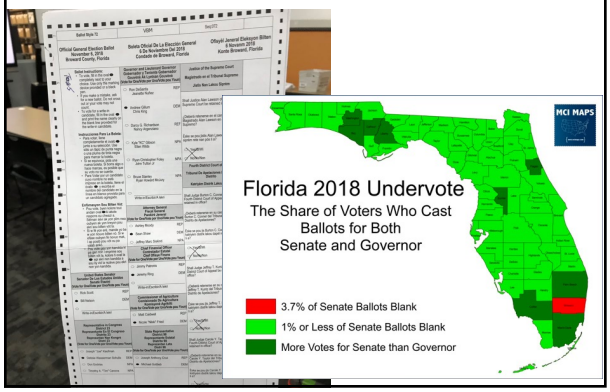
Some voters likely chose someone they did not intend.

Make controls line up with their labels.

6

8

Florida, 2018: Hidden vote



9

Hawaii's alert tool invited a false alarm



Live and test shared one interface

A drop-down chose the template. Nothing marked live as dangerous.

One generic confirmation

"Are you sure?" looks the same every time, so people click yes.

For 38 minutes, a whole state believed it. The sender's screen was a drop-down and an "Are you sure?" box.

No fast way to take it back

The correction went out 38 minutes later.

8

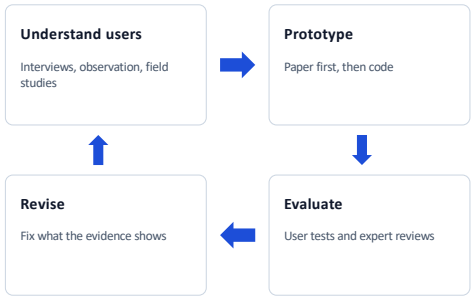
10

Achieving usability

- User testing and field studies
  - having users use the product and gathering data
- 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

11

Usable UIs come from a loop, not a guess



10

12

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
  - provides basis for testing
  - team-building
  - allows interaction with user to ensure satisfaction

13

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



14

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

15

Where does paper prototyping fit?

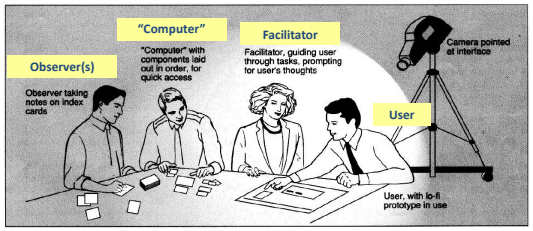
When in the software lifecycle is it most useful to do (paper) prototyping?

- 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

16

Paper prototyping usability session

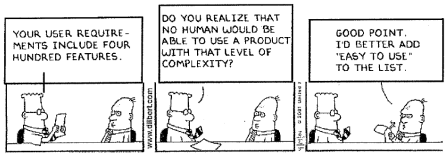
- user gets tasks to perform on a paper prototype
- observed by people and/or recorded
- a developer can "play computer"



17

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.



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

18

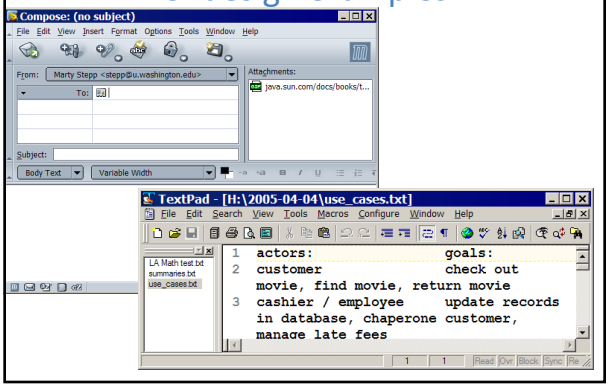
Shneiderman's eight golden rules

- **Consistency**: same actions, same words, same layout
- **Shortcuts** for frequent users
- **Feedback**: every action gets a visible response
- **Closure**: a group of actions has a clear end
- **Prevent errors**, and make recovery simple
- **Easy reversal**: undo lets people explore safely
- **User in control**: people start the actions
- **Less memory load**: show options, do not make users recall

30

19

UI design examples



20

Each control exists for one job

Save Cancel

Button

Starts one action

Switch, checkbox

Toggles one option

Boston

Select

Picks one from a list

Email address

Text field

Takes free-form input

Inbox Sent Drafts

Tabs

Switch between views

File deleted Undo

Toast

Confirms, offers undo

11

21

Label buttons with the action they do

- One primary action per screen
- Style destructive actions differently
- Show shortcuts, such as Ctrl+S or underlined letters

Before

Save changes to report.docx?

Yes No Cancel

After

Save changes to report.docx?

Save Don't save Cancel

12

22

Toolbars for common, menus for rare

Toolbar

Save Undo Redo Share

Actions used constantly. One click, always visible.

Menu

New window Ctrl+N

Export as PDF Ctrl+E

Page setup

Document properties

Rare actions. Shortcuts teach themselves.

People avoid menus, so give every menu action a second route.

13

23

Pick checkbox, radio, or switch by rule

Checkbox

- [x] Email
- [x] Text message
- [ ] Push

Any number can be on. Changes apply when saved.

Radio buttons

- (•) Small
- ( ) Medium
- ( ) Large

Exactly one of a few. Show every option.

Switch

Wi-Fi [on]

Bluetooth [off]

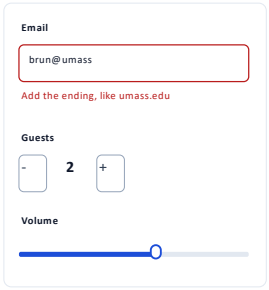
On or off, and the change applies immediately.

14

24

Let people type only when they must

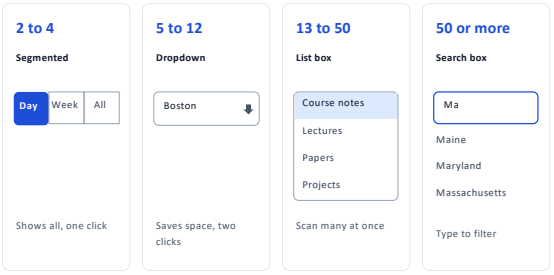
- Show a visible label, not only a placeholder
- Use the right input type: email, number, date
- Validate as people type, and say how to fix it
- Use a stepper or slider for numbers



15

25

Match control to number of choices



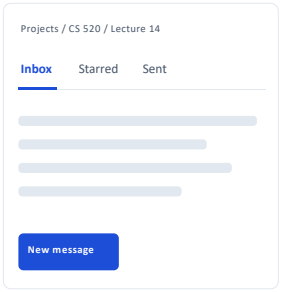
Rules of thumb, not laws. Check them with real users.

16

26

Tabs switch views, dialogs interrupt

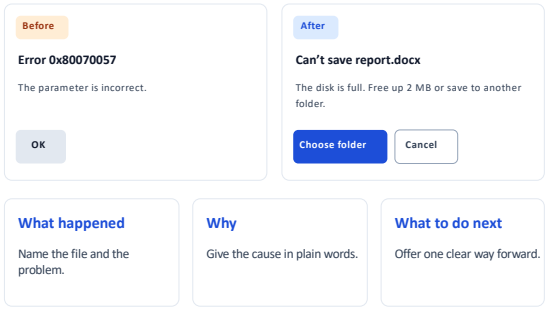
- **Tabs**: peer views the user may switch between at any moment
- **Dialogs**: temporary tasks or decisions, kept short
- **Breadcrumbs and nav bars**: show where you are and where else you can go



17

27

Good errors say what, why, and the fix



What happened
Name the file and the problem.

Why
Give the cause in plain words.

What to do next
Offer one clear way forward.

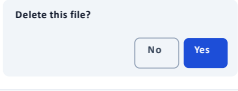
18

28

Prefer undo to "Are you sure?"

Confirmation dialog

Interrupts every time, so people click through without reading.



Undo toast

Instant and recoverable, so a slip costs one click.



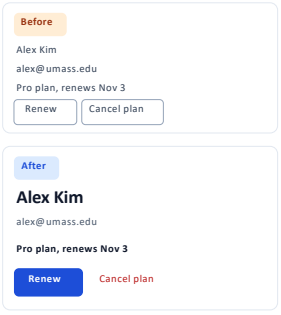
Confirm only what cannot be undone, and name the consequence.

19

29

Space, size, and contrast set hierarchy

- **Proximity**: related items together, unrelated apart
- **Alignment**: shared edges let the eye scan
- **Contrast**: size and weight show what matters
- **Consistency**: same look means the same thing



20

30

Accessible design helps every user

- **Contrast**
- **Target size**: at least 24 by 24 pixels
- **Keyboard**: every control reachable, focus visible
- **Labels**: screen readers announce each control
- **Color**: never the only signal

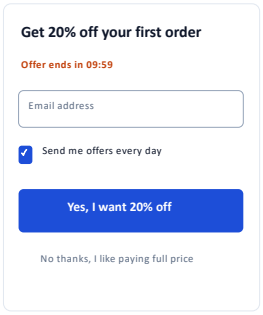


21

31

Dark patterns turn the UI against users

- **Confirmshaming**: guilt-trip wording on the decline button
- **Preselection**: opt-in boxes already checked
- **Roach motel**: easy to join, hard to cancel
- **Fake urgency**: countdown timers that reset



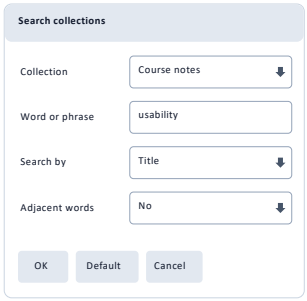
22

32

Spot the five control mistakes

Did the designer choose the right components?

Assume there are 20 collections and 3 ways to search.

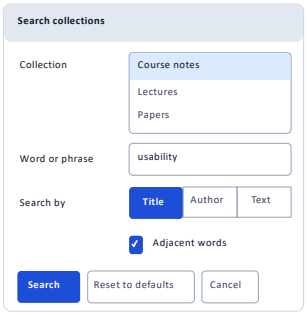


23

33

Same dialog, right controls

1. **Collection**: a list shows many choices at once
2. **Search by**: three choices fit a segmented control
3. **Adjacent words**: one checkbox replaces Yes and No
4. **OK**: becomes the verb, Search
5. **Default**: becomes Reset to defaults



24

34

Start with users, tasks, context

1

Who uses it?

Novices, experts, students, nurses. Skills and goals differ.

2

What do they do?

List the frequent and the critical tasks. Design for those first.

3

Where is it used?

A noisy ward, a phone on a bus, a desk. Context shapes the UI.

Ask them and watch them. Do not guess.

25

35

Sketch on paper before you code



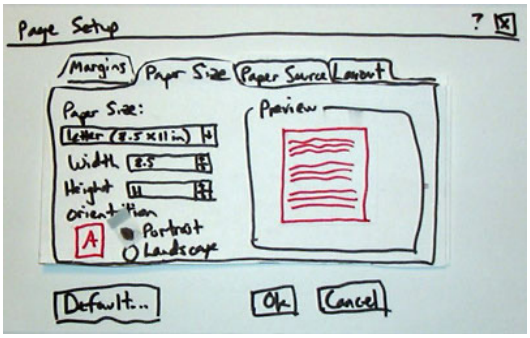
- Sketch each screen in minutes
- Let a person play the computer
- Change it with an eraser, not a refactor

Illustrative stock photo of a paper wireframe

26

36

Example paper prototype screen



37

Follow conventions people know

- Users expect your app to work like the others they use
- Use platform controls and standard icons with labels
- Break a convention only when testing shows a better way



29

38

Design for the full range of users

N

Novices

Clear labels, visible options, guidance at first use.

E

Experts

Shortcuts, bulk actions, and a command palette save time.

A

Everyone else

Accessible design also helps one-handed or sunlit users.

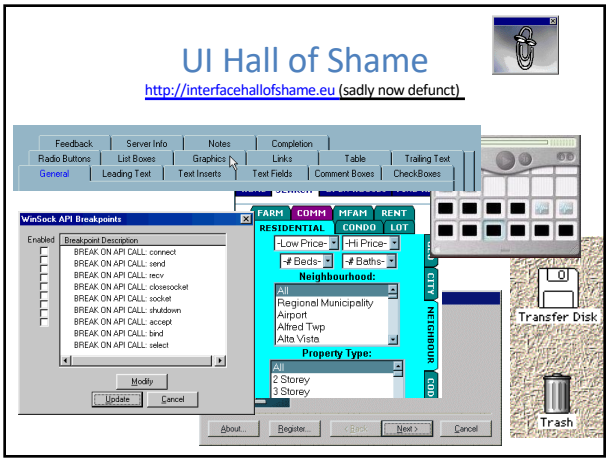
Test with people unlike you.

32

39

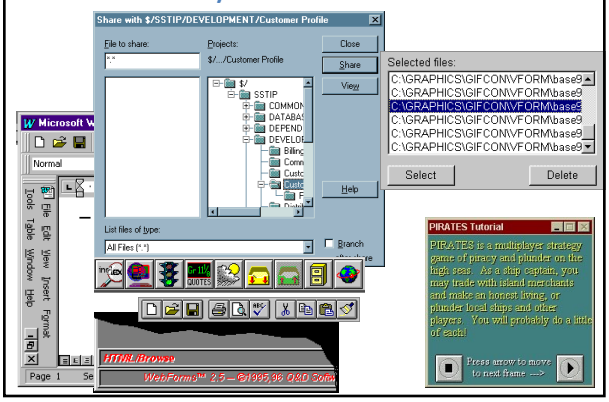
UI Hall of Shame

http://interfacehallofshame.eu (sadly now defunct)



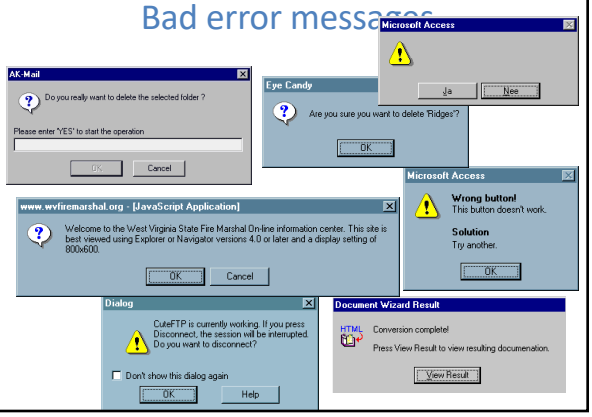
40

Layout and color



41

Bad error messages



42

Five design habits to keep

1. Know the users and tasks
2. Sketch cheaply, test early
3. Follow conventions
4. Give feedback and allow undo
5. Design for everyone

Recall: which habit would have helped most in the Hawaii alert?

33

46

Two ways to find usability problems

Inspection

- Experts review against heuristics
- Fast and cheap
- No users needed

User testing

- Real users attempt real tasks
- Slower, more realistic
- Finds problems experts miss

Use both: inspect first, then test.

34

47

Nielsen's ten usability heuristics

1 Visibility of system status	6 Recognition over recall
2 Match the real world	7 Flexibility and efficiency
3 User control and freedom	8 Minimalist design
4 Consistency and standards	9 Help users recover from errors
5 Error prevention	10 Help and documentation

35

48

Heuristics catch real problems

Uploading photo.jpg

62%

Visible status

Show progress so users know the system heard them.

Email address

Submit

Enter an email

Error prevention

Guard actions that cannot succeed yet, and say why.

Search

ui design

heuristics

think aloud

Show, don't recall

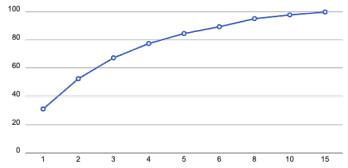
Show recent items instead of making users remember.

36

49

Five users find most problems

Problems found, by number of test users (model with L = 31%)



85%

of problems found by five users

Run several small tests, not one big one.

39

50

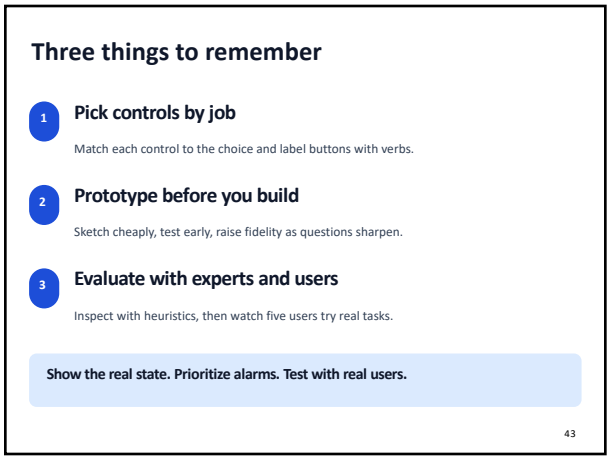
Run a think-aloud usability test

- Write three to five realistic tasks
- Ask users to think aloud
- Stay quiet and do not help
- Log where they struggle

"Tell me what you are thinking as you go. We are testing the app, not you."

40

51


Three things to remember
1
Pick controls by job
Match each control to the choice and label buttons with verbs.
2
Prototype before you build
Sketch cheaply, test early, raise fidelity as questions sharpen.
3
Evaluate with experts and users
Inspect with heuristics, then watch five users try real tasks.
Show the real state. Prioritize alarms. Test with real users.
43

52