basic issues in dialog design

◆ choosing an appropriate interaction style
  – direct manipulation, menus, forms, commands, ...

◆ organizing the interaction dialog to suit the task
  – addressing Norman’s gulfs
  – minimizing cognitive demands on the user
  – internal and external consistency
  – feedback, error prevention and recovery

◆ supporting different types of users
  – backgrounds, expertise level, work styles, disabilities

user interaction categories

◆ direct manipulation
  – affordances, point-and-do, visibility, incremental, immediate feedback, reversible

◆ menus (pop-up, pull-down, palettes, cascading, hot links)
  – leveraging recognition (term selection), use as a task structuring mechanism, short-cuts for frequent users

◆ forms and dialog boxes
  – composite of techniques, perhaps modal, often specialized input

◆ command and natural languages
  – efficiency and flexibility, programmability, expert (commands) or novices (natural language), speech input/output

◆ visualizations, 3D interactions, virtual reality
  – real world metaphor, navigation or exploration emphasis, the feeling of immersion, novel input/output
so when to use what?

- direct manipulation
- menus
- forms or dialog boxes
- command languages
- natural language
- 3D or immersive environments

where to go for guidance?

—see also Web pointers collected by Vlad——

- user interface standards
  - long, arduous process of consensus, buy-in
  - sometimes mandated by organizations, countries, etc
  - assumes an associated compliance procedure
- user interface guidelines
  - relatively high-level, intended to be generic
  - more detailed guidelines tend to lag practice (why?)
- style guides
  - more concrete, assume interaction techniques
  - may be commercial or customized for a specific product (or product family)

*Note that aids such as these focus on how to implement a user interface, and not much on what it should be*
customized style guide: examples

- display the time in hh:mm AM/PM (e.g., “10:13PM) in the upper right corner of the screen
- center the name of the application in the middle of the title bar, at the top of the window
- if a list does not fill the on-screen area allocated to it, do not display an elevator in the vertical scroll bar
- if a list does fill the on-screen area allocated to it, display an elevator, such that its location in the scroll bar is relative to the portion of the list that is visible
- use a blue background color for all displays
- when the cursor is positioned over a movable border (e.g., of a window), change its icon from a pointer to a hand

overview of homework 2
—note that the write-up on the Web has been updated—

1. each member of a team chooses a UI concern
   - e.g., messages/prompts, help, feedback, forms, menus, icons, screen organization, navigation
   - try to identify concerns most relevant to the interactive system you are starting to design
2. each member then proposes a minimum of 5 UI style recommendations
   - state the rule, then give 2-3 sentences of rationale: why are you recommending this, what impact will it have?
3. make two copies: one to turn in, one to share as part of project notebook
guidelines overview (Ch 2)

- user-centeredness
- individual differences
- issues from human cognition
- simplicity
- consistency
- feedback and messages, anthropomorphism
- prevent (and help users recover from) errors
- modal interactions
- getting users’ attention

*as we go through: can you relate to Norman's stages? can you think of examples from your own experience?*

user-centeredness

“Know the user!”

- understand users’ needs and preferences
  - often addressed through participatory methods
- develop designs that reflect this understanding
  - easy, quick, start up for new users
  - optimized operations for experienced users
- give users the feeling of control
  - flexible start, stop, resume
  - possible to change goals in midstream
  - system waiting for user’s initiative, not vice versa
designing for individual differences
(an obvious corollary)

◆ physical or cognitive challenges
  − alternate I/O devices, vocabulary choice
◆ attending to user experience differences
  − usually at a minimum: novice, intermittent, frequent, power user (can you really satisfy them all?!)
◆ personal abilities or working style issues
  − spatial skills, problem-solving, logic, concrete vs. abstract reasoning skills
◆ one important approach: user preferences
  − moving more and more into end-user programming

issues from human cognition
Much more than the old maxim “7 +/- 2”...

◆ minimize short-term memory requirements
  − chunking (grouping) strategies can help
  − break complex tasks into simpler subtasks (closure)
  − context, status info for resuming after interruption
◆ recognition rather than recall when possible
  − mnemonics for improved memory
◆ semantic and articulatory directness
  − the important concept of affordances, i.e., what an object suggests about how to use it (or what it does)
◆ leveraging the real world: metaphor, analogy
simplicity

Just keep telling yourself: “Less is more!”

◆ easy tasks **should** be easy!!
  – difficult tasks get broken into manageable subtasks
◆ try to apply minimalist design:
  ✔ cut the verbiage (do you really need to say it all?)
  ✔ expect, encourage inference (users can & will think!)
  ✔ organize into real world tasks (familiar, engaging)
  ✔ more guidance at points where errors are expected

  *This will be hard to do ... much iteration assumed.*
◆ think of Ben Shneiderman’s mantra: “overview, browse and zoom, details on demand”

consistency

But: Foolish consistency is the hobgoblin of little minds.

◆ the intuition: “principle of least astonishment”
  – a variant: do similar things in similar ways
◆ the design problem: how is similarity measured?
  – (like beauty, it’s in the mind of the beholder)
  – operates at many different levels: keystrokes, word choice, icon design, screen layout, help, ...
  – affects all of Norman’s levels of action (examples?)
◆ also, must understand trade-offs well enough to know when to **break** the consistency
  – e.g., “inconsistent” positions or defaults for critical or frequent dialogs
feedback, messages, and prompts
Tell the user what she is about to do, then tell her what she did!

◆ feed-forward to aid users in planning, action
  – e.g., icons, word choice, affordances in general

◆ feedback to aid in interpretation, evaluation
  – ranges from low-level (dynamic outlines, highlighting)
    to high-level (attribute updates, confirmations)
  – status during long operations, or after interruptions

◆ message content is a design task in itself
  – task-oriented wording; be specific and constructive
  – tone is important too: don’t be critical or threatening,
    avoid patronizing or being too “friendly”

blocking and recovering from errors
“To err is human; forgive by design!”

◆ first challenge is to anticipate errors
  – not the natural thing to design for! (But scenario-based reasoning can help here)

◆ then try to prevent them as much as possible
  – indicate allowable input, deactivate inappropriate choices
  – destructive actions: Help users avoid destroying their data!

◆ where possible, design error recovery mechanisms
  – provide an undo, simple and natural reversibility
  – context-specific help text (possibly even help agents)
  – again, note that this demands iterative user testing, to find
    the task steps most likely to need recovery assistance
modal interactions
Remember, we actually are experts in modal interaction!

◆ *generally*, same UI action has the same effect
  – e.g., typing a key causes input, moving the mouse changes the cursor location, clicking a menu opens it
◆ in reality, impossible to avoid modes altogether
  – e.g., rectangle vs. line creation, spell-check, dialogs
◆ but rarely should you design *preemptive* modes
  – i.e., where user must do something to “escape” (OK for time-critical alerts, confirming destructive actions, etc.)
◆ for the rest, help users to *manage* modes
  – salient status indicators of current mode
  – e.g., single click of another option or window

getting users’ attention
Like modes, use judiciously and for task-relevant reasons.

◆ humans are experts at managing attention!
◆ avoid blinking or other forms of animation unless you have very good reasons
  – alerts, process or 3D demonstrations, entertainment
◆ color (in moderation, e.g. 4-5) can be effective
  – but be aware of color conventions, and perception issues (blue hard to resolve, color-blindness issues)
  – consider doing gray-scale design first
◆ audio also can be very useful
  – directionless (unlike vision), so good for alerts
  – map audio characteristics to task needs
yes, there will be conflicts

◆ often associated with exception-handling
  – e.g., the consensus default doesn’t make sense
◆ how to be simple, concise, and incredibly helpful?
  – rarely can you satisfy all users all the time
◆ common to encounter problems stemming from legacy systems or platforms

your job: understand the trade-offs; use sound task-based reasoning to facilitate adjudication