cisc3650
human-computer interaction
spring 2012
lecture # I.2
theories, principles and guidelines

topics:

- Norman:
  - everyday things; design principles; activity theory; gulf of execution; gulf of evaluation
- Shneiderman:
  - theories; principles; guidelines

references:

- The Design of Everyday Things, by Donald A. Norman, *Ch 1, The psychopathology of everyday things* and *Ch 2, The psychology of everyday things*
- Designing the User Interface: Strategies for Effective Human-Computer Interaction (5th edition), by Ben Shneiderman and Catherine Plaisant. *Ch 2, Guidelines, principles and theories*.

Norman: Design of Everyday Things

- Issue: Everyday items that are poorly designed and are hard to figure out how to use *This is BAD!!*
- examples:
  - microwave oven
  - washing machine
  - doors
  - audio/visual equipment
  - telephones
- Typically, things that are designed to do *everything* are usually hard to figure out how to do *anything* with
- "doors" example:
  - doors can be pushed, pulled, slid
  - "user" is faced with two questions:
    - which direction does the door move?

    - on which side do you operate the door (to make it move)?
- Common Design Problems
  - poor instructions
  - poor "visibility" (see ahead) of the operation of the technology (i.e., lacking appropriate visual cues for how to use something)
  - lack of "feedback" (see ahead)
- When simple things need instructions, the design has failed!

Norman: Everyday Things

- There are somewhere between 20,000 and 30,000 everyday things!
- These include:
  - lights, sockets, screws, watches, etc.
  - writing devices / utensils
  - fasteners (e.g., on clothing, like buttons, zippers, snaps)
  - furniture
  - food utensils (e.g., fork, knife, chopsticks, spatula, egg beater)
  - tools (e.g., hammer, screwdriver)
- How many parts do everyday things have?
  e.g., Norman's stapler has 16 parts!
  How many parts do you need to know about?
- How long does it take a person to learn to use all these things?

Norman: Design Principles

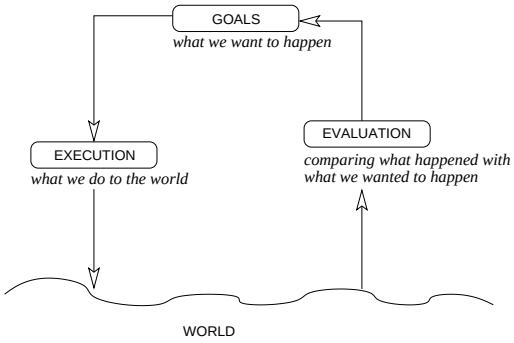
- Principles of design for *Understandability* and *Usability*
  - provide good "Conceptual Model"
  - make things "Visible"
- Visibility
  - mapping between intended actions and actual operation
  - feedback indicationg action succeeded (or didn't)
  - *natural design* ⇒ natural signs (for how to use something), should be naturally interpreted
  - be aware of cultural references (e.g., stop sign)
- Affordance
  - the perceived and actual properties of an object provide clues to the operation of the object
  - examples:

    - buttons are for pressing
    - knobs are for turning
    - slots are for inserting things into
- Conceptual Model
  - visible structure that implies:
    - affordance (what an object was designed to do)
    - constraints (what cannot be done with an object)
    - mappings (between what the object can do and how to use the object to do it)
  - "mental model" — models people have of themselves, others, devices, etc.
  - "system image" — visible part of a device
- Mapping
  - relationship between two things
  - correspondance between what you want to do and how to do it
- Feedback
  - send user information about their action
  - information can be auditory and/or visual

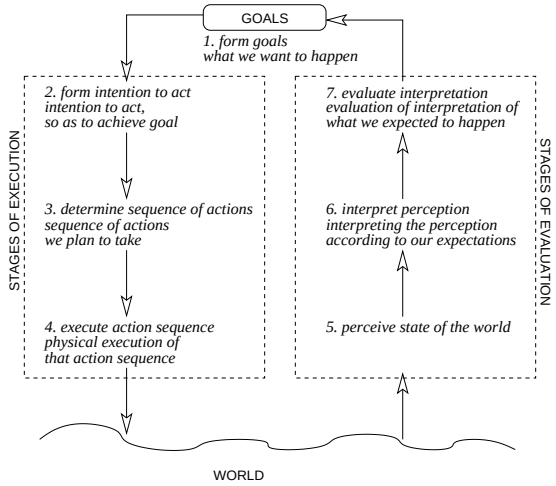
- Causality
  - response after an action indicates that the action *caused* the response
  - can we predict the effects of our actions?
  - can we figure out what happened when something goes wrong?

Norman: Activity Theory

- Action Cycle:



- Seven Stages of Action



Norman: Gulfs Execution and Evaluation

- Gulf of Execution
  - difference between user's intention and allowable actions
  - how well does a system allow a user to perform their intended actions?
- Gulf of Evaluation
  - amount of effort required to interpret state of the system and how well the user's intention has been met
  - how easy is it to get information about the system and interpret that information, and match that interpretation with the user's expectations?

Shneiderman and Plaisant: Theories

- User Interface (UI) design theories can be categorized in either of the following ways:
  - *descriptive* vs *explanatory* vs *prescriptive* vs *predictive*
  - *motor* vs *perceptual* vs *cognitive*
- descriptive = provide consistent terminology
- explanatory = describe sequence of events, cause and effect
- prescriptive = give guidance
- predictive = evaluate design's execution time, error rates, trust level, conversion rates; depends on type of user (novice, expert); these values can have high standard deviations across user types
- motor = pointing with mouse, using keyboard; theories such as Fitt's Law (to be discussed later in the term) desribe human capabilities
- perceptual = finding items on a display
- cognitive = planning sequences of actions; memory load requirements

- good "scent" $\Rightarrow$ which way to go; helps user find what they are looking for
- "taxonomy" = classification of complex set of ideas/phenomena/actions into categories
- Challenges:
  - Theories should be more central to research and practice.
  - The power of a good theory si to shape design practice.
  - Theories should lead practice, rather than drag behind.
- Theories:
  - Design-by-levels (Foley)
    natural decomposition of a system
    - conceptual: user's (mental) model
    - semantic: meanings conveyed by interface design
    - syntactic: how users conveys semantic meaning
    - lexical: hardware, device dependent
  - GOMS (Card)
    - Goals: user's goals
    - Operators: actions

    - Methods: decomposed actions
    - Selection rules: how to select between methods
  - Stages-of-action (Norman)
    activity theory, described earlier
  - Context (Suchman)
    "situated action"

Shneiderman and Plaisant: Principles

- Determine user's skill level
  - novice or first-time user
  - knowledgable intermittent user
  - expert frequeny user
- Identify the user's tasks
  - high-level, mid-level, low-level tasks ⇒ *Task Analysis*
  - reduce frequency of actions
- Choose interaction style
  - direct manipulation
  - menu selection
  - form fill-in
  - command language
  - natural language

- Eight rules of interface design
  1. Strive for consistency.
  2. Cater to universal usability.
  3. Offer informative feedback.
  4. Design dialogs to yield closure.
  5. Prevent errors.
  6. Permit easy reversal of actions.
  7. Support internal locus of control.
  8. Reduce short-term memory load.
- Prevent errors
  - correct actions
  - complete sequences
- Ensure human control while increasing automation

Shneiderman and Plaisant: Guidelines

- Guidelines:
  - provide *consistency*
  - promote *best practices*
- Navigating the interface
  - a good interface should "reduce the user's workload"
  - some examples:
    - Standardize task sequences
    - Ensure embedded links are descriptive
    - Use unique and descriptive headings
    - Use radio buttons for mutually exclusive choices
    - Develop pages that will print properly
    - Use thumbnail images to preview larger images
  - consider *accessiblity*:
    - Use text alternatives
    - Provide alternate ways of conveying time-based media (e.g., movies, animations)

    - Make elements distinguishable from one another
    - Design for predictability
- Organizing the display
  - Consistency of data display (e.g., colors, fonts, capitals, etc)
  - Efficient information assimilation by user
  - Minimal memory load on user
  - Compatible data display with data entry
  - Flexibility of user controls for data display
- Getting user's attention
  - intensity (e.g., brightness of color)
  - markings
  - size
  - fonts
  - inverse video
  - blinking (use sparingly)
  - color
  - audio

- Facilitating data entry
  - consistent data entry transactions
  - minimal input actions
  - minimal memory load
  - compatible data entry and display
  - flexible user control for display

to do

- read the first handout: Norman *The Psychopathology of Everyday Things* (from the first class)
- read the second handout: Shneiderman and Plaisant *Guidelines, principles and theories* (from today's class)