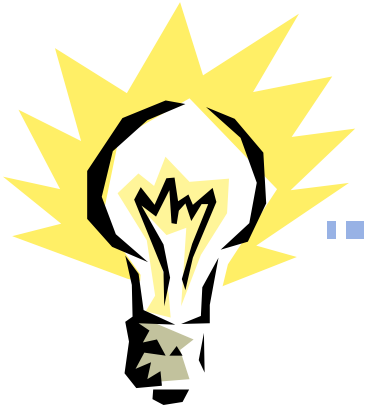


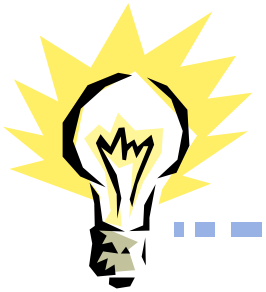
User Centered Design (UCD) Process

Avoid Bad Design, Use UCD

Evidence-based Design

Hypothesis testing!

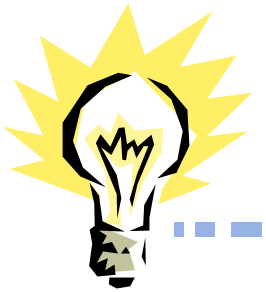
Fall 2018



Good Design (reminder!)

"Every designer wants to build a high-quality interactive system that is admired by colleagues, celebrated by users, circulated widely, and imitated frequently." (Shneiderman, 1992, p.7)

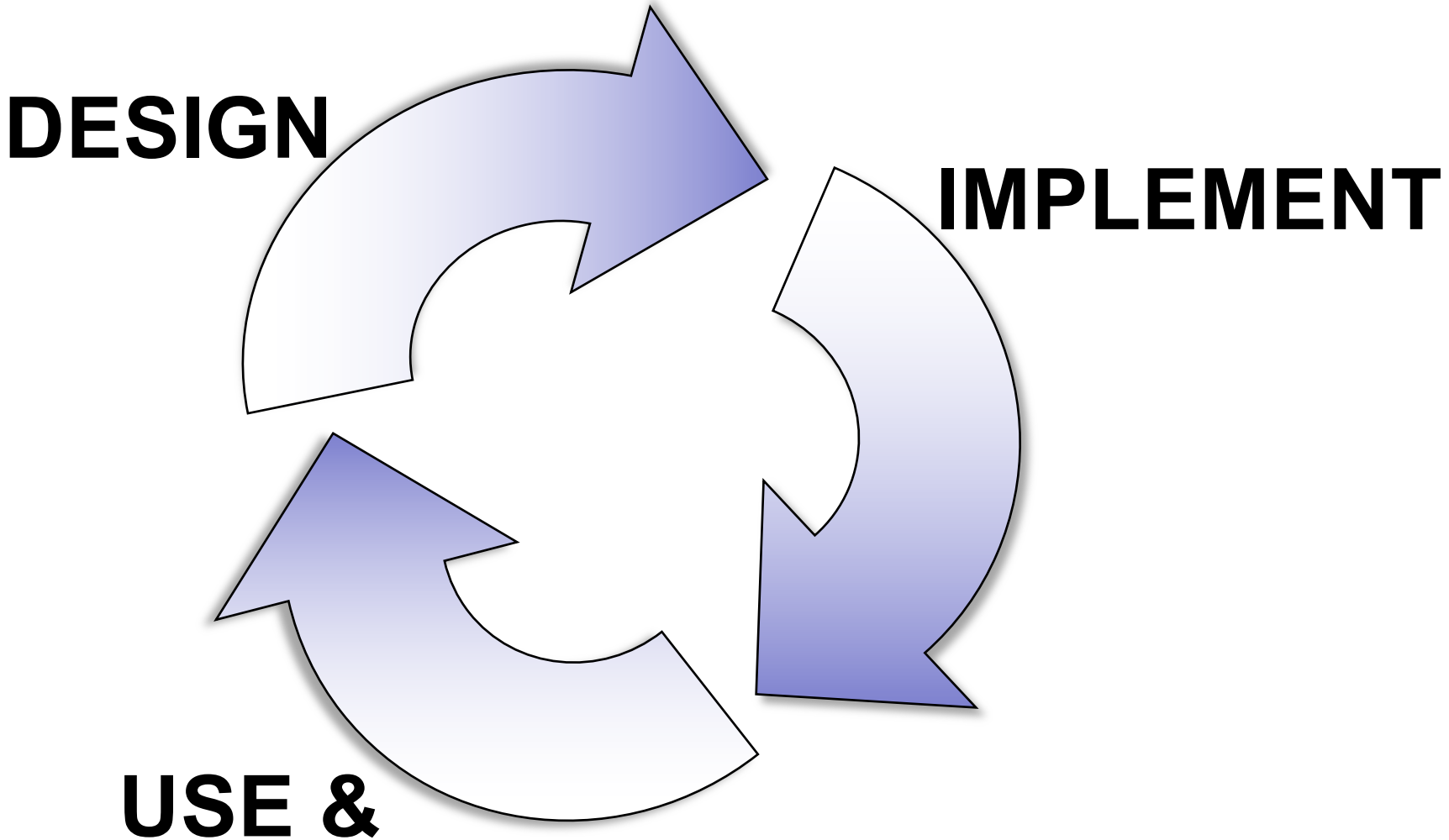
*…and **ALMOST** anything goes!…*

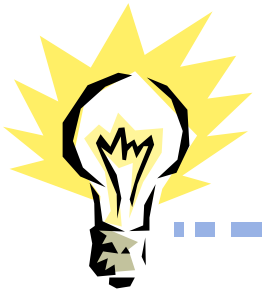


User Centered Design

- A way to force yourself to identify and consider the relevant human factors in your design
- Helps reduce the number of decisions made out of the blue, and helps focus design activities
- Helps document and defend decisions that may be reviewed later

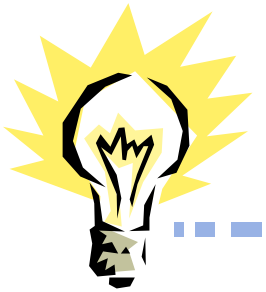
The Tao of UCD





UCD: 9 Step Overview

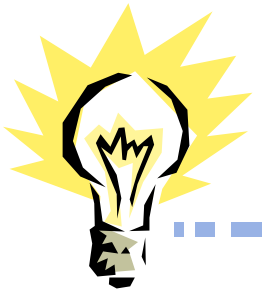
1. Define the Context
2. Describe the User
3. Needs Analysis and Task Analysis
4. Function Allocation & Information Architecture
5. System Layout / Basic Design
6. Mockups & Prototypes
7. Design Evaluation
8. *Iterative* Test & Redesign
9. Updates & Maintenance



Design Implications

- At each stage, consider how the details of your discovery process affect your design

Finding/Data	**Design Implications**
Users 16-80 yrs	Range of text sizes Range of grip strength
Some French speakers	Multilingual interface
Astronaut users	Extensive training available
Military context	Aesthetics less of an issue Ruggedness is critical



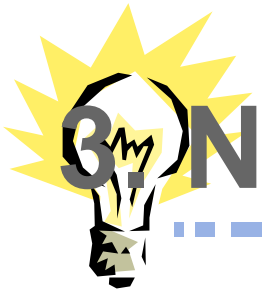
1. Define the Context

- Context: the "type" of uses, applications
  - Life critical systems, applications
  - Industrial, commercial, military, scientific, consumer
  - Office, home, entertainment
  - Exploratory, creative, cooperative
- Market
- Customer (not the same as the User)

…Design Impacts?…

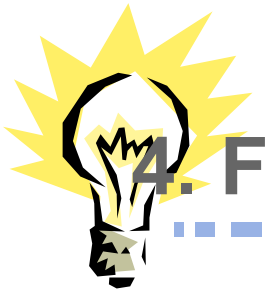
2. Describe the User (!!)

- Physical attributes
  (age, gender, size, reach, visual angles, etc…)
- Perceptual abilities
  (hearing, vision, heat sensitivity…)
- Cognitive abilities
  (memory span, reading level, musical training, ma…)
- Physical work places
  (table height, sound levels, lighting, software version…)
- Personality and social traits
  (likes, dislikes, preferences, patience…)
- Cultural and international diversity
  (languages, dialog box flow, symbols…)
- Special populations, (dis)abilities



3. Needs Analysis & Task Analysis

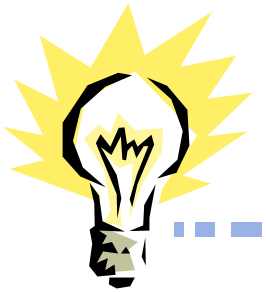
- Interviews, surveys, wants&needs study, field studies, etc.
- Talk to and observe **users** (NOT customers) doing what they do
- List each and every TASK
- Break tasks down into STEPS
- ★ABSTRACT into standard tasks
  (monitor, diagnose, predict, control, inspect, transmit, receive, decide, calculate, store, choose, operate, etc.)



4. Function Allocation & Information Architecture

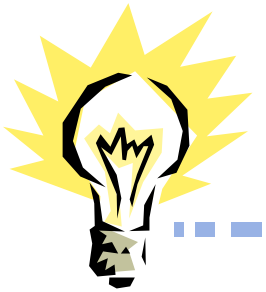
- Decide who or what is best suited to perform each task (or each step)
  - e.g., system remembers login id, and reminds the user, but user remembers the password
- Base this on knowledge of system hardware, software, users' abilities, culture, comm. protocols, privacy, etc.
- Allocation constraints: Effectiveness; Cognitive/affective; Cost; Mandatory
- Information Architecture
  - Determine information inputs and outputs

…Don't forget the design implications!…



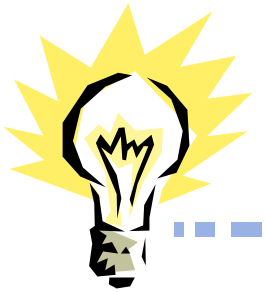
5. System Layout / Basic Design

- Summary of the components and their basic design
- Cross-check with any Requirements Documents; Human Factors refs; Hardware specs; Budgets; Laws (ADA); etc.
- Ensure that the system will support the design and comply with constraints
- (Verification and Validation, in the language of software engineering)



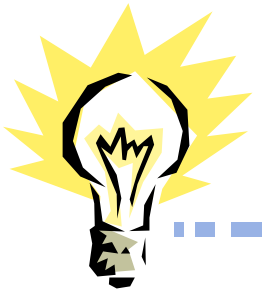
6. Mockups & Prototypes

- "Informed Brainstorming"
- Widely divergent, but still compliant
- RAPIDLY mock up the user interfaces for testing with real people
- Pen and paper or whiteboard to start
- Iterate, iterate, iterate!!
- Increasingly functional & veridical
  - List audio & visual details at same levels of detail in the prototypes



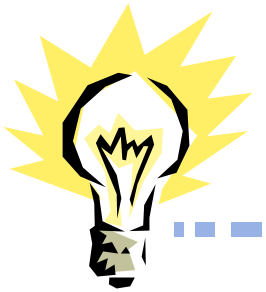
7. Design Evaluation

- Get real (or representative) users to do what they do, using the prototypes
- Subjective and objective feedback. Sometimes users “want” features that actually yield poor performance
- Video, screengrabs, lots of notes, etc.
- Be rigorous wherever possible (stats, etc.)
- Feedback into the iterative evaluation & redesign of the system
- “Discount” usability testing can be very effective, using fewer subjects, more rapid results



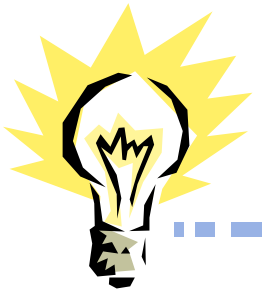
8. *Iterative* Test & Redesign

- Repeat cycles of testing and reworking the system, subject to cost/time constraints
- Focus on Functionality First !
- Plan for several versions during development



9. Updates & Maintenance

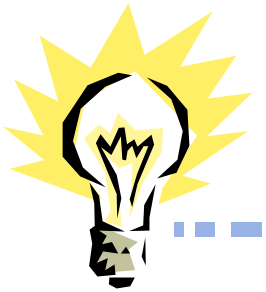
- Sustainability of system
- In-the-field feedback, telemetry, user data, logs, surveys, etc.
- Analyze and make iterative redesign/test recommendations
- Updates and maintenance plan as part of the design!
  - (design it so it can be fixed or updated)



UCD: 9 Step Overview

1. Define the Context
2. Describe the User
3. Needs Analysis & Task Analysis
4. Function Allocation & Information Architecture
5. System Layout / Basic Design
6. Mockups & Prototypes
7. Design Evaluation
8. *Iterative* Test & Redesign
9. Updates & Maintenance

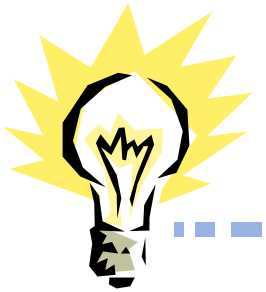
Design Implications?!!



UCD: Focusing Your Efforts

- There are real-world constraints
- Cutting out steps is not the way to economize!
- Optimize the efficiency of each step

- *Here:* Focus on the context and the user, to get the most value for the time spent



Concepts, Principles, Guidelines

- Remember…
  - No “cookbooks” (sorry!)
  - No simple, universal checklists
  - Think from perspective of user
  - There are many concepts, principles, and guidelines to help you
  - Focus on higher level principles that apply across situations, display types, etc.

*…and (**almost**) anything goes…*