Working as part of a group

see McCarthy, Dynamics of Software Development

- establish a shared vision
  - what is OOGA? what can we add?
  - harmonious sense of purpose

- develop a creative environment
  - the more ideas the better, ideas are infectious
  - don’t flip the BOZO bit

- scout the future
  - what’s coming, what’s the next project
  - what new technologies will affect this project
Scheduling/Slipping

- **McCarthy page 50, Group Psyche, TEAM=SOFTWARE**
  - anything you need to know about a team can be discovered by examining the software and vice versa
  - leadership is interpersonal choreography
  - greatness results from ministrations to group psyche which is an “abstract average of individual psyches”
  - mediocrity results from neglect of group psyche

- **Slipping a schedule has no moral dimension (pp 124-145)**
  - no failure, no blame, inevitable consequence of complexity
  - don’t hide from problems
  - build from the slip, don’t destroy
  - hit the next milestone, even if redefined (“vegetate”)
Towards being a hacker

- See the hacker-faq (cps 108 web page)
  - Hackers solve problems and build things, and they believe in freedom and voluntary mutual help. To be accepted as a hacker, you have to behave as though you have this kind of attitude yourself. And to behave as though you have the attitude, you have to really believe the attitude.

- The world is full of fascinating problems
  - no one should have to solve the same problem twice
  - boredom and drudgery are evil
  - freedom is good
  - attitude is no substitute for competence

You may not work to get reputation, but the reputation is a real payment with consequences if you do the job well.
Aside: ethics of software

- What is intellectual property, why is it important?
  - what about FSF, GPL, copy-left, open source, ...
  - what about money
  - what about monopolies

- What does it mean to act ethically and responsibly?
  - What is the Unix philosophy? What about protection? What about copying? What about stealing? What about borrowing?
  - No harm, no foul? Is this a legitimate philosophy?

- The future belongs to software developers/entrepreneurs
  - what can we do to ensure the world’s a good place to be?
Software Design

See Alan Cooper, *The Essentials of User Interface Design*

- who designs the software?

Implementation is view of software developer, user’s view is mental model, software designer has to bridge this gap

- Example: copy/move files in a Windows/Mac environment, what’s the difference in dragging a file/folder between two folders on the same device and dragging between devices, e.g., c: to a:? Is this a problem? To whom?

- Implications in OOGA? What’s a game? What’s a button?
Comfort with technology and mathematics

- “Show me all the first year students who live in Pegram and in Brown”
  what does “and” mean here? Does the average user understand Boolean? Does the average programmer understand Boolean? Recursion? Threads? Queues?

- How you solve a problem in your program isn’t (necessarily) how the user solves the problem, keep these distinctions clear

- “Saying that someone is ‘computer literate’ is really a euphemism meaning he has been indoctrinated and trained in the irrational and counter-intuitive way that file systems work, and once you have been properly subverted into thinking like a computer nerd, the obvious ridiculousness of the way the file system presents itself to the user doesn’t seem so foolish.”
Is what’s familiar necessarily what’s best?

- Hierarchical/cascading menus
  - what’s the purpose
  - are they good? (“not user friendly, but very software.dot.dweeb friendly”)
- What’s in the File menu? Why?
  - Graphics in menus?
- What about redundancies, e.g., menu and shortcut and toolbar
  - users become more expert
  - pedagogic vector (the program should instruct)
- What about ‘OK’, ‘Apply’, ‘Cancel’
  - what do these mean?