Today

- Quiz 9
- Estimation (cont.)

Homework

- 1st iteration (due)
- HW 4 (due)
- HW 5 (out)
Estimation questions ...

- Why worry about accurate estimates?
- Where do estimation errors come from?
- How should estimates be presented?
- What is being estimated?

Exercise ...
Why worry about accurate estimates?

IEEE/ACM Software Engineering Code of Ethics:

“Software engineers shall ensure that their products and related modifications meet the highest professional standards possible. In particular, software engineers shall, as appropriate:

3.09 **Ensure realistic quantitative estimates** of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work and provide an **uncertainty assessment** of these estimates.”
Q: What are some benefits of accurate estimates?

- improved status visibility
  - track progress by comparing planned with actual progress
  - not helpful if planned progress is “fantasy”!
- higher quality
  - put less stress on developers
  - shown to be $4 \times$ fewer defects than when under extreme schedule pressure
- better coordination
  - e.g., with other business functions, like testing, documentation, marketing, etc.
- better budgeting
  - forecast cost of project
- increased credibility
  - good for developers! (who often are blamed)
The Overestimation vs. Underestimation Debate

- overestimate ≡ estimate too higher (higher than actual)
- underestimate ≡ estimate too low (lower than actual)

Q: Which is worse?
- see below for answer

Q: What are some arguments against overestimating?

“Parkinson’s Law”:

Work expands to fill the time available for its completion

- if you give a developer 5 days and they finish in 3 ...
- ... they’ll find something to do to fill the 2 days!

⇒ so “squeeze” the schedule to avoid this

“Student Syndrome”

- given too much time developers will procrastinate until late in the project
- then have to rush to complete the work
- and probably not finish on time, or else end up with low quality product
Shorter schedules get work done faster ... i.e., create a sense of urgency

- developers say they need 6 months
- so, insist it be done in 3 months
- although it can’t realistically be done in 3 ...
  - even if done in 4–5 months that is better than 6
  - and worst case it will take 6 months

⇒ this is really arguing for underestimates

Q: What are some arguments against underestimation?

Reduced chance of on-time completion (i.e., challenged or failed)

Reduced effectiveness in project planning

- too small of team
- don’t meet any deadlines

Can shortchange upstream activities

- this leads to rework ... making project take even longer

Destructive late-project dynamics

- more status meetings with upper management
- more meetings to apologize to customers
- quick fixes / workarounds ... low quality
• overtime and team motivation issues

Q: So which are worse?

• overestimates can be an issue (but easily dealt with)

• underestimates are much worse
  – harder to deal with
  – more damaging to project
Exercise Results ...

“You never have to fear that estimates created by developers will be too pessimistic, because developers will always generate a too-optimistic schedule”

- Chris Peters, Microsoft

Developers’ estimates tend to be too low!

- single point estimates tend to be “best case” estimates
- even worst case estimates tend to be the “best worst case”
- more on this later ...