Tonight

• A few brief technical topics
• Fred Brooks video
• Course evaluations (during or after video)

Unassigned papers in course pack


Spiral model

• A risk-reduction software process
• In the waterfall model, “bad” decisions are identified very late
  Vastly increasing the cost of fixing them
• In the spiral model, the risks are explicitly identified and accounted for
  Iteratively

Reexamining the fault density: Component size connection

• Conventional wisdom is that smaller components contain proportionately fewer faults
• Hatton empirically found that medium-sized components were proportionately more reliable than large or small ones
• That is, there may be a “sweet spot” in component size with respect to faults

More

• It seems plausible, on another basis, that this sweet spot might exist
• Many system faults occur because of problems that arise at the interfaces of components
• Having a large number of small components, like in OO, may increase the number of interfaces and hence potential problems at the interfaces
• None of this should recommend building components of specific sizes
  But it should keep our minds open about conventional wisdom, among other things
Software safety

- Software safety concerns reducing the risk of harm to people, the environment, or the economy when systems containing software fail
- It is distinct from reliability, which focuses on increasing the chances that a system works properly
- It is rooted in system analysis, with a focus on software requirements and specifications
- She’s also worked more recently on human-centered design for safety

Rational design process

- A rational design process is one in which every step has a reason
- Even if a design is achieved through a muddled (not fully top-down nor bottom-up) process, it doesn’t mean there aren’t reasons
- Faking a rational design process after the fact allows revisionist history to be used to write clear documentation
- The inability to do so indicates a lack of clarity and conceptual integrity in the design

Active design reviews

- There is significant evidence that design reviews reduce errors in software systems
  - Boehm/Basili’s rule of thumb is that peer reviews catch 60% of the defects
- Active design reviews are (in essence) disciplined and structured reviews
  - Not just a bunch of geeks in a room looking over a design
  - Another Boehm/Basili rule of thumb is that perspective-based reviews catch 35% more defects than nondirected reviews

Cost

- The cost of reviews is a key question
- The cost of preparing and holding the reviews is reasonably easy to identify
- The scheduling cost to a project due to delays in scheduling reviews is less clear

Bohem/Basili rules of thumb:
IEEE Computer January 2001

- Finding and fixing a software problem after delivery is often 100 times more expensive than finding and fixing it during the requirements and design phase
- Current software projects spend about 40-50% of their effort on avoidable rework
- About 80% of avoidable rework comes from 20 percent of the defects
- About 80% of the defects come from 20 percent of the modules, and about half the modules are defect free
- About 90% of the downtime comes from, at most, 10 percent of the defects
- Disciplined personal practices can reduce defect introduction rates by up to 75%
- All other things being equal, it costs 50% more per source instruction to develop high-dependability software products than to develop low-dependability software products. However, the investment is more than worth it if the project involves significant operations and maintenance costs
- About 40-50% of user programs contain nontrivial defects
Comments? Questions?
If not, let’s roll the movie...

Thanks for the quarter...