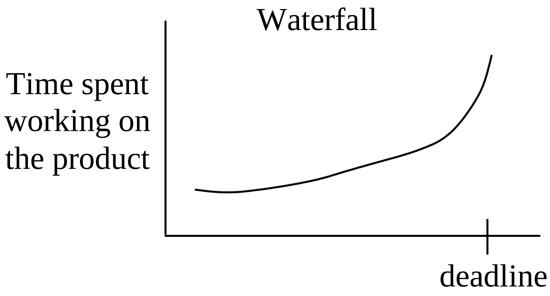
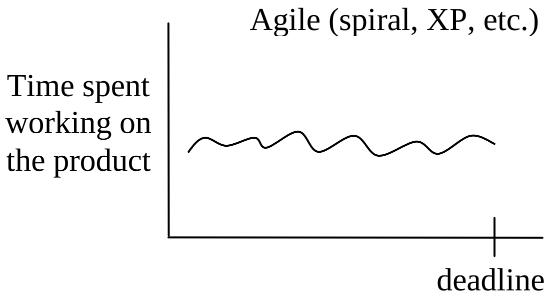
Section 07: Discussion Points

(covered in 45 minutes)

Issues ahead of the Beta Release:

- Time distribution in projects developed under the waterfall process vs. in projects that use agile processes (spiral, XP, etc.)





Refer back to 'Speech Acts' handout:

"Who will do it (the producer)? This is a person, *not* a team."
"A request is not completed until: 1. The producer has informed the customer that it is done..."

- Have each team elect a point-of-contact person, responsible for sending us a notification when their submission is complete.
- When they are ready, I write down the contact person's name and team affiliation.

Who makes the important product-wide decisions in your team?

(One person, or all by unanimous consent, or all using the Roman Rule, or ...?) Is this an unspoken or an explicit agreement among team members?

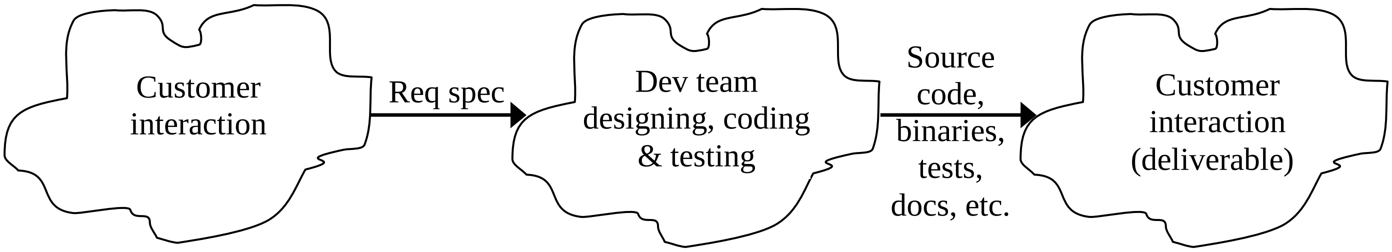
- Managerial structures:
  - Dominion model
    - Pros: clear chain of responsibility; people are used to it
    - Cons: single point of failure at the commander; less or no sense of ownership by everyone
  - Communion model
    - Pros: a community of leaders, each in his/her own domain; inherent sense of ownership
    - Cons: people aren't used to it as much (and this scares them)
- Both (communion and dominion) models allow for the possibility of having a point-of-contact person, responsible for completing a particular task.

Comments on the post-LCA meetings and discussions with Doug and me:

- What did they find useful? (Write down answers.)
- What did they find missing? (Write down answers.)
  - If anything, why did they not raise the issue/question during the meeting?
- What we found useful:
  - Getting up-to-date on how the projects are moving and what their perceived current challenges are.
  - Getting to know students a little better in an informal atmosphere.
  - Seeing an impromptu demo from the Grocery team (on the pocket PCs). It was perfectly okay that some functionality wasn't working; we loved it.
- What we found missing:
  - No team sent us up-to-date docs to look at prior to the meetings. It would have helped you to use the meetings as an opportunity to gather feedback from us, rather than as a Q&A session led by us.
    - Use us more for usability evaluations; do not just try it on fellow students in the lab.
    - One of the biggest risks is that what you build may not be what the customer wants.
      - One relevant measure of success is how much value your product brings to the customer.
  - Very few students took notes (Doug did).
  - Experience and intuition tells me that student teams almost always stumble on the first step. This is okay -- that's why there is a beta release -- so that you can improve the process before the final release. But I only heard one team (out of 6) that openly expressed a concern about a slipping schedule.

- Other things to note:
  - All students sat on the opposite side of the table from Doug.
    - A sense of *power relationship* despite the intent for informality.
    - We are there to help you; we are not out to get you.
      - "Since you've paid the tuition, you might as well take the lesson."
    - There will always be a power relationship with your bosses (e.g., they determine who gets what raise when), but that does not mean that you cannot talk openly to them, off the record.

Waterfall and Outsourcing


Customer interaction
Req spec
Dev team designing, coding & testing
Source code, binaries, tests, docs, etc.
Customer interaction (deliverable)

- Software development is not about specs slipped under your door and a CD with the source code and binaries coming out in a few months. That would be the waterfall process model, and in today's dynamic and competitive business environment it would likely result in a product failure. ("I feel so much better since I've given up hope.")
  - If waterfall were possible, most software companies would already be in China and India, because once the spec is out and in the hands of programmers, management can go on an extended Hawaii vacation. Or can't they?
  - Communication and frequent feedback from all stakeholders is the key to keeping your job. These things are hard to outsource. The moment it becomes unnecessary, your job goes away because it would be significantly cheaper (~3 times) to move it overseas.
  - The hard part is communicating with customers and peers, and keeping them all satisfied throughout the process of producing software.
    - Even with outsourcing, this raises the cost of software development to at least 90% of what it would be without having geographically distributed teams.
- Outsourcing and offshoring
  - Discuss the difference between outsourcing and offshoring.
  - "Outsourcing abroad has proved profitable primarily for jobs that can be routinized and sharply defined. For the foreseeable future, most high-value work will require creative interaction among employees, interaction that is facilitated by physical proximity, personal contact, and shared cultural experiences."
    - Only 7-10% of content delivered with no face-to-face contact gets across to the audience. The rest is communicated via body language or is lost.