Today

• Quiz 6
• Exam Overview
• Estimation (cont.)
• Proj. Planning (start)

Reading

• Ch 3: 69–99
Exam Overview

Basics

- next Thursday
- closed book and note
- done individually
- around 7–8 questions
- worth 10% of final grade

Topics

- everything is fair game
- issues in soft. eng.
- projects vs. process vs. products
- scope, time, cost, quality
- classic mistakes
- software development processes
- requirements
- estimation and planning (including today)
**Terminology**

**Estimate**
- a prediction of how long a project will take
- usually at a certain cost, scope, and quality

**Target**
- an external (business) objective
- “We need version 2.1 ready for a tradeshow in May”
- “We need the release before the holiday sales cycle”

**Commitment**
- a promise to deliver product by a specific date
- also can include agree to a specific scope, cost, and quality
Commitments vs. estimates may:

- be the same ... or be more aggressive or conservative
- the commitment though should be based on the estimate

Problems arise when there is a large gap between

- the target (or commitment)
- and the actual effort needed to complete the project
- ideally within 20% of each other (to adjust scope, team size, schedule)

Successful projects do not need perfectly accurate estimates

- this is good, but “exactimation” is not realistic

Instead ...

- need good estimates
- realistic target setting
- and good planning and control
Milestone 1.0

The first major release of a software product

- will be delivered to the customer

In Milestone 1.0

- not all stories will (usually) be implemented
- work with customer, make sure stories prioritized
- stories not nec. being dropped ... just deferred to later milestones

Defining milestones:

- take all the stories
- order by priority
- work with customer to select top-priority stories for milestone
- sanity-check your milestone estimate

For estimates, we need to factor in the number of developers ...
**First approach** at estimating milestones

- if your milestone estimate is 273 dev days of work
- and you have 3 developers:
  - $273 \text{ dev days} / 3 \text{ devs} = 91 \text{ dev days}$
  
  Q: Is this realistic as a commitment?
  - hint: think of the days on a calendar!!!
  - don’t want to sign up for working weekends (at least yet)

**Second approach** at estimating milestones

- lets say your target is 90 (calendar) days
- there are only **60 developer days** in 90 calendar days
  
  - note lower than the book ...

  \[
  \begin{array}{ccccccc}
  M & T & W & Th & F & Sa & Su \\
  3 & 4 & 5 & 6 & 7 & 8 & 9 \\
  10 & 11 & 12 & 13 & 14 & 15 & 16 \\
  17 & 18 & 19 & 20 & 21 & 22 & 23 \\
  24 & 25 & 26 & 27 & 28 & 29 & 30 \\
  \end{array}
  \]

  - $273 \text{ dev days} - (3 \text{ devs} \times 60 \text{ dev days}) = 93 \text{ dev days}$
  - we’re short 93 dev days (more than 20%)!!!

Q: what should we do here?

- get more developers (cost)
- extend target (time)
- reduce stories (scope)
Third approach at estimating milestones

Q: We still have a problem! ... What is it?

- assuming everyone works 100% of the time every day
  - vacation, paperwork, sickness, software/hardware updates, etc.
- we’re also assuming our estimates reflect actual development times
- we need to account for our teams “velocity”
  - in general, a measure of how “productive” the team is
  - depends on past estimates and actual results

Velocity

- measured as percentage of productive work
- determined from past iterations or entire projects
  - Velocity = EstimatedDaysOfWork / DaysRequired
- using velocity, given \( n \) days, how many are productive days?
  - DaysRequired = EstimatedDaysOfWork / Velocity
- for new projects/teams, you have to guess velocity
  - e.g., the book suggests 70%
- notice that DaysRequired keeps on growing (actual time is shrinking)!
  - 30 calendar days = 20 days of work
  - 20 work days = 14 productive days (for \( v = 70\% \))
Milestone 1.0

What **typically** happens ...

- milestone estimate is **much longer** than the target
- go back to the user stories (scope)
  - cut out more functionality (remove and/or reduce stories)
  - focus on **baseline** functionality (what is needed for working version)

Planning Milestone 1.0

- a series of iterations (called **sprints** in **scrum**)
- usually short ... 1 week to 1 month
- each sprint has a goal (in terms of functionality)
- after each iteration something to show the customer (for feedback)
- when planning ... focus only on first couple of iterations
  - pick stories for first and second iterations
  - save the rest for later
- “**product backlog**”: the current milestone stories
- “**sprint backlog**”: the stories in the current iteration
Q: What is the tradeoff between short and long iterations?

- short iterations:
  - more visibility
  - short feedback cycles
  - less time spent going in wrong direction
  - learn and improve as a team faster

- long iterations:
  - less overhead (planning, meetings, demos, etc)
  - developers can build momentum
  - can tackle bigger pieces of project
  - more time to recover from problems

Who decides what?

- project team (developers) decides how many stories to include in the sprint
  - based on velocity and story estimates
  - with help from the customer

- the customer
  - can reprioritize stories (e.g., to bump up stories)
  - can change the scope (e.g., reduce scope of a story)
  - can split a story and reprioritize pieces
Defining Tasks

Once sprint backlog created, break stories into tasks

- a task is a unit of work to get a story implemented
- a task alone is a non-deliverable
  - stories are deliverables since user cares about them
  - user doesn’t really care about tasks
  - tasks are what developers work on
- tasks are usually carried out by one developer
  - OK to team up for your projects ...

Stories fit on 3x5 cards, tasks on post-it notes

- title + terse description
- an estimate ... using planning poker
- each one is like a technical “todo” item
- include design and test tasks

Task estimates are usually more accurate

- summing task estimates gives better estimates than stories
- thus, may need to revise iteration based on task estimates
Total Task Estimate for Story is 9 ½ days