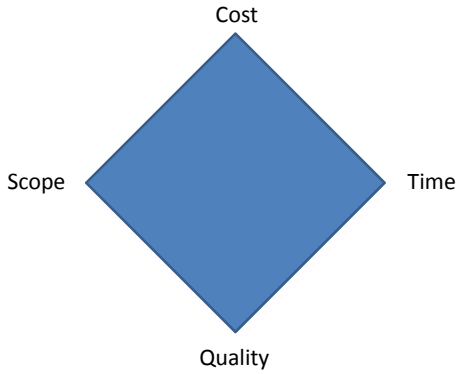
Interacting with the Client

User Stories, Planning, Estimating

The Client in XP

- Other methodologies
  - Client comes in for milestone meetings
  - Given PowerPoint presentations and demos
  - Sometimes no interaction until product done
- XP
  - Considered part of the team, but not a developer
  - Open communication required between developers and client
  - Client makes decisions but is informed about the cost of change
  - Client must make some commitment to spend time with the developers

Variables in Software Development



Client controls scope, has final say on quality
Developers in charge of technical decisions
Developers shouldn't take away client decisions (Project Greenlight example)

Relationship

- The client may be tempted to make comments or suggestions about the code, but this is not their role
  - (exception for this class, with the "professor" hat on, but will try not to overstep bounds)
- Client more like a black-box tester

Planning

- Meeting to Determine User Stories
- User Stories written on 3x5” cards, initially with just:
  - Title
  - Description
- Stories can be fairly general (unlike detailed requirements) but the client needs to be able to determine if a user story is completed successfully or not
- Fuzzy user stories make you uncomfortable?
  - Must be phrased in terminology the client understands
  - E.g. “persistence requirement” may not make sense to them

User Stories

- Interview client to collect user stories.
- Write down stories on index cards and read back to the client
  - If possible have the client write them down as well
  - Mutual process between developers and client to come up with the stories; suggestions can come from both sides but the client has the final say
- When you have enough to get going on the project, stop.
  - New stories can be added any time during the project
- Have the client rank the stories in order of importance.
  - Many rankings, one is: Critical, Important, Nice to Have, Distant Future

In-Class Exercise

- I am the client and all of you are the development team
- Help develop user stories for a web-based grade calculator

Analysis and Estimation

- You now have a stack of user stories
- Identify stories that require clarification
- Estimate coding time required for each story, but not in actual time, but in “units”.
  - Joshua Kerievsky uses NUTs: Nebulous Units of Time
  - Idea is to convey the relative sizes of stories
  - Tough to do because you don’t know what units represent until a few iterations are done, but they will shape up as time goes on

Questions to Yourself for Each User Story

- Do you understand what is being described?
- Can you estimate the story?
  - Can be a failsafe for the previous question; sometimes the process of estimating means you don't quite understand what's being described
- Is there a way for the client to test if you've finished the story?
- Is the story too large?
  - May be possible to break up into smaller stories, each of which is estimated and prioritized

In-Class Exercise

- Estimate units for grade calculator user stories

Determining Workload

- Clients are initially not happy to get estimates in terms of units
  - Client: What's a unit?
  - Developers: We don't know.
  - Client: How many units can you do this week?
  - Developers: We don't know, but we can make an initial estimate, and it will get better every iteration and even within an iteration.
- If you estimated 1 unit for story 1, and it took 3 hours, then story 2 at 2 units would now have an estimate of 6 hours
- If you estimated 3 hours/unit the first iteration but it really took 6 hours/unit then you can generate a better estimate for the second iteration
  - Project spike useful here to get an initial estimate

Estimating NUTs

- Estimate for the first iteration how many person-hours the group can collectively commit per week
  - Remember coding must be done in pairs
  - Allow for time when you're not coding and not working
  - Make an estimate; the next one can be better
- Estimate time for unit
- Go to client and say how many units you can do per week
  - E.g. if 1 unit/hour and 20 person hours then you can do 20 units per week

Client Reevaluation

- Give client the user stories, estimates, and the total number of units you can do per week
- Client gets to pick the stories that add up to the total number of units
- Client doesn't get to add more stories beyond the total number of units
  - Important not to let the client get away with this, remind the client they can do different stories the next iteration
  - Have to prioritize and drop something if another being added

In-Class Exercise

- Estimate total hours, units per week for the grade calculator
- Client to prioritize

Choosing and Estimating Tasks

- You now have prioritized stories for the iteration
- Next job is to break the user stories into tasks; discrete steps to complete the story
  - E.g. to save a document, you may have the task of creating the GUI to initiate the task, another task for the disk operation
- Members of your group bid for tasks with an estimate of units; must add up to the total units they committed to for that week
  - Low bid wins
- Tasks aren't shared with the client

Tasks

- In defining tasks it may happen that you underestimated the user story and it will take more units than you though
  - You need to go back to the client and ask to reduce the total so it is within the estimated number of units you have
  - Not pleasant but the alternative is to press forward and pretend you can do more than you can; better early notification than late

Pairing and Tasks

- You may have signed up for 10 hours worth of tasks, but since you are paired those 10 hours will include working on someone else's tasks
- Don't worry, keep track of how many units you completed so you know how many to sign up for next week

In-Class Exercise

- Break grade calculator stories up into tasks
- Bid on tasks with number of units

Dealing with Disappointment

- After a week perhaps you see your estimates weren't accurate
  - Usually programmers underestimate the time required
  - Reassess where you are with your group and immediately go to the client so he or she can determine how you should spend your remaining time
- Sometimes this is good news
  - If you only got 30 hours/week in and you estimated 40, then you have better data for the next iteration
  - Estimates should get better each iteration; "surprises" are early, not later

Rinse and Repeat

- Even if you didn't complete as many stories as estimated the first iteration, the client should be happy with your honesty
- As the project progresses you should get better at knowing what you can do in an iteration
- Continue to keep the client informed and track where you are at all times
- Client may be unhappy the product is going slowly, but it's hard to argue with the data you are gathering and sharing

Communication

- Use the ProjectPier site to share information with your team members
- Good place to keep track of
  - Meeting notes
  - Issues or problems
  - Assigned tasks, estimates, actual time taken
    - Compare with actual time

Rules

1. The developers will be truthful in their estimates and the customers will believe these estimates
2. The developers will refine their estimates and the customers will refine their expectations based on the actual achievements in each iteration
3. During the iteration the developers will update the client as to the progress of the iteration. The client will use this information to quickly refine what is required in the current iteration.