Today’s Lecture

• Discuss aspects of the Extreme Programming Model
  – As presented in “Extreme Programming Explained: Embrace Change” by Kent Beck
  – Why “Extreme”?
    • Extreme Programming (XP) takes commonsense principles and practices to extreme levels

The Basic Problem: Risk

• Beck argues that “risk” is the main problem of software development
  – Schedule slips
  – Project canceled
  – Business Changes
  – Staff Turnovers
• XP is a methodology that “addresses risk at all levels of the development process”

Four “Control” Variables

• Beck defines four control variables in software development
  – Cost
  – Time
  – Quality
  – Scope
• External forces get to pick the values of any three variables; the development team picks the value of the fourth
Four Variables, cont.

• Beck argues that the values of all four variables need to be “visible”
  – If stakeholders can see all four variables they can consciously choose which variables to control
  – If they do not like the resulting value of the fourth variable, they can choose to change the inputs or choose to control a different set of three

“Scope” is Important

• Beck argues that “scope” is the most important of the four
  – By adjusting project scope based on the values of the other three, you increase your chance of success
• This perspective is backed by XP practices
  – Practice making estimates
  – Implement most important requirements first

Cost Curve

• Cost of Change increases exponentially over time
  – its cheaper to fix a bug if its caught early in the life cycle
• XP is predicated on the notion that given the right set of practices, the cost curve can be flattened
• This is a BIG assumption and may make adoption of XP impossible for some organizations

How to Flatten the Curve?

• Technology
  – Objects
    • Used correctly they provide extreme flexibility
  – Object Databases
• Practices
  – Simple Design, Automated Tests, Refactoring
• ...
Learning to Drive

• Beck tells a story of learning to drive
  – Mom first told him “line the car up in the middle of the lane, straight toward the horizon”
    • Beck drives car off the road!
  – Mom then tells him “Driving is not about getting the car going in the right direction. Driving is about constantly paying attention, making a little correction this way, a little correction that way.”
• This is the paradigm for XP. Change is constant and must be constantly monitored and adapted to

Four “Values” underlying XP

• Communication
  – via several mediums: conversation, code, tests, metrics
• Simplicity
  – Beck says “Simplicity is not easy”
• Feedback
  – Tests as well as user feedback
• Courage
  – XP resembles a hill-climbing algorithm; you can get stuck in local optima

Basic Principles

• Rapid Feedback
• Assume Simplicity
• Incremental Change
• Embracing Change
• Quality Work

XP Practices

• The Planning Game
• Small Releases
• Metaphor
• Simple Design
• Testing
• Refactoring
• Pair Programming
• Collective Ownership
• Continuous Integration
• 40-hour Week
• On-site Customer
• Coding Standards