Agile Development Practices

Agile programming doesn't just mean doing more work with fewer people.
Outline

- Review of life cycles
- Agile processes
- Software development classic mistakes
XP

- “Features” are the control variable of the triangle
  - Can deliver less, but on time and on budget
- Development cycle (2 week period)
  - Meet with client to elicit requirements
  - Planning – break requirements into tasks
  - Implementation
    - Write tests first
    - Simplest possible design to pass the tests
    - Can refactor occasionally
  - Evaluate progress
- Expert customer is part of the team, involved in all stages; No complete up-front analysis and design

Feels a little like evolutionary prototyping
Pair Programming

WE'RE GOING TO TRY SOMETHING CALLED EXTREME PROGRAMMING.

FIRST, PICK A PARTNER. THE TWO OF YOU WILL WORK AT ONE COMPUTER FOR FORTY HOURS A WEEK.

THE NEW SYSTEM IS A MINUTE OLD AND I ALREADY HATE EVERYONE.

Copyright © 2003 United Feature Syndicate, Inc.
When to use XP

- Use for:
  - A dynamic project done in small teams
  - Projects with requirements prone to change
  - Customer is available

- Not for:
  - Requirements are truly fixed
  - Cost of late changes is very high
  - Customer is not available
Activity 1 - Review

Work with your row to describe:

- Waterfall (row 1 and 2)
- Spiral (row 3)
- Evolutionary Proto (flavor of XP) (row 4)
- Staged Delivery (row 5 and 6)

1. What it is, one advantage, one limitation
2. One example project suited to each and why
Activity 2 - Select

Which model would you choose:

1. College enrollment system (for UW)

2. Vive la Taco family restaurant online ordering system (for your family)

3. Word plugin to handle English to Spanish translation (for MS)
Agile principles

Agility is:

- **Observing** and **Adapting** to your environment. Moving faster than things that will harm your project

- Keeping up with relevant changes
  - In requirements
  - In priorities
  - In environment
  - In budget

**How Can We Make This Work?** Goal of Agility - develop software in the face of changing environment and constrained resources
Agile principles

Satisfy the customer through early and continuous delivery
  - Studies show a strong correlation between final quality and frequent deliveries of increasing functionality. More frequent == higher quality.

Welcome changing requirements, even late in development
  - Statement of attitude
  - Agile team works hard to keep the structure of software flexible

Deliver *working* software frequently
  - Working software is the primary measure of progress
Agile Principles (2)

Customers, developers, stakeholders work together daily
  - An agile project must be continuously guided

Build projects around motivated individuals
  - Other things may need to change to enable this

Promote self-organizing teams, and teams that reflect and adjust as needed

High value on face to face conversations
  - Primary mode of communication is conversation
  - Written docs are not required  (GA: yikes!)
Promote sustainable development
  - Teams work at a rate that allows them to maintain the highest quality standards for the duration

Continuous attention to technical excellence
  - High quality is the key to high speed
  - “Do no harm” creed
  - Plug for refactoring

Simplicity is essential
  - Take the simplest path that is consistent with the goals (and be confident that it will be easy to change if needed)
  - YAGNI (you ain’t gonna need it) (GA: shortsighted?)
Standish data argues for simplicity...

54% of features were rarely or never used! That effort could have been productively spent elsewhere! (1994 report)
References

Agile
- Agile processes-XP lecture, CS19 Necula, Aiken, Keutzer CS169 L3, UCB
- NetObjectives: Intro to Scrum-Based Agile Software Development 8/24/06
- Agile Processes, R.C. Martin

Classic Mistakes
- Rapid Development, 3.3, Steve McConnell
  http://stevemcconnell.com/rdenum.htm
- Survival Guide: 254-257, Steve McConnell