Writing and using software sucks

- From the users' point of view
  - Rarely does what it's intended to do
  - Does things that are not needed, bloated
  - It's not intuitive
  - Too difficult to use
  - Unsafe/Insecure
  - Crashes
  - Doesn't evolve with users' needs
  - Upgrades are infrequent / expensive

Writing and using software sucks

- From the developers' point of view
  - Process is stressful
  - Process isn't enjoyable
  - We are pressed to do more than what we can do
  - Following abstract requirements is confusing / hard / unsatisfying
  - Sometimes boring / repetitive
  - We feel that users don't like it --> no sense of pride
  - As it grows, we feel that we don't control/know it anymore
  - Afraid to make changes
  - Lonely

Why is writing good software so hard?

- Because change is an unavoidable property of software development:
  - Changing requirements
  - Changing competition
  - Changing expectations
  - Changing platforms
  - Changing schedules
  - Changing staff

XP Core Values to embrace change

- XP embraces change in the software development process with a set of four core values:
  - Communication: Let's involve the customer in the entire process. And let's encourage programmers to talk
  - Feedback: Know at all times how we're doing: is the software stable? Does it break? Is it what the customer expects?
  - Simplicity: Do not over-generalize. Design the simplest component to solve a particular need. If we need to generalize later, we'll generalize when we see the need --that is, the software will evolve with our needs.
  - Courage: Have courage to change (architecture, requirements, code)... and also let's make sure that we have a safety net below.

XP Principles

- Fine scale feedback
  - Test Driven Development
  - Planning Game
  - Whole Team
  - Pair Programming

XP Principles

- Continuous process rather than batch
  - Continuous Integration.
  - Design Improvement
  - Small Releases

XP Principles

- Shared understanding
  - Simple Design
  - System Metaphor
  - Collective Code Ownership
  - Coding Standard or Coding Conventions
- Programmer welfare
  - Sustainable Pace

Taking ideas to the eXtreme

- We have learned that code reviews are good --> review all the time
- We have learned that testing is good --> test all the time, writing tests first
- We have learned that design is good --> keep on designing all the time
- We have learned that simplicity is good --> always strive for the simplest design, without generalizing
- We have learned that short iterations are good --> make new releases very very frequently

Some eXtreme practices

- Test-driven development
  - Guides the design process
  - Keeps our tendency to over-engineer in check.
  - Unit tests stay with the system, and can be reused
  - Gives more confidence that the code is correct.
  - Gives courage to change the code: if we can re-run the tests, we'll know that the new code is correct.

Some eXtreme practices

- Pair-programming
  - Hard at the beginning, but introduces feedback.
  - Fewer errors. Higher code quality.
  - More people know the code.
  - More fun
  - Can learn (code, domain, etc.) from each other.

XP Resources On-line

- Extreme Programming, February 25, 2003, By Jay Munro
  - http://www.pcmag.com/print_article/0,1761,a=36115,00.asp]
- Extreme Programming: A gentle introduction.
  - http://www.extremeprogramming.org/
- Extreme Programming Roadmap (original WIKI)
  - http://c2.com/cgi/wiki?ExtremeProgrammingRoadmap, esp. http://c2.com/cgi/wiki?ExtremeProgrammingCorePractices
- Extreme Programming (XP) FAQ
  - http://www.jera.com/techinfo/xpfaq.html
- JUnit

XP Resources Books

- Extreme Programming Explained: Embrace Change, by Kent Beck
  - http://www.amazon.com/exec/obidos/tg/detail/-/0201616416
- Test Driven Development: By Example, by Kent Beck
  - http://www.amazon.com/exec/obidos/tg/detail/-/0321146530
- Refactoring to Patterns, by Joshua Kerievsky
  - http://www.amazon.com/exec/obidos/ASIN/0321213351

COMPUTER SCIENCE

Contact Information

- Matthew Cornell - mailto:cornell@cs.umass.edu
- Agustin Schapira - mailto:schapira@cs.umass.edu