CISC327 - Software Quality Assurance

Lecture 7
XP in Practice
XP in Practice

• Outline
  – Today we look at the actual practices of the XP process, and how they can be applied in the context of our project
  – The key ideas to keep in mind at all times are:
    • metaphor
    • simplicity
    • testing
    • automation
    • collective work
    • standards
Agile Development

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

Although there is value in the items on the right, agile software developers value the items on the left more.

http://agilemanifesto.org
XP 1: The Planning Game

• The Planning Game
  – Refers to the practice of having a continuous dialog between business and technical people on the project
  – Often in the form of weekly meetings, where business people bring business constraints, and technical people bring technical constraints
    • Business people bring issues of scope, priority, releases
    • Technical people being estimates, consequences, scheduling
  – Forces the project members to continually balance between what is possible (the technical aspects) and what is desirable (the business aspects)
    • Unfortunately we won't really be able to practice this in the project, the closest we come is our dialog in class and email
XP 2: Small Releases

• Small Releases
  – Refers to the practice of addressing only the most pressing business requirements, and getting them addressed by releasing a new version quickly
  – Means that we should bring the first version into production as quickly as possible
  – Means that we should shrink the cycle to the next version as much as possible
  – In practice this means shrinking software cycles to a month or two instead of six months or a year
    • In our project, we will shrink to quick releases at roughly two week intervals
XP 3: Metaphor

• Metaphor
  – Refers to the practice of understanding and speaking of the system in real-world terms independent of its programmed solution
  – An example of a metaphor is the "desktop" of modern operating systems
    • The goals in building such an operating system can be understood in terms of an office desk
  – The metaphor drives the design of the architecture and interfaces of the system
    • In our project, the metaphor is "native", that is, there is a natural physical understanding of what we are doing, our front end is simply a retail console
XP 4: Simplicity

• Simplicity
  – Refers to the practice of always using the simplest possible design and code that can handle the tests
  – Do not speculate or try to guess what will be needed in the future, address only the current test suite
  – Do not implement any features that do not affect the test results
    • In our project, the simplest, smallest solution will be considered the best
XP 5: Testing

• **Testing**
  – The only required program features are those for which there is an *automated test*
  – Always create tests *first*, and treat them as the goal *(specification)*
  – Programmers create *unit tests* (tests for each method or segment of code)
  – Customers create *functional (acceptance) tests* that check that the product has the required functionality
    • In our project, we will create explicit tests *first* as we go along, beginning with assignment #1, and program to meet them
• Refactoring
  – Refers to the practice of continually looking for ways to simplify the architecture and coding of the system as new features and changes are made
  – When a new feature or change is needed, we first look to see if there is a way to rearchitect the system to make it easier or simpler to add - if so, we rearchitect first
  – Once the new feature has been added or changed, we look to see if the resulting new program can be simplified by rearchitecting or merging similar code
    • In our project, we will face changes that may require refactoring
XP 7: Pair Programming

• **Pair Programming**
  – Refers to the practice of having all production code written with two people working *together* on one terminal
  – One partner works *tactically* on the specific part of the code (e.g. method) being coded at the moment
  – The other partner works *strategically*, considering higher level issues such as:
    • is this *approach* going to work?
    • can we *simplify* this by restructuring?
    • what other *tests* do we need to address here?
  – In our project, we will do all programming in *pairs*
XP 8: Collective Ownership

• Collective Ownership
  – Refers to the practice of having everyone responsible for the quality of the software, and no one to blame for failures of the software
  – Everyone is responsible for identifying opportunities to improve things and to act upon them at any time
  – No one owns the code, it belongs to everyone together - there is no notion of "my code", only the universal notion of "our code"
    • In our project, both team members will be collectively responsible for all parts of both phases
XP 9: Continuous Integration

• Continuous Integration
  – In XP, new code is always integrated and tested within a day
  – Changes are not allowed to go on without being continually tested in context to catch integration failures before they happen
• In our project, starting with assignment #2, we will model this by testing again immediately after each day's changes
XP 10: On-site Customer

• **On-site Customer**
  – A real customer must be a part of the development team at all times
  – Must be available to answer questions, resolve disputes, set short-term priorities based on business knowledge
  • In our project, we will model this by having the customer available every day by email (not quite right, but it will have to do!)
XP 11: Coding Standards

• Coding Standards
  – Project-wide conventions about the coding of programs
  – Necessary since everyone is responsible for all of the code, and may have to read or change any part of it at any time
  – Usually specifies
    • Commenting standards, e.g., every method must have a comment of the form ...
    • Naming conventions, e.g., variables representing dates will always be named ending in "Date", all constant will be named with a two letter prefix indicating their business type
  – In our project, you will be required to specify your coding standards, and they will be judged according to the clarity, readability, and consistency of your code.
Summary

• XP Practices
  – XP uses a set of standard practices that together form an easy to apply practical system for team development of software
  – Emphasis is on collective responsibility, continuous improvement, and high quality standards
  – We will try to apply these practices in the course project
Summary

• **References**
  - Beck chapter 10 (1st ed.)

• **Reading Assignment**
  - Beck chapters 11, 12 (1st ed.)

• **Next Time**
  - Introduction to *Systematic Testing*