Crowdsourcing Software Development

Greg Hilton and Chris Ellison
Carnegie Mellon University Silicon Valley
SE Practicum Project Advisor: Dr. Edward Katz
Sponsor: Codective

Concept and Vision

Vision
An environment where all software development project stakeholders can meet to discuss and share ideas, ensuring contributors are fairly rewarded for their work.

Concept
A lightweight web application as a framework on which ideas may be posted and projects hosted. Track users’ contribution and use community feedback to determine the quality of contributions.

Materials and Methods

Materials
- Ruby on Rails 2
- Sun/Oracle VirtualBox virtual machine
- Ubuntu 10 Linux

Methods
Processes based on Extreme Programming:
- Test-driven Development (TDD)
- Behavioral-driven Development (BDD)
- Short (1-week) iterations
- Rapid Client Feedback
- Pair Programming

Key Challenges
- Fluid Requirements
- Remote Client, International
- New Technologies
- Short Timescale

Related Projects
- There are other sites that attempt to draw upon the “wisdom of crowds” to build useful products:
  - Wikipedia: A crowdsourced encyclopedia, and one of the most well-known examples of crowdsourcing in action.
  - Cambrian House: Began as a software crowdsourcing community. Now produces the Chaordix crowdsourcing platform.
  - TopCoder Inc: Competitive crowdsourcing of software and UI design for cash prizes.
  - Quirky.com: Users submit ideas and designs for physical products, which are then refined and bid to manufacturers. Once a critical mass of sales has been reached, a product is deemed successful and is manufactured and sold.
  - Waze: Crowdsourced competitor to commercial turn-by-turn navigation systems.

Road Map

Course Goal
Implement a maintainable and extensible alpha version of the Codective platform. As the requirements for the platform were very unclear at the beginning of the course, we also wanted to be able to rapidly trial various options to clarify the product requirements during development.

Development Approach
- Followed most of the engineering practices described by Extreme Programming (best suited for dynamic requirements)
- Early setup of testing infrastructure
- Worked with the client to develop user stories as backlog
- Met with the clients via telecom twice a week:
  - Demonstrate progress and proof-of-concept
  - Refine development plan
- Keep feature priorities in order
- Used TDD, BDD, and pair programming practices

Next Steps
At the end of the course, the Codective platform alpha was released to the public (and can be found at http://www.codective.com). The site is available and is accepting test projects and user feedback.

Lessons Learned

What Worked Well
- Pair Programming
- Test-Driven Development
- Short Iterations
- Begin-week and end-week client meetings

What to Improve
- Set up the continuous integration framework earlier – as the chart below shows, our productivity increased greatly as we improved test coverage.
- Behavior-Driven Development – the current BDD tools are rough around the edges and require a large time investment before you see much payoff.

Quality v. Productivity

Bibliography