Agile Unified Process (UP): Introduction to an OOA/D Process

Eunjee Song
Dept. of Computer Science
Baylor University

Slide Sources: Applying UML and Patterns by C. Larman and Introduction to OOA/D Process slides by Dr. R. France

Overview

- Agile Process Overview
- Agile Unified Process (UP) Basics
- Introduction to an OOA/D Process
- Agile Process Principles

What is an agile process?

A process that continuously adapts and adjusts to
- changes derived from experiences gained during the development,
- changes in software requirements and
- changes in the development environment.

Why agile processes?

- Internet “age” characterized by rapid changes in technologies and demands
- Increased pressure to develop new services and enhance existing services quickly to gain competitive advantage
- Focus is on producing “quality” code quickly and economically

Agile UP Basics

- No detailed overall plan
  - High-level plan that outlines major outlines and estimates end-time (Phase Plan)
- Iterative, incremental, time-boxed
  - Each iteration is a mini-project
  - Detailed plan for each iteration (Iteration Plan)
  - Each iteration has a non-extensible fixed duration
- Risk-driven, quality-focused
  - Tackle high-risk, high-value issues in early iterations
  - Build a cohesive core architecture in early iterations
  - Continuously verify quality

Iterative/incremental process

- Time-boxed iterations
- Feedback from iteration N leads to refinement and adoption of the requirements and design in iteration N+1
- The system grows incrementally
- Iterations are fixed in length or timeboxed
- Requirements are refined
Phases
Activities and iterations organized around 4 major phases in UP:
- Inception
  - Early exploration of problem to determine project feasibility
  - What’s the perceived business value?
  - What are the risks?
- Elaboration
  - Requirements detailing (major requirements identified)
  - Iterative implementation of “core” architecture
- Construction
  - Iterative development of remaining low-risk elements
  - Prepare for deployment
- Transition
  - Beta tests
  - Deployment

Principles - 1
- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
Principles - 2

- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity—the art of maximizing the amount of work not done—is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Conclusions

- Some aspects of a software development project can benefit from an agile approach while others can benefit from a less-agile or more predictive approach.
- Practical software development processes can be classified along a spectrum depending on their degree of "agility".
  - At one extreme are the predictive processes in which the process steps are defined in detail early in the project, and project goals remain relatively stable throughout the execution of the process.
  - At the other end are the purely agile processes in which process steps and project goals are dynamically determined.
  - The agility of a process is determined by the degree to which a project team can dynamically adapt the process based on changes in the environment and the collective experiences of the developers.
  - Current agile processes are close to the purely agile end.