Chapter 3

Supplementary Notes

The Agile Software Process

- Implementations tend to be “customized”
- Several Agile Models:
  - XP— user stories, pair-programming, refactoring, and continuous integration, incremental delivery
  - Adaptive Software Development
    - adaptive cycle planning, time-boxing, risk-driven planning, collaborative learning, self-organizing teams
  - Dynamic Systems Development Method
    - operationalized prototyping
    - 80% deliverable in 20% of time
  - Scrum
    - backlog, sprints, scrum meetings
  - Crystal
    - a set of example agile processes, useful principles
  - Feature Driven Development
    - plan, design, and build by feature
  - Lean Software Development
  - Agile Unified Process
    - Serial in the large, iterative in the small
- In real world, hybrids abound
  - A little of this, a little of that
  - What is the project? Who are the people? When is the deadline? These (and more) are all factors in determining the right process model for the project, team, or company.

Fact/Fallacy Tidbit

- Fact 6

  New tools & techniques cause an initial *loss* of productivity and/or quality

- Discussion
  - Operational changes made today for improved productivity tomorrow
  - Learning curve causes productivity/quality loss until tool or technique is fully mastered
  - This gap poses dilemma:
    - Timing of the change (when can we do this?);
    - Evaluating expected benefits;
    - Cost to make the change;
    - Duration of learning curve (proportional to benefit);
    - Collecting metrics to evaluate decision (once fully adopted)
  - Real benefit typically between 5% and 35% (see Fact 5, next lecture)

From Robert Glass, "Facts & Fallacies of Software Engineering"