
Software Engineering: A Practitioner's Approach, 6/e

Chapter 4: Agile Development

copyright © 1996, 2001, 2005

R.S. Pressman & Associates, Inc.

The Manifesto for Agile Software Development

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

That is, while there is value in the items on the right, we value the items on the left more.”

Kent Beck et al

What is "Agility"?

- ❑ Effective (rapid and adaptive) response to change
- ❑ Effective communication among all stakeholders
- ❑ Drawing the customer onto the team
- ❑ Organizing a team so that it is in control of the work performed
- ❑ Rapid, incremental delivery of software

An Agile Process

- ❑ Is driven by customer descriptions of what is required (scenarios)
- ❑ Recognizes that plans are short-lived
- ❑ Develops software iteratively with a heavy emphasis on construction activities
- ❑ Delivers multiple 'software increments'
- ❑ Adapts as changes occur

Agile Software Development

- ❑ **Agile software** engineering represents a reasonable alternative to conventional **software** engineering for certain classes of **software** and certain types of **software** projects
- ❑ **Agile** development processes can deliver successful systems quickly
- ❑ **Agile** development stresses continuous communication and collaboration among developers and customers

Agile Software Development

- ❑ **Agile software** engineering embraces a philosophy that encourages customer satisfaction, incremental **software** delivery, small project teams (composed of **software** engineers and stakeholders), informal methods, and minimal **software** engineering work products
- ❑ **Agile software** engineering development guidelines stress on-time delivery of an operational **software** increment over analysis and design

Traits of **Agile** Team Members

- Competence
- Common focus
- Collaboration
- Decision-making ability
- Fuzzy-problem solving ability
- Mutual trust and respect
- Self-organization

Agile Process Models

- Extreme Programming (XP)
- Adaptive **Software** Development (ASD)
- Dynamic Systems Development Method (DSDM)
- Scrum
- Feature Driven Development (FDD)
- Agile** Modeling (AM)

1. Extreme Programming (XP)

The most widely used agile process, originally proposed by Kent Beck.

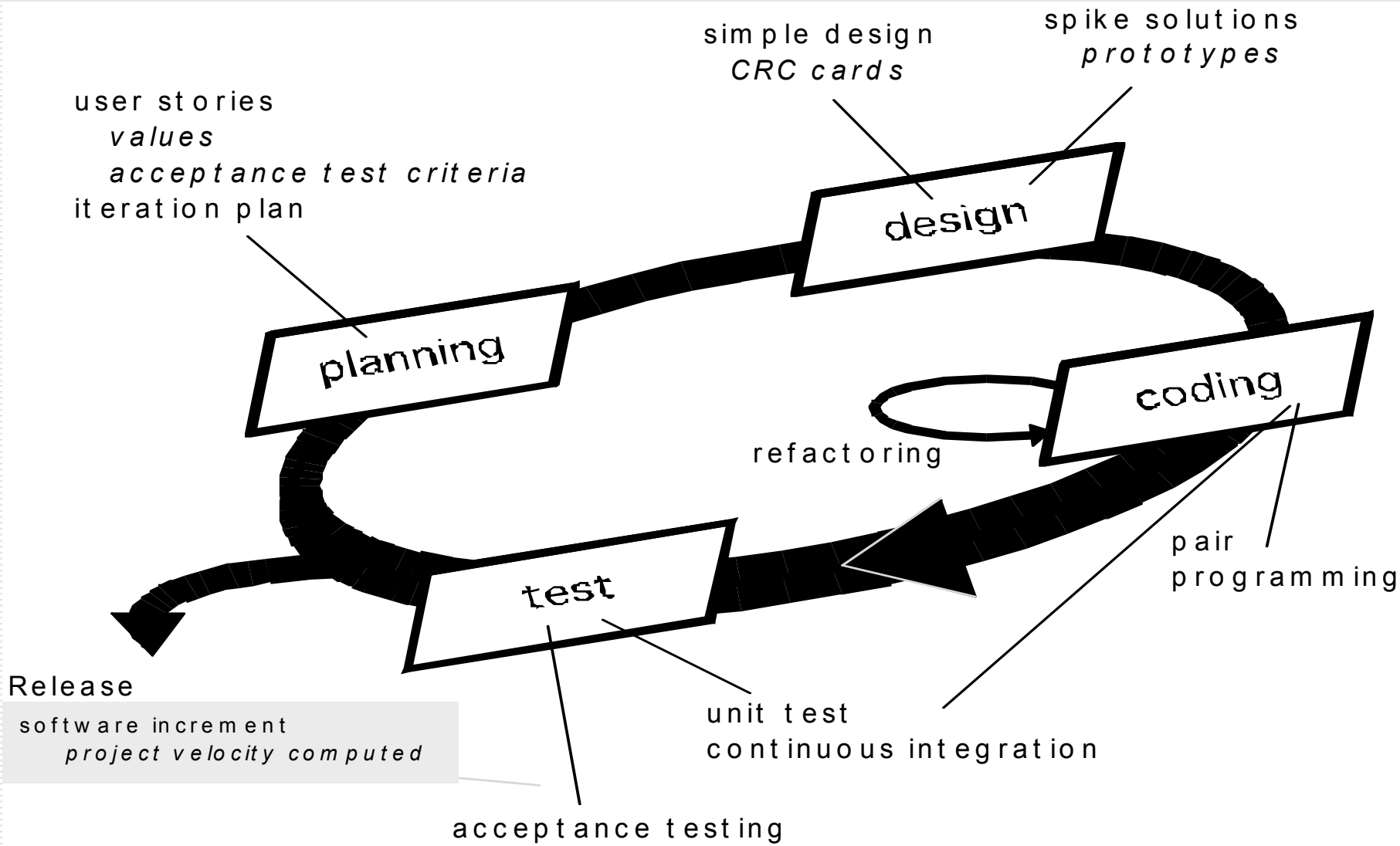
XP Activities are:

□ XP Planning

- Begins with the creation of “**user stories**”, describing features and functionalities to be built.
- Based on business value priority to stories are given
- XP Team members assess each story and assign cost (measured in development weeks)
 - More than 3 weeks, it is recommended to split the story into smaller stories
 - New stories can be written at any time
- Stories are grouped to for a **deliverable increment**
- Story with highest priority/or the riskiest one will be implemented first
- A **commitment** is made on delivery date
- After the first increment “**project velocity**” is used to help define subsequent delivery dates for other increments

Extreme Programming (XP)

- XP Design
 - Follows the *KIS principle*
 - Encourage the use of *CRC cards* (see Chapter 8)
 - For difficult design problems, suggests the creation of “*spike solutions*”—a design prototype
 - Encourages “*refactoring*”—an iterative refinement of the internal program design
- XP Coding
 - Recommends the *construction of a unit test* for a store *before* coding commences
 - Encourages “*pair programming*”
- XP Testing
 - All *unit tests* are executed daily
 - “*Acceptance tests*” are defined by the customer and executed to assess customer visible functionality



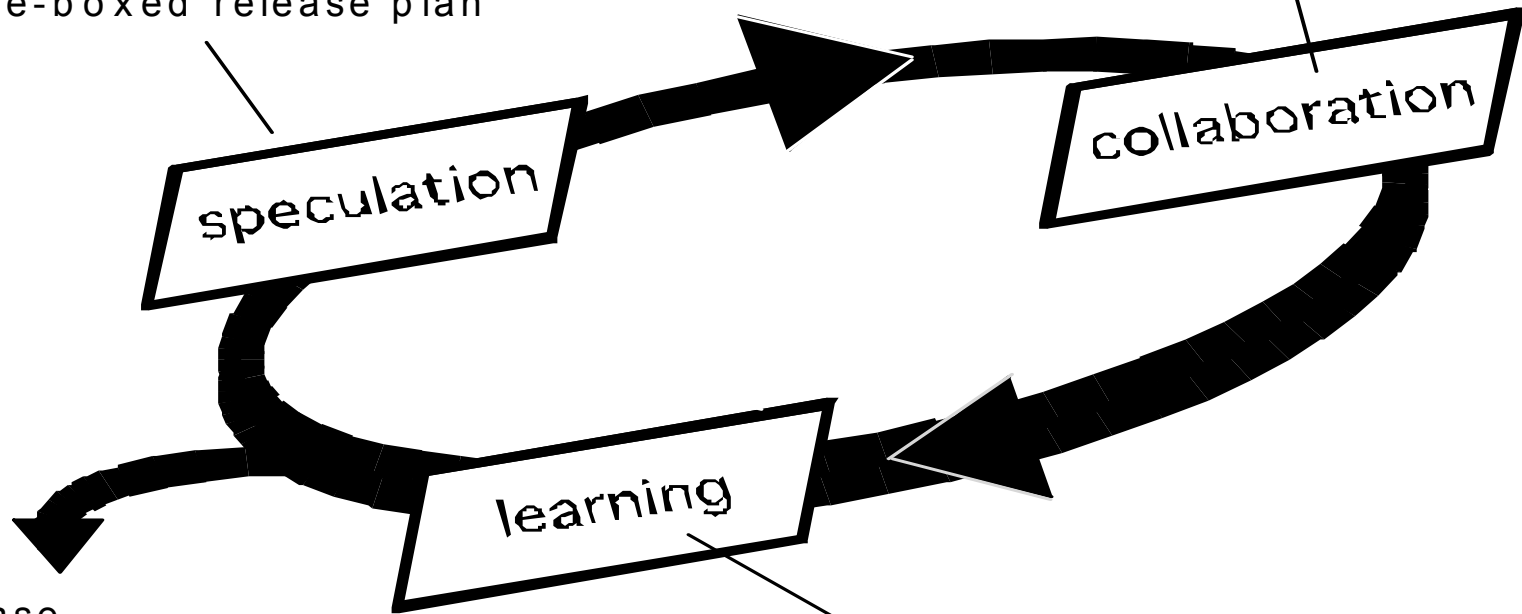
2. Adaptive Software Development

Technique for building complex software and systems!

- Self-organization arises when independent agents cooperate to create a solution to a problem that is beyond the capability of any individual agent
- ASD – distinguishing features
 - Mission-driven planning
 - Component-based focus
 - Uses “time-boxing” (See Chapter 24)
 - Explicit consideration of risks
 - Emphasizes collaboration for requirements gathering
 - Emphasizes “learning” throughout the process

adaptive cycle planning
uses mission statement
project constraints
basic requirements
time-boxed release plan

Requirements gathering
JAD
mini-specs



Release

software increment
adjustments for subsequent cycles

components implemented/ tested
focus groups for feedback
formal technical reviews
postmortems

ASD

Speculation:

- Project initiation and adaptive cycle planning.
 - Define set of release cycles for the project.

Collaboration:

- People work together must trust one another to
 - Criticize without aggression;
 - Assist without resentment;
 - Work as hard or harder as they do
 - Have skills set to contribute to the work and at hand.
 - Communicate problem or concern in a way that leads to effective action.
-

ASD -III

- Learning:
 - Emphasis on learning as much as on progress.
 - Improve level of real understanding.
 - **Focus groups** – customers/ users feedback provides a direct indication of whether or not the product is satisfying business needs.
 - **Formal technical reviews** – review software components that are developed, improving quality and learning.
 - **Postmortem** – team becomes introspective, addressing its own performance and process.
-

Dynamic System Development Method

- ❑ Provide a framework for building and maintaining systems which meet tight time constraints through the use of incremental prototyping in a controlled project environment.
 - ❑ 80 % work is delivered in 20 % time.
 - ❑ Idea borrowed from RAD model.
 - ❑ Iterative software process and each iteration follows the 80% rule.
-

Dynamic Systems Development Method

- Promoted by the DSDM Consortium (www.dsdm.org)
- DSDM—distinguishing features
 - Similar in most respects to XP and/or ASD
 - Nine guiding principles
 - Active user involvement is imperative.
 - DSDM teams must be empowered to make decisions.
 - The focus is on frequent delivery of products.
 - Fitness for business purpose is the essential criterion for acceptance of deliverables.
 - Iterative and incremental development is necessary to converge on an accurate business solution.
 - All changes during development are reversible.
 - Requirements are baselined at a high level
 - Testing is integrated throughout the life-cycle.

Dynamic Systems Development Method

Feasibility study

Establishes requirements and constraints

Business study

Establishes functional and information requirements needed to provide business value

Functional model iteration

Produces set of incremental prototypes to demonstrate functionality to customer

Design and build iteration

Revisits prototypes to ensure they provide business value for end users

may occur concurrently with functional model iteration

Implementation

Latest iteration placed in operational environment

Scrum

- Originally proposed by Schwaber and Beedle
- Scrum – distinguishing features
 - Development work is partitioned into “packets”
 - Testing and documentation are on-going as the product is constructed
 - Work occurs in “sprints” and is derived from a “backlog” of existing requirements
 - Meetings are very short and sometimes conducted without chairs
 - “demos” are delivered to the customer with the time-box allocated

Scrum

❑ Backlog

Prioritized list of requirements or features the provide business value to customer
Items can be added at any time)

❑ Sprints

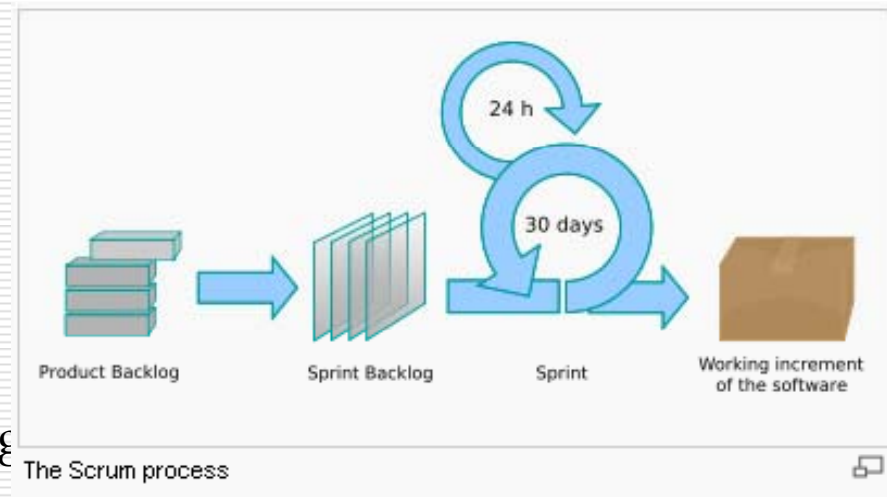
Work units required to achieve one of the backlog items
must fit into a predefined time-box
Affected backlog items frozen

❑ Scrum meetings

15 minute daily meetings
What was done since last meeting?
What obstacles were encountered?
What will be done by the next meeting?

❑ Demos

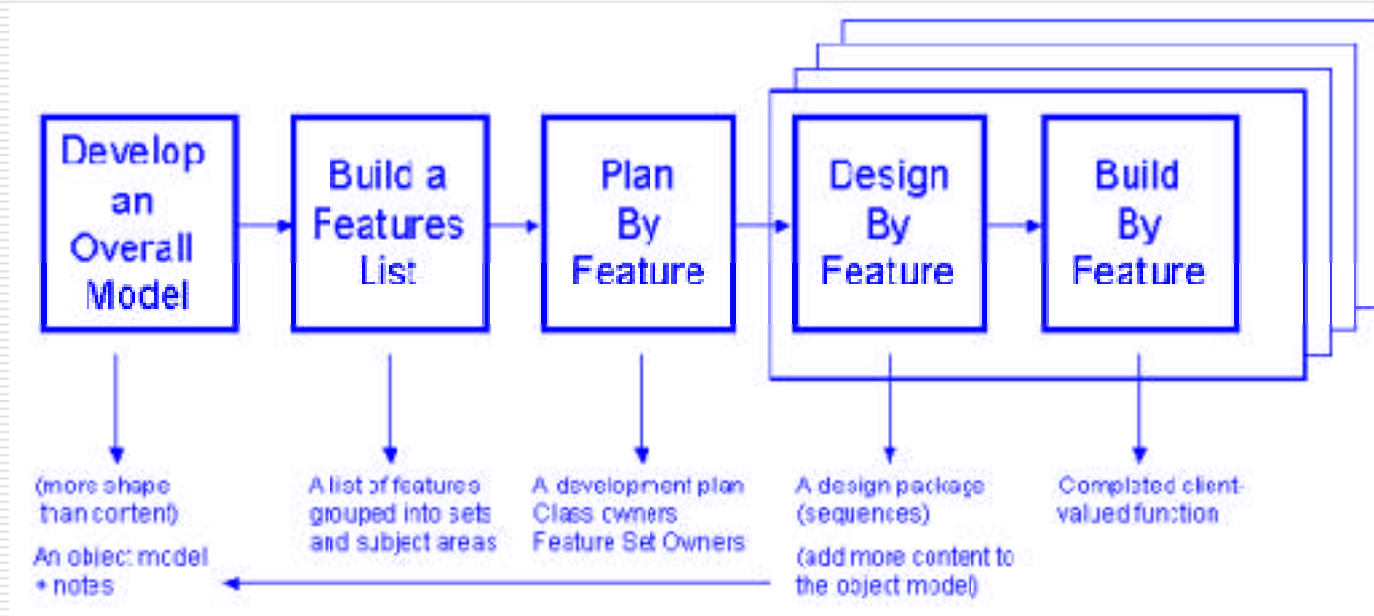
Deliver **software** increment to customer for evaluation



Feature Driven Development

- Originally proposed by Peter Coad et al
- FDD – distinguishing features
 - Emphasis is on defining “features”
 - a *feature* “is a client-valued function that can be implemented in two weeks or less.”
 - Uses a feature template
 - <action> the <result> <by | for | of | to> a(n) <object>
 - A features list is created and “plan by feature” is conducted
 - Design and construction merge in FDD

Feature Driven Development



Reprinted with permission of Peter Coad

Agile Modeling

- Originally proposed by Scott Ambler
 - Suggests a set of agile modeling principles
 - Model with a purpose
 - Use multiple models
 - Travel light
 - Content is more important than representation
 - Know the models and the tools you use to create them
 - Adapt locally
-