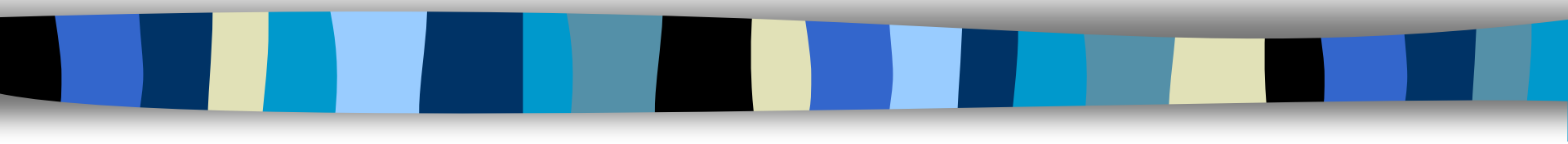


An Agile View of Process

Software Engineering: A Practitioner's Approach

6th Edition

Roger S. Pressman

Agile???

- **Agile software development** is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.
- It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages **rapid and flexible response to change.**
- It is a **conceptual framework that promotes foreseen interactions** throughout the development cycle.

The Manifesto (policy) 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."

What is Agility?

- Effective response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team; **eliminate** the “us and them” attitude
- Organizing a team so that it is in control of the work performed
- Rapid, incremental delivery of software

Principles to achieve agility – by the Agile Alliance (1)

1. Highest priority -> satisfy the customer
2. Welcome changing requirements
3. Deliver working software frequently
4. Business people and developers must **work** together
5. Build projects **around motivated individuals**
6. **Emphasize face-to-face** conversation

Principles to achieve agility – by the Agile Alliance (2)

7. Trust that **working software** is the primary measure of progress
8. Agile processes promote **sustainable development(maintain constant pace)**
9. Continuous attention to technical excellence and good design enhances agility
10. **Simplicity** – the art of maximizing the amount of work not done – is essential
11. The **best designs** emerge from self-organizing teams
12. The **team tunes and adjusts its behavior** to become more effective.

Agile Software Process – Addresses Three Key Assumptions

- Difficulty in predicting changes of requirements and customer priorities
- For many types of s/w, design and construction are **interleaved**
- Analysis, design, construction, and testing **are not as predictable as we might like (from planning point of view)**

Comparison with other methods..

- Methods exist on a continuum from ***adaptive* to *predictive***
- Agile methods **lie on the *adaptive* side**
- Adaptive methods focus on adapting quickly to changing realities. When the needs of a project change, an adaptive team changes as well. An adaptive team **will have difficulty describing exactly what will happen in the future**.

Predictive methods

- Predictive methods, in contrast, **focus on analyzing and planning the future in detail and cater for known risks.** In the extremes, a predictive team can report exactly what features and tasks are planned for the entire length of the development process.
- Predictive methods rely on **effective early phase analysis and if this goes very wrong, the project may have difficulty changing direction**.
- Predictive teams will often institute a change control board to ensure that **only the most valuable changes are considered**.

How an agile Software Process should be…

- An agile process must be adaptable
- It must adapt incrementally
- Requires customer feedback
- An effective catalyst for customer feedback is an operational prototype

The Politics of Agile Development

- There is considerable debate about the benefits and applicability of agile software development
- No one is against agility. The real question is:
  - What is the best way to achieve it?

Human Factor

- Key point:
  - The Process molds to the needs of the people and team, not the other way around
- A number of key traits must exist among the people on an agile team and the team itself
  - Competence
  - Common focus
  - Collaboration
  - Decision-making ability
  - Fuzzy problem-solving ability
  - Mutual trust and respect
  - Self-organization

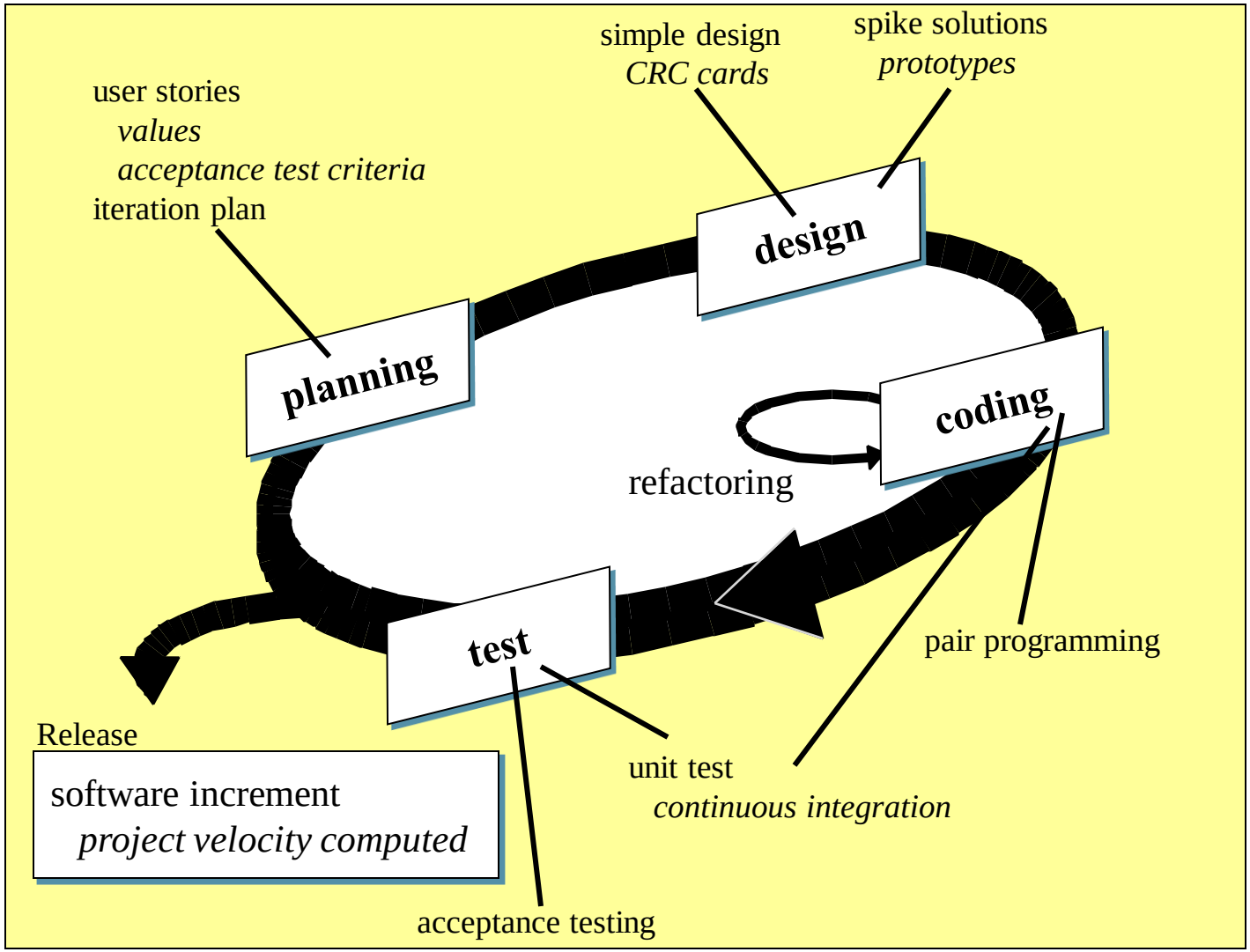
Agile Process Models

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

Extreme Programming (XP) - 1

- The most widely used agile process, originally proposed by Kent Beck [BEC99]
- XP uses an object-oriented approach as its preferred development paradigm
- Defines four (4) framework activities
  - Planning
  - Design
  - Coding
  - Testing

Extreme Programming (XP) - 2



XP - Planning

- Begins with the creation of a set of stories (also called *user stories*)
- Each story is written by the customer and is placed on an index card
- The customer assigns a value (i.e. a priority) to the story
- Agile team assesses each story and assigns a cost
- Stories are grouped to for a deliverable increment
- A commitment is made on delivery date
- After the first increment "project velocity" is used to help define subsequent delivery dates for other increments

XP - Design

- Follows the KIS (keep it simple) principle
- Encourage the use of CRC (class-responsibility-collaborator) cards
- For difficult design problems, suggests the creation of "*spike solutions*"—a design prototype
- Encourages "*refactoring*"—an iterative refinement of the internal structure of code
- Design occurs both before and after coding commences

XP - Coding

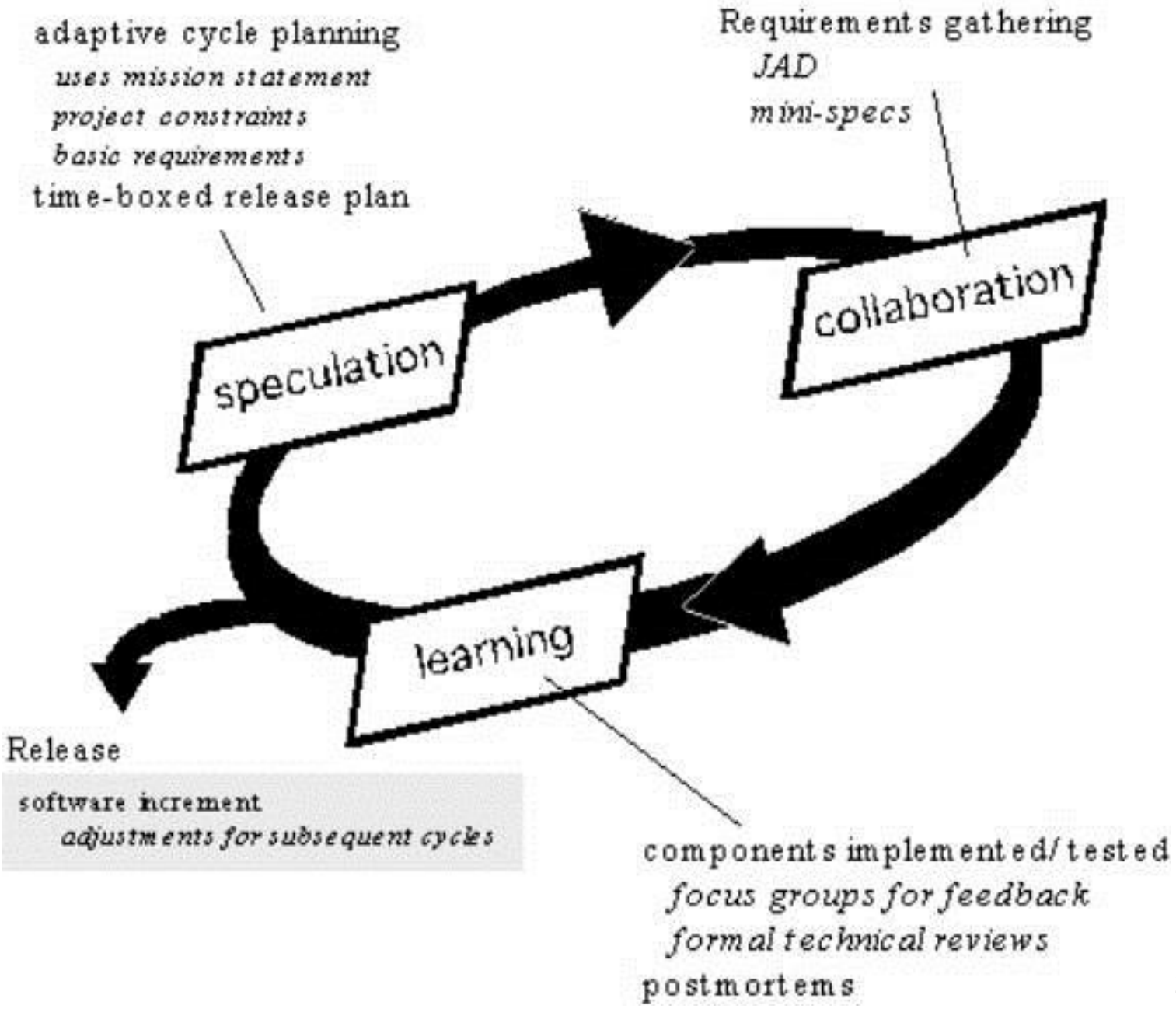
- Recommends the construction of a series of unit tests for each of the stories before coding commences
- Encourages "*pair programming*"
  - Mechanism for real-time problem solving and real-time quality assurance
  - Keeps the developers focused on the problem at hand
- Needs continuous integration with other portions (stories) of the s/w, which avoids compatibility and integration problems.

XP - Testing

- Unit tests should be implemented using a framework to make testing automated. This encourages a regression testing strategy.
- Integration and validation testing can occur on a daily basis
- *Acceptance tests*, also called *customer tests*, are specified by the customer and executed to assess customer visible functionality
- Acceptance tests are derived from user stories

Adaptive Software Development(ASD)

- Focus on human collaboration and team self-organization
- Incorporates three phases namely speculation, collaboration, and learning
- During *speculation,* the project is initiated and *adaptive cycle planning* is conducted
- Collaboration encompasses communication and teamwork, but it also emphasizes individualism, because individual creativity plays an important role in collaborative thinking
- ASD teams learn in three ways: focus groups, technical reviews and project postmortems


adaptive cycle planning
uses mission statement
project constraints
basic requirements
time-boxed release plan
Requirements gathering
JAD
mini-specs
speculation
collaboration
learning
Release
software increment
adjustments for subsequent cycles
components implemented/tested
focus groups for feedback
formal technical reviews
postmortems

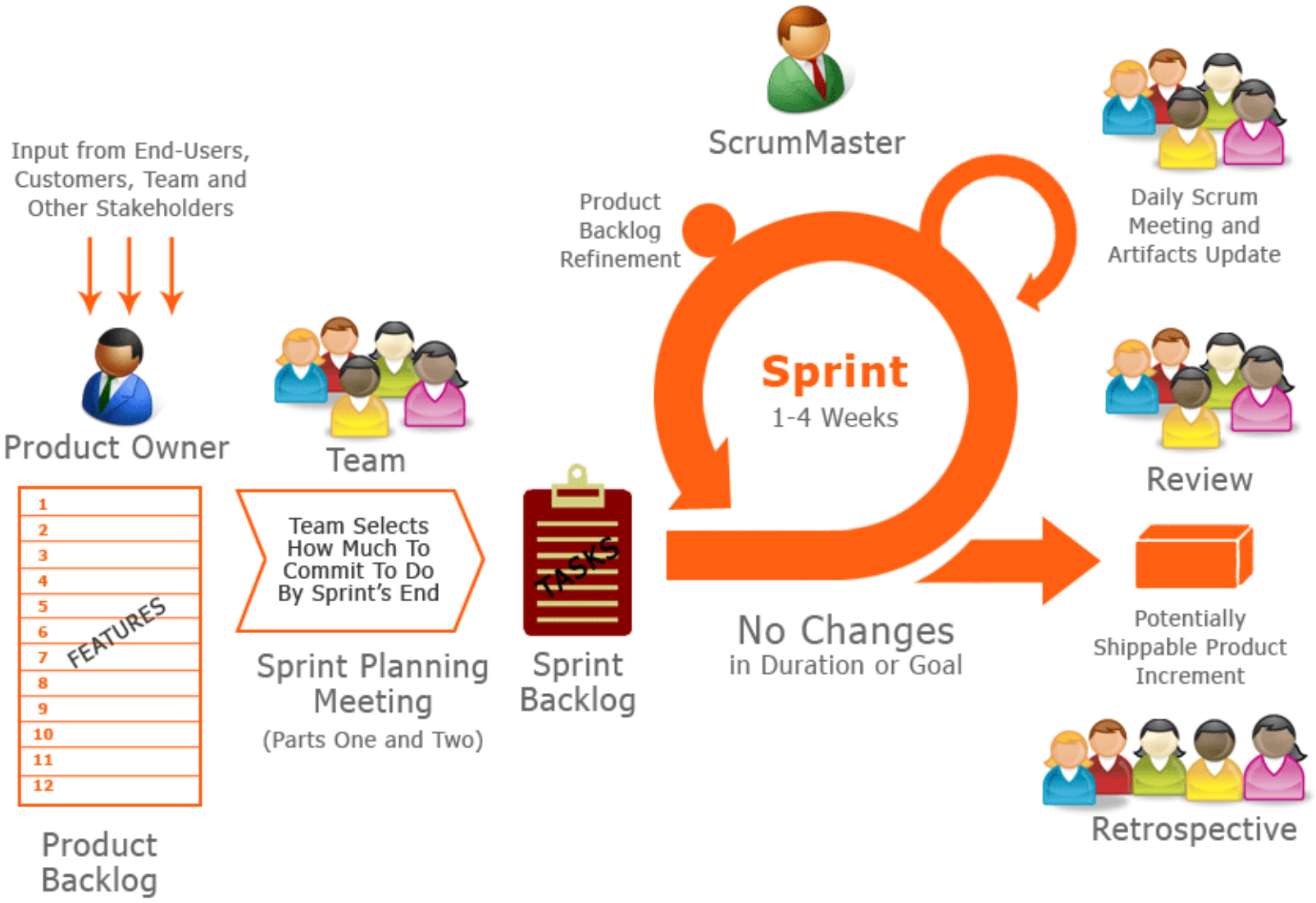
Dynamic Systems Development Method (DSDM)

- Provides a framework for building and maintaining systems which meet tight time constraints
- 80 percent of an application can be delivered in 20 percent of the time it would take to deliver the complete (100 percent) application
- DSDM Life Cycle has following activities:
- *Feasibility Study*: assesses whether the application is a viable candidate for the DSDM process
- *Business study:* establishes the functional and information requirements that will allow the application to provide business value

DSDM Cont.

- *Functional model iteration:* produces a set of incremental prototypes that demonstrate functionality for the customer
- *Design and build iteration:* revisits prototypes built during *functional model iteration* to ensure that each has been engineered in a manner that will enable it to provide operational business value for end users
- *Implementation:* places the latest software increment (an “operationalized” prototype) into the operational environment

SCRUM


Input from End-Users, Customers, Team and Other Stakeholders
Product Owner
Team
1
2
3
4
5
6
7
8
9
10
11
12
FEATURES
Product Backlog
Team Selects How Much To Commit To Do By Sprint's End
Sprint Planning Meeting
(Parts One and Two)
TASKS
Sprint Backlog
ScrumMaster
Product Backlog Refinement
Sprint
1-4 Weeks
No Changes
in Duration or Goal
Daily Scrum Meeting and Artifacts Update
Review
Potentially Shippable Product Increment
Retrospective

Feature Driven Development (FDD)

- An adaptive agile process that can be applied to moderately sized and larger software projects
- Manages problem and project complexity using feature-based decomposition
- A *feature* is a client-valued function that can be implemented in two weeks or less
- Features provide benefits like:
  - Because features are small blocks of deliverable functionality, users can describe them more easily

FDD Cont.

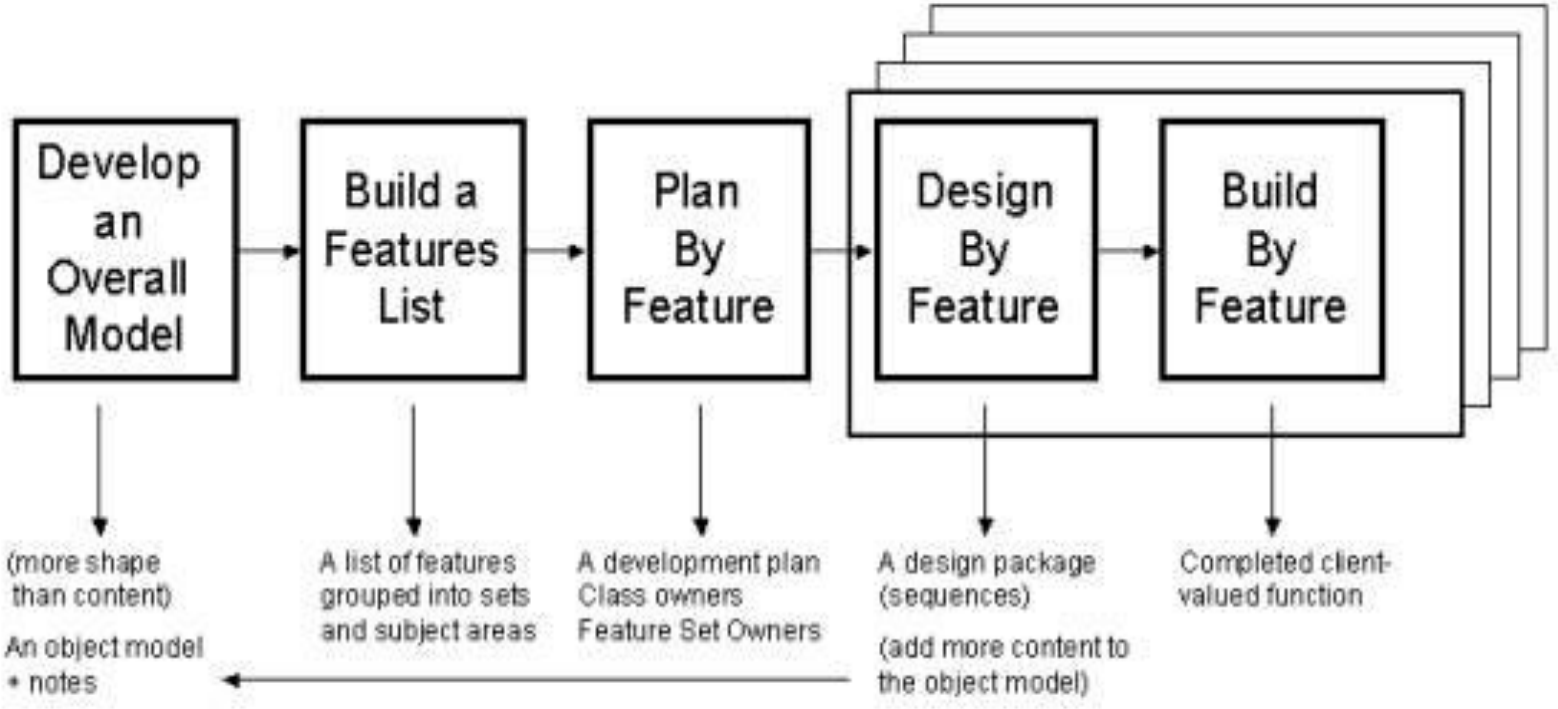
- Features can be organized into a hierarchical business-related grouping
- Since a feature is the FDD deliverable software increment, the team develops operational features every two weeks
- Project planning, scheduling, and tracking are driven by the feature hierarchy

How to define a feature?

<action> the **<result> <by|for|of|to>** a(n) **<object>**

Where <object> is a person, place or thing

FDD Framework Activities



Agile Modeling (AM)

- Agile Modeling is a practice-based methodology for effective modeling and documentation of software-based systems
- The agile team must have courage and humility

Principles of Agile Modeling

- 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

When to use Agile model:

•When new changes are needed to be implemented. The freedom agile gives to change is very important. New changes can be implemented at very little cost because of the frequency of new increments that are produced.

•To implement a new feature the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.

When to use Agile model(2)

•Unlike the waterfall model in agile model very limited planning is required to get started with the project. Agile assumes that the end users' needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This effectively gives the customer the finished system they want or need.

•Both system developers and stakeholders alike, find they also get more freedom of time and options than if the software was developed in a more rigid sequential way. Having options gives them the ability to leave important decisions until more or better data or even entire hosting programs are available; meaning the project can continue to move forward without fear of reaching a sudden standstill.

Advantages of agile process

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Close, daily cooperation between business people and developers.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.
- Even late changes in requirements are welcomed

Disadvantage of agile process

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.