# Agile Development (Part 2)

## Agile Methodologies & Tooling

Software Engineering | Chapter 3

# Introduction & Objectives

The Agile universe is vast! What if you need more structure than XP? Or want to apply manufacturing principles?

*"Today, we tour the Agile landscape, exploring how one size does NOT fit all."*

**By the end of this lecture:**

- Compare major Agile models: Scrum, Lean, FDD, ASD, Crystal.
- Explain core Scrum elements (Roles, Artifacts, Ceremonies).
- Understand Agile Modeling & Unified Process.
- Identify key Agile tooling categories.

# Adaptive Software Development (ASD)

## Philosophy

Replace the traditional "Plan-Build-Revise" with a dynamic cycle that acknowledges uncertainty.

**Key Focus:** Mission-driven, feature-based iterations with a heavy emphasis on learning.

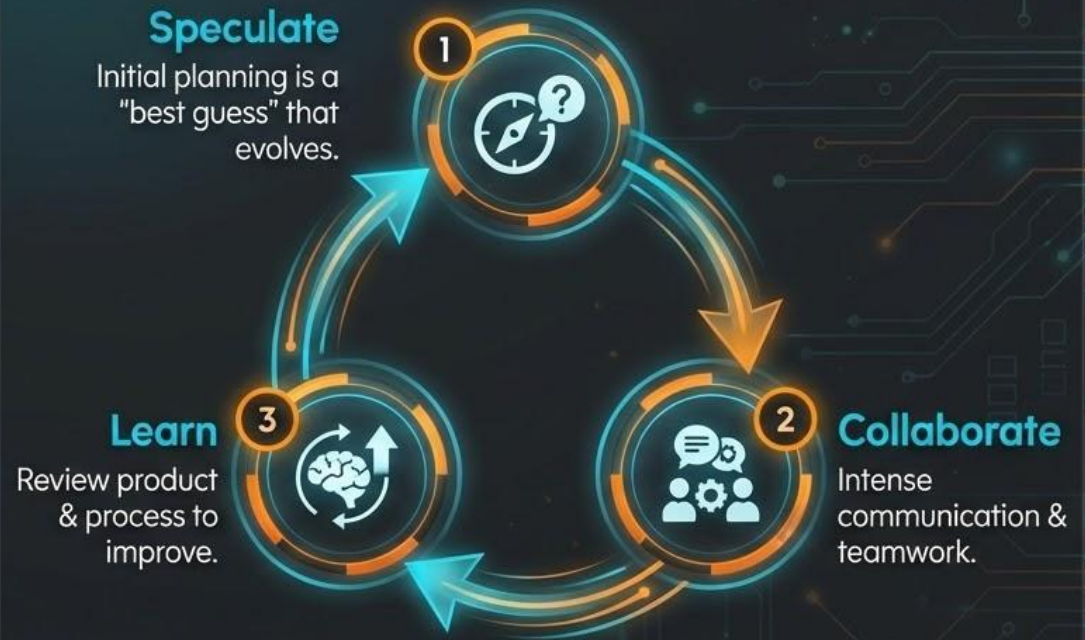
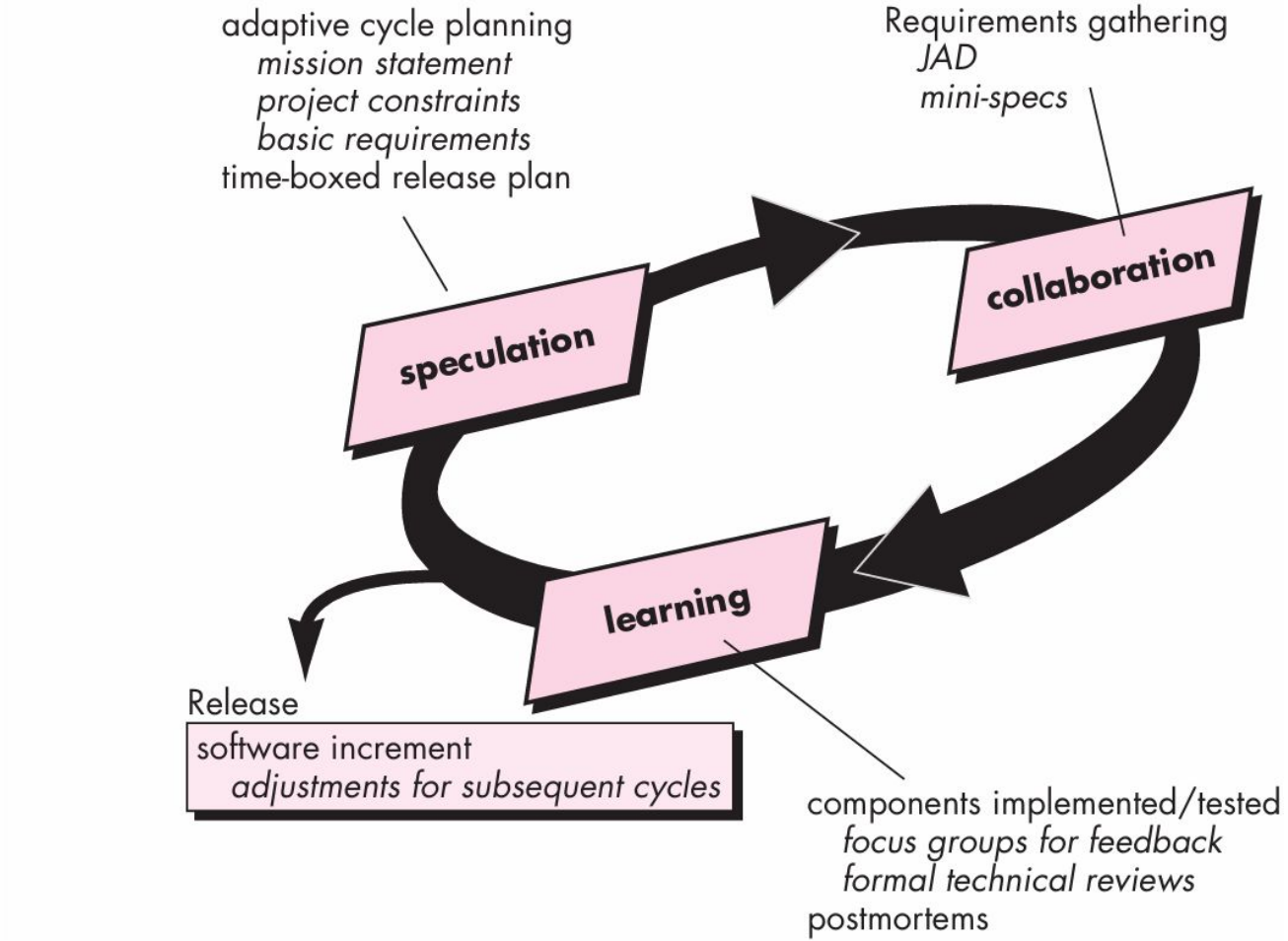


**FIGURE 3.3**

**Adaptive software development**



# Scrum: Roles & Artifacts


## Product Owner

Visionary. Manages the Product Backlog and defines 'Done'.

## Scrum Master

Servant-leader. Removes impediments and protects the team.

## </> Dev Team

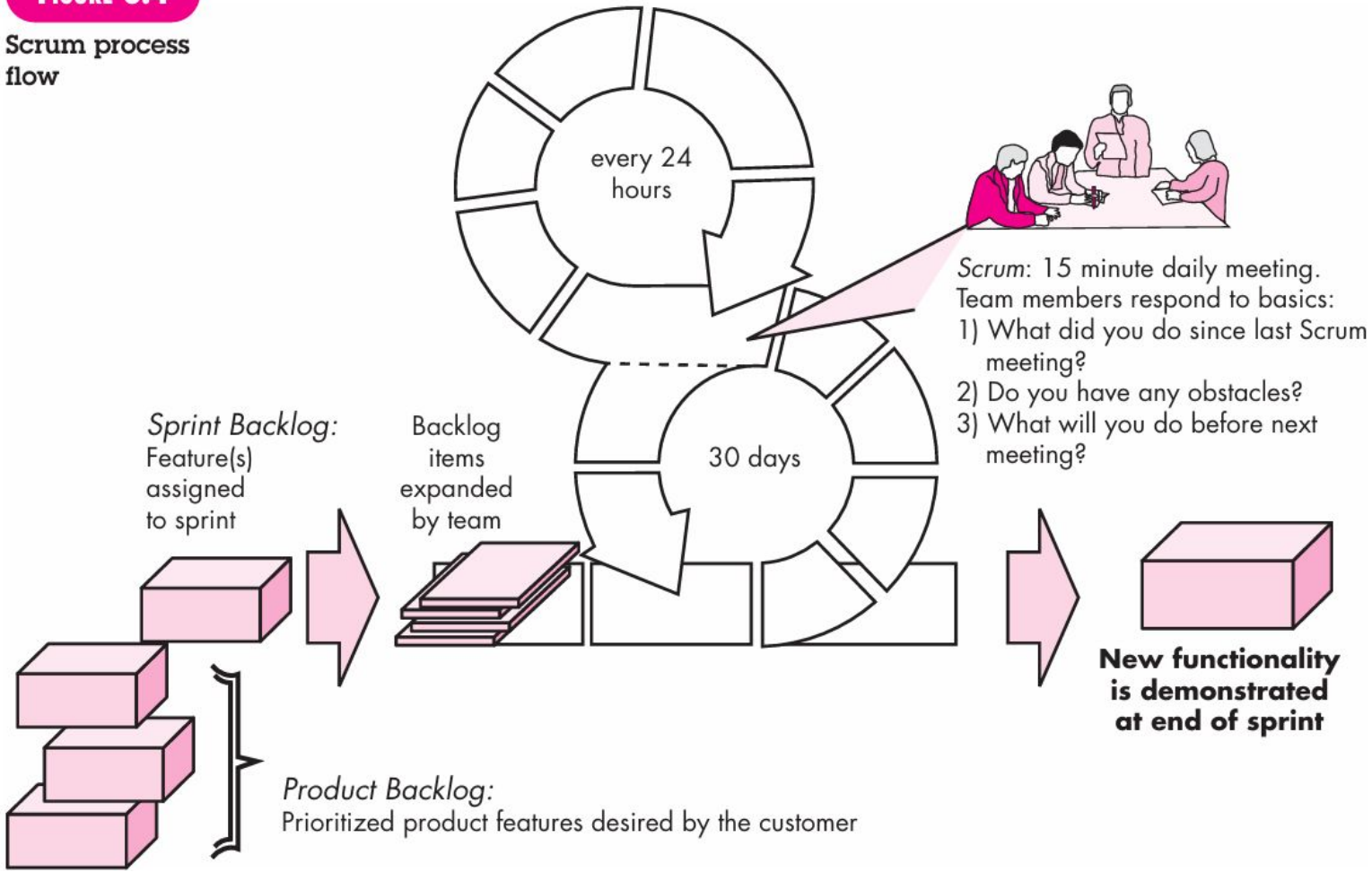
Self-organizing, cross-functional group delivering the Increment.

## Artifacts:

Product Backlog — Sprint Backlog — Increment

**FIGURE 3.4**

Scrum process flow



# The Sprint Cycle (2-4 Weeks)

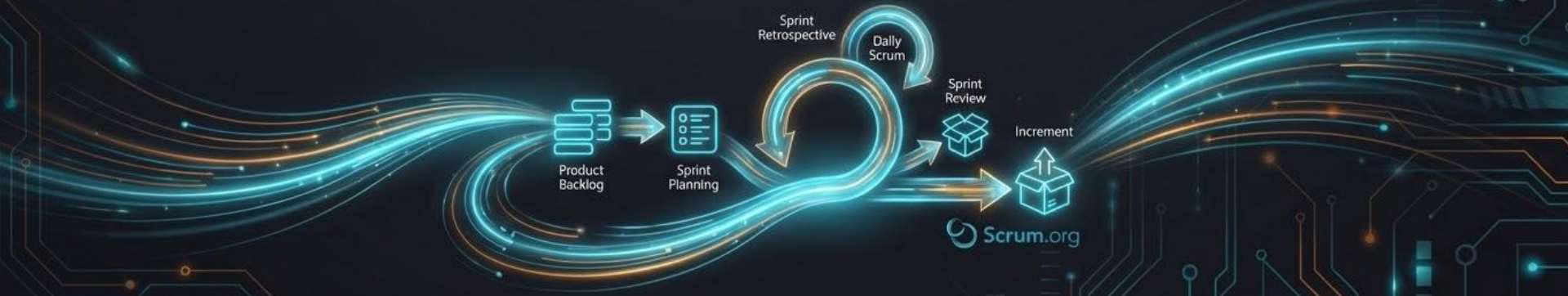
**Plan**
Select Backlog items.

**Daily**
15-min Sync.

**Develop**
Create Increment.

**Review**
Demo to stakeholders.

**Retro**
Process improvement.



# DSDM & Crystal

## DSDM


Focuses on business needs and on-time delivery.

### MoSCoW Prioritization

- Must have (Critical)
- Should have (Important)
- Could have (Desirable)
- Won't have (Later)

## Crystal


Methodology 'weight' is tailored to project size and criticality.

**Core Idea:**

People over process. Lightweight, adaptable, and communication-centric.

# Feature Driven Development (FDD)

A model-driven, short-iteration process blending Agile with upfront planning.

**Distinctive:**
Strong emphasis on clear roles (Chief Programmer, Class Owner) and feature-centric delivery.

## 5 Core Activities

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

where an **<object>** is "a person, place, or thing (including roles, moments in time or intervals of time, or catalog-entry-like descriptions)." Examples of features for an e-commerce application might be:

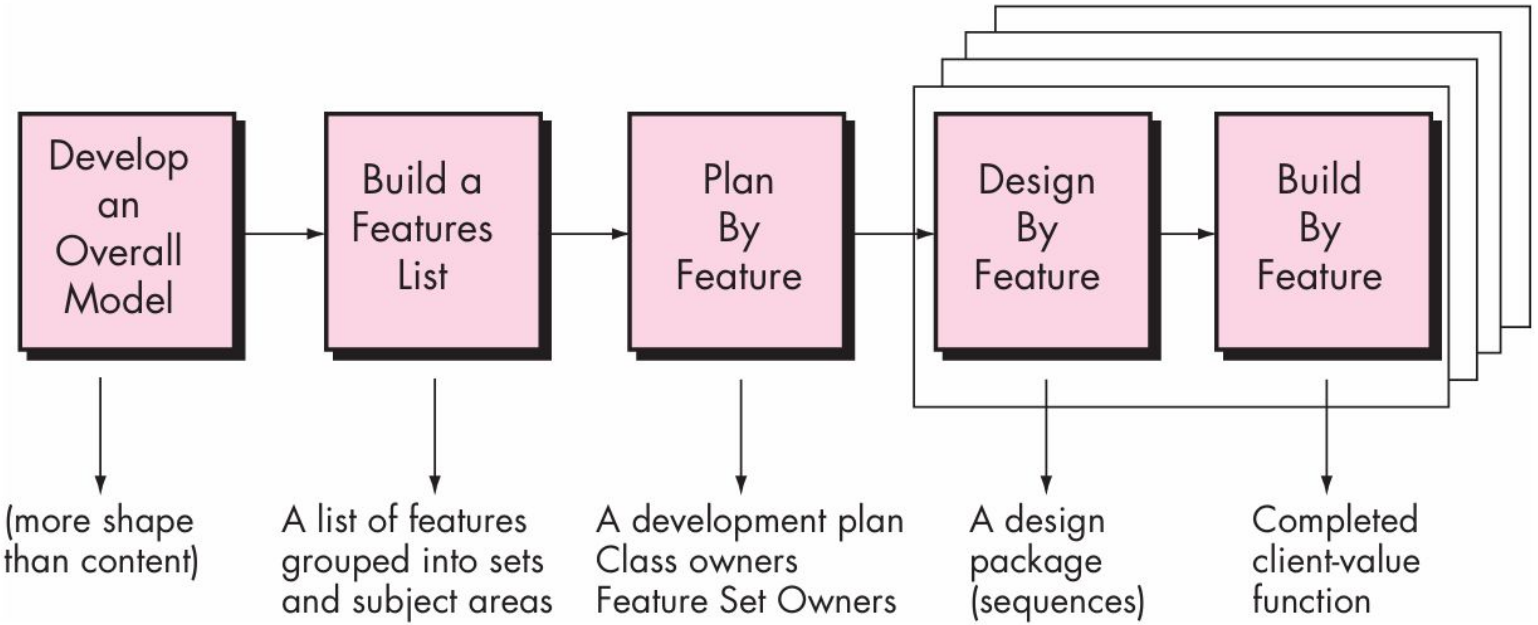
*Add the product to shopping cart*

*Display the technical-specifications of the product*

*Store the shipping-information for the customer*

**FIGURE 3.5**

**Feature Driven Development [Coa99] (with permission)**



# Lean Software Development

## The 7 Principles

Applying manufacturing wisdom to optimize the whole system.


**Philosophy:** Focus on flow, pull systems, and continuous improvement (Kaizen).

# Modeling & Unified Process

## Agile Modeling (AM)

**Practice, not process**. "How do we model without heavy docs?"

Model with a Purpose.

Travel Light (create just enough).

Content > Representation (whiteboard sketches are fine).

## Agile Unified Process (AUP)

Streamlined version of RUP.


Retains phases: Inception, Elaboration, Construction, Transition.

Applies Agile concepts: TDD, Refactoring.

Bridges discirlined architecture with responsiveness.

# Activity: Match the Methodology

| Methodology | Characteristic |
| --- | --- |
| 1. Scrum | A) Sprints, Product Backlog, Scrum Master |
| 2. FDD | B) Develop Overall Model, Build by Feature |
| 3. Lean | C) Eliminate Waste, Optimize the Whole |
| 4. Crystal | D) Tailored to Team Size and Criticality |
| 5. ASD | E) Speculate-Collaborate-Learn Cycle |

# Agile Tooling: Planning & Comms

Tools should enable agility, not introduce bureaucracy.

## Planning & Tracking

Manage Backlogs, track velocity, burn-down charts.

**Jira, Trello, Azure DevOps**

## Collaboration

Enable daily scrums, remote pairing, info radiators.

**Slack, MS Teams, Miro**

# Agile Tooling: Dev & Quality

## Development Environment

**Purpose:** Enable TDD, Refactoring, CI/CD.

**Examples:** Jenkins, GitHub Actions, JUnit, IDE Refactoring tools.

## Testing & CM

**Testing:** Automated unit & acceptance tests (Selenium, Cucumber).

**Config Management:** Essential for collective ownership (Git, Subversion).

**Principle:** Automate the mundane, amplify communication, make progress visible.

# Conclusion & Key Takeaways

## Spectrum

Different methods solve different problems (Management vs. Engineering vs. Flow).

## Lean

Focuses on waste elimination and flow.

The true **Agile** practitioner is a **methodologist**, selecting and adapting practices to fit the **context**.

## Scrum

Dominant framework with fixed roles and sprints.

## Adaptation

FDD and AUP add structure; Crystal adapts to team size.