Module 2 – Topic 2.3

Agile Practices

Topic Outline

- Make connections from Agile to the process models examined
- Extreme Programming – XP
  - An agile methodology that focuses on development
- Scrum
  - An agile methodology that focuses on management

Using Agile with Process Models

Recall The Agile Manifesto

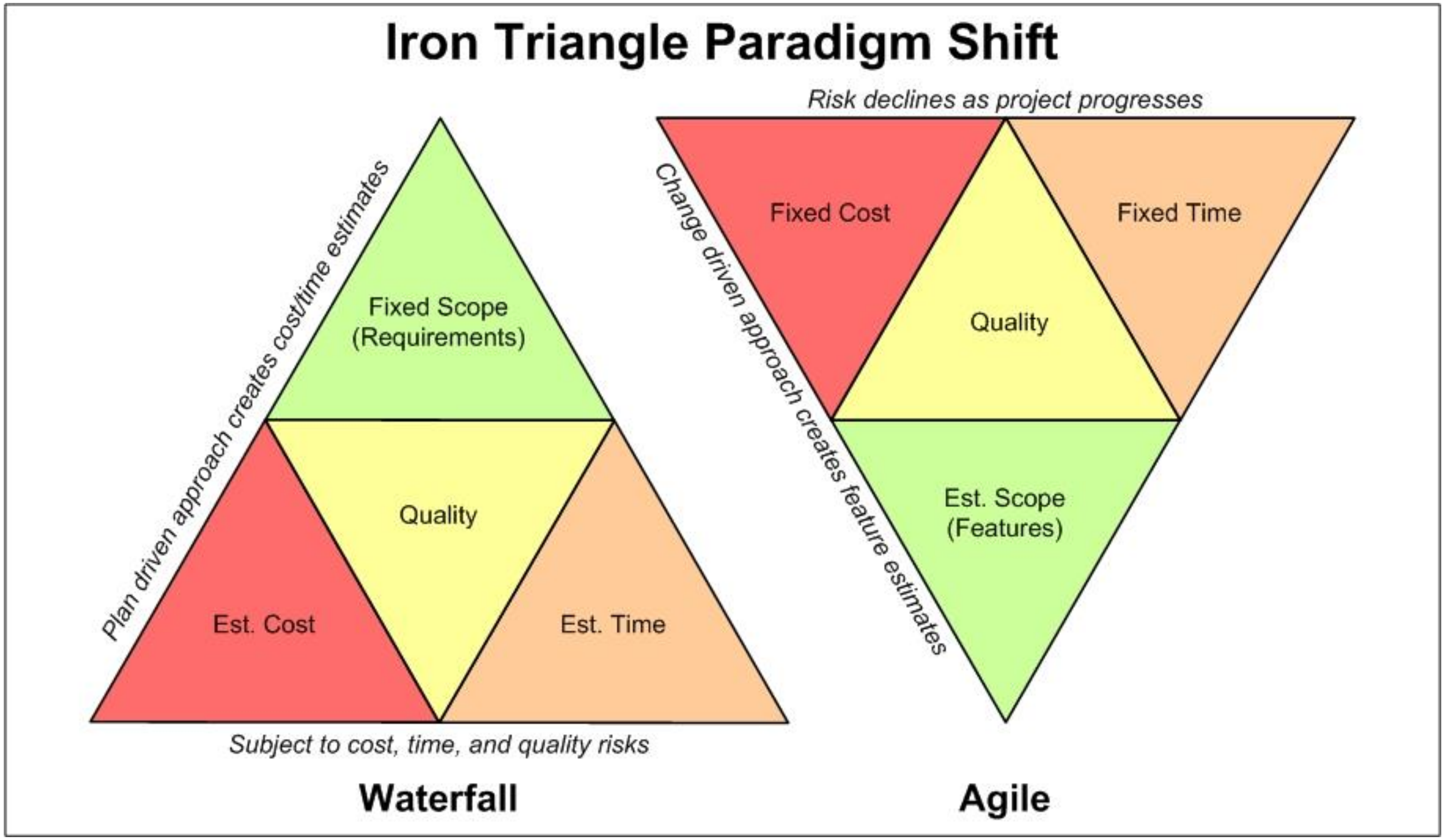
Individuals & Interactions	over	Process & tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

Agile Practices and Waterfall Model

- The waterfall model does not work well with agile practices

Waterfall Model	**Agile Practices**
Heavily reliant on documentation	Value working software
The product is only delivered at the end	Early and continuous delivery of working software
Requires contracts or sign-offs	Value customer collaboration
Not adaptable to changes	Responding to change

Agile Practices and Waterfall Model


Iron Triangle Paradigm Shift
Risk declines as project progresses
Fixed Scope (Requirements)
Quality
Est. Cost
Est. Time
Plan driven approach creates cost/time estimates
Subject to cost, time, and quality risks
Waterfall
Fixed Cost
Fixed Time
Quality
Est. Scope (Features)
Change driven approach creates feature estimates
Agile

Agile Practices and V-Model

v-moel is a linear model

- The linear aspect of the V-model does not fit well with agile practices

- But the V-model does encourage verification to ensure the product is working the way it is supposed to

V-Model	Agile Practices
Organize each level of verification to appropriate phases	Verification is an important part of ensuring working software

Agile Practices and Sawtooth Model

- Sawtooth is closer to Agile than Waterfall and V-model because it allows client interactions and feedback

- But just a couple of prototypes is not enough

Sawtooth Model	Agile Practices
Not enough opportunity for close collaboration and feedback to include the suggested improvements or changes to the product	Satisfy the customer through early and continuous delivery of valuable software

iteration is the fundemantal to the agile practices

Agile Practices with Spiral Models

- You could use agile practices along with the Spiral model

Spiral Model	Agile Practices
Has iteration	Iteration is a core concept
Continuous releases to gather feedback	Early and continuous delivery of working software
Value risk management	Value risk management

- But there are differences

Spiral Model	Agile Practices
Longer iterations	Prefer shorter Iteration
Not frequent client interactions	Frequent client interactions
Heavyweight risk planning	Lightweight risk management

Agile Practices and Unified Process

- The unified process is the most compatible with agile values and principles
  - Agile Unified Process

Unified Process	Agile Practices
Large projects where a great deal of refinement is needed	Small to medium projects where a great agility is needed

Extreme Programming - XP

An agile methodology that focuses on development

XP and Agile Manifesto

- XP values
  - Client satisfaction
  - Constantly delivering software
  - Fast responding to change
  - Effective team work and self organization

12 Code-Centric Practices for Development

🟩 The Planning Game	▨ Simple Design	▨ Pair Programming	▨ 40 Hour Work Week
□ Small Releases	🟩 Continuous Testing	▨ Collective Code Ownership	▨ On-Site Customer
□ System Metaphor	🟩 Refactoring	🟩 Continuous Integration	□ Coding Standards

to let client understand the program

□ Rules in Agile Manifesto

🟩 Widely adopted practices

▨ Controversial practices

Pair Programming

- This takes code review to the extreme
  - Two developers work side by side at one computer to develop code
- But personality conflicts will play a huge factor

We will learn more about this in Module 4 Topic 4.1 Software Reviews.

You will practice pair programming in the Laboratory 1 in the Week 6.

Collective Code Ownership

Danielle is the product manager working with a five-person development team to create a payroll application. After delivering a prototype, the client found a bug in the application that calculates the wages incorrectly. Timmy and Maria were the programmers who co-wrote the unit test for this feature. Steven and Maria were the programmers who pair programmed the source code. Danielle, the product manager, conducted the acceptance test for the product before the release.

Who is to blame for this error?

But this works for small development teams

A. Danielle (product manager, conducted acceptance test)
B. Timmy (co-wrote unit test)
C. Maria (pair-programmed the source code, co-wrote unit test)
D. Steven (pair-programmed the source code)
E. The other two developers on the team

Simple Design

- Design what you need to make the requirements you're working on today
- Lack of up front software architecture planning could lead to a lot of reworking in the future

On-Site Client

- The clients are always around to clarify and answer questions about anything that might come up
- It's often unrealistic to expect the client to be there every time an issue or question arises

The Planning Game 🟩

- The client and the development team work together to plan the product
  - Comp up with a list of new features for the product. Each of these features is expressed as a user story
- The development team creates estimates for how much effort they can allot in the next iteration
- The client and development team then prioritizes the stories, and decide when to release a working version of each feature

We will learn more about this in Module 3 Right Product – Clients Needs and Software Requirements

Continuous Testing 🟩

测试写在代码之前

- Test-driven development – tests are written for a feature before the source code is written

- Tests are executable forms of requirements
  - Acceptance tests for the client to test that each feature of the overall product works as specified
  - Unit test for the developers to test lower-level functionality

We will learn acceptance test in Module 3 Topic 3.3 Writing Requirements, and learn test-driven development in Module 4 Topic 4.2 Test-Driven Development.

You will practice test-driven development and refactoring in the Laboratory 2 in Week 9.

Refactoring

- Restructuring the design of the code without changing the behaviour of the code
- Improve the design so that new features can be added easily
- Must be performed together with automated unit testing

We will learn more about this in Module 4 Topic 4.1 Software Reviews.

You will practice refactoring and test-driven development in the Laboratory 2 in Week 9.

Continuous Integration

- Combining code frequently (at least once daily). All tests should pass 100% before and after the integration
- Whenever a developer commits a code change it will be built, tested, integrated, and released

Which XP practice must be used together with continuous integration?

Industry example:
Microsoft Daily Build

Quick Question

You're the manager of the development team who's producing an app for a cinema that allows users to check show times and buy tickets.

Which of the following, if any, would be acceptance tests?

A. a test that confirms when a user buys a ticket, a valid ticket appears
B. a test that confirms when a user creates an account, their information is added to the database
C. a test that confirms when the user logs in, that their log-in information matches an account in the database
D. a test that confirms when a user selects a show time, movies playing at that time appear

Quick Question

Which of the following statements is true about the Extreme Programming practice of continuous testing?

A. Test are written for a required feature to validate the product
B. Test are written for a required feature by the client writhing unit tests

option B is in term of validation

C. Test are written for a required feature before its source code is written
D. Test are written for a required feature just after its source code is written

Quick Question

Continuous delivery mainly aims to achieve ________ by the end of each iteration?

A. a meeting with the client to gain feedback on the working software

B. the product requirements are received for the next iteration

[✓ C. working software that is tested, ready-to-run, and releasable to others]

Quick Question

In the XP practice of refactoring, which two of the following statements are true?

[✓ A.] Developers run unit tests while refactoring

B. The behaviour of the source code is changed while restructuring its design

C. A new feature is added during refactoring

[✓ D.] The design of the source code is restructured without changing its behaviour

Scrum

An agile methodology that focuses on management

Scrum and Agile Manifesto

- Scrum is iterative and incremental
  - You never go very long before assessing where you are with the product

- Scrum has three pillars
  - Transparency – everyone can see every part of the project
  - Inspection – inspection of work products and progress
  - Adaptation – the team must adjust and adapt to prevent deviation

All scrum practices take the values and principles from the Agile Manifesto and turn them into guidelines and rules to use in software development.

Three Roles in a Scrum Team

- Product owner
  - Make decisions about the product and the product backlog
- Development team
  - Turn the backlog of features into increments of working software
- Scrum master
  - Make sure that the Scrum team is adhering to Scrum theory, practices, and rules

Product Owner

- The product owner is one person and not a committee
- Responsibilities
  - Prioritizes the requirements in the Product Backlog and decide what actually gets built (learn about the Product Backlog in the Module 3 – Topic 3.3 Writing Requirements)
  - Makes sure that the development team understands what is expected for the features in the backlog

Development Team

- Self organizing
  - No "assigning" work. The team members communicate and coordinate which tasks they want to work on.
- Cross functional
  - Do not depend on anyone who is not part of the team
  - Team members take on a variety of tasks, so the members have no titles (like coders or testers)
- Small
  - Ideally, 3 – 9 members

Scrum Master

- Responsibilities to the Product Owner
  - Finding techniques to manage the backlog
  - Ensuring the product owner knows how to prioritize the backlog to get maximum value
  - Facilitating scrum events
- Responsibilities to the development team
  - Coaching the team to self-organize
  - Removing development roadblocks
  - Felicitating scrum events

Quick Question

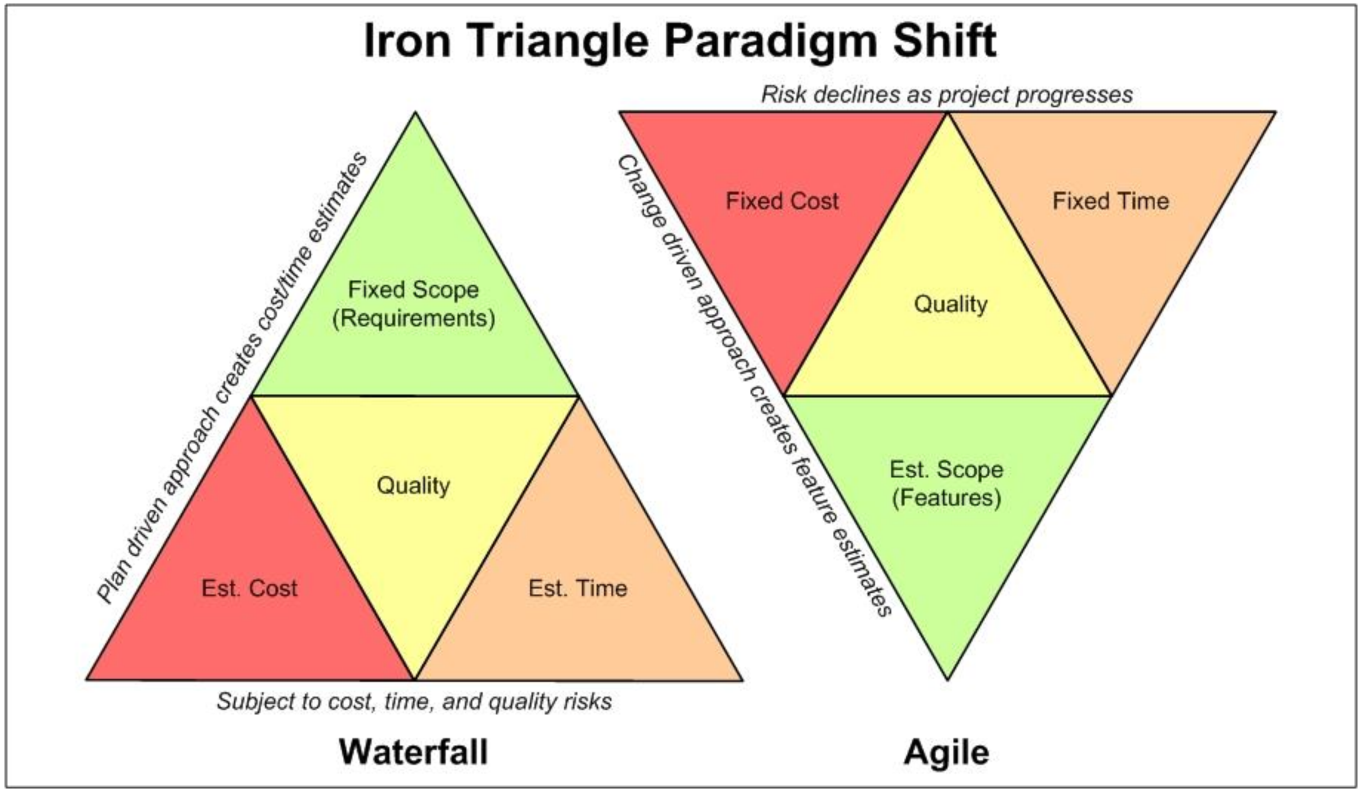
Penny, the representative from the company, was appointed as the Product Owner. Your team is halfway into development and you get an email from the CEO of the company. He is requesting a pre-made running playlist users can listen to when they run. You know that your Development Team had already talked to Penny about this feature. And they decided as a team that they wouldn't implement this feature in favor of another one.

Following the practices of Scrum, what should you do?

A. Send the CEO an email back saying, "No, you get no say in this project"
B. Tell your development team to add the feature to the backlog
C. Tell your development team to develop the feature for this iteration
[✓ D. Send the CEO back an email asking for all feature requests to come through Penny]

Scrum Events

- Sprint, sprint planning, daily scrum, spring review meeting, spring retrospectives


Iron Triangle Paradigm Shift
Plan driven approach creates cost/time estimates
Fixed Scope (Requirements)
Quality
Est. Cost
Est. Time
Subject to cost, time, and quality risks
Waterfall
Risk declines as project progresses
Change driven approach creates feature estimates
Fixed Cost
Fixed Time
Quality
Est. Scope (Features)
Agile

All of these events are **time boxed**

Scrum Events

- A Sprint
  - A development phase of a specific amount of time, lasting one month or less, typically 1 or 2 weeks
  - At the end of each Sprint, the client is delivered a working prototype
- Each Sprint contains the four Scrum events
  - At the beginning of the Sprint – Sprint planning
  - At the beginning of each day – Daily scrum
  - At the end of the Sprint – Spring review and Sprint retrospectives

Learn about the Spring review in Module 3 Topic 3.5 Validating Requirements and the Daily scrum in Module 5 – Topic 5.2 Monitoring Project Plan

The Definition of Done

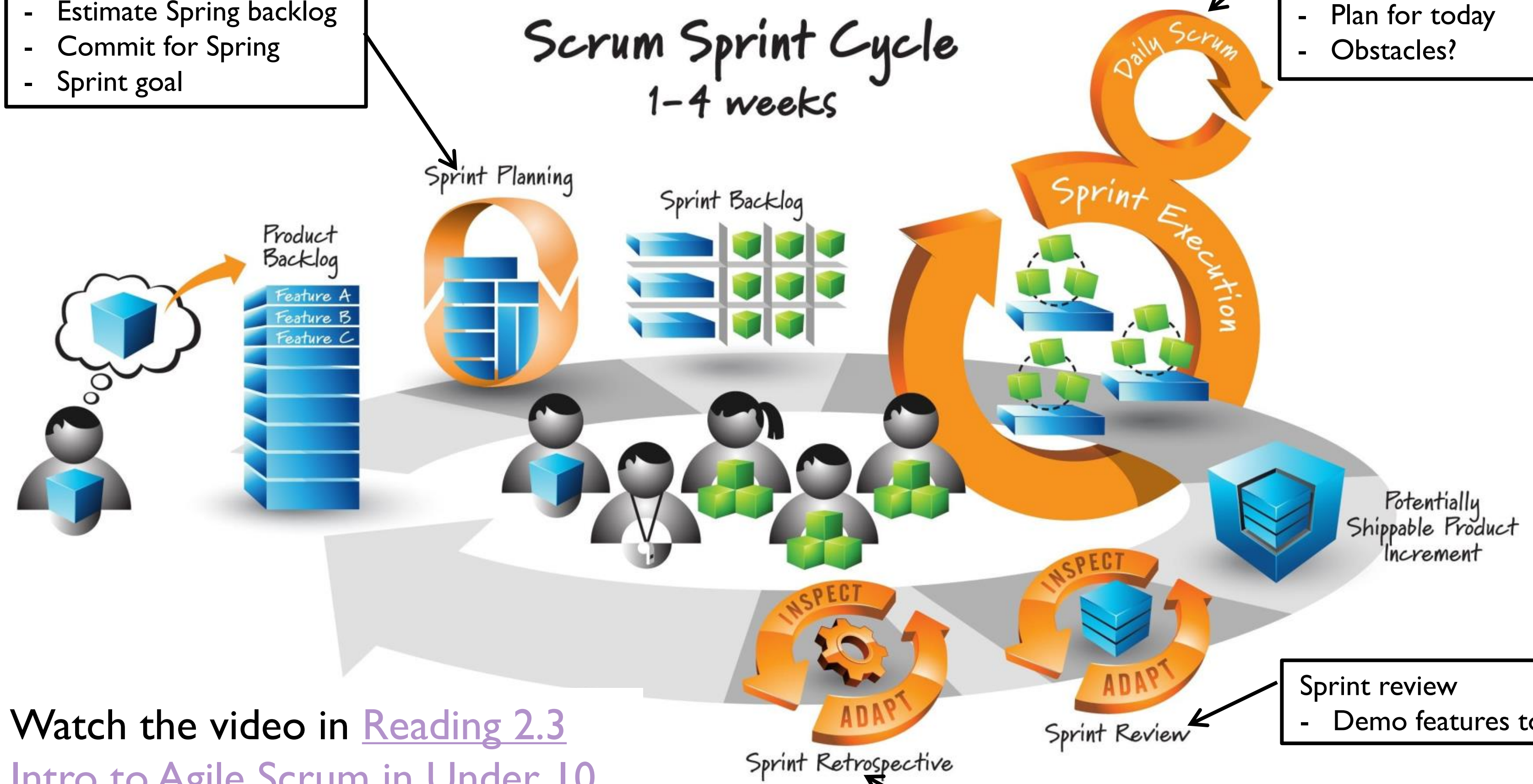
- The working prototype that is delivered at the end of a Sprint must be Done
- Typically, a feature is considered Done when the feature is coded, tested, and documented

during the sprint meeting, the content inculded should be done

A Summary of Scrum Practices

Spring Planning
- Review product backlog
- Estimate Spring backlog
- Commit for Spring
- Sprint goal

Daily scrum
- Done since last meeting
- Plan for today
- Obstacles?



Sprint review
- Demo features to all

Sprint retrospectives
- Inspect and adapt

Watch the video in [Reading 2.3 Intro to Agile Scrum in Under 10 Minutes]

Quick Question

The Sprint goal for this Iteration is to set up the user interface. This is everything that the end user will see and interact with. Halfway through the Sprint, the client sends you an email saying that they've changed their mind. They want the main colours of the application to be blue and white instead of red and black. The client also requests that they want the application to now support ads that pop up on the screen.

Which of these changes can be made in the current sprint?

A. The colour change
B. Popup ads
C. Both of them
D. Neither of them

in this question, the goal of sprint is to develop the UI interface, thus the change of the goal of the current sprint is not allowed