Software Engineering

Chapter 3
(Agile Software Engineering:
Agile methods, Scrum, and Extreme Programming)

Course Learning Outcome

CLO1: Recognize Software Engineering principles, **life cycle phases, processes**, and activities.

Agile software engineering

- Software products must be brought to market quickly so rapid software **development and delivery** is essential.
- Virtually all software products are now developed using an **agile approach**.
- **Agile** software engineering focuses on **delivering** functionality **quickly**, **responding** to **changing** product specifications and **minimizing** development **overheads**.

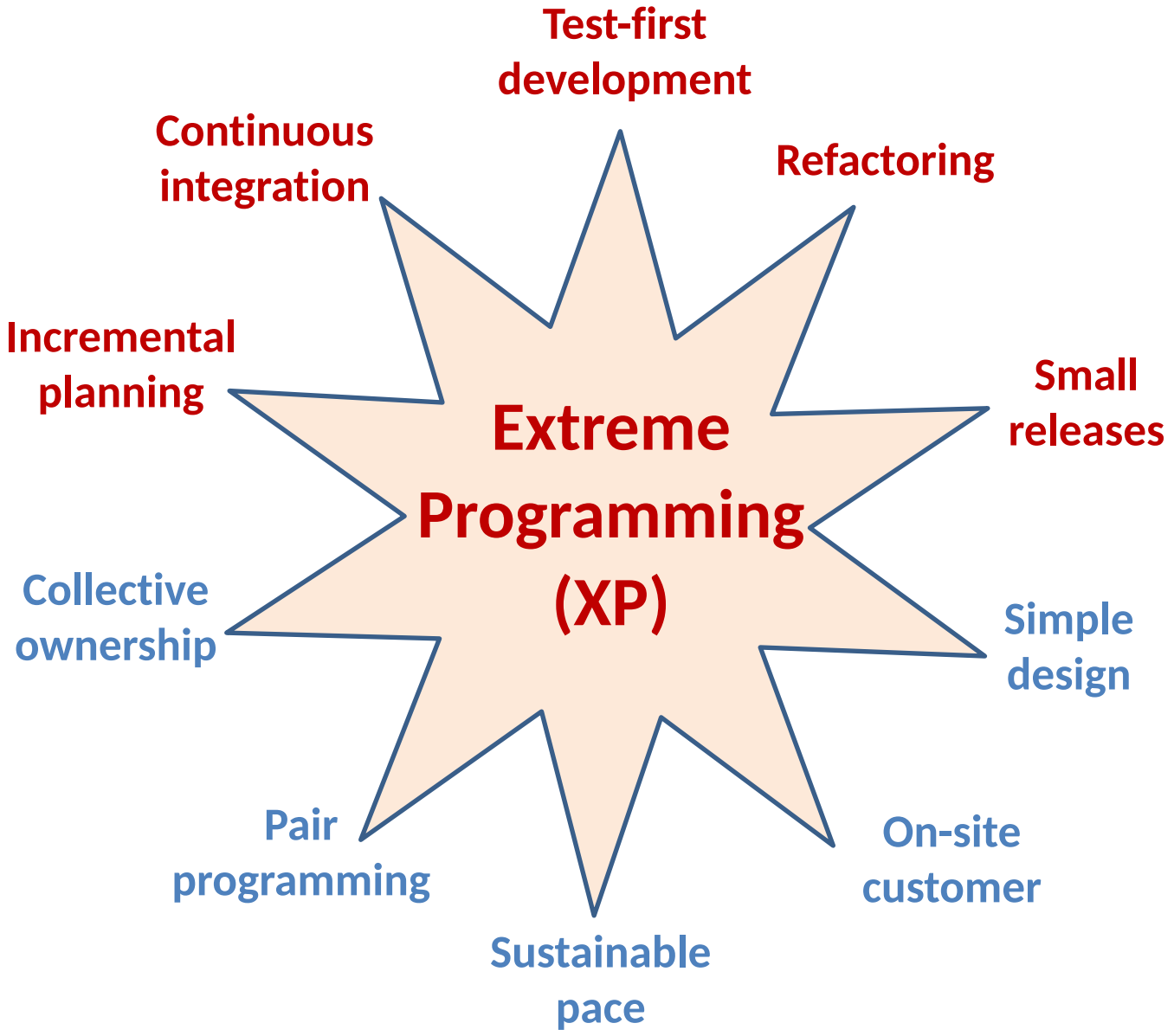
Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Extreme Programming (XP)

- The most influential work that has changed software development culture was the development of **Extreme Programming (XP)**.
- The name was coined by **Kent Beck in 1998** because the approach was developed by pushing recognized good practices, such as iterative development, to 'extreme' levels.
- Extreme programming focused on 12 new development techniques that were geared to rapid, incremental software development, change and delivery.
- Some of these techniques are now widely used; others have been less popular.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Extreme Programming Practices



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Widely adopted XP practices

- Incremental planning/user stories
  - There is no ‘grand plan’ for the system. Instead, what needs to be implemented (the requirements) in each increment are established in discussions with a customer representative.
  - In XP, a customer or user is part of the XP team and is responsible for making decisions on requirements.
  - User requirements are expressed as user stories or scenarios.
  - The customer chooses the stories for inclusion in the next release based on their priorities and the schedule estimates.

A ‘prescribing medication’ story

Prescribing medication

The record of the patient must be open for input. Click on the medication field and select either ‘current medication’, ‘new medication’ or ‘formulary’.

If you select ‘current medication’, you will be asked to check the dose; If you wish to change the dose, enter the new dose then confirm the prescription.

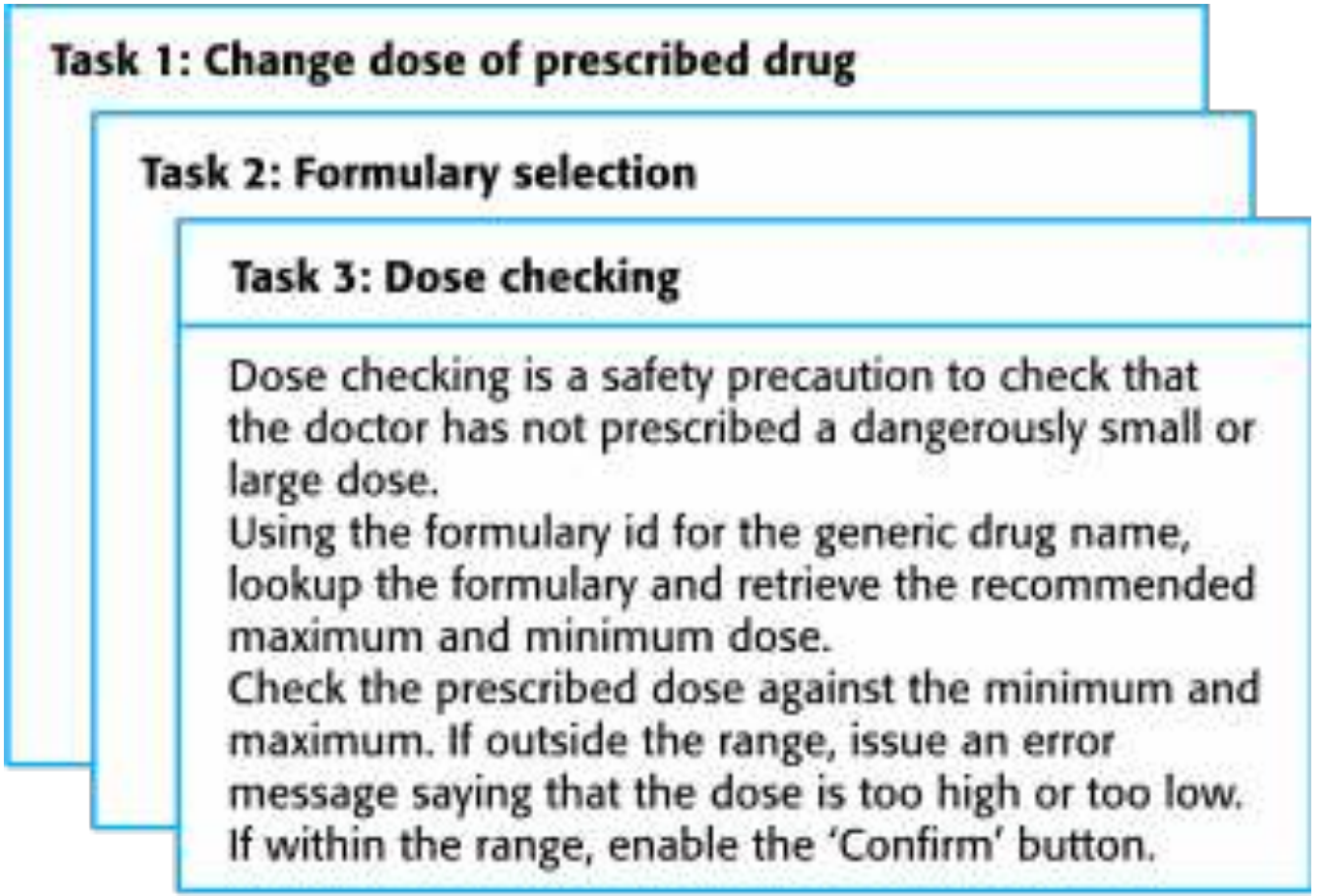
If you choose, ‘new medication’, the system assumes that you know which medication you wish to prescribe. Type the first few letters of the drug name. You will then see a list of possible drugs starting with these letters. Choose the required medication. You will then be asked to check that the medication you have selected is correct. Enter the dose then confirm the prescription.

If you choose ‘formulary’, you will be presented with a search box for the approved formulary. Search for the drug required then select it. You will then be asked to check that the medication you have selected is correct. Enter the dose then confirm the prescription.

In all cases, the system will check that the dose is within the approved range and will ask you to change it if it is outside the range of recommended doses.

After you have confirmed the prescription, it will be displayed for checking. Either click ‘OK’ or ‘Change’. If you click ‘OK’, your prescription will be recorded on the audit database. If you click ‘Change’, you reenter the ‘Prescribing medication’ process.

Examples of task cards for prescribing medication



Widely adopted XP practices

- Small releases
  - The minimal useful set of functionality that provides business value is developed first.
  - Releases of the system are frequent and incrementally add functionality to the previous release.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Widely adopted XP practices

- Test-driven development
  - Instead of writing code then tests for that code, developers write the tests first.
  - This helps clarify what the code should actually do.
  - An automated unit test framework is used to run the tests after every code contribution.

```
void testMultiply() {
    assertEquals(20, calculator.multiply(4, 5),
            "Regular multiplication should work");
}
```

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Test case description for dose checking

Test 4: Dose checking

Input:
1. A number in mg representing a single dose of the drug.
2. A number representing the number of single doses per day.

Tests:
1. Test for inputs where the single dose is correct but the frequency is too high.
2. Test for inputs where the single dose is too high and too low.
3. Test for inputs where the single dose * frequency is too high and too low.
4. Test for inputs where single dose * frequency is in the permitted range.

Output:
OK or error message indicating that the dose is outside the safe range.

Widely adopted XP practices

- Continuous integration
  - As soon as the work on a task is complete, it is integrated into the whole system and a new version of the system is created.
  - All unit tests from all developers are run automatically and must be successful before the new version of the system is accepted.

Widely adopted XP practices

Refactoring

- Refactoring means improving the structure, readability, efficiency and security of a program.
- Conventional wisdom in software engineering is to design for change. It is worth spending time and effort anticipating changes and add generality to a program to cope with change, which might reduce costs later in the life cycle.
- XP, however, maintains that this is not worthwhile as changes cannot be reliably anticipated.
- All developers are expected to refactor the code as soon as potential code improvements are found.
- This keeps the code simple and maintainable.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.



SCRUM

Scrum

- The principal responsibility of software project managers is to manage the project so that the software is delivered **on time** and **within the planned budget** .
- The standard approach to project management is plan-driven: Managers draw up a plan for the project showing what should be delivered, when it should be delivered and who will work on the development of the project deliverables.
- Agile project management requires a different approach, which is adapted to incremental development and the practices used in agile methods: Plans always change so anything apart from short-term plans are unreliable.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

What is Scrum?

- Scrum is an agile method that focuses on managing iterative development rather than specific agile practices.
- There are three phases in Scrum.
  - The initial phase is an outline planning phase where you establish the general objectives for the project and design the software architecture.
  - This is followed by a series of sprint cycles, where each cycle develops an increment of the system.
  - The project closure phase wraps up the project, completes required documentation such as system help frames and user manuals and assesses the lessons learned from the project.

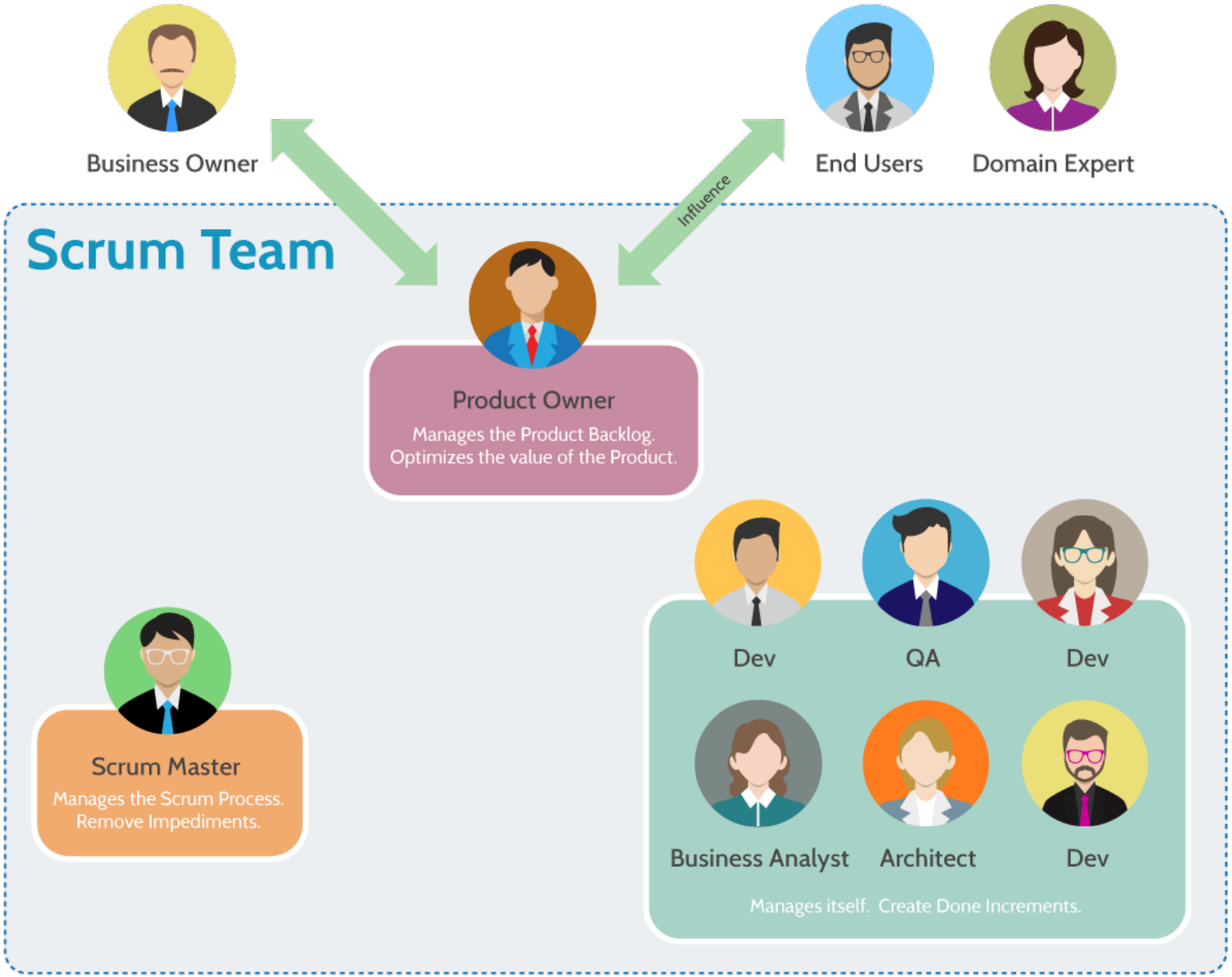
Scrum Terminology (a)

Scrum term	Definition
Development team	A self-organizing group of software developers, which should be no more than 7 people (up to ten in the scrum official guide). They are responsible for developing the software and other essential project documents.
Potentially shippable product increment	The software increment that is delivered from a sprint. The idea is that this should be 'potentially shippable' which means that it is in a finished state and no further work, such as testing, is needed to incorporate it into the final product. In practice, this is not always achievable.
Product backlog	This is a list of 'to do' items which the Scrum team must tackle. They may be feature definitions for the software, software requirements, user stories or descriptions of supplementary tasks that are needed, such as architecture definition or user documentation.
Product owner	An individual (or possibly a small group) whose job is to identify product features or requirements, prioritize these for development and continuously review the product backlog to ensure that the project continues to meet critical business needs. The Product Owner can be a customer but might also be a product manager in a software company or other stakeholder representative.

Scrum Terminology (b)

Scrum term	Definition
Scrum	A daily meeting of the Scrum team that reviews progress and prioritizes work to be done that day. Ideally, this should be a short face-to-face meeting that includes the whole team.
Scrum Master	The Scrum Master is responsible for ensuring that the Scrum process is followed and guides the team in the effective use of Scrum. He or she is responsible for interfacing with the rest of the company and for ensuring that the Scrum team is not diverted by outside interference. The Scrum Master should not be thought of as a project manager. However it may not always find it easy to see the difference.
Sprint	A development iteration. Sprints are usually 2-4 weeks long.
Velocity	An estimate of how much product backlog effort that a team can cover in a single sprint. Understanding a team's velocity helps them estimate what can be covered in a sprint and provides a basis for measuring improving performance.

Key roles in Scrum



Key roles in Scrum

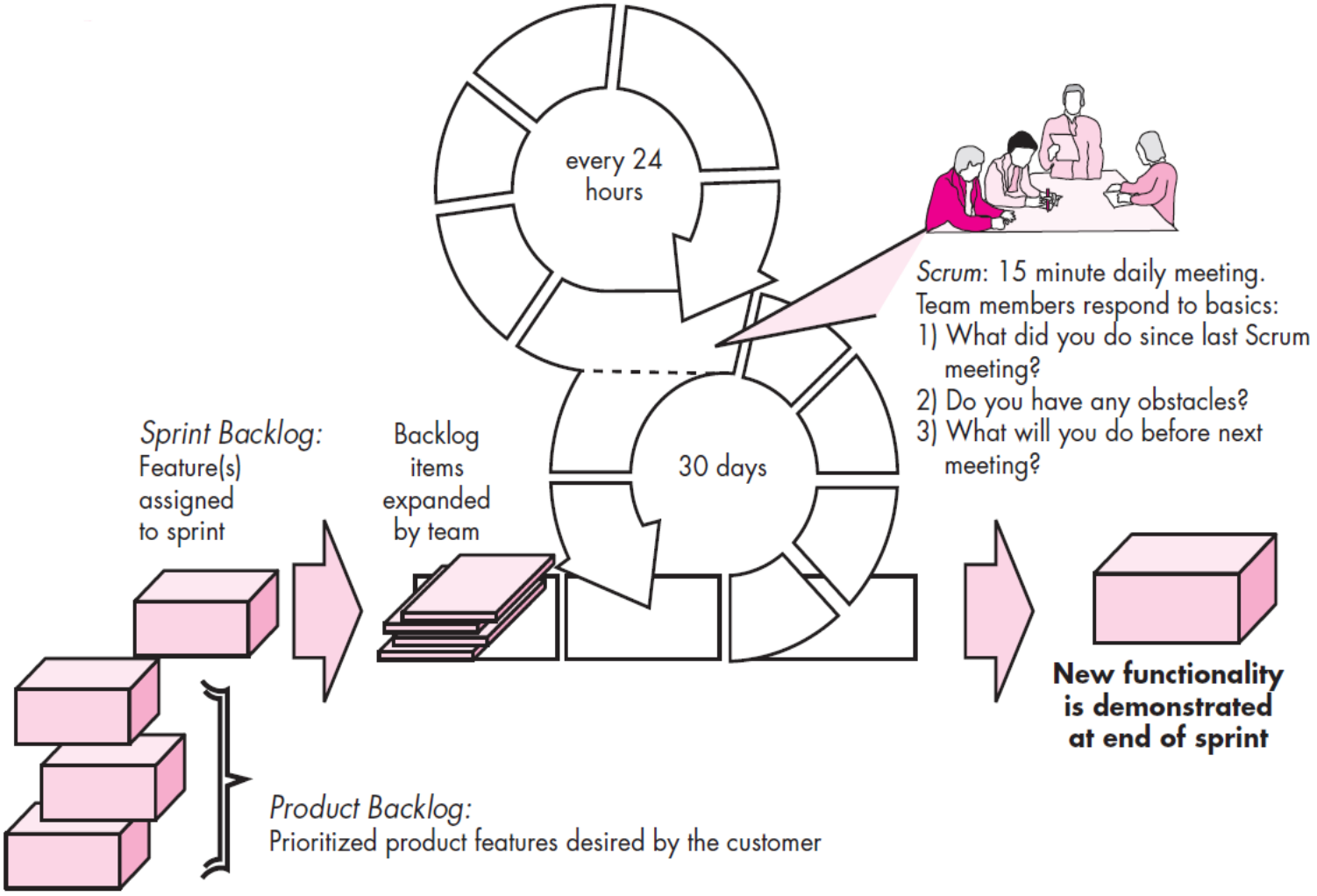
- **The Product Owner** is responsible for ensuring that the development team are always focused on the product they are building rather than diverted into technically interesting but less relevant work.
  - In product development, the product manager should normally take on the Product Owner role.
- **The Scrum Master** is a Scrum expert whose job is to guide the team in the effective use of the Scrum method. The developers of Scrum emphasize that the Scrum Master is not a conventional project manager but is a coach for the team. They have authority within the team on how Scrum is used.
  - In many companies that use Scrum, the Scrum Master also has some project management responsibilities.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

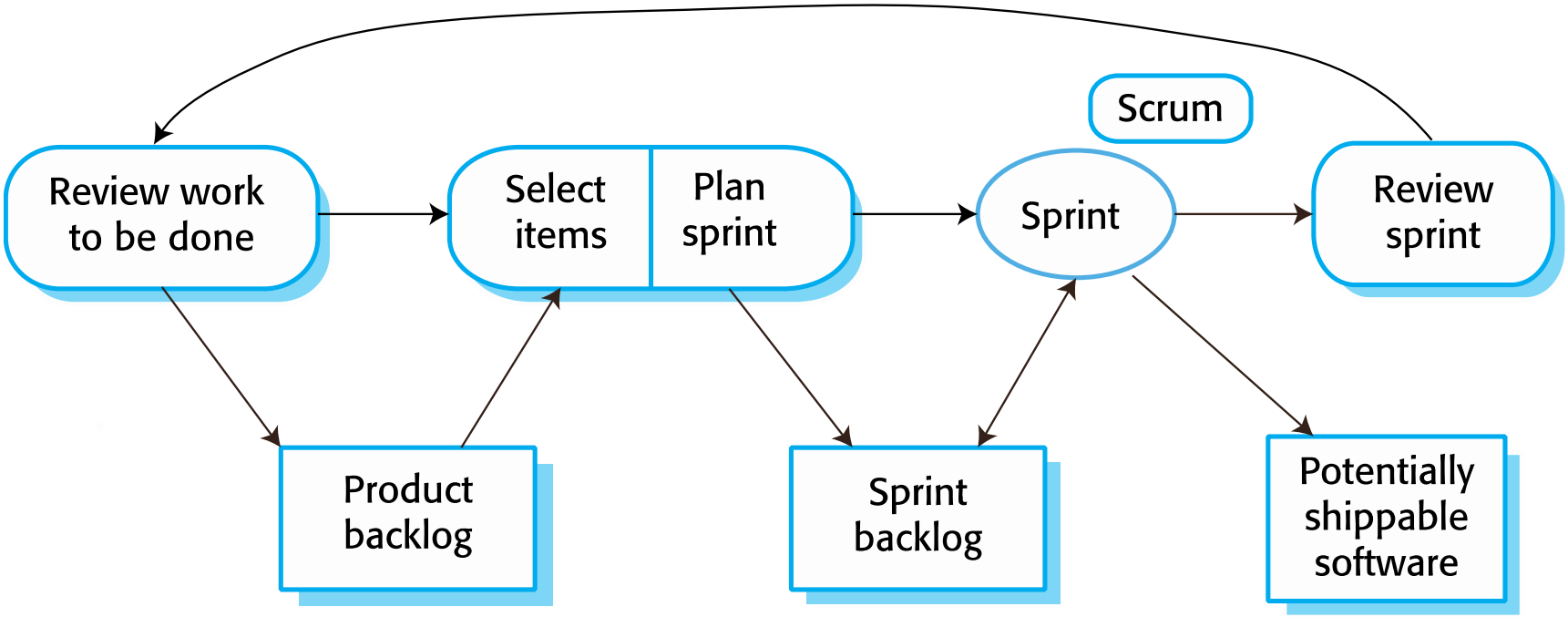
Scrum and sprints

- In Scrum, software is developed in sprints, which are fixed-length periods (2 - 4 weeks) in which software features are developed and delivered.
- During a sprint, the team has daily meetings (Scrums) to review progress and to update the list of work items that are incomplete.
- Sprints should produce a 'shippable product increment'. This means that the developed software should be complete and ready to deploy.

Scrum Process Flow



Scrum Sprint Cycle



Key Scrum practices

- **Product backlog**
  This is a to-do list of items to be implemented that is reviewed and updated before each sprint.
- **Timeboxed sprints**
  Fixed-time (2-4 week) periods in which items from the product backlog are implemented,
- **Self-organizing teams**
  Self-organizing teams make their own decisions and work by discussing issues and making decisions by consensus.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Product backlogs

- The product backlog is a list of what needs to be done to complete the development of the product.
- The items on this list are called product backlog items (PBIs).
- The product backlog may include a variety of different items such as product features to be implemented, user requests, essential development activities and desirable engineering improvements.
- The product backlog should always be prioritized so that the items that have to be implemented first are at the top of the list.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Examples of product backlog items (PBIs)

1. As a teacher, I want to be able to configure the group of tools that are available to individual classes. (feature)
2. As a parent, I want to be able to view my children's work and the assessments made by their teachers. (feature)
3. As a teacher of young children, I want a pictorial interface for children with limited reading ability. (user request)
4. Establish criteria for the assessment of open source software that might be used as a basis for parts of this system. (development activity)
5. Refactor user interface code to improve understandability and performance. (engineering improvement)
6. Implement encryption for all personal user data. (engineering improvement)

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

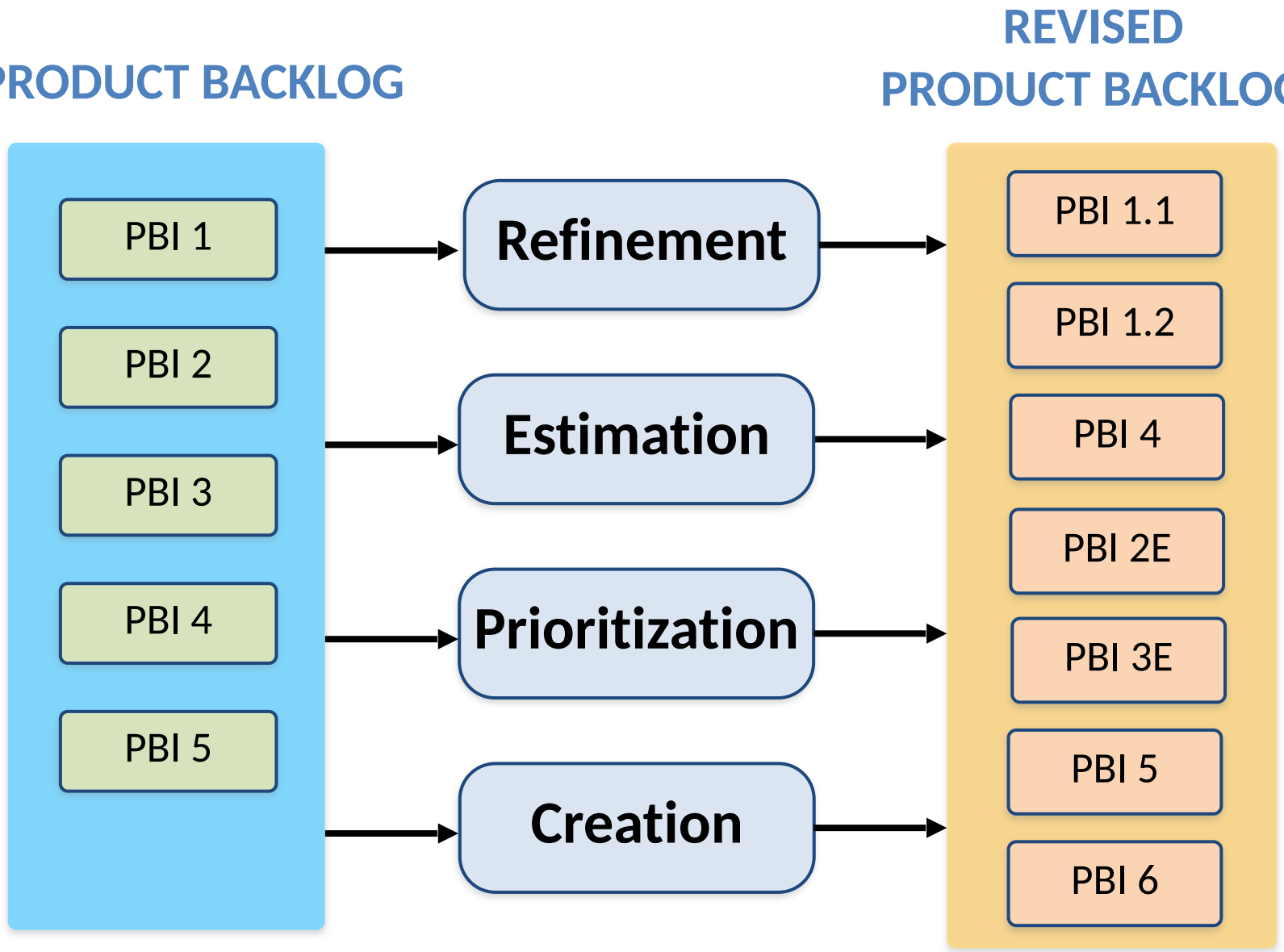
Product Backlog Item States

The definition of "Ready"

- **Ready for consideration**
  These are high-level ideas and feature descriptions that will be considered for inclusion in the product. They are tentative so may radically change or may not be included in the final product.
- **Ready for refinement**
  The team has agreed that this is an important item that should be implemented as part of the current development. There is a reasonably clear definition of what is required. However, work is needed to understand and refine the item.
- **Ready for implementation**
  The PBI has enough detail for the team to estimate the effort involved and to implement the item. Dependencies on other items have been identified.

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Product Backlog Activities



Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Sprint Activities

- The highest-priority items that are ready for implementation in the product backlog are selected for implementation in the next sprint.
- Team members work together to plan the sprint by analyzing the selected items to create the sprint backlog. This is a list of activities to be completed during the sprint.
- During implementation, the team implements as many of the sprint backlog items as they can in the fixed time period (same for each iteration) allowed for the sprint.
- Incomplete items are returned to the product backlog. Sprints are never extended to finish an incomplete item.

Managing External Interactions

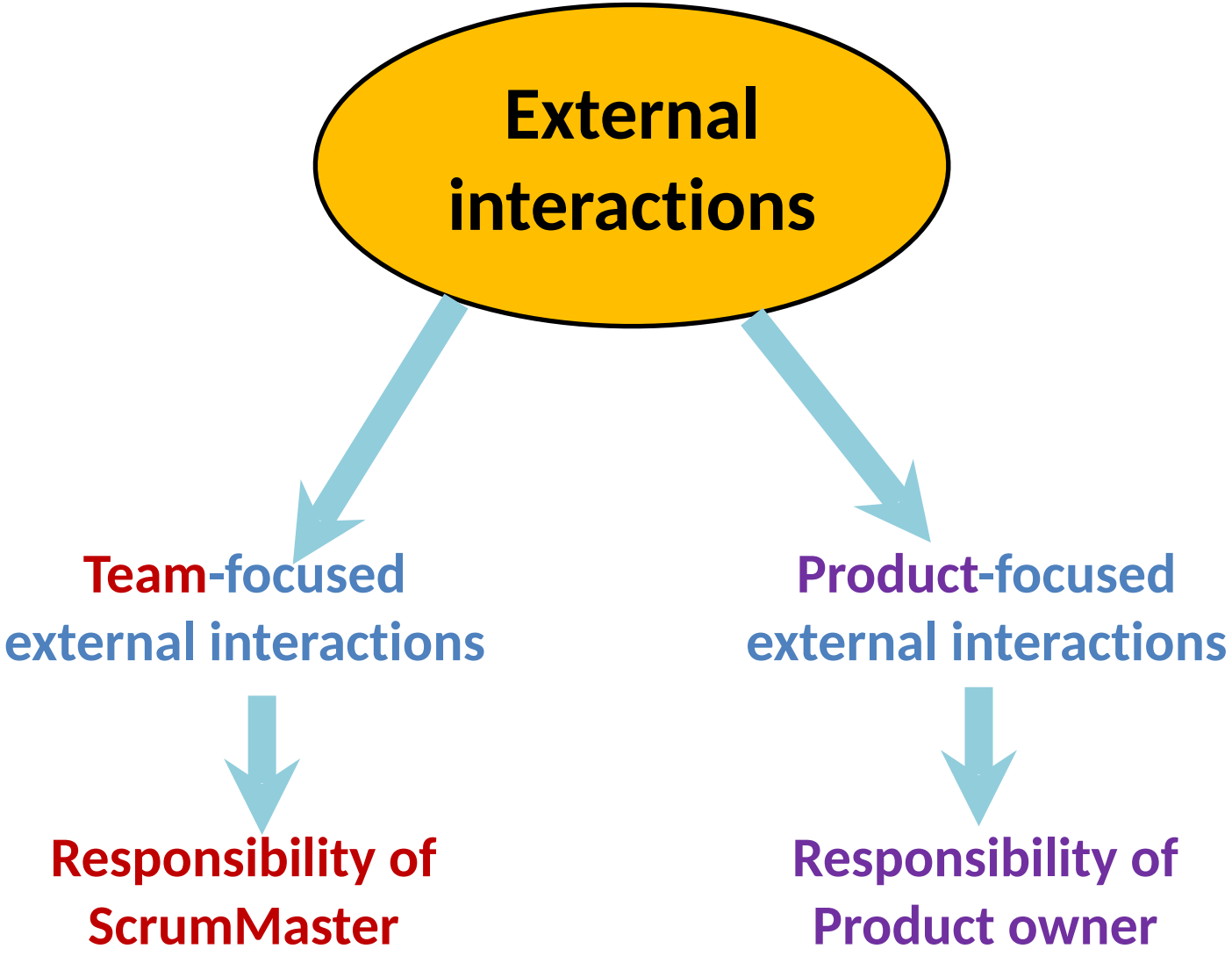
All development teams, even those working in small start-ups or non-commercial development, have some external interactions. Some interactions will help the team understand what customers require from the software product being developed.

Others will be with company management and other parts of the company, such as human resources and marketing.

In a Scrum project, the ScrumMaster and the Product Owner should be jointly responsible for managing interactions with people outside the team

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson

Managing External Interactions

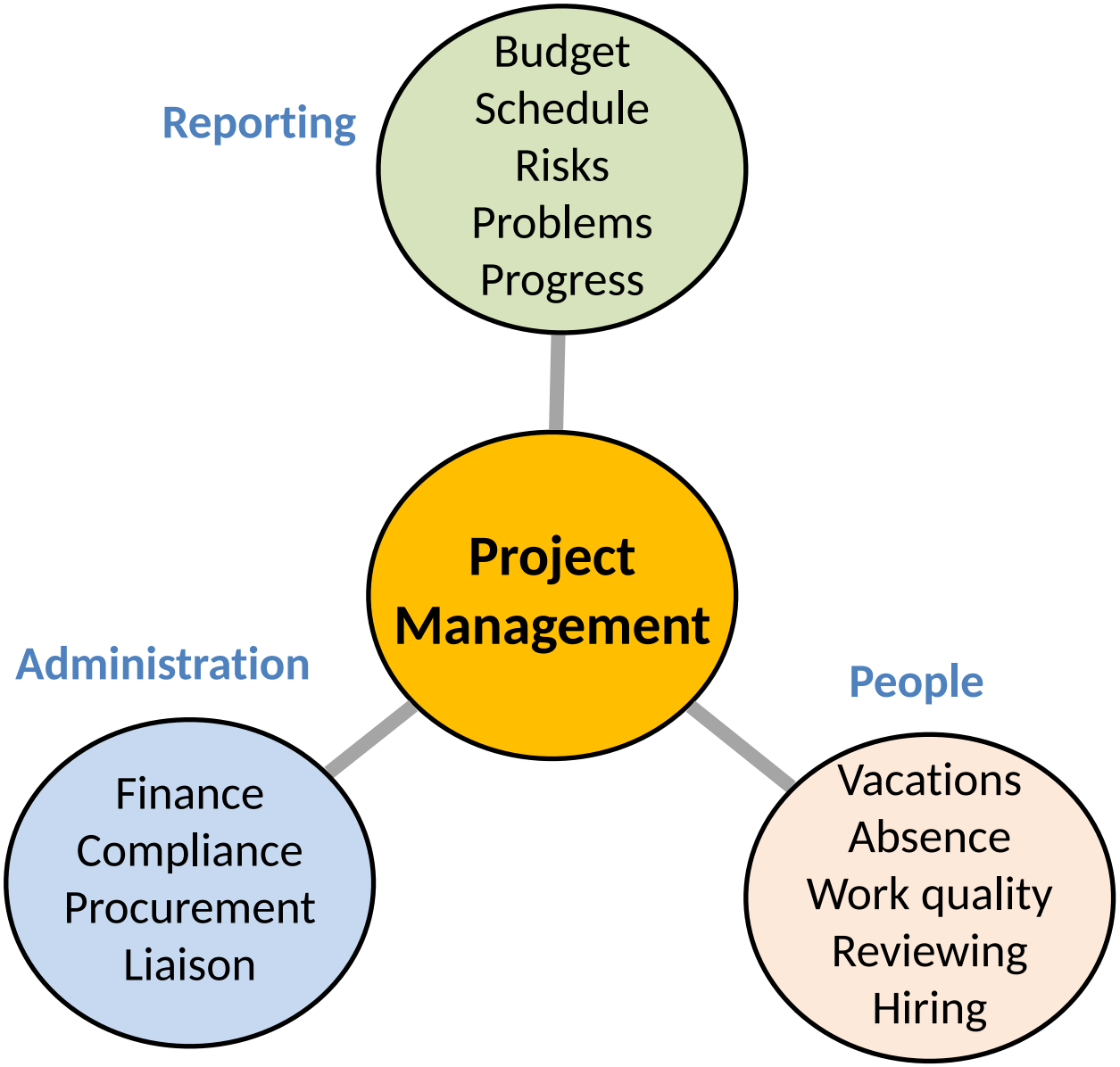


Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

Project management

- In all but the smallest product development companies, there is a need for development teams to report on progress to company management.
- A self-organizing team has to appoint someone to take on these responsibilities.
  - Because of the need to maintain continuity of communication with people outside of the group, rotating these activities around team members is not a viable approach.
- The developers of Scrum did not envisage that the ScrumMaster should also have project management responsibilities.
  - In many companies, however, the ScrumMaster has to take on project management responsibilities.
  - They know the work going on and are in the best position to provide accurate information and project plans and progress.

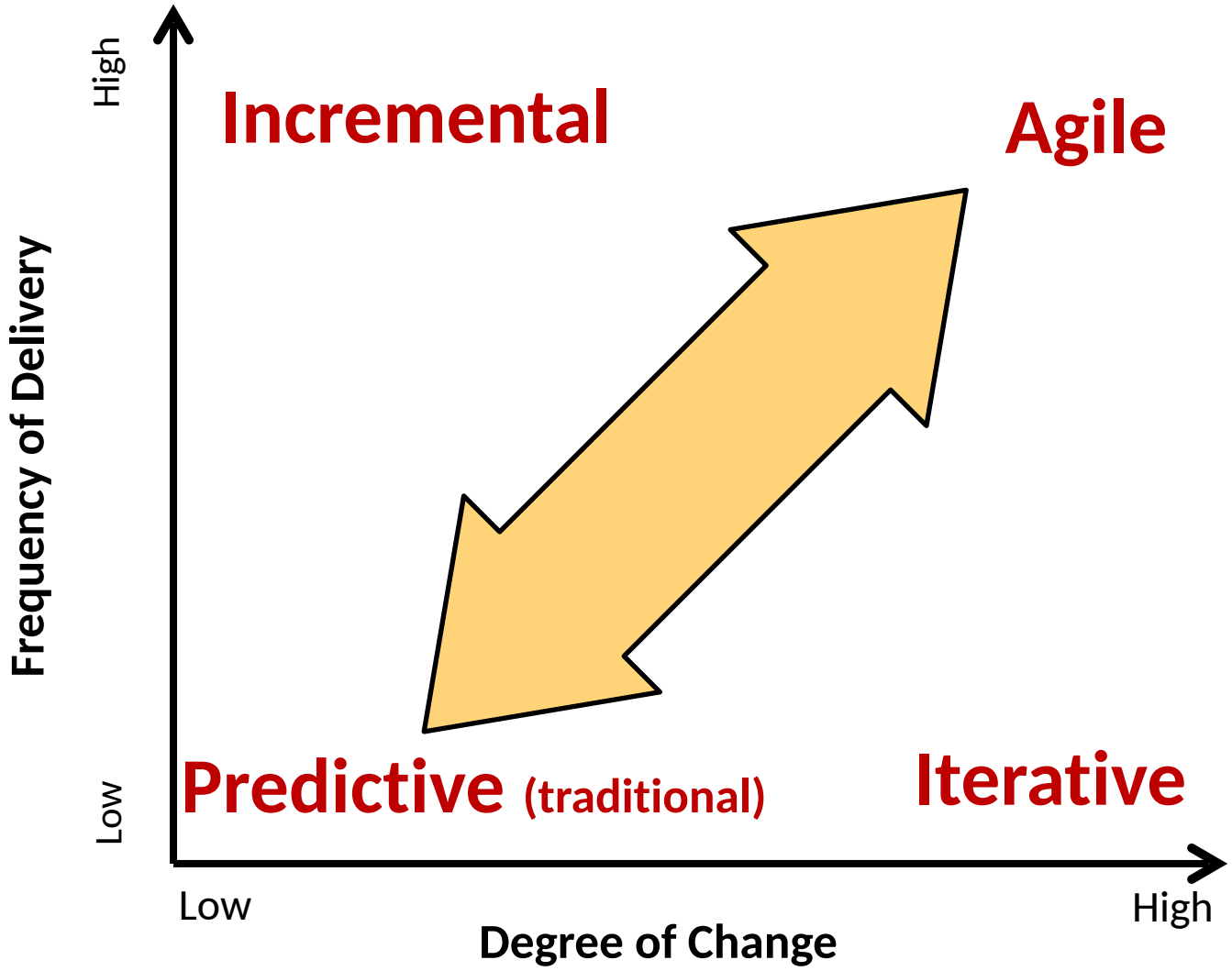
Project Management Responsibilities



h

Source: Ian Sommerville (2019), Engineering Software Products: An Introduction to Modern Software Engineering, Pearson.

The Continuum of Life Cycles



Source: Project Management Institute (2017), Agile Practice Guide, Project Management Institute

Summary

- Agile methods are based around iterative development and the minimization of overheads during the development process.
- Extreme programming (XP) is an influential agile method that introduced agile development practices such as user stories, test-first development and continuous integration. These are now mainstream software development activities.

Summary

- Scrum is an agile method that focuses on agile planning and management. Unlike XP, it does not define the engineering practices to be used. The development team may use any technical practices that they believe are appropriate for the product being developed.
- In Scrum, work to be done is maintained in a product backlog – a list of work items to be completed. Each increment of the software implements some of the work items from the product backlog.

Summary

- Sprints are fixed-time activities (usually 2–4 weeks) where a product increment is developed. Increments should be 'potentially shippable' i.e. they should not need further work before they are delivered.
- A self-organizing team is a development team that organizes the work to be done by discussion and agreement amongst team members.
- Scrum practices such as the product backlog, sprints and self-organizing teams can be used in any agile development process, even if other aspects of Scrum are not used.