Agile Software Development

Agile software development

- Rapid software development and delivery is the most critical requirement for most business systems.
- Rapid software development became known as agile development or agile methods.
- Common characteristics of all the agile methods:
  - The processes of specification, design and implementation are interleaved.
  - The system is developed in a series of increments
  - Extensive tool support is used to support the development process
- Agile methods are incremental development methods in which the increments are small. They involve customers in the development process and minimize documentation.

Agile software development

- Agile methods do not have a separate requirements engineering activity. Rather, they integrate requirements elicitation with development.
- Agile methods have been particularly successful for two kinds of system development:
  - Product development where a software company is developing a small or medium-sized product for sale.
  - Custom system development within an organization, where there is a clear commitment from the customer to become involved in the development process.

Scrum Method

- Agile development has to be managed so that the best use is made of the time and resources available to the team.
- Scrum agile method was developed (Schwaber and Beedle 2001; Rubin 2013) to provide a framework for organizing agile projects
- Scrum has emerged as the most widely used agile method.

User stories

- Requirements are expressed as a scenario of use that might be experienced by a system user, called user stories.
- Break down stories to tasks.
- Estimates the effort and resources required for implementing each task.
- The customer then prioritizes the stories for implementation, to identify useful functionality that can be implemented in about two weeks.

User stories

- The User Story can take the form:

  As a < type of user >, I want < some goal > so that < some reason >.

- Examples:
  - As a manager, I want to be able to generate progress report, so that I can understand my employees' progress.

User stories

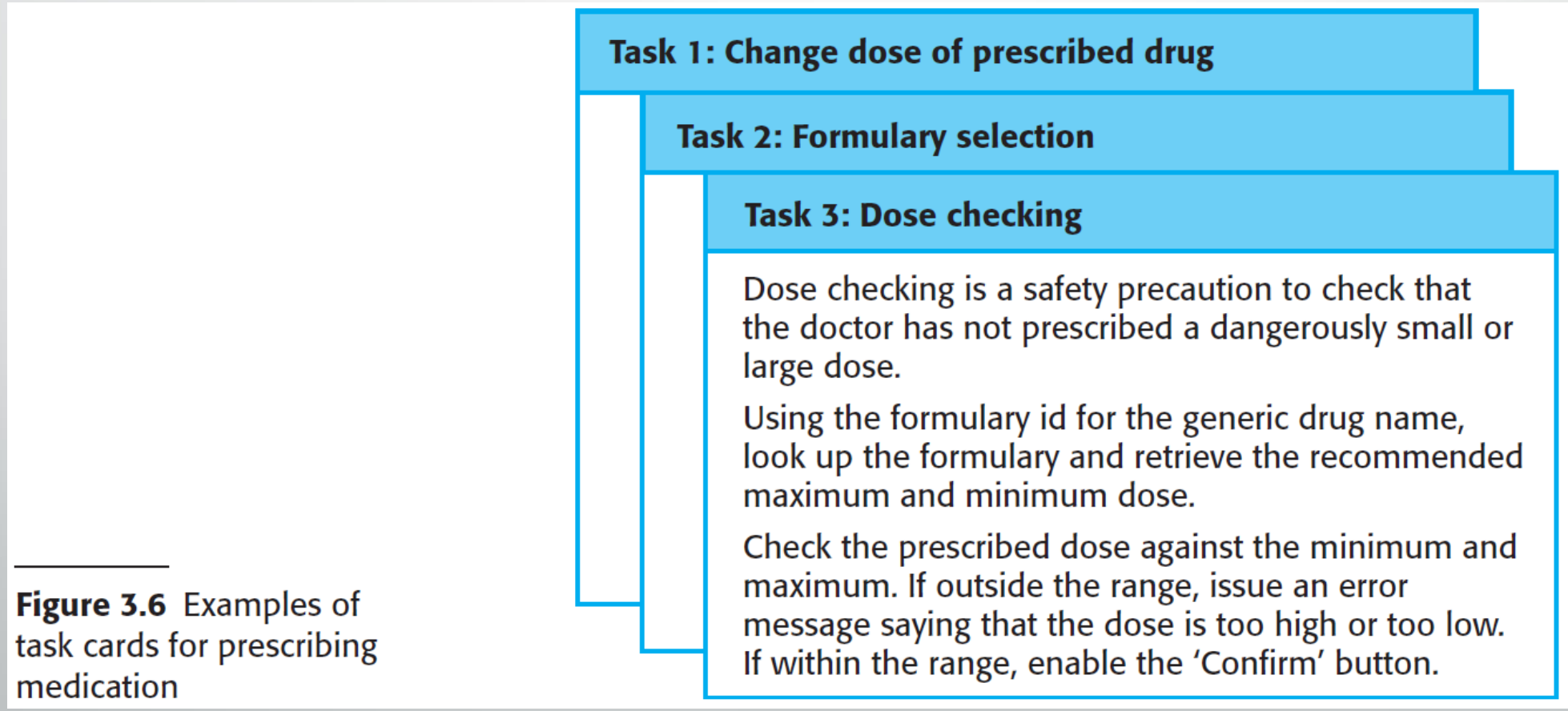


Figure 3.6 Examples of task cards for prescribing medication

Test Case

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.

Figure 3.7 Test case description for dose checking

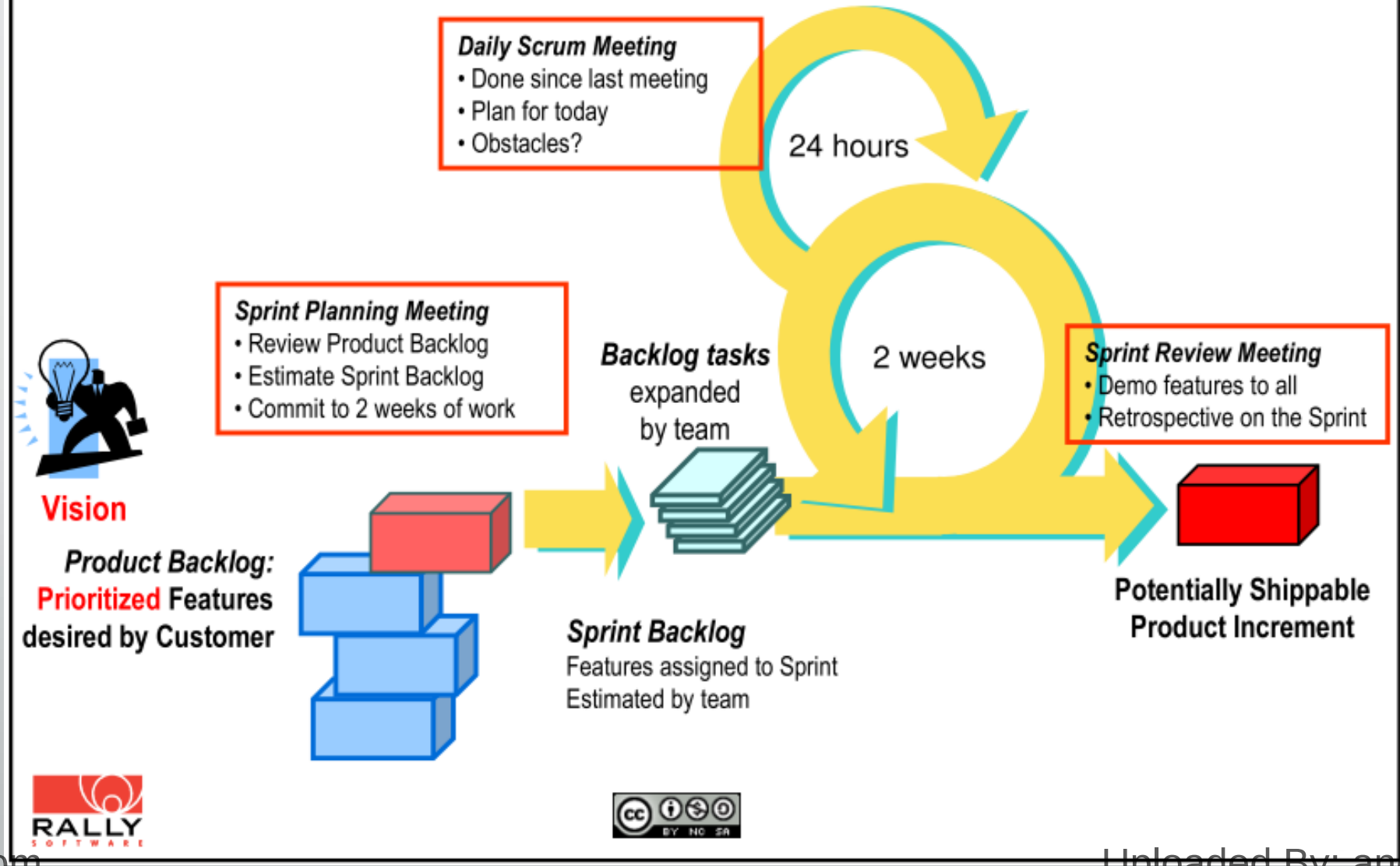
Scrum terminology

Scrum term	Definition
Development team	A self-organizing group of software developers, which should be no more than seven people. 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 that 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	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 terminology

ScrumMaster	The ScrumMaster 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 developers are adamant that the ScrumMaster should not be thought of as a project manager. Others, however, may not always find it easy to see the difference.
Sprint	A development iteration. Sprints are usually 2 to 4 weeks long.
Velocity	An estimate of how much product backlog effort 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.

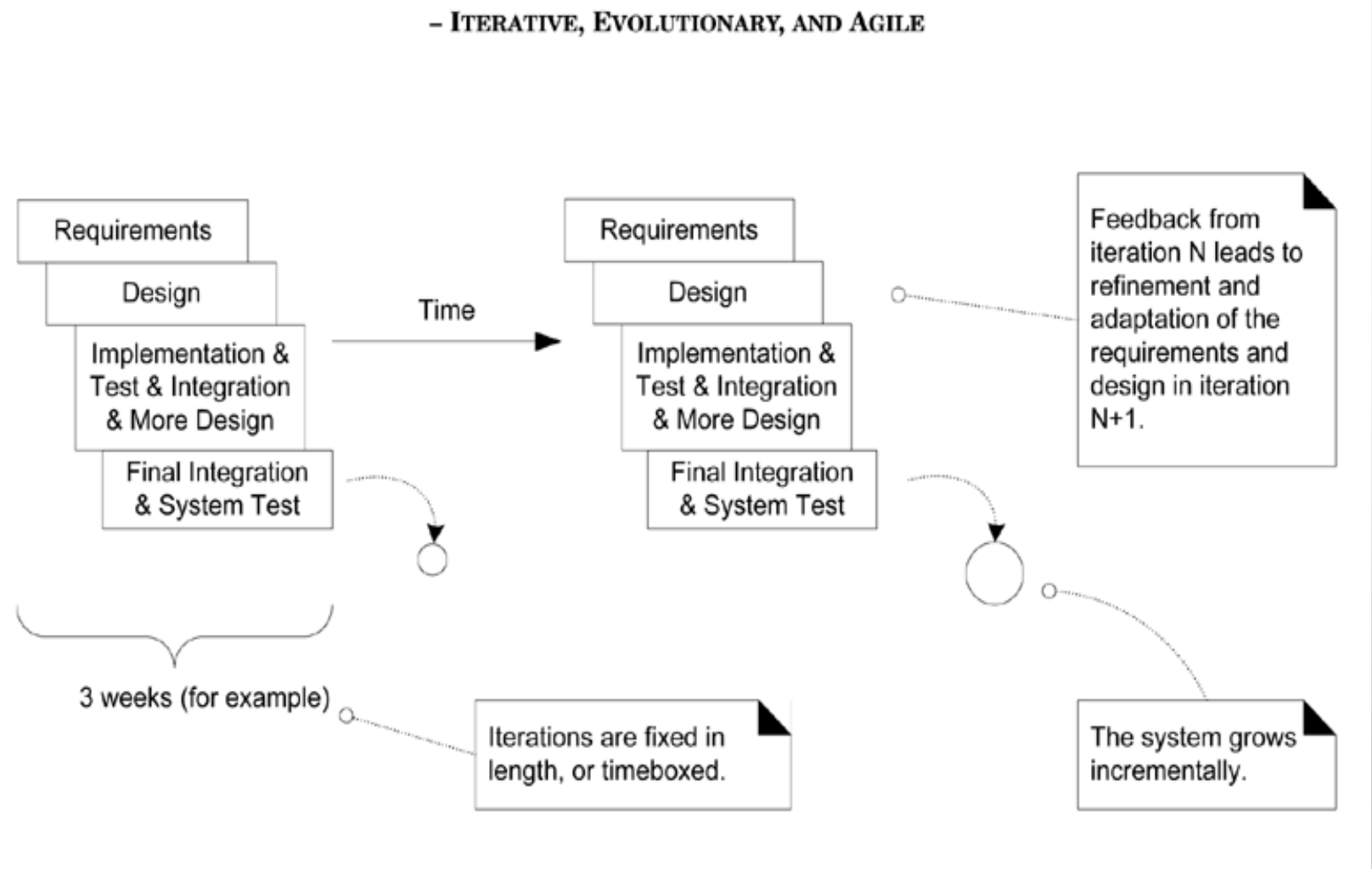
Presto: The Scrum Framework!

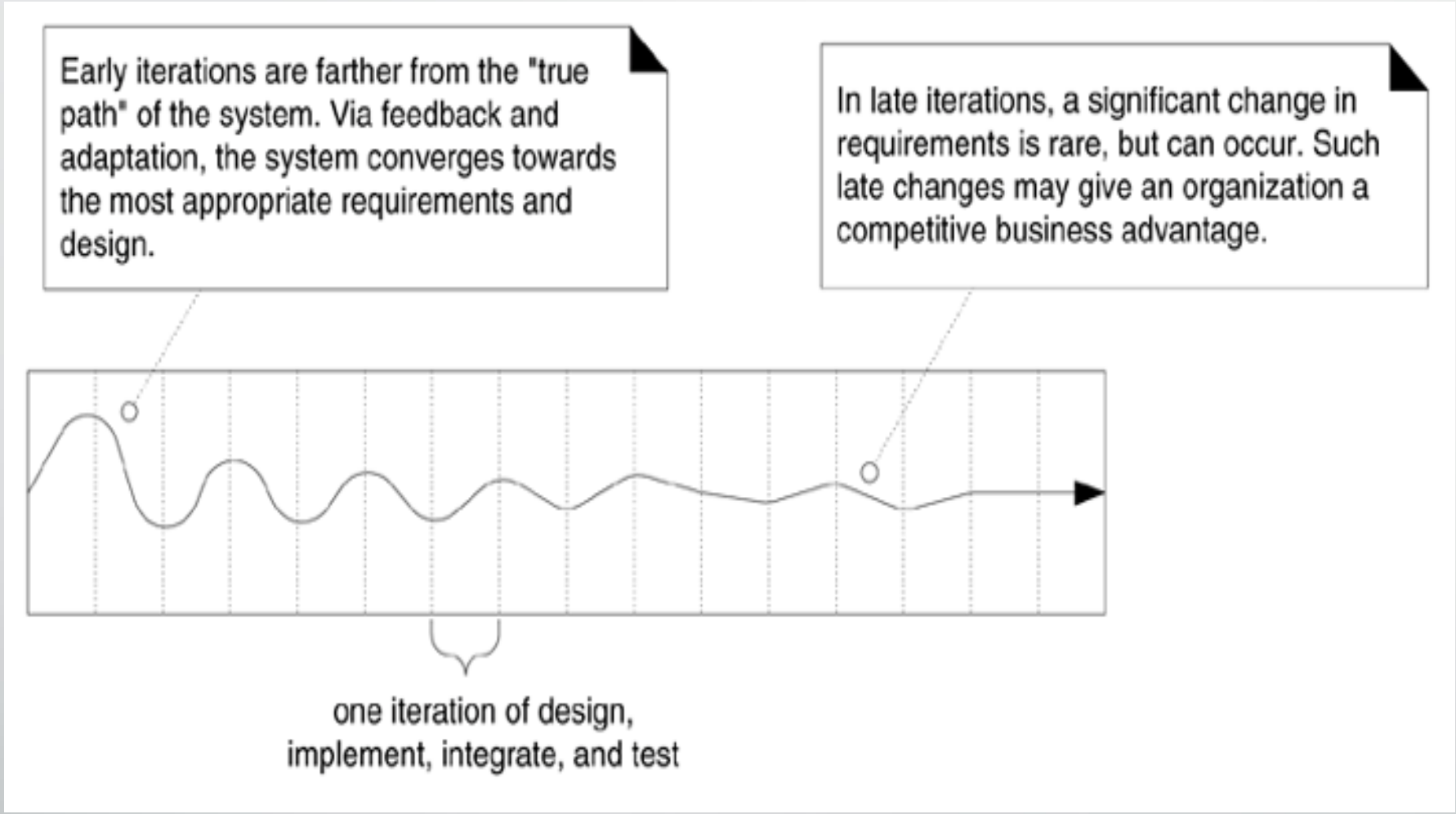


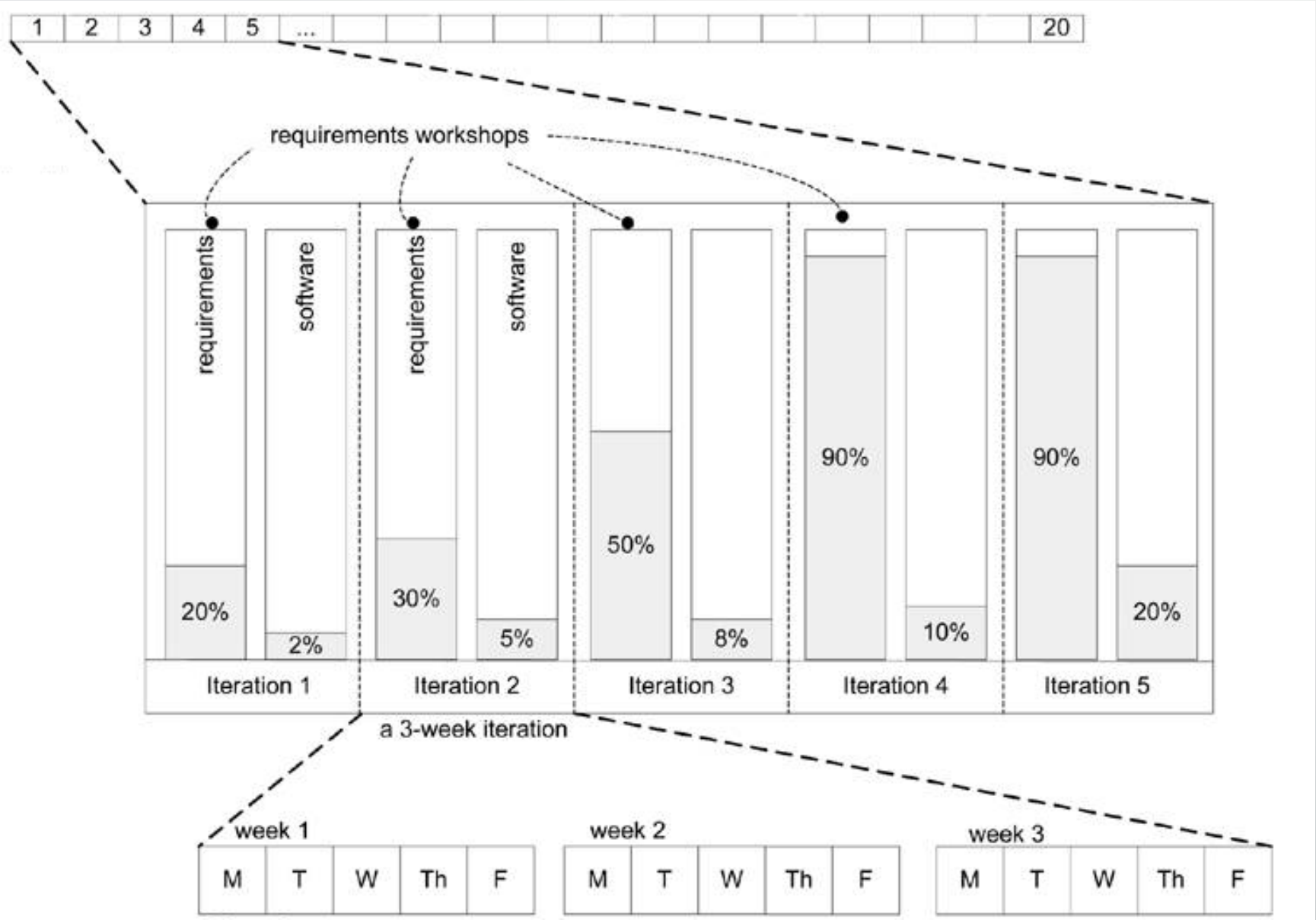
Scrum

- At the beginning of each cycle, the Product Owner prioritizes the items on the product backlog to define which are the most important items to be developed in that cycle.
- They then estimate the time required to complete these items. To make these estimates, they use the velocity attained in previous sprints
- This leads to the creation of a sprint backlog—the work to be done during that sprint. The team self-organizes to decide who will work on what, and the sprint begins.
- During the sprint, the team holds short daily meetings (Scrums or Stand-up meetings) to review progress and, where necessary, to re-prioritize work. During the Scrum, all team members share information, describe their progress since the last meeting, bring up problems that have arisen, and state what is planned for the following day.
- At the end of each sprint, there is a review meeting, which involves the whole team. This meeting has two purposes. First, it is a means of process improvement. Second, it provides input on the product and the product state for the product backlog review that precedes the next sprint.

Core Practices - Iterations







Extreme programming-XP

- Small releases: very small increments of functionality (often no longer than two weeks).
- Collective ownership: anyone can change anything.
- Continuous integration: as soon as the work on a task is complete, it is integrated into the whole system. After any such integration, all the unit tests in the system must pass.
- Incremental planning.
- On-site customer: a representative of the end-user of the system (the Customer) should be available full time for the use of the XP team. In an extreme programming process, the customer is a member of the development team.
- Pair programming: a Coder and a Reviewer work together.
- Refactoring.
- Simple design.
- Test first Development: an automated unit test framework is used to write tests for a new piece of functionality before that functionality itself is implemented.

Extreme programming-XP