Agile Software Development for Scalable Internet Services

Andrew Mutz
Director of Software Engineering, Appfolio
October 9th, 2013

Introduction

- What is Agile?
- The Agile workflow
- Supporting Tools and Techniques
  - Burndown Charts & Scrum Board
  - Test-Driven Development
  - Continuous Integration
  - Pair Programming

What is Agile?

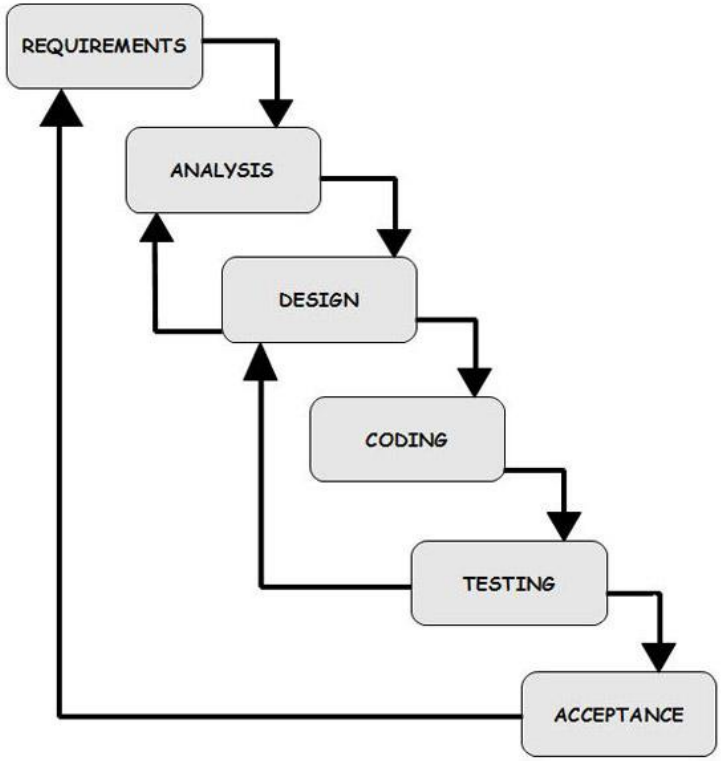
Responding to change > Following a plan

versus

What is Agile?

Individual & interactions > Process

versus



What is Agile?

Customer collaboration > Frozen requirements

"Here's what we've got so far."

"Uh... Now that I see it in action, I've changed my mind."

versus

"I'm sure if I just build what they've asked for, they'll love it."

What is Agile?

Code > Documentation

- Demonstration to customer
- Automated acceptance tests
- Descriptive unit tests
- Centralized styling & view code

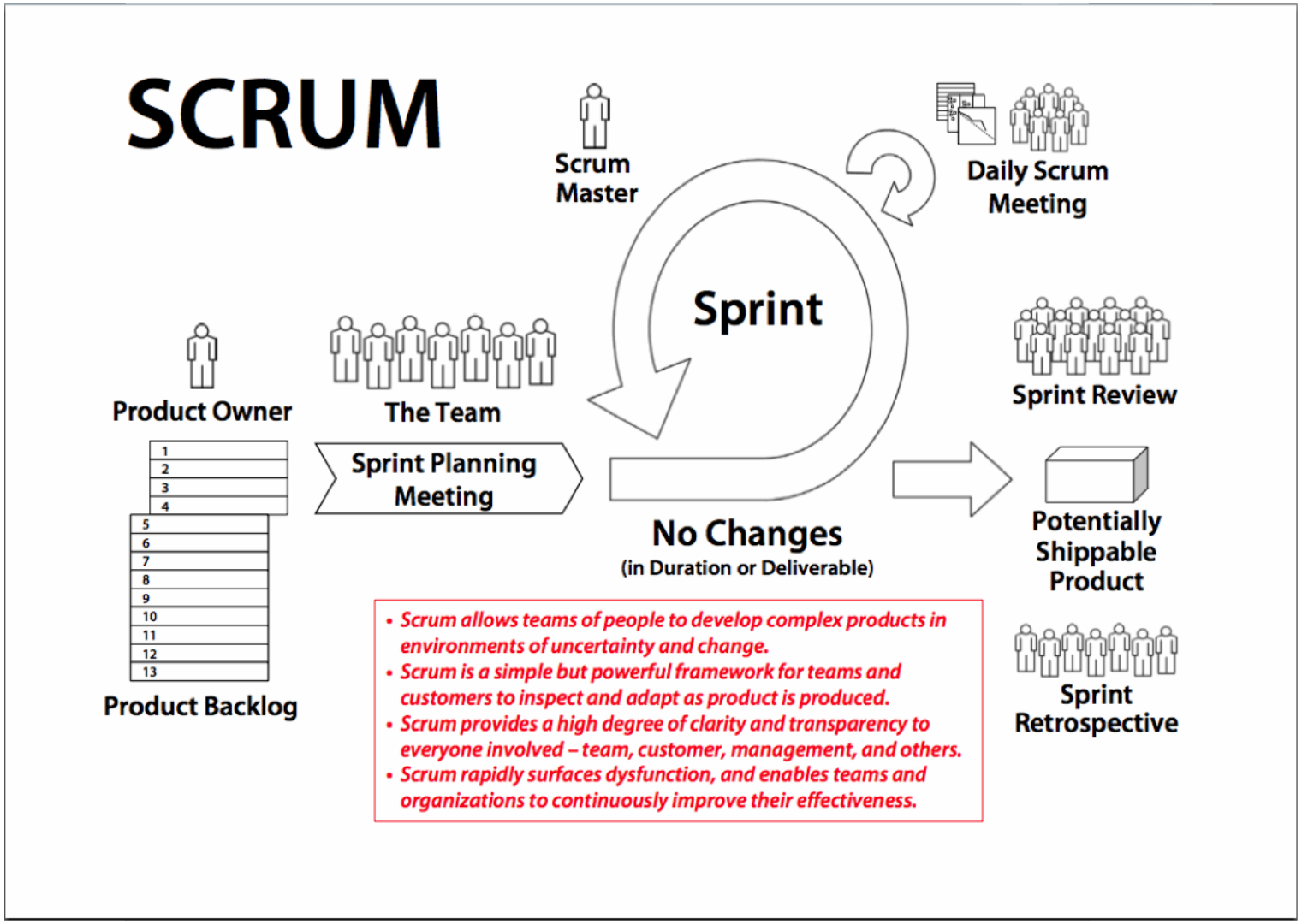
versus

- Description to customer
- Extensive documented requirements
- Code comments
- Style guide documentation

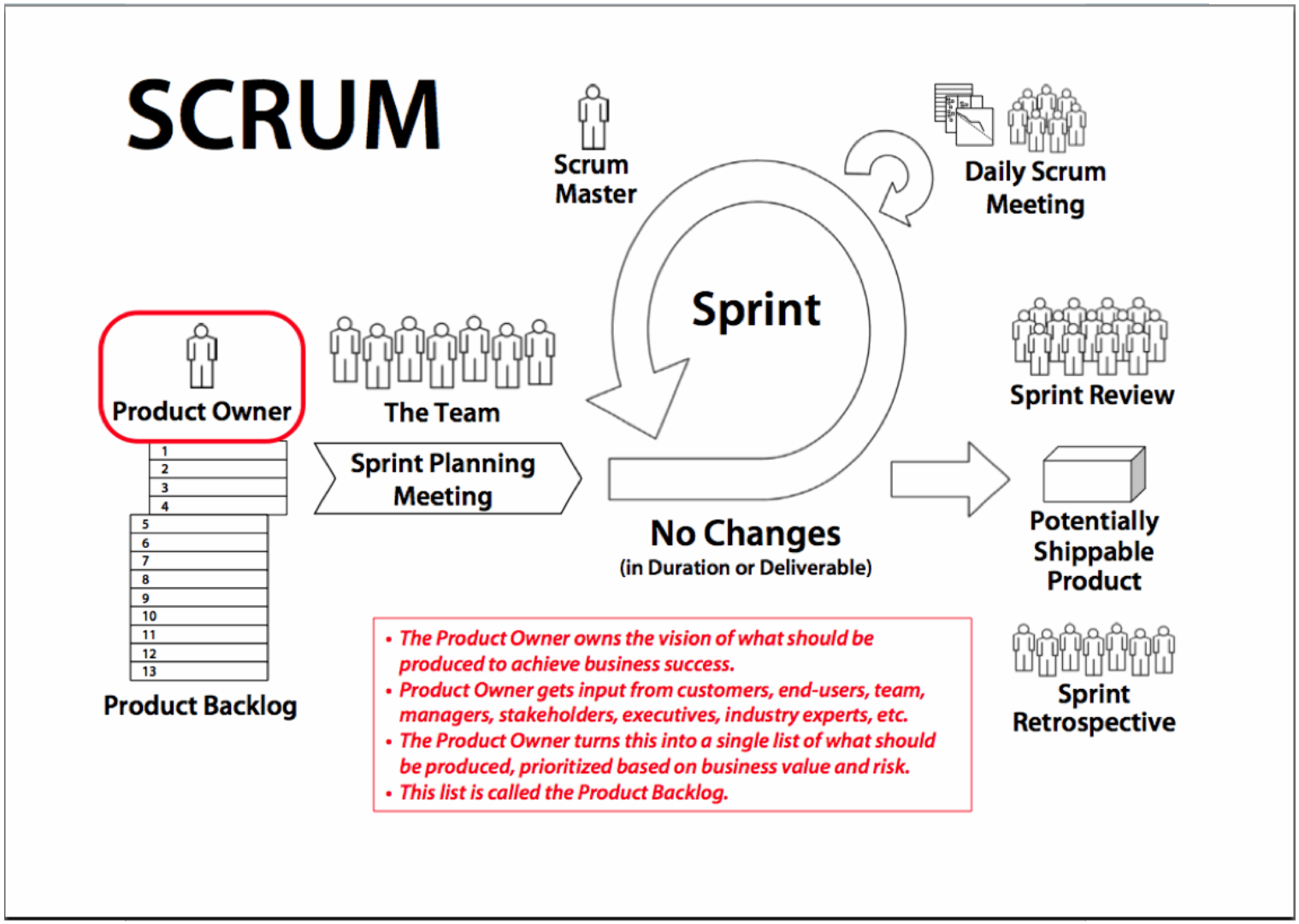
What is Agile?

Waterfall	Scrum
Write documents.	Write code to demo.
Predictive - Frozen requirements months in advance. Fight change.	Empirical. Accept change. Re-prioritize and adapt
Silos - QA, UI, product separate depts.	All one cross-functional team.
Everything is required.	Prioritized backlog of work.
Update biz people infrequently.	Customer accepts work daily. Demo at end.
Individual responsibility for features.	Shared responsibility for features

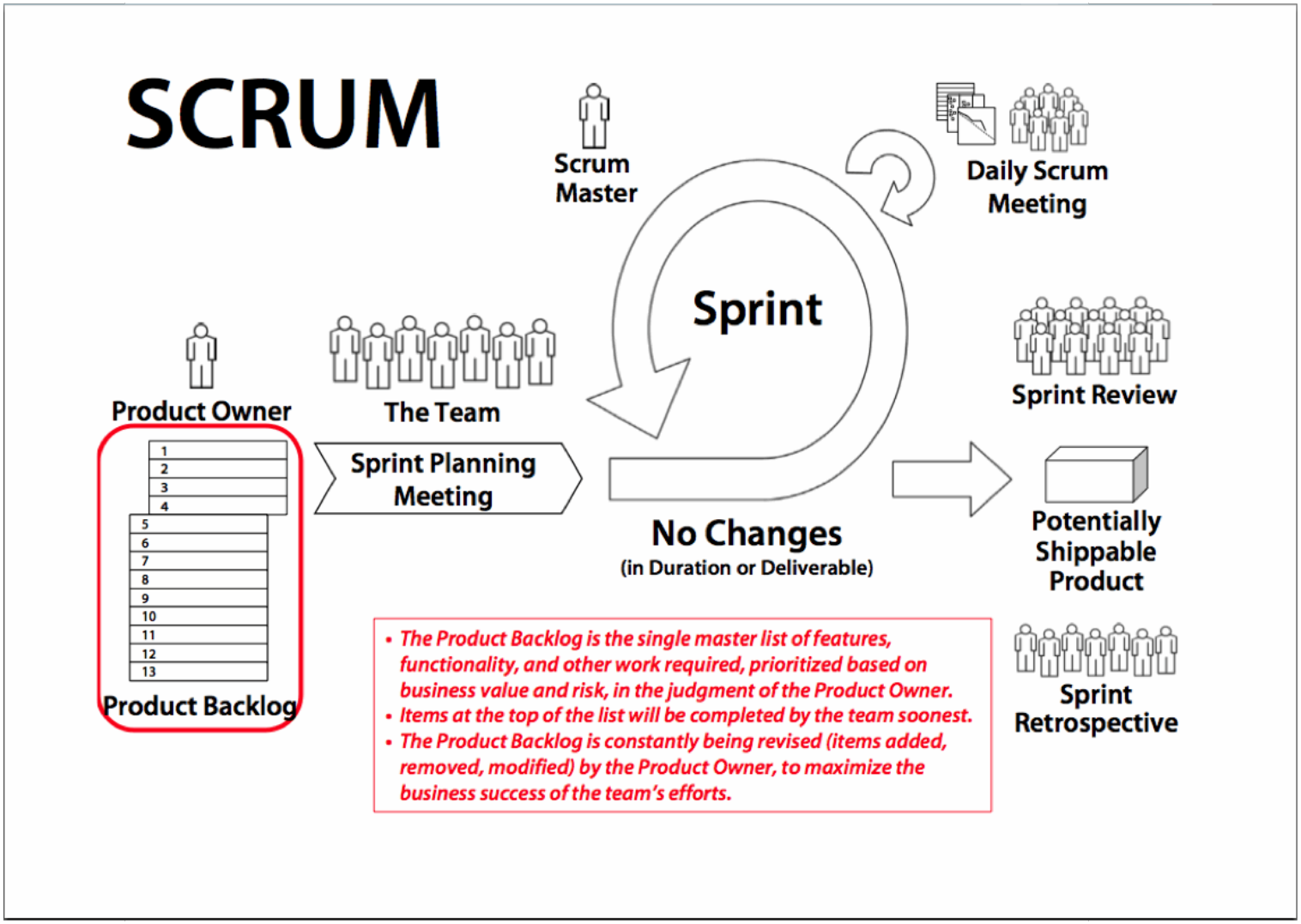
Agile Workflow with Scrum



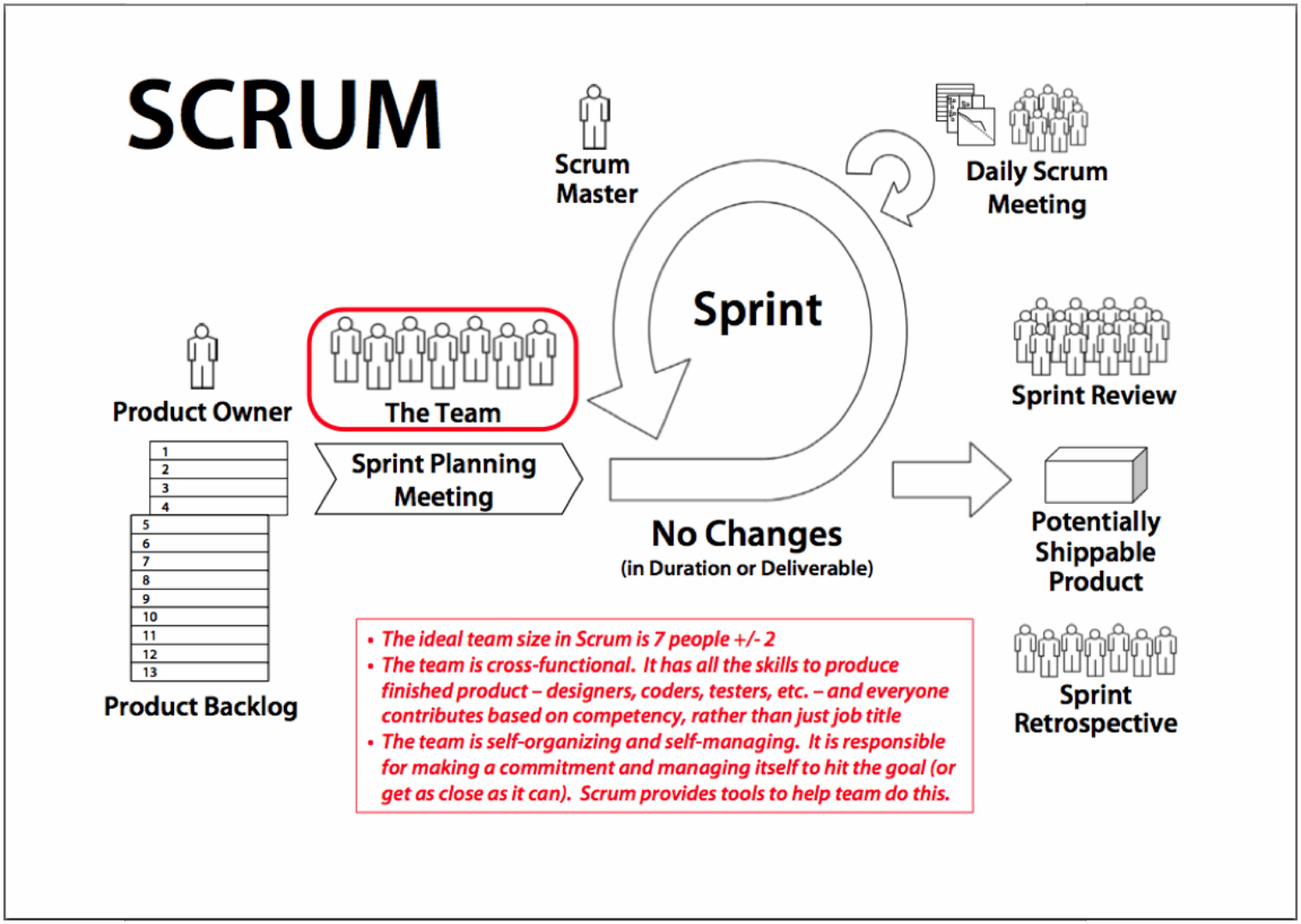
Agile Workflow with Scrum



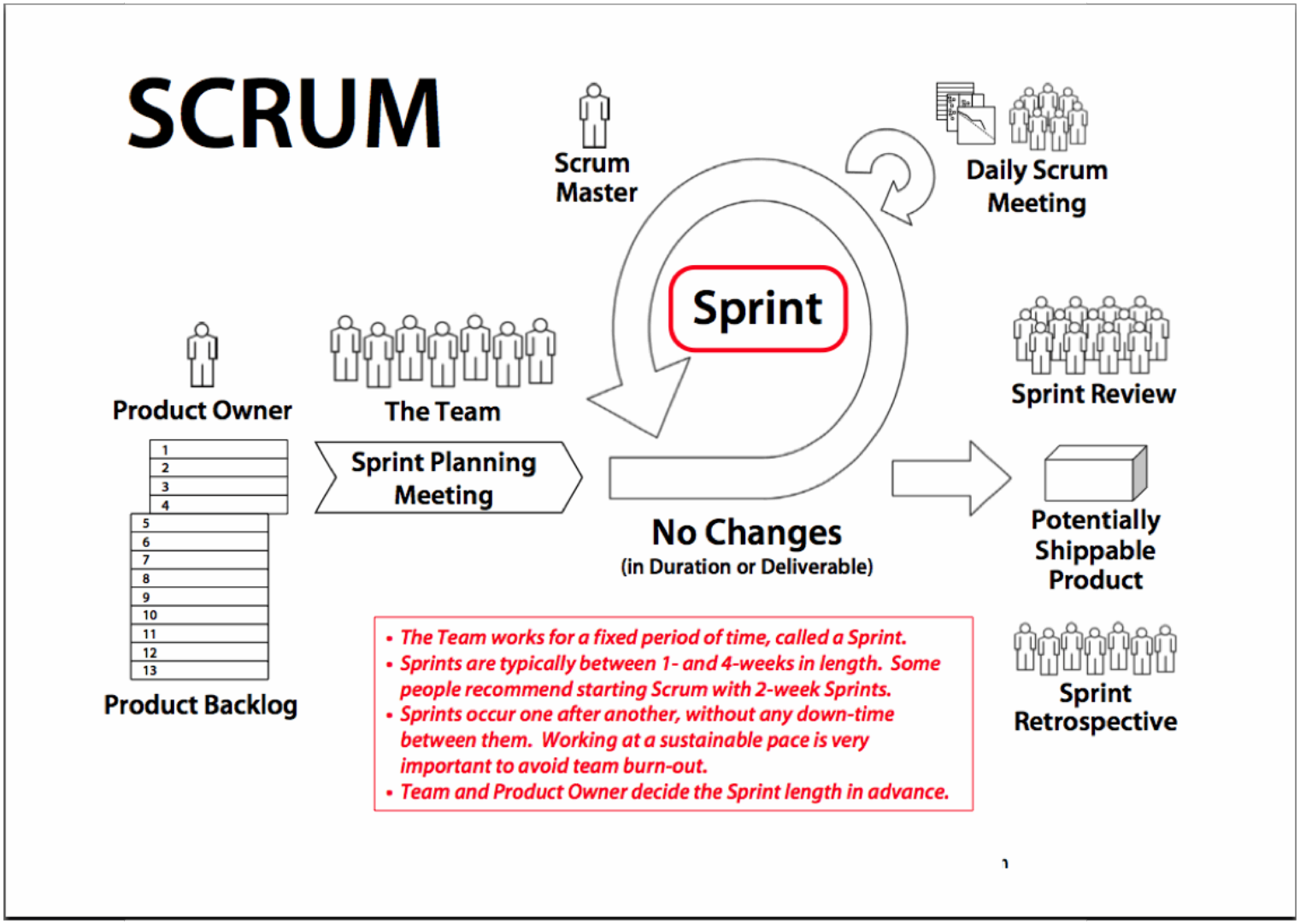
Agile Workflow with Scrum



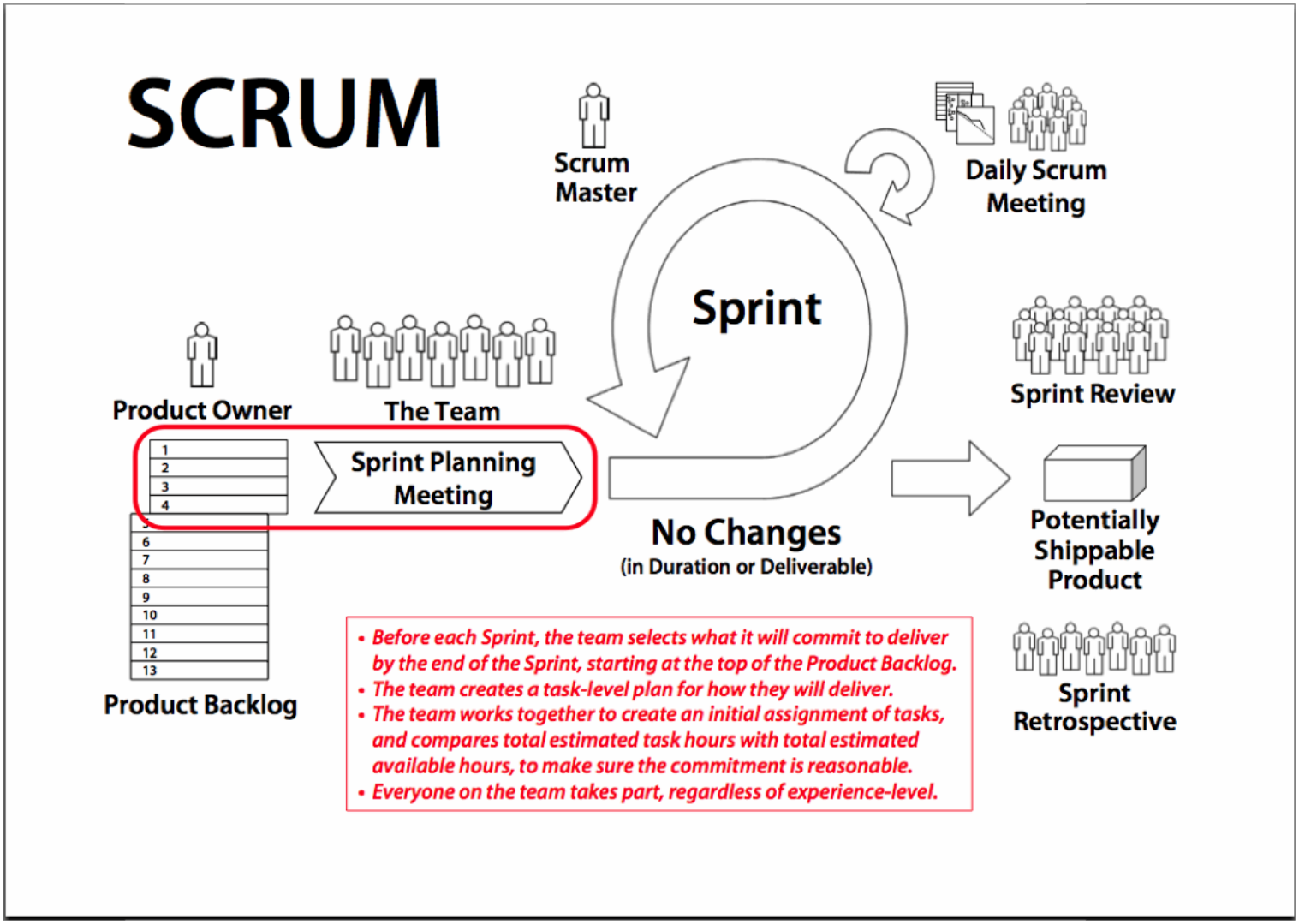
Agile Workflow with Scrum



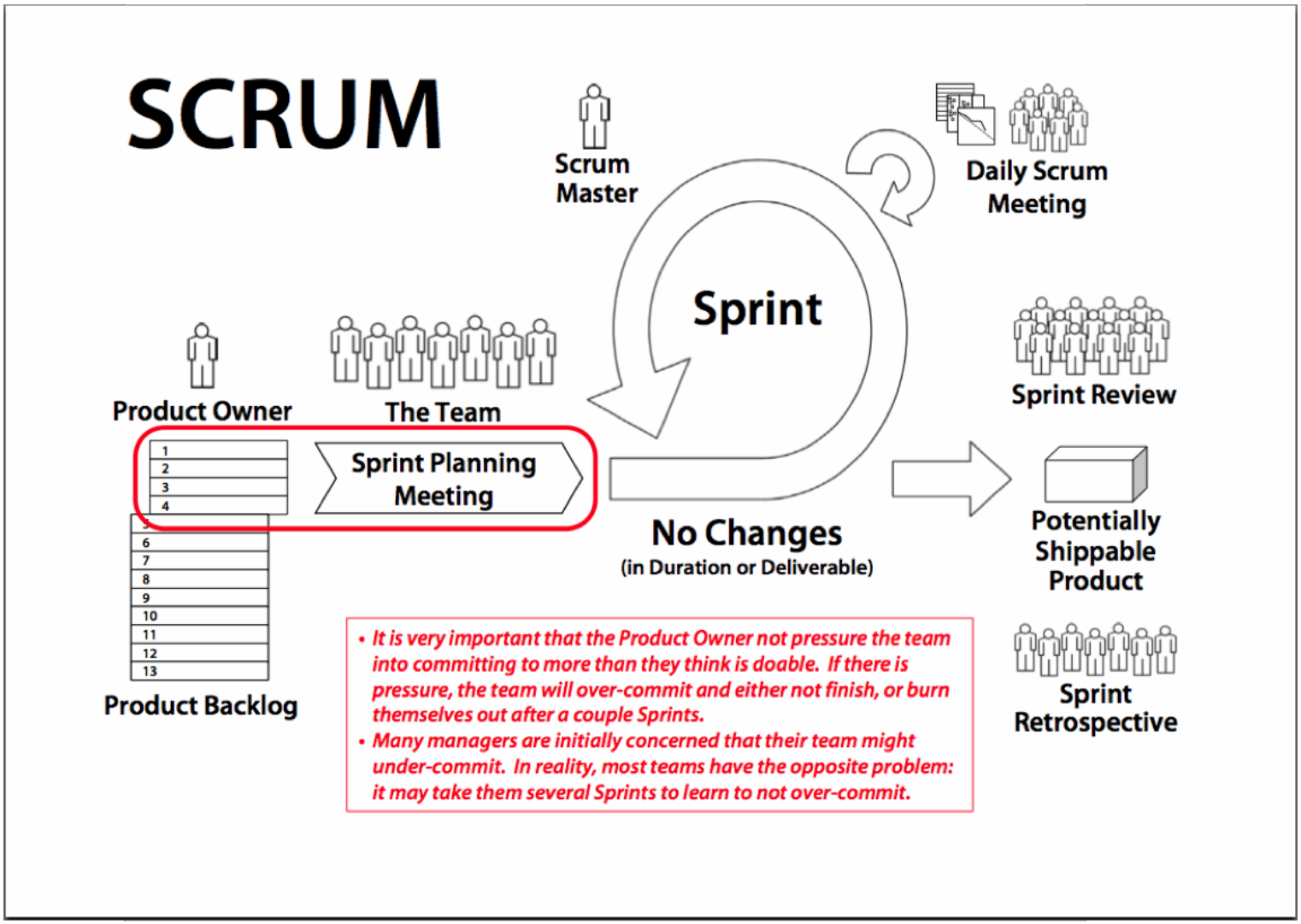
Agile Workflow with Scrum



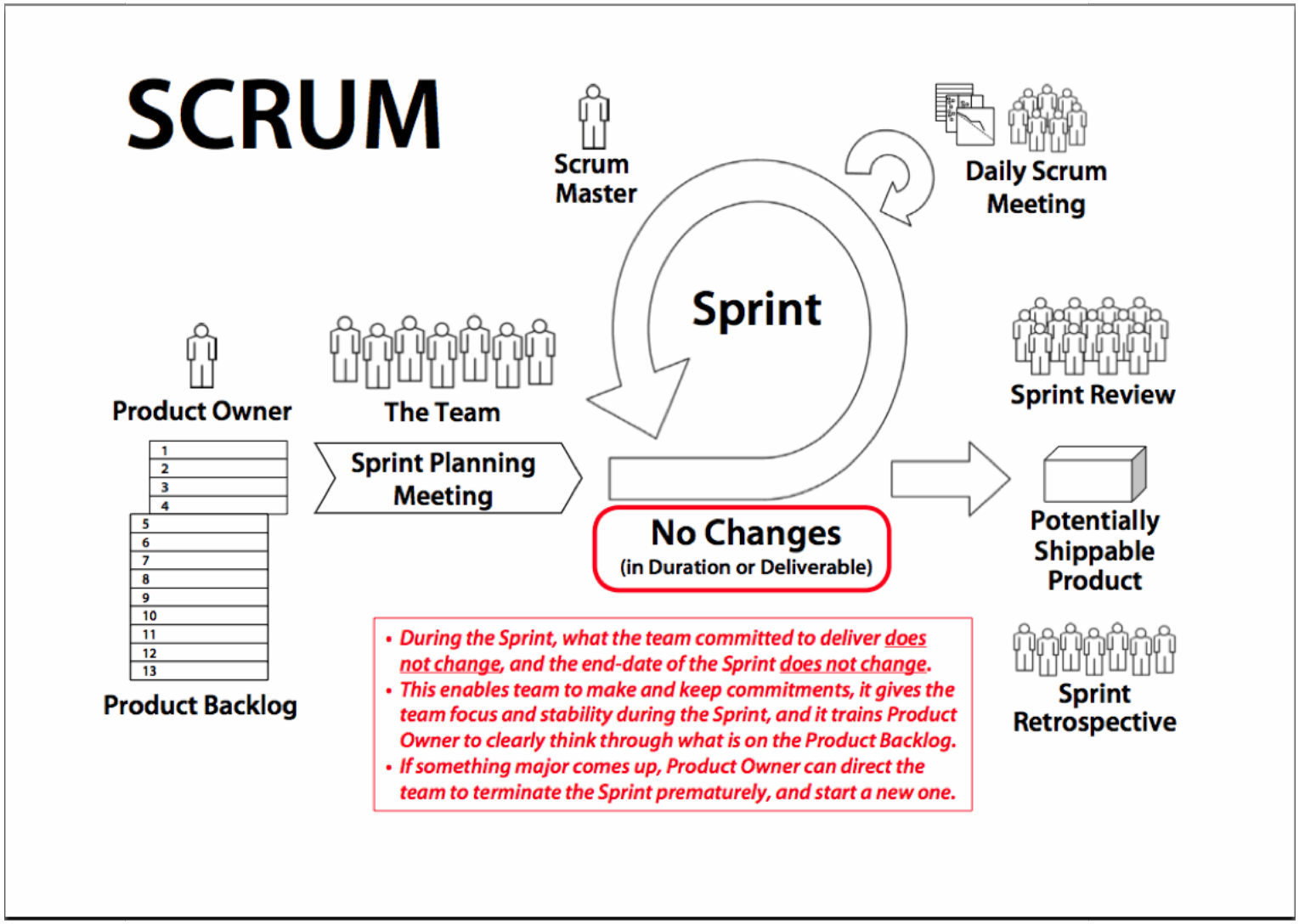
Agile Workflow with Scrum



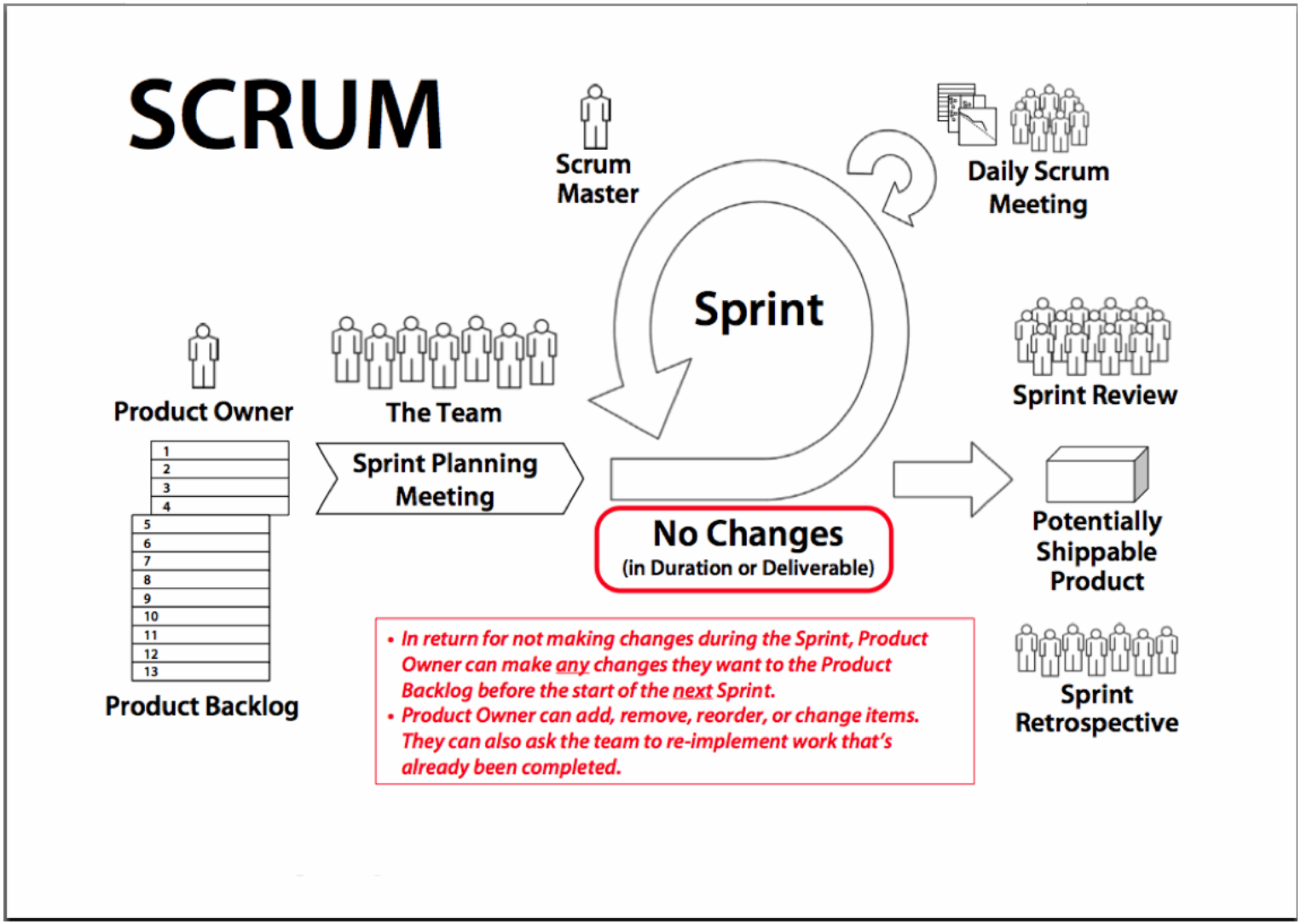
Agile Workflow with Scrum



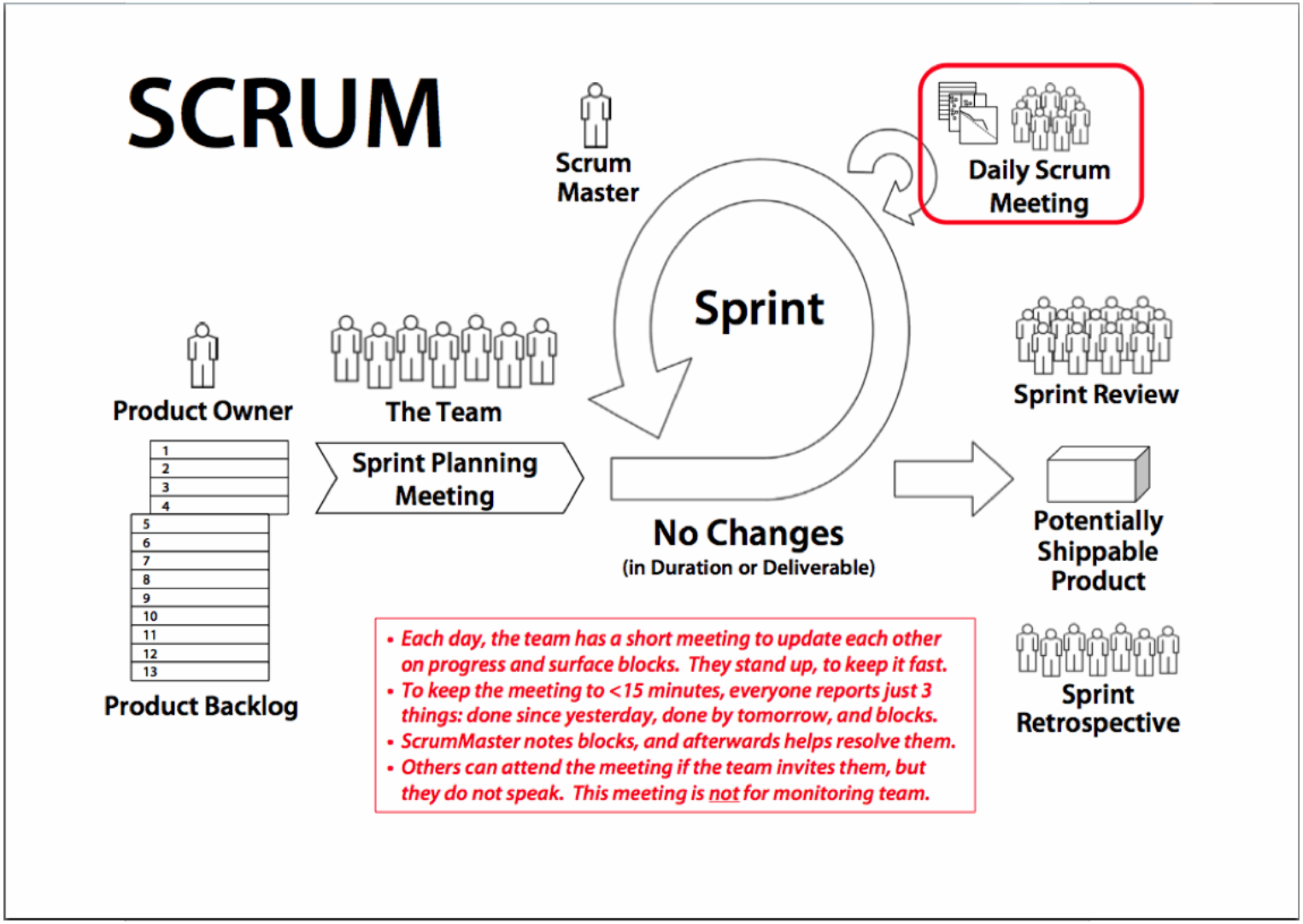
Agile Workflow with Scrum



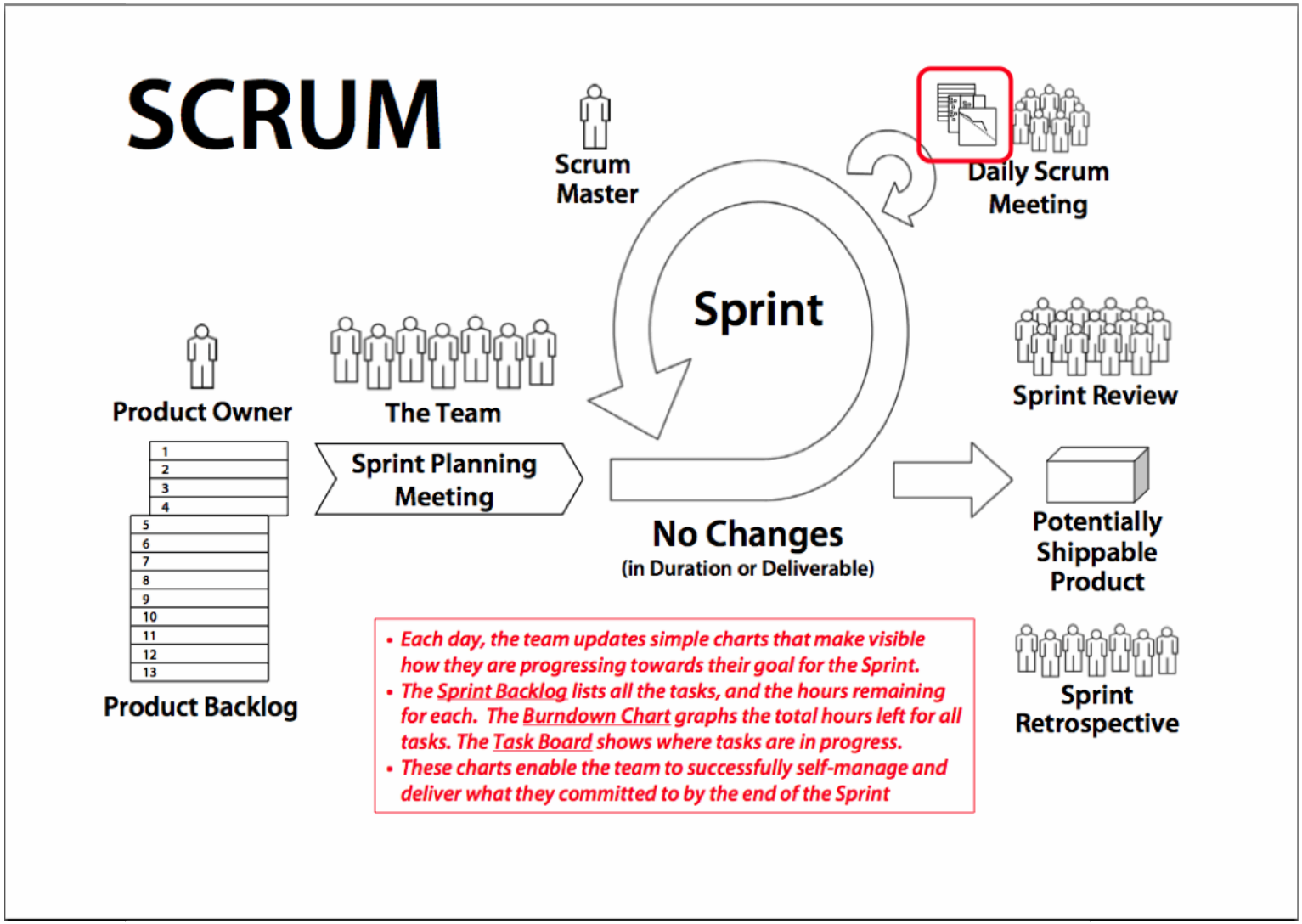
Agile Workflow with Scrum



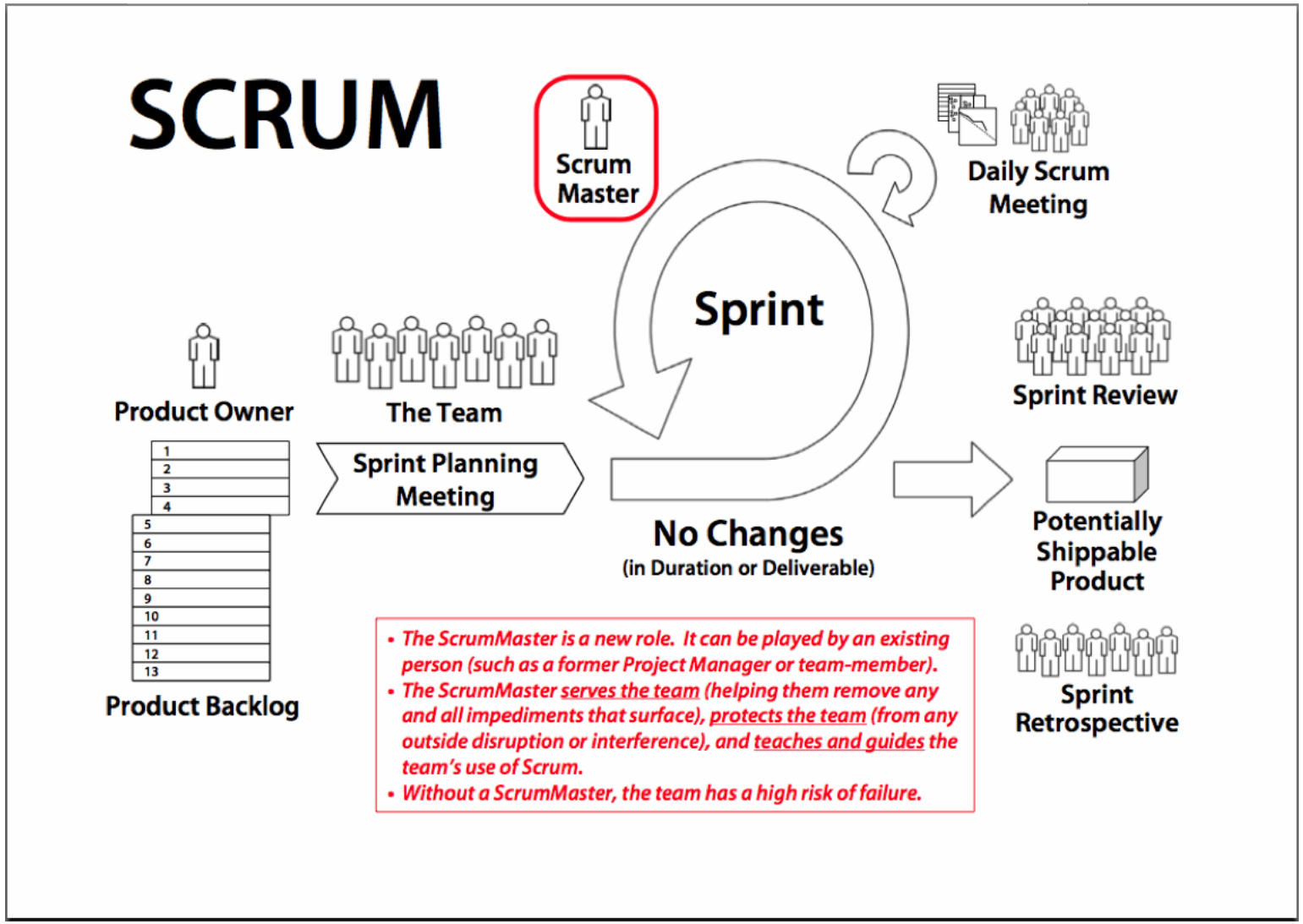
Agile Workflow with Scrum



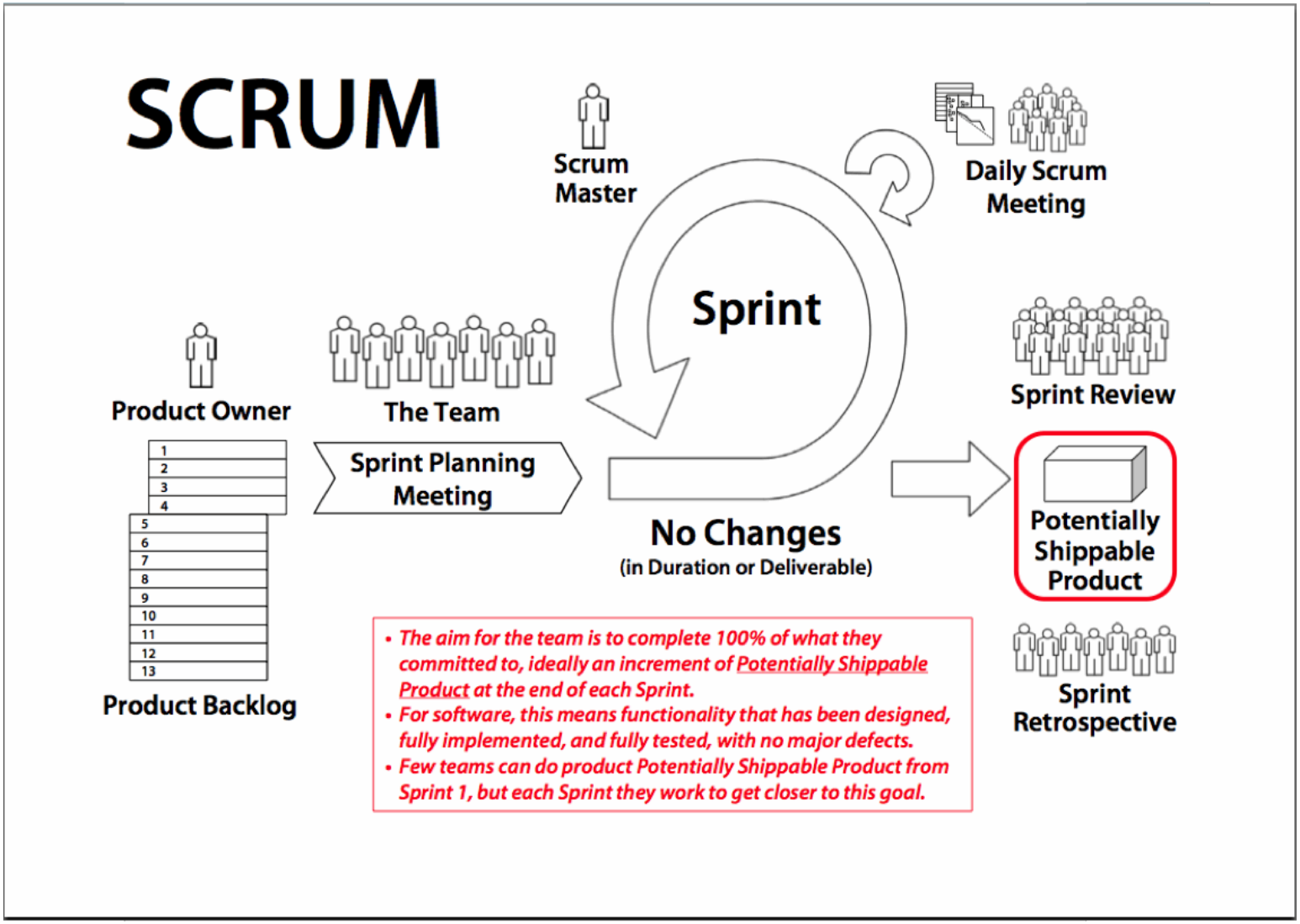
Agile Workflow with Scrum



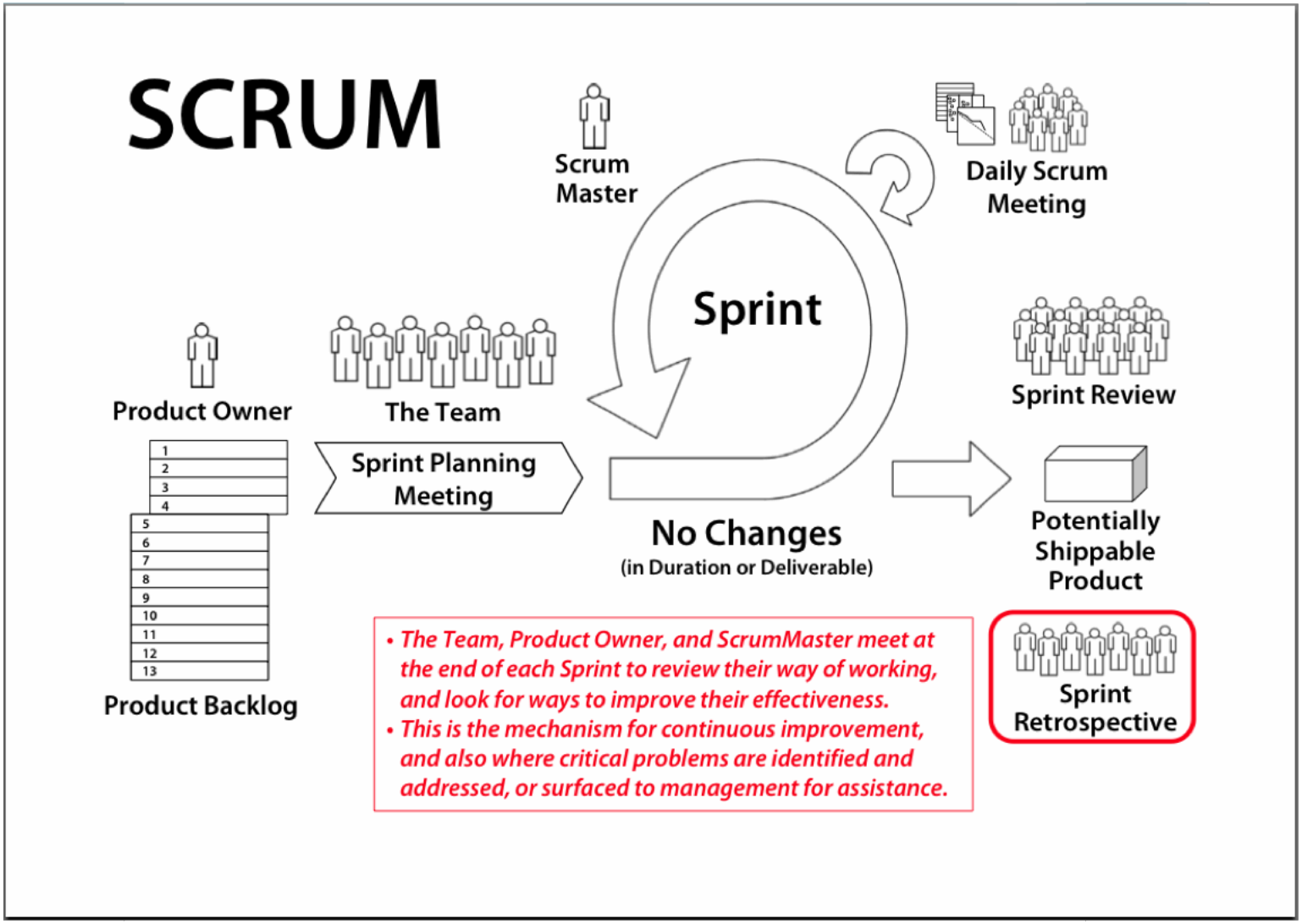
Agile Workflow with Scrum



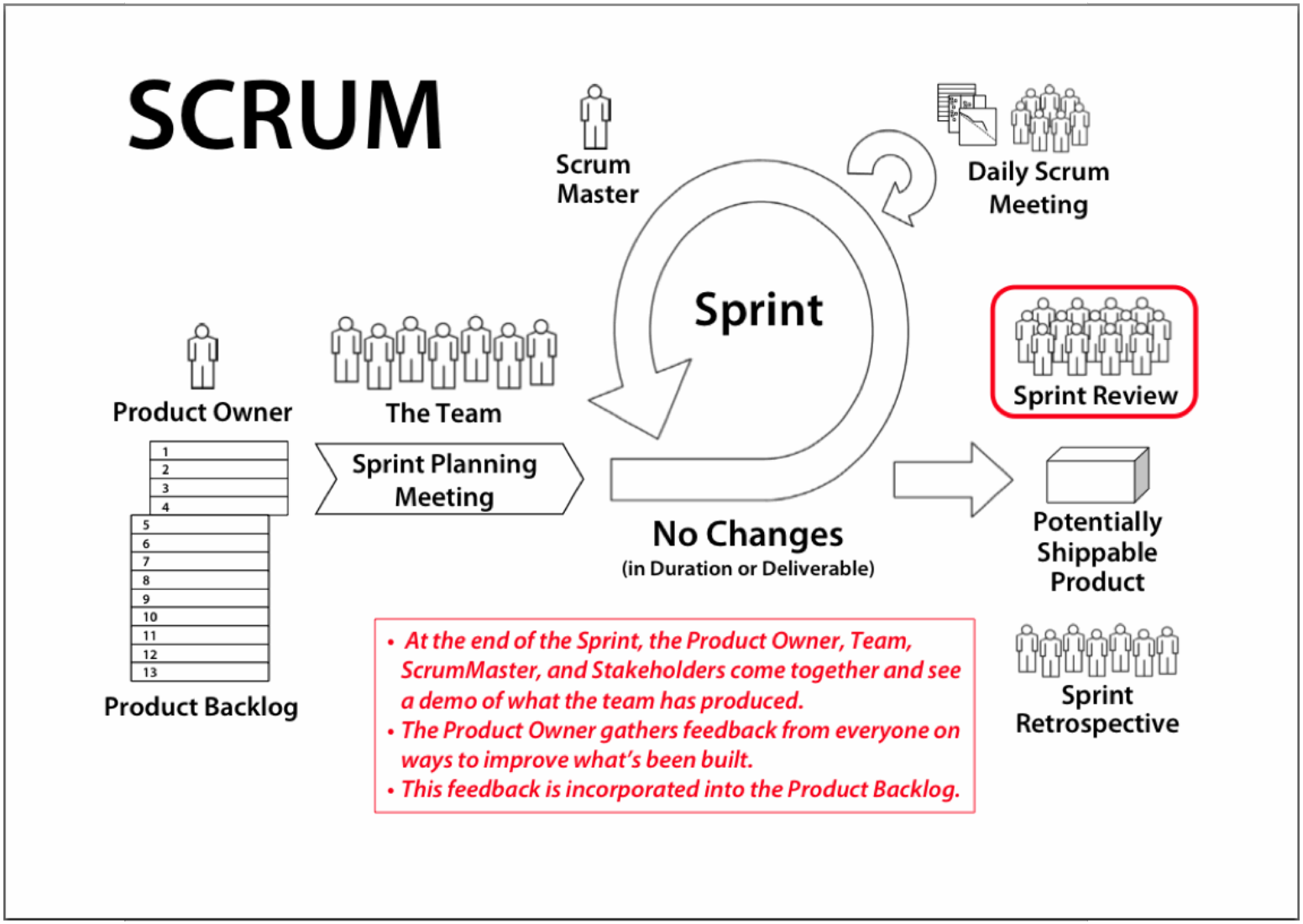
Agile Workflow with Scrum



Agile Workflow with Scrum

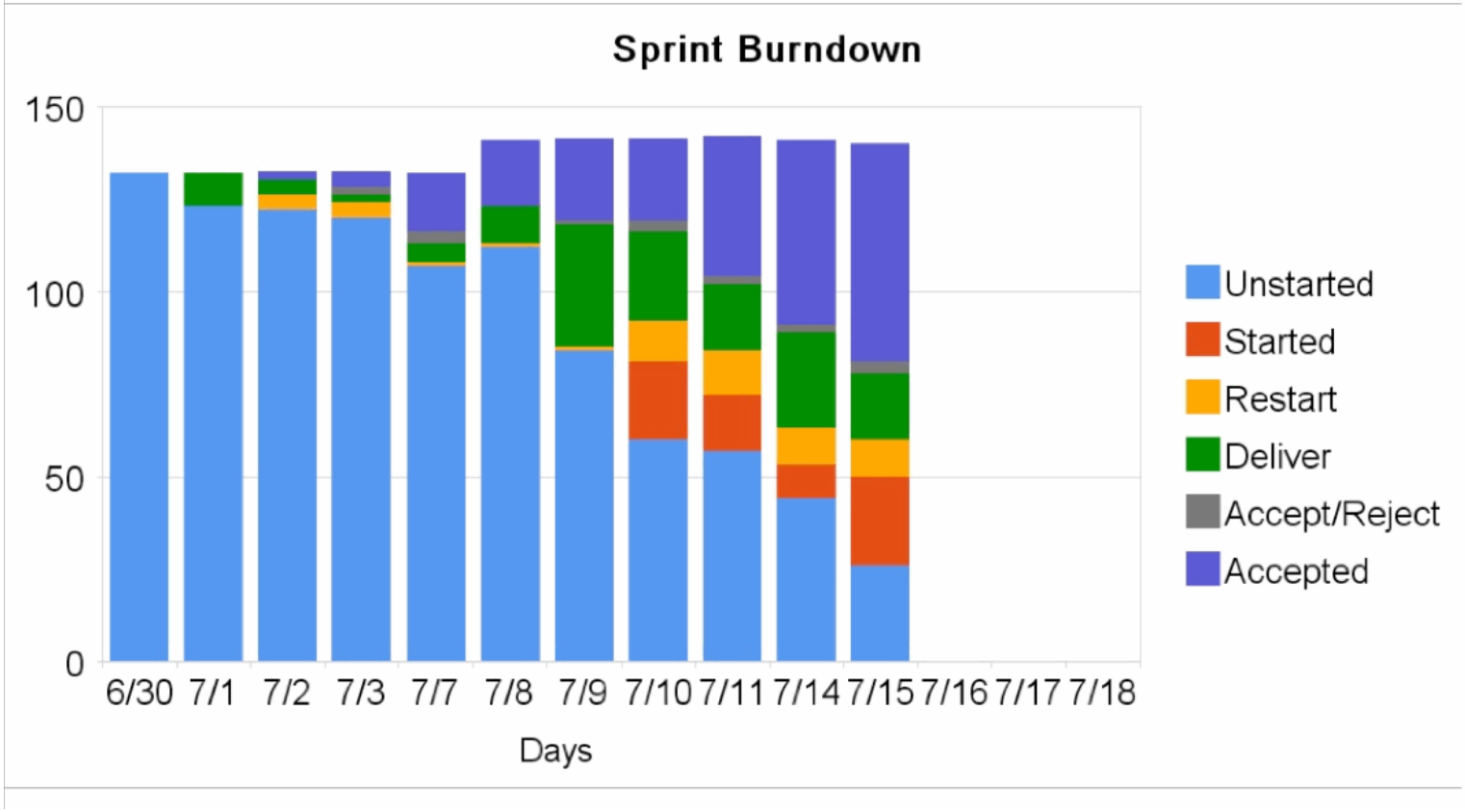


Agile Workflow with Scrum

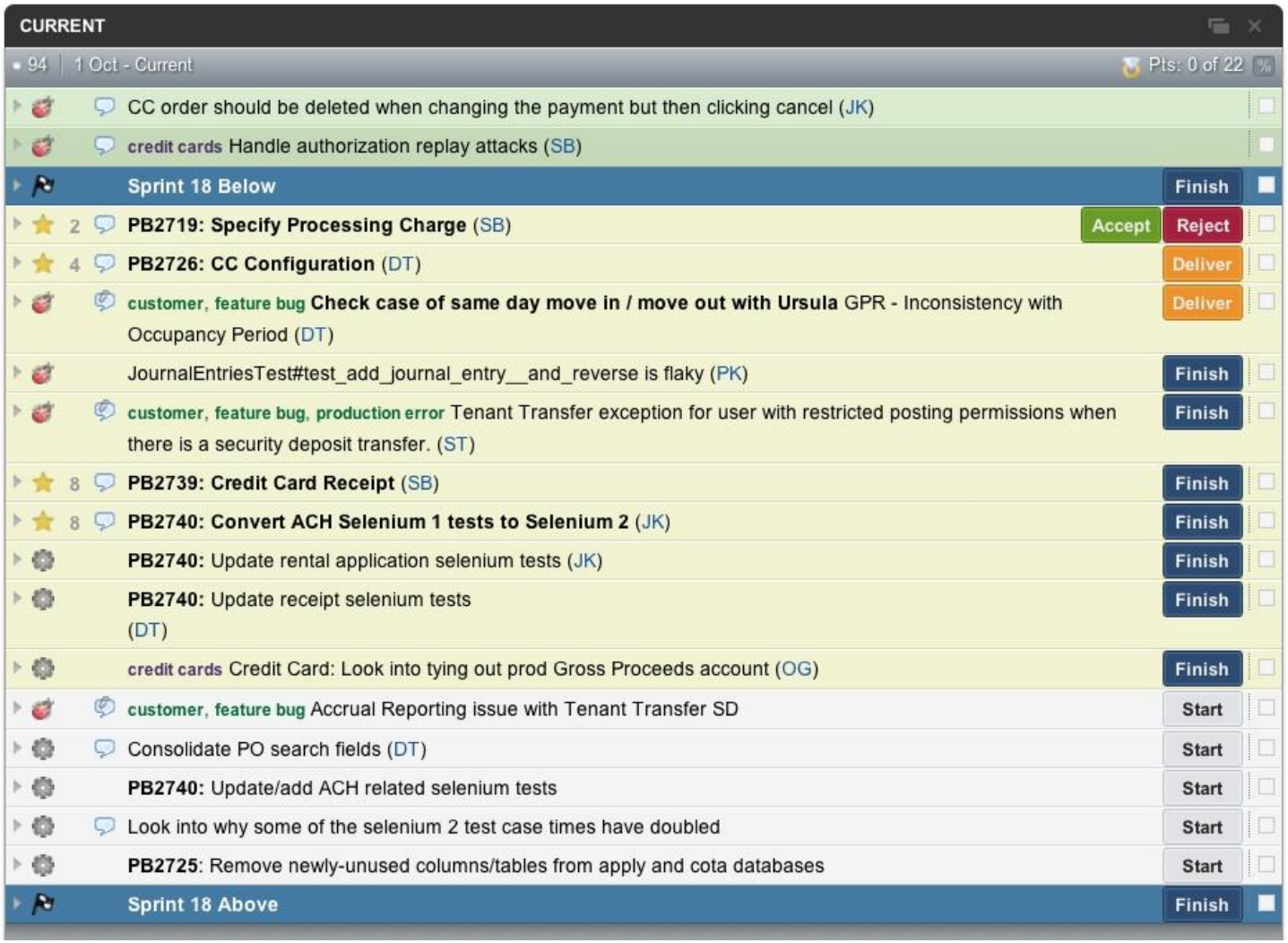


Burndown Charts & Scrum Boards

- Sprint burndown chart is a place to see daily progress.



Burndown Charts & Scrum Boards



Test-Driven Development

"You can not inspect quality into the product; it is already there."

W. Edwards Deming
Columbia University

Test-Driven Development

- Why automated testing?
  - Agile practices work best with frequent releases. Automating routine regression testing is efficient.
  - Tests serve as documentation for how the author intends the code to behave.
  - Large software systems are very complex. Any change can contain unforeseen consequences.

Test-Driven Development

So, we want high test coverage. How do we get there?

- **Only write code if there is a test to cover it.**

1. Write a failing test, watch it fail.
2. Write code to make the test pass.
3. Refactor.

Test-Driven Development

- Types of automated testing:
  - Unit tests
  - Functional tests
  - Selenium tests (page objects)
  - Javascript unit tests
  - Inter-application testing
  - Appearance tests

- **All bug fixes should include a previously-failing automated test!**

Continuous Integration

"The effort of integration is exponentially proportional to the amount of time between integrations."

Martin Fowler
ThoughtWorks

Continuous Integration

- Integration is the process of taking many independently designed changes and reconciling their conflicts.

- Recipe for smooth code integration:
  - Develop on trunk (or merge from trunk early/often)
  - Commit frequently.
  - Run all automated tests on each commit.
    - Detect integration problems as soon as possible.

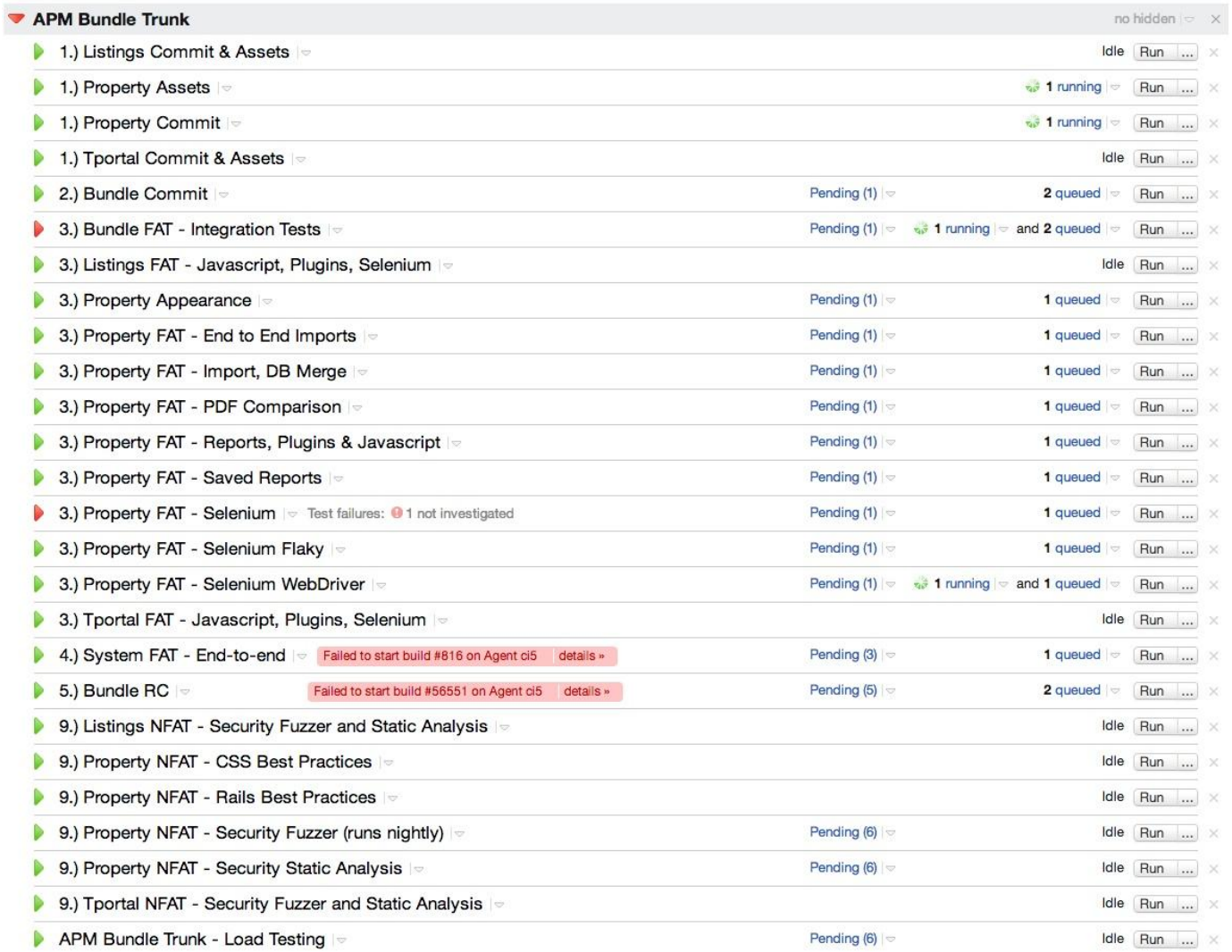
Continuous Integration

- More CI best practices
  - Have a fast commit suite (< 5 minutes)
  - On green, automatically deploy to testing servers.
  - On red, immediately fix or revert changes.
  - Run extended suite to verify release candidates.
  - Perform more extensive (or optional) testing nightly.

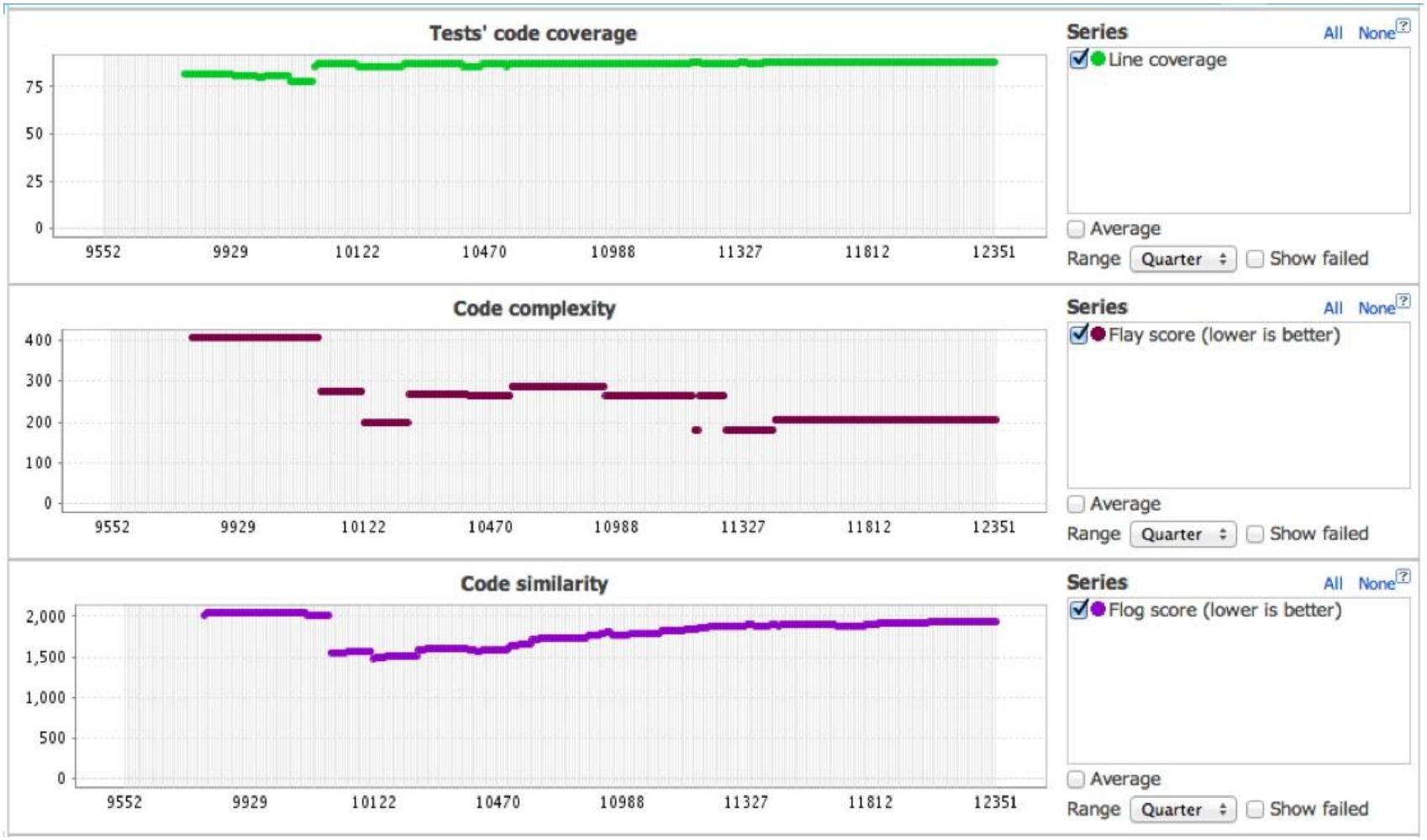
Continuous Integration

- Continuous Integration at AppFolio:
  - Commit suite:
    - Unit & Functional tests
  - Release candidate suite:
    - Selenium & Javascript tests
    - PDF appearance
    - Inter-app Communication
  - Additional Testing:
    - Rails & CSS best practices
    - Security static analysis
    - Test coverage, code complexity

Continuous Integration



Continuous Integration



Pair Programming

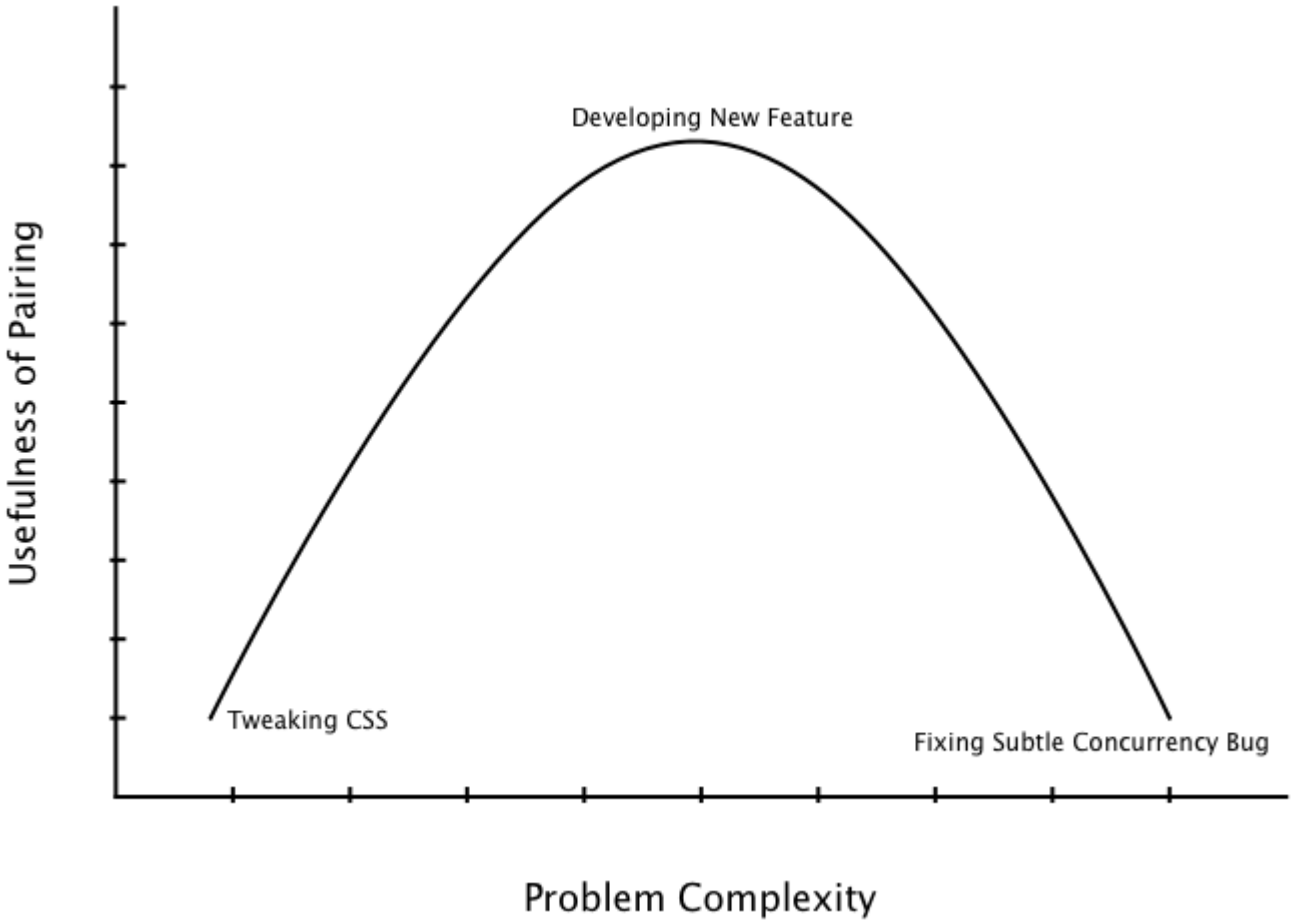
Pair programming: two developers share one computer and discuss all code that is being written.

Supplants code reviews.

- Different flavors:
  - Driver-Navigator: One person does most of the implementation while the other watches, discusses, thinks of consequences.
  - Ping-pong pairing: One person writes the test, the other makes it pass
    - Frequently used while learning TDD and pair programming

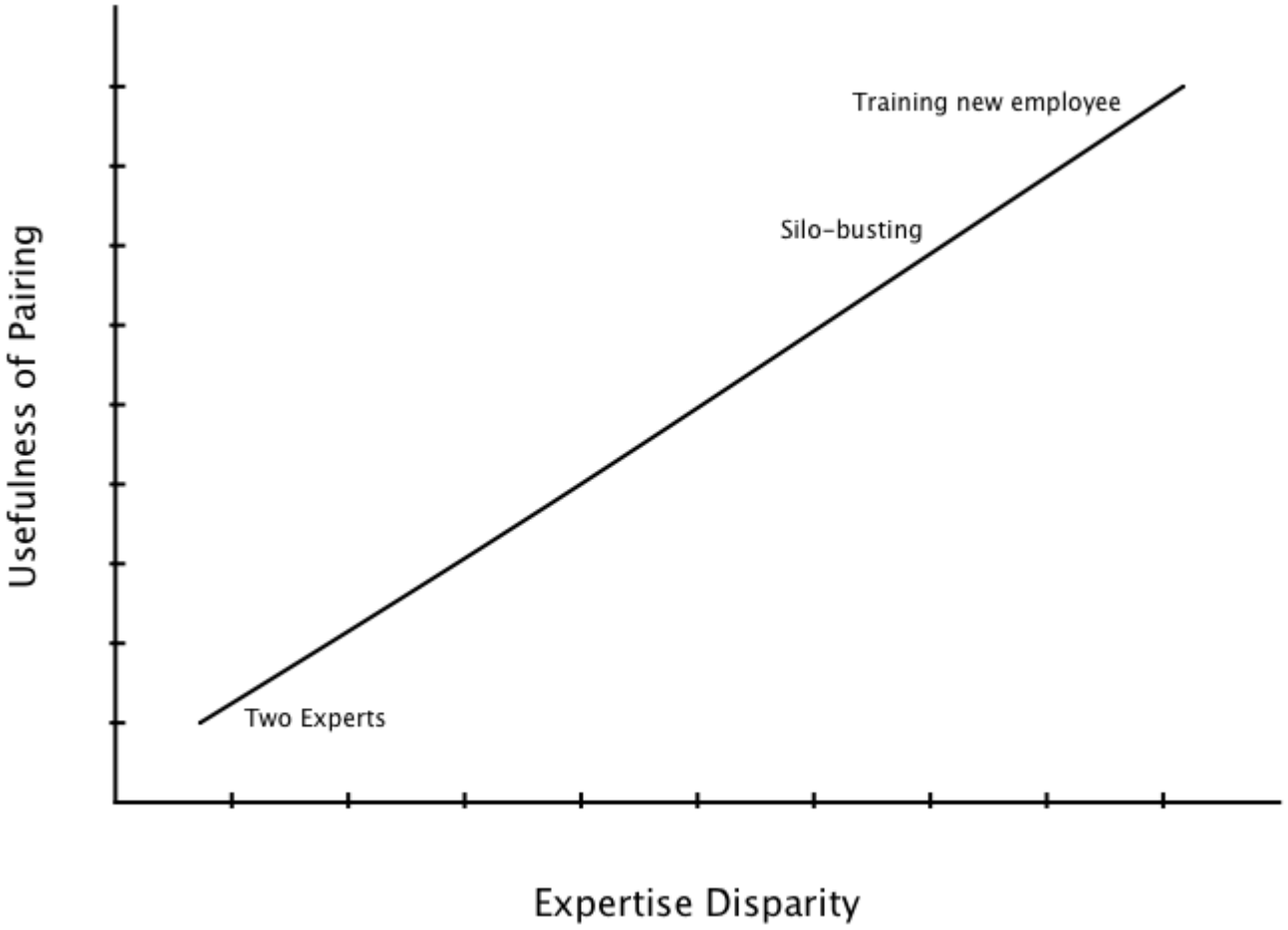
Pair Programming

Like any tool, it has strengths and weaknesses.



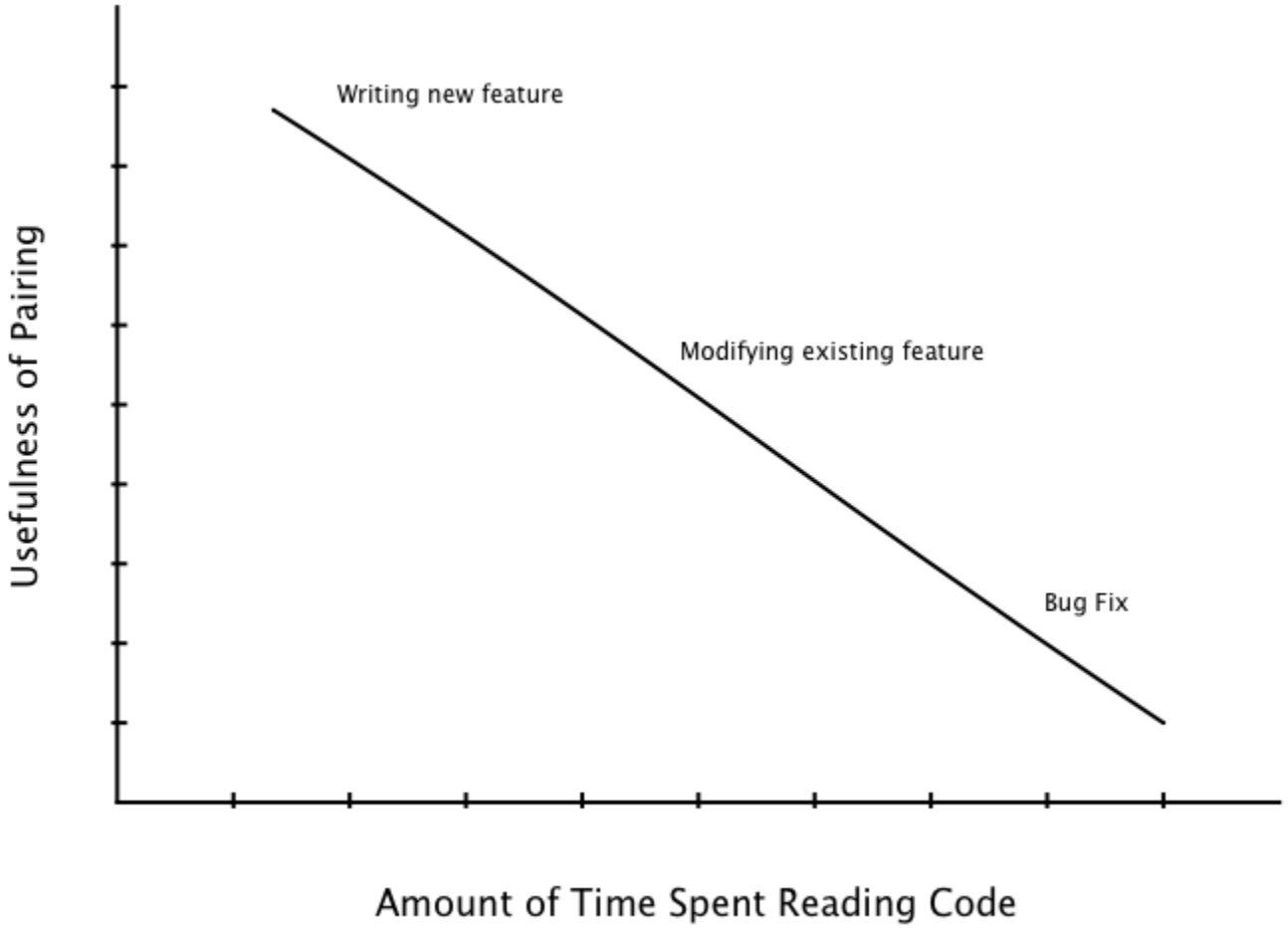
Pair Programming

Like any tool, it has strengths and weaknesses.



Pair Programming

Like any tool, it has strengths and weaknesses.



Conclusion

- Along with the rest of computing, software development techniques have been rapidly improving

- Applying Scrum, TDD, CI and pair programming will set your team on a course for success.