

Agile Software Development for Scalable Internet Services

Andrew Mutz

Director of Software Engineering, Appfolio

October 9th, 2013

Introduction

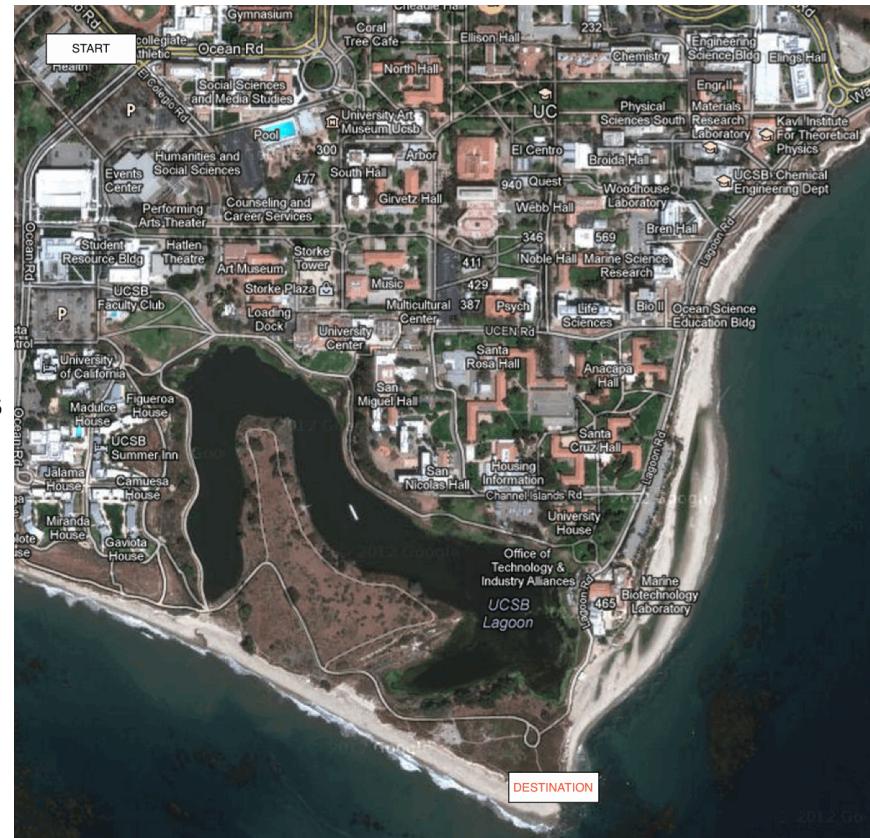
- What is Agile?
- The Agile workflow
- Supporting Tools and Techniques
 - Burndown Charts & Scrum Board
 - Test-Driven Development
 - Continuous Integration
 - Pair Programming

What is Agile?

Responding to change > Following a plan



versus

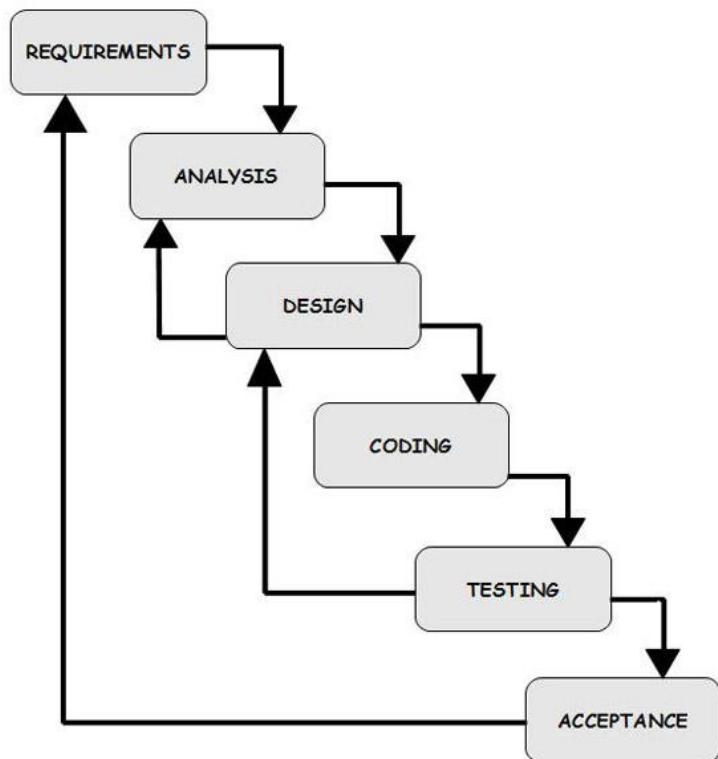


What is Agile?

Individual & interactions > Process



versus



What is Agile?

Customer collaboration > Frozen requirements



versus



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

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

"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

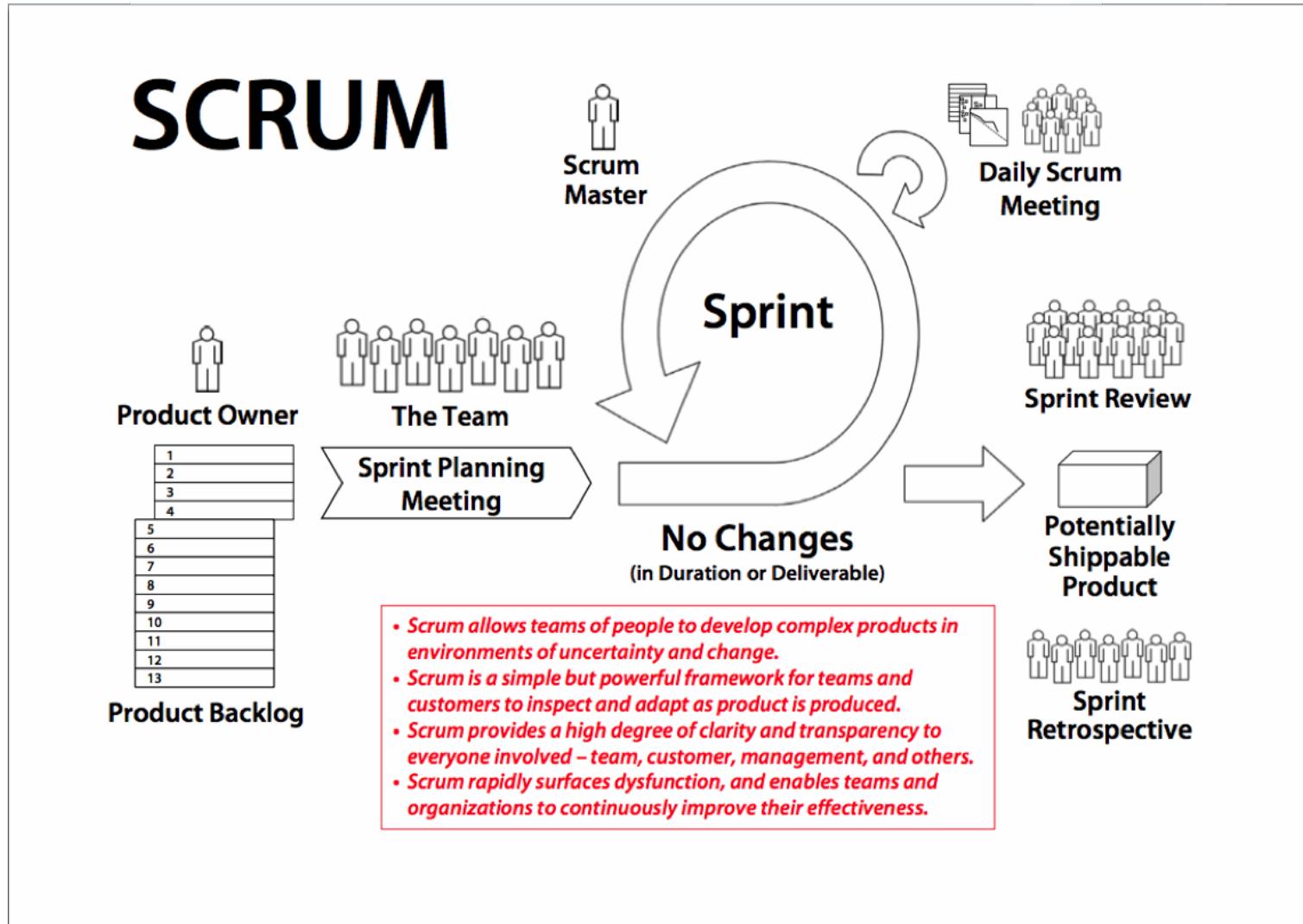
versus

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

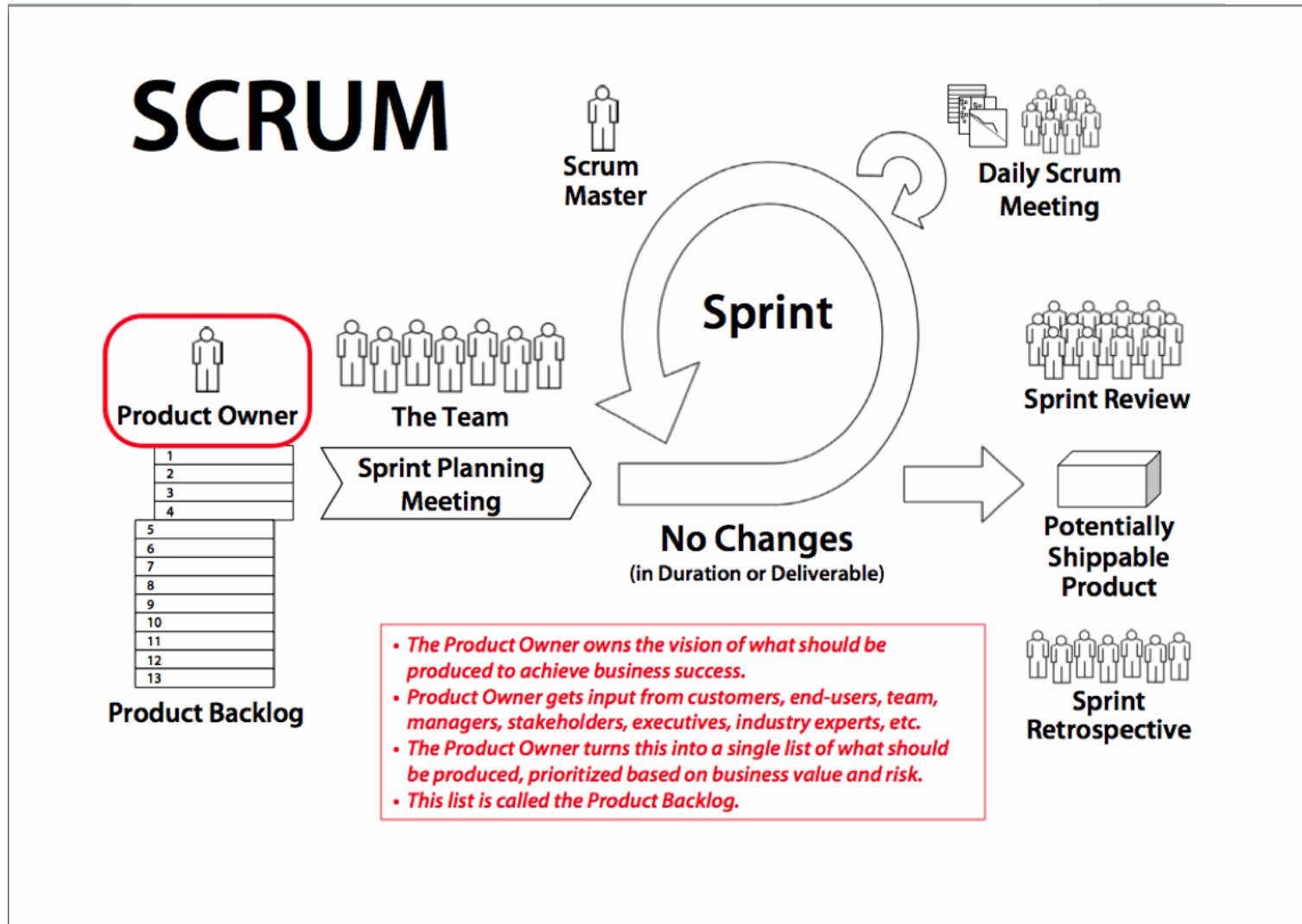
What is Agile?

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

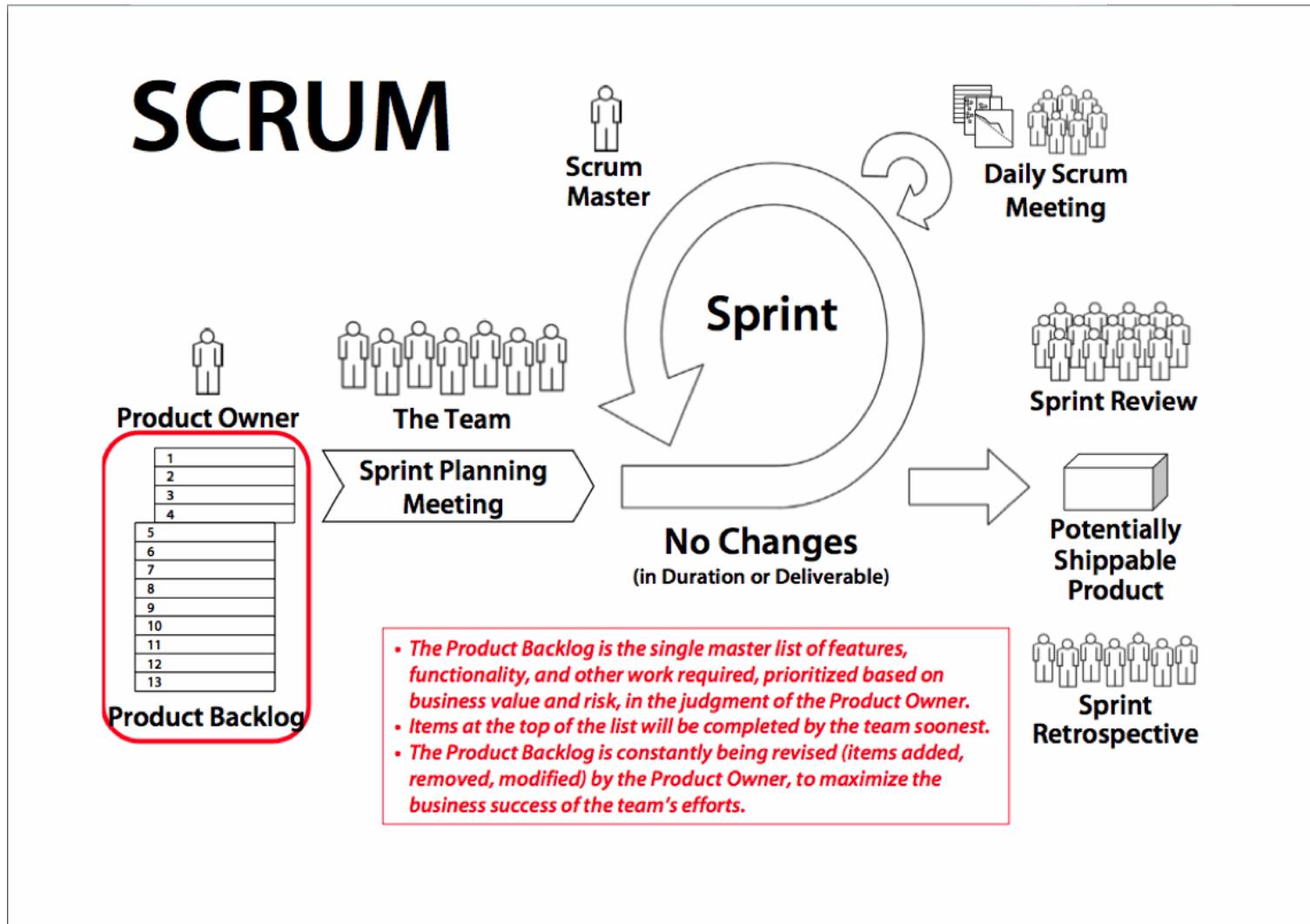
Agile Workflow with Scrum



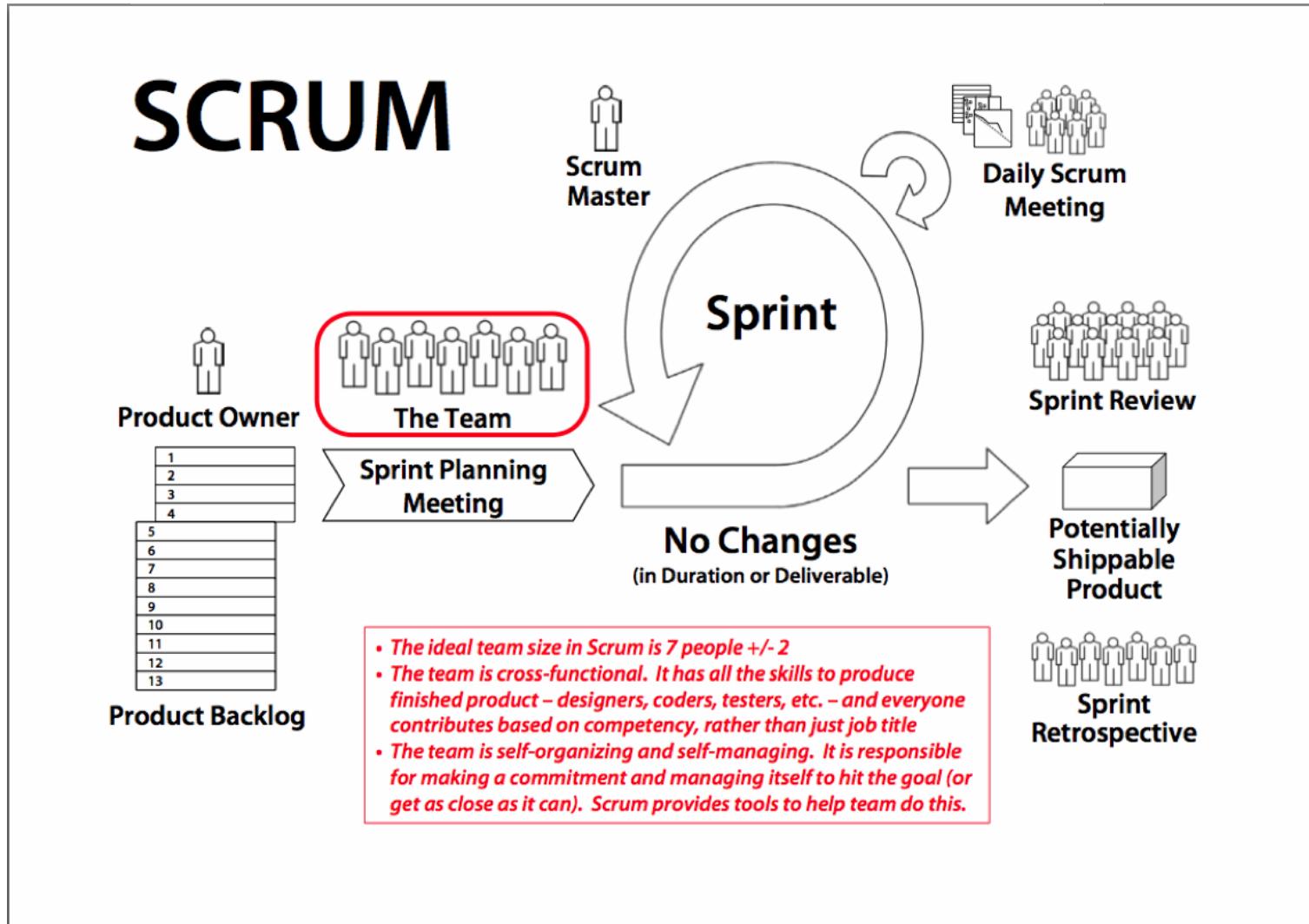
Agile Workflow with Scrum



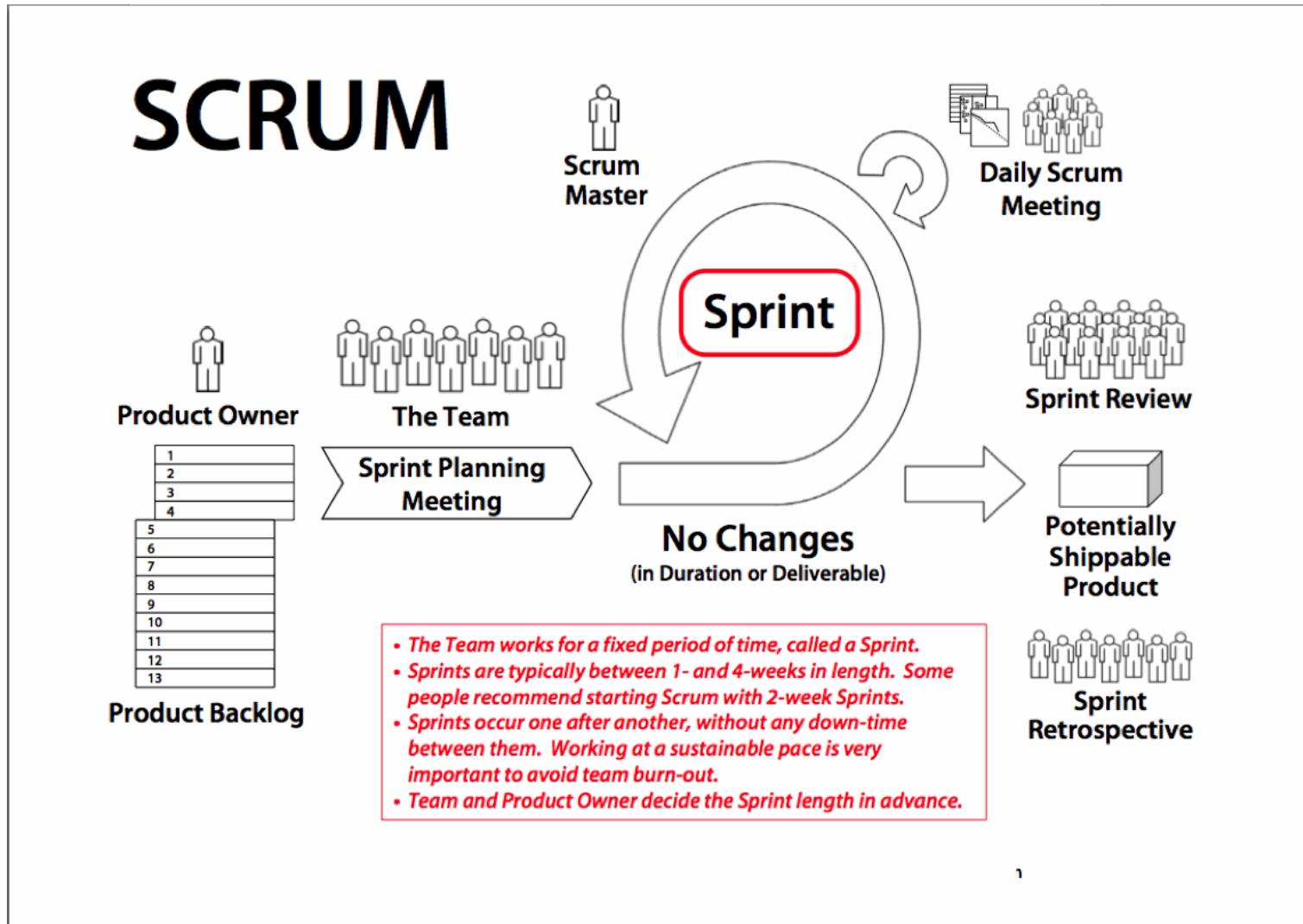
Agile Workflow with Scrum



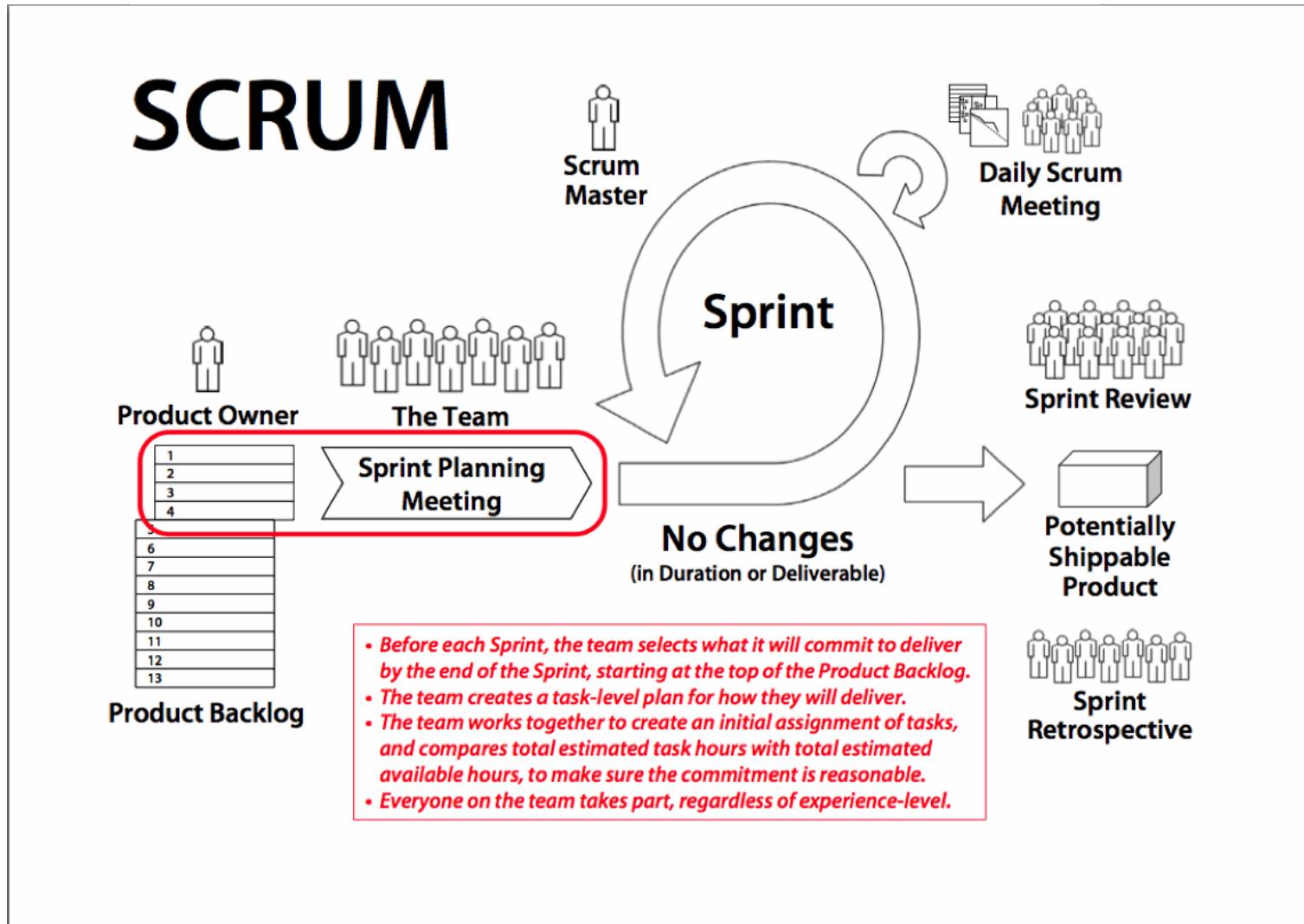
Agile Workflow with Scrum



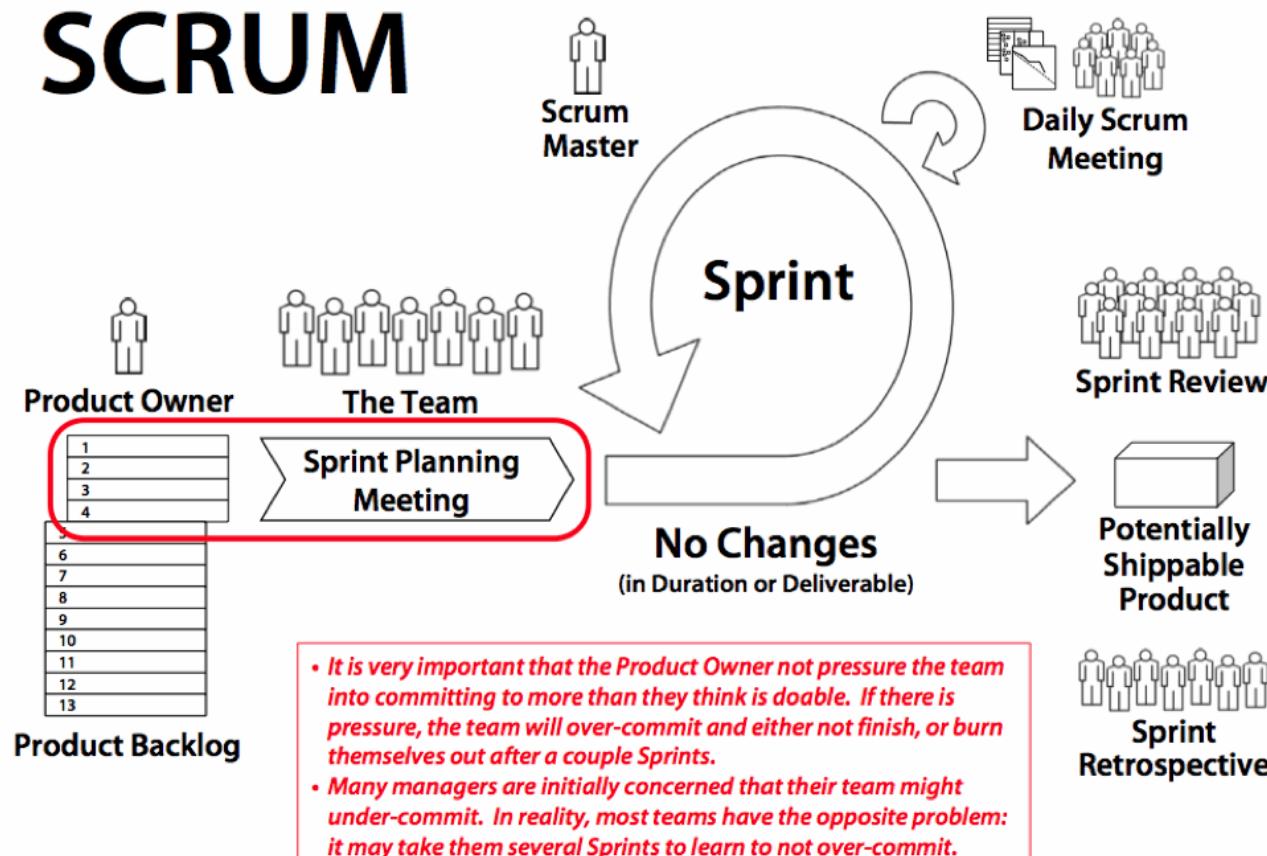
Agile Workflow with Scrum



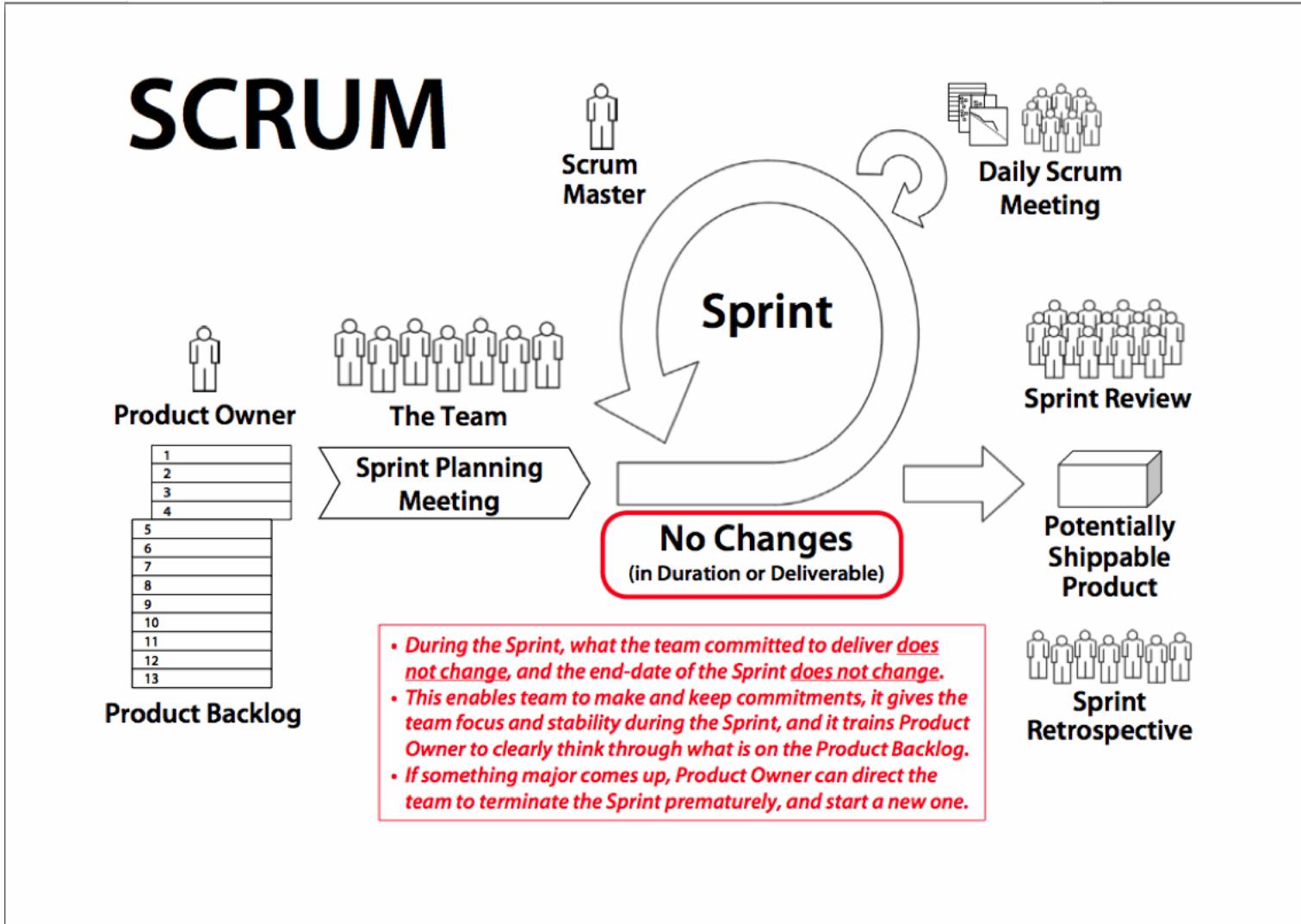
Agile Workflow with Scrum



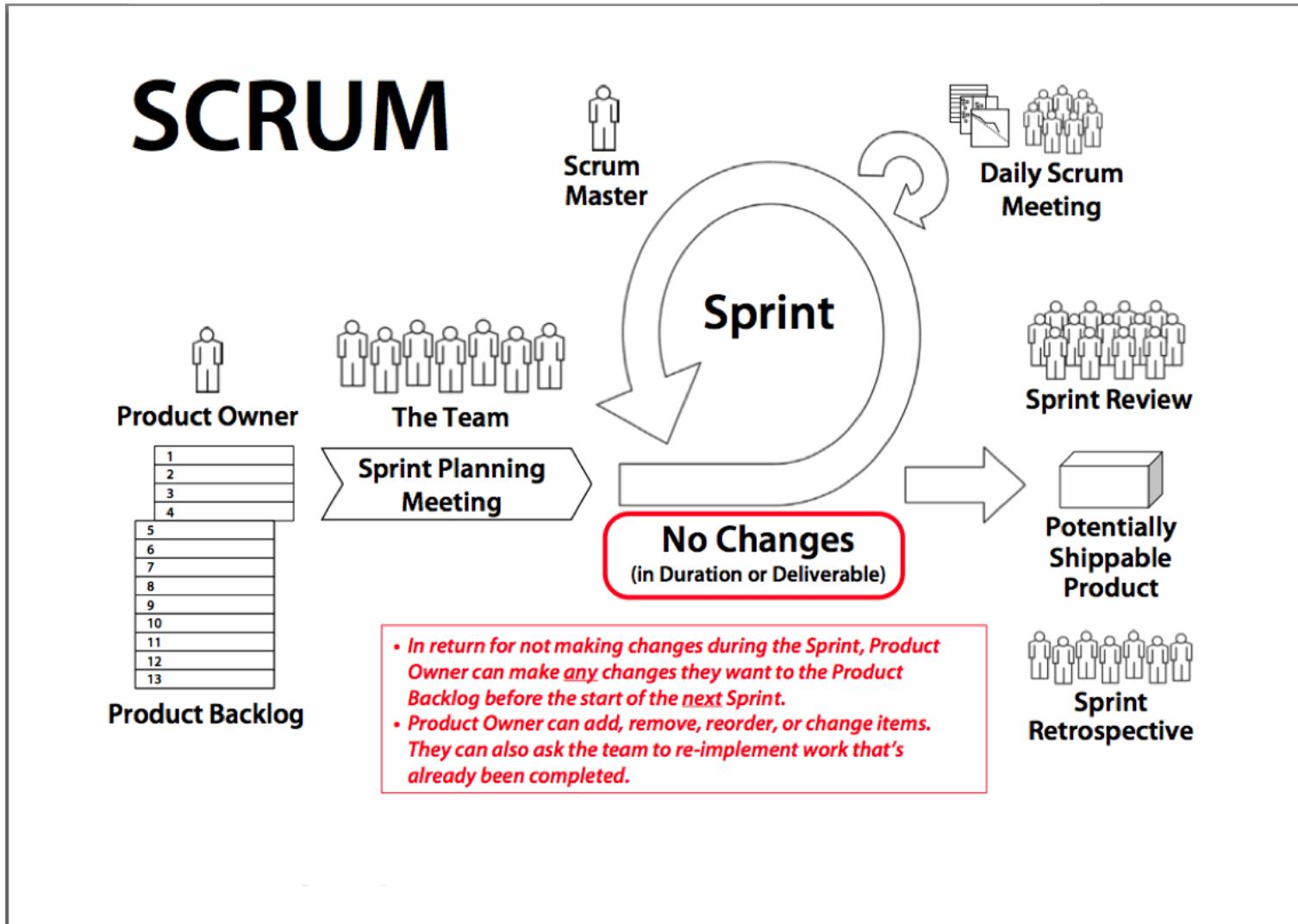
Agile Workflow with Scrum



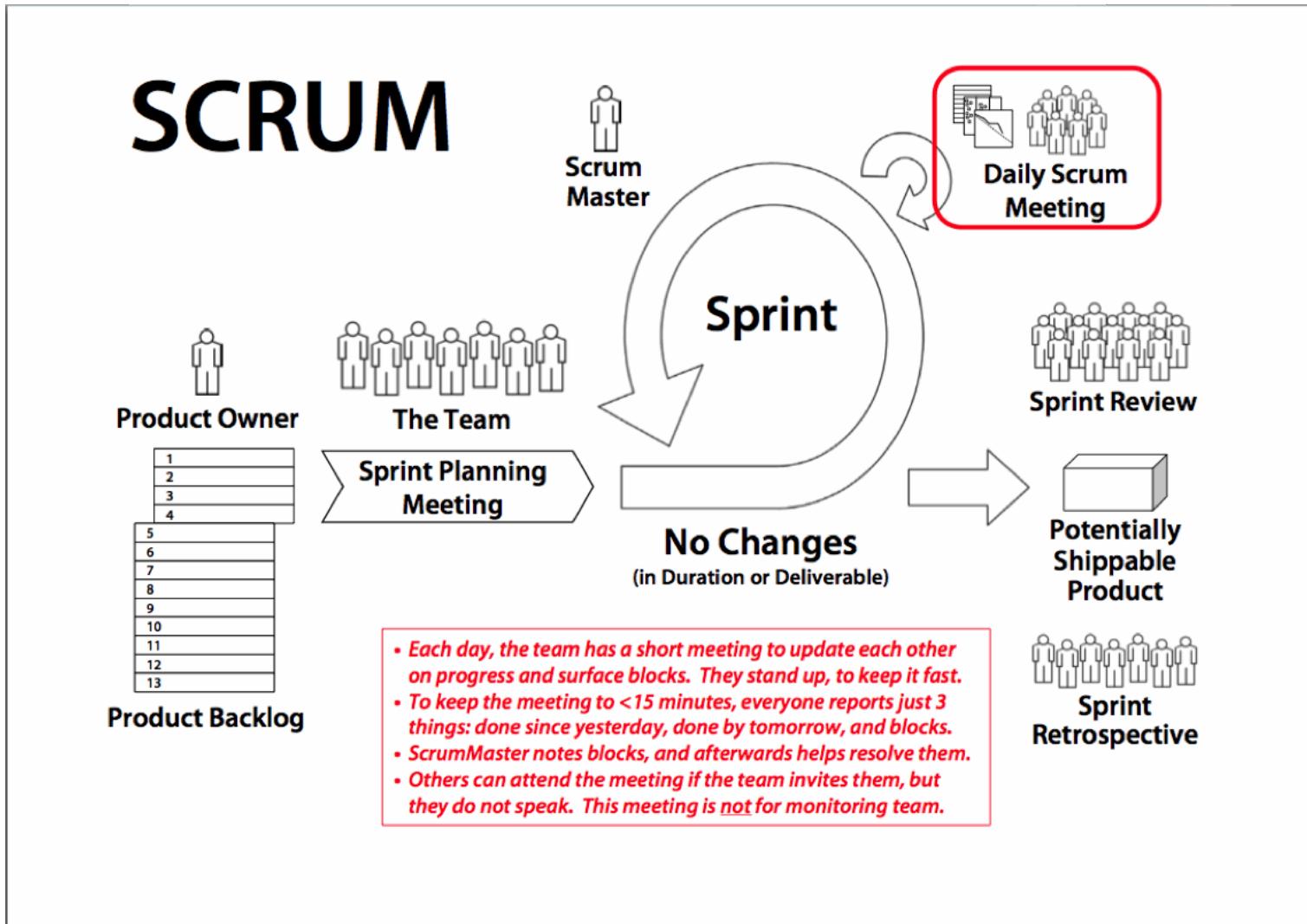
Agile Workflow with Scrum



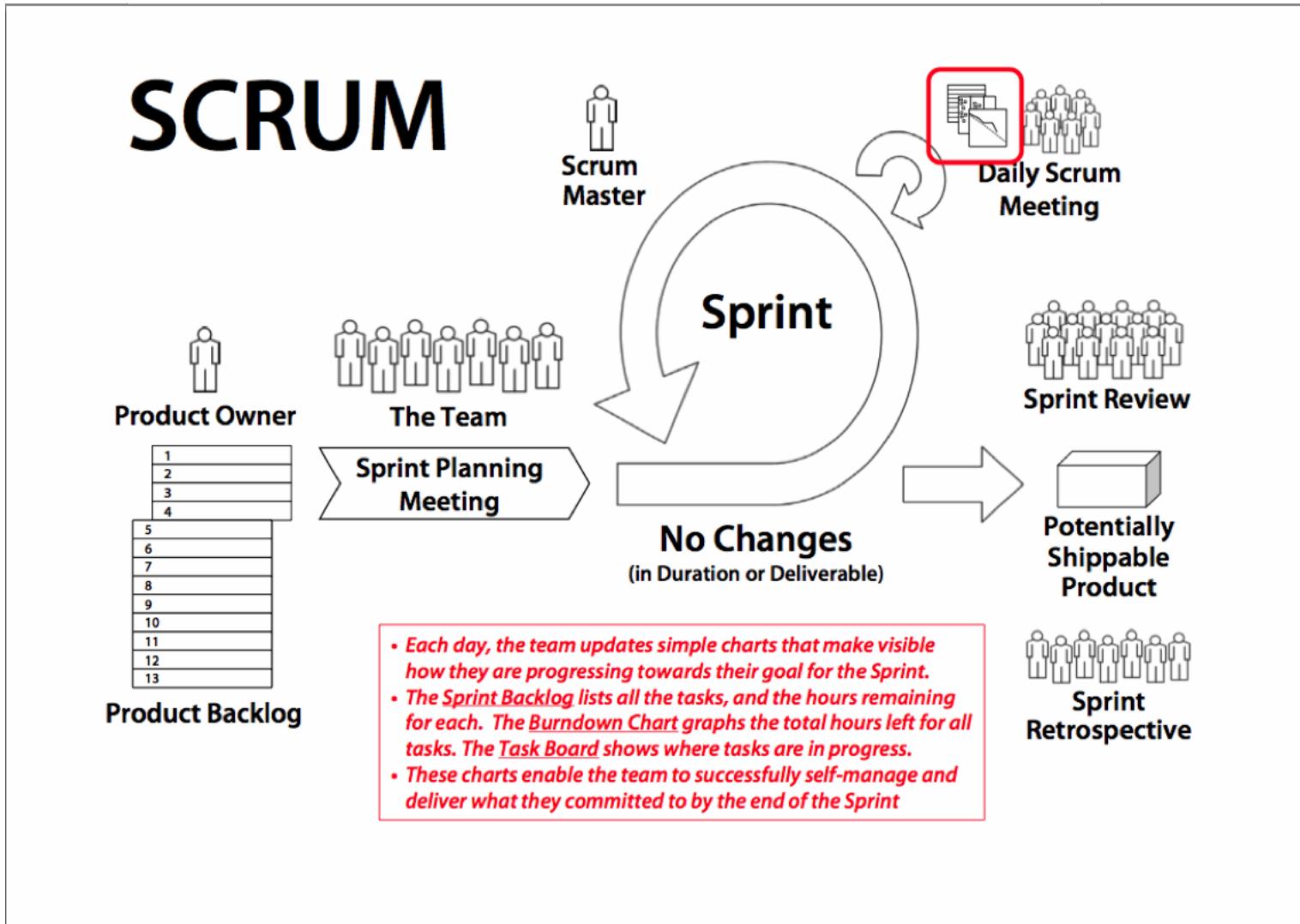
Agile Workflow with Scrum



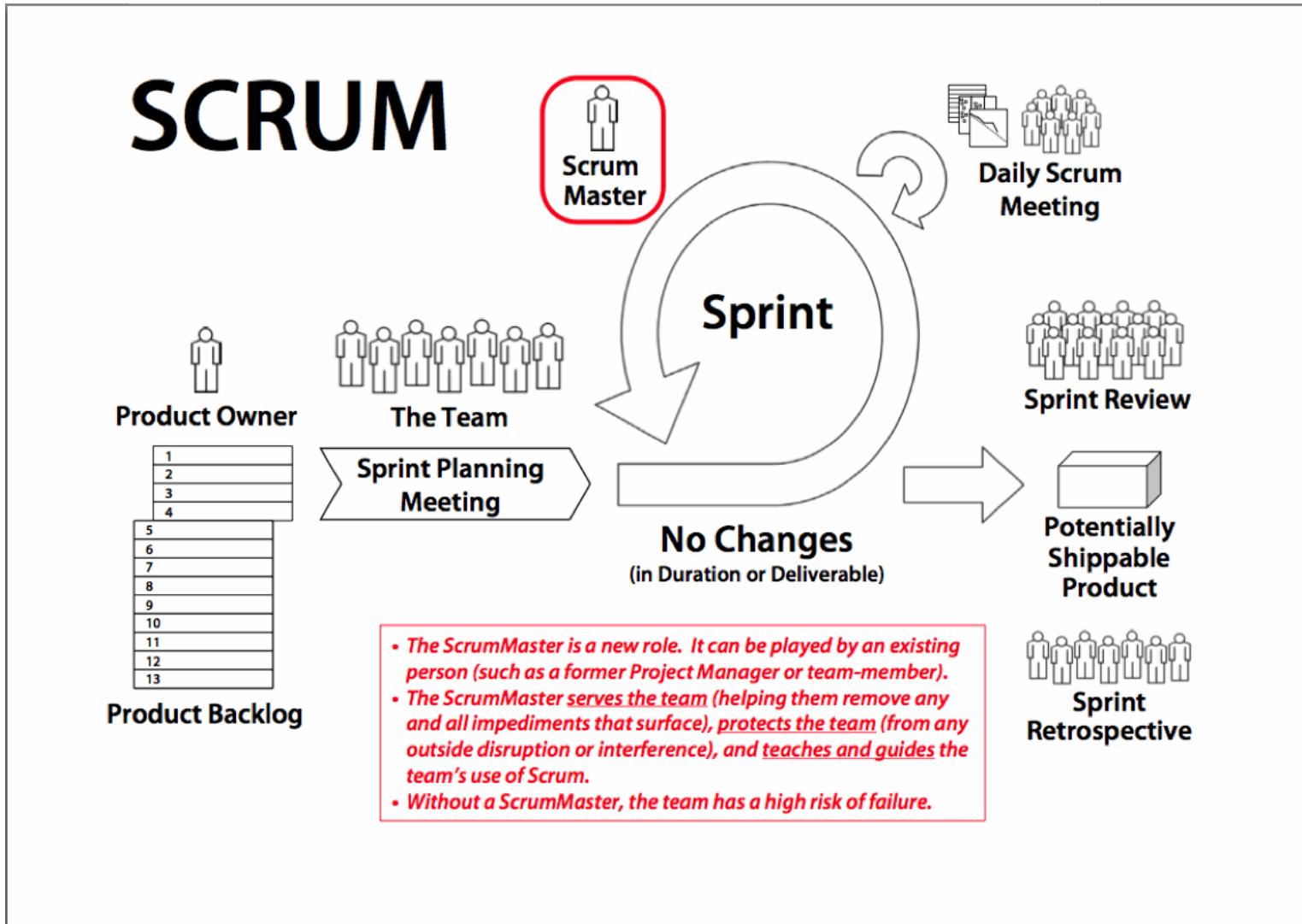
Agile Workflow with Scrum



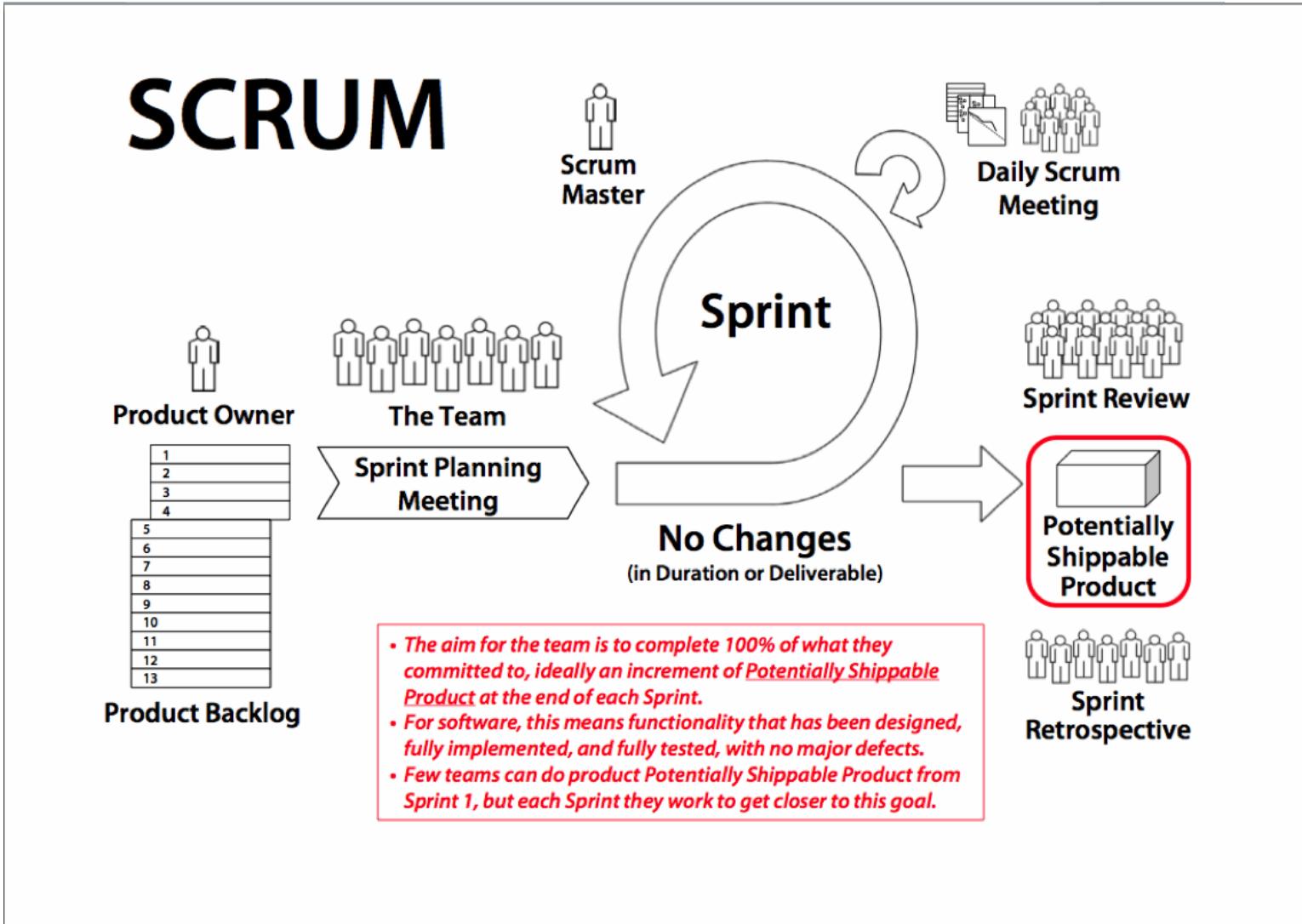
Agile Workflow with Scrum



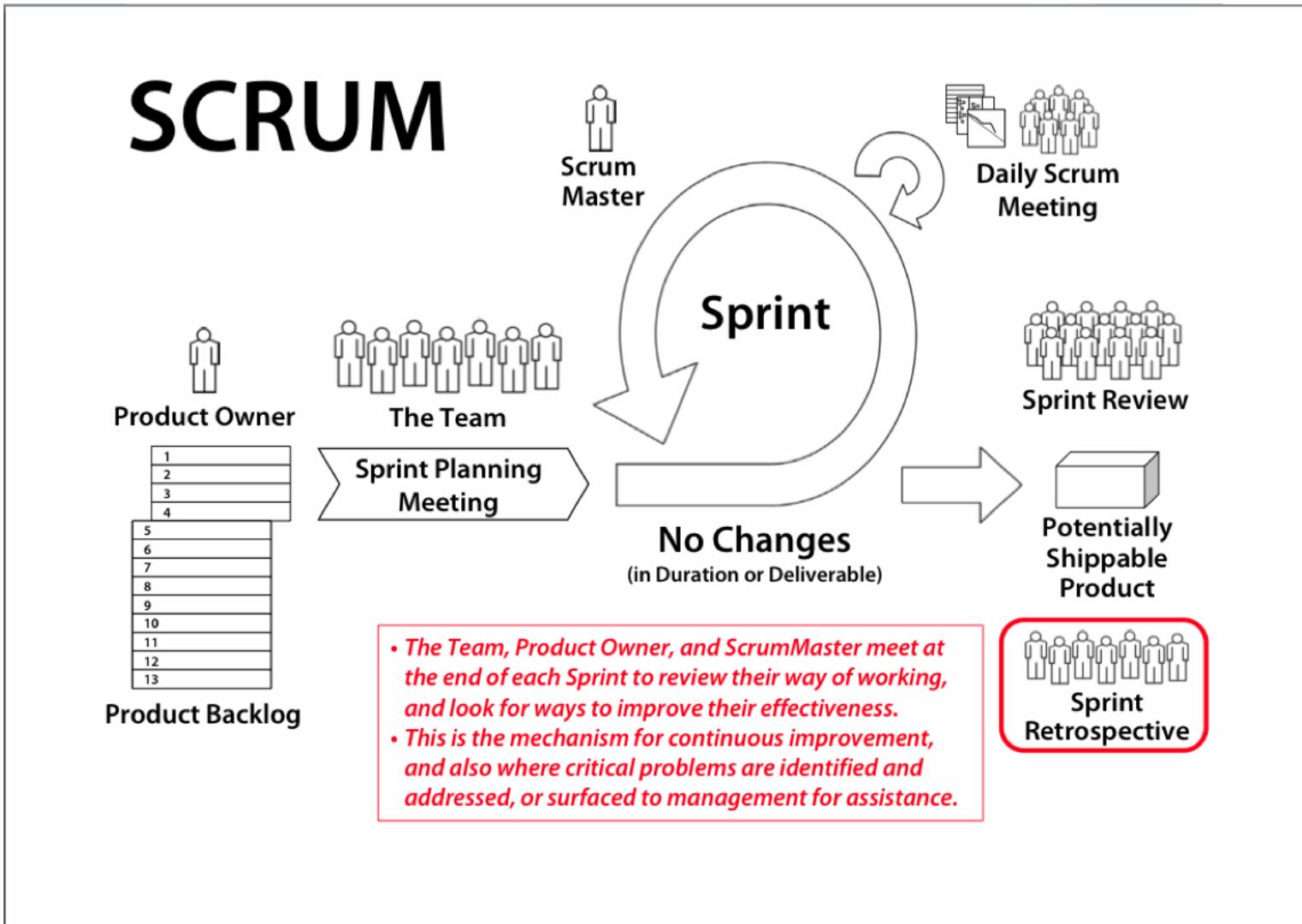
Agile Workflow with Scrum



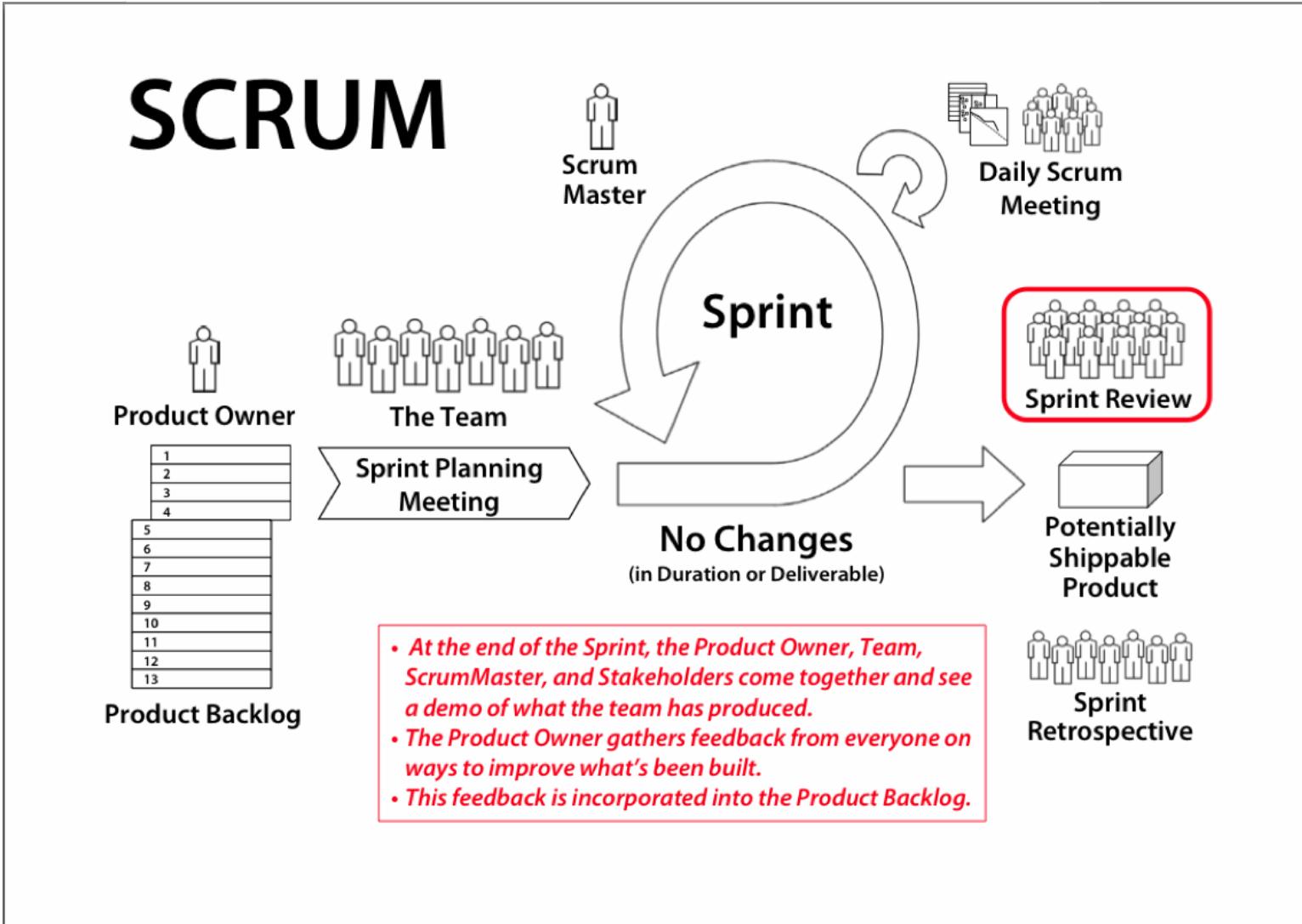
Agile Workflow with Scrum



Agile Workflow with Scrum

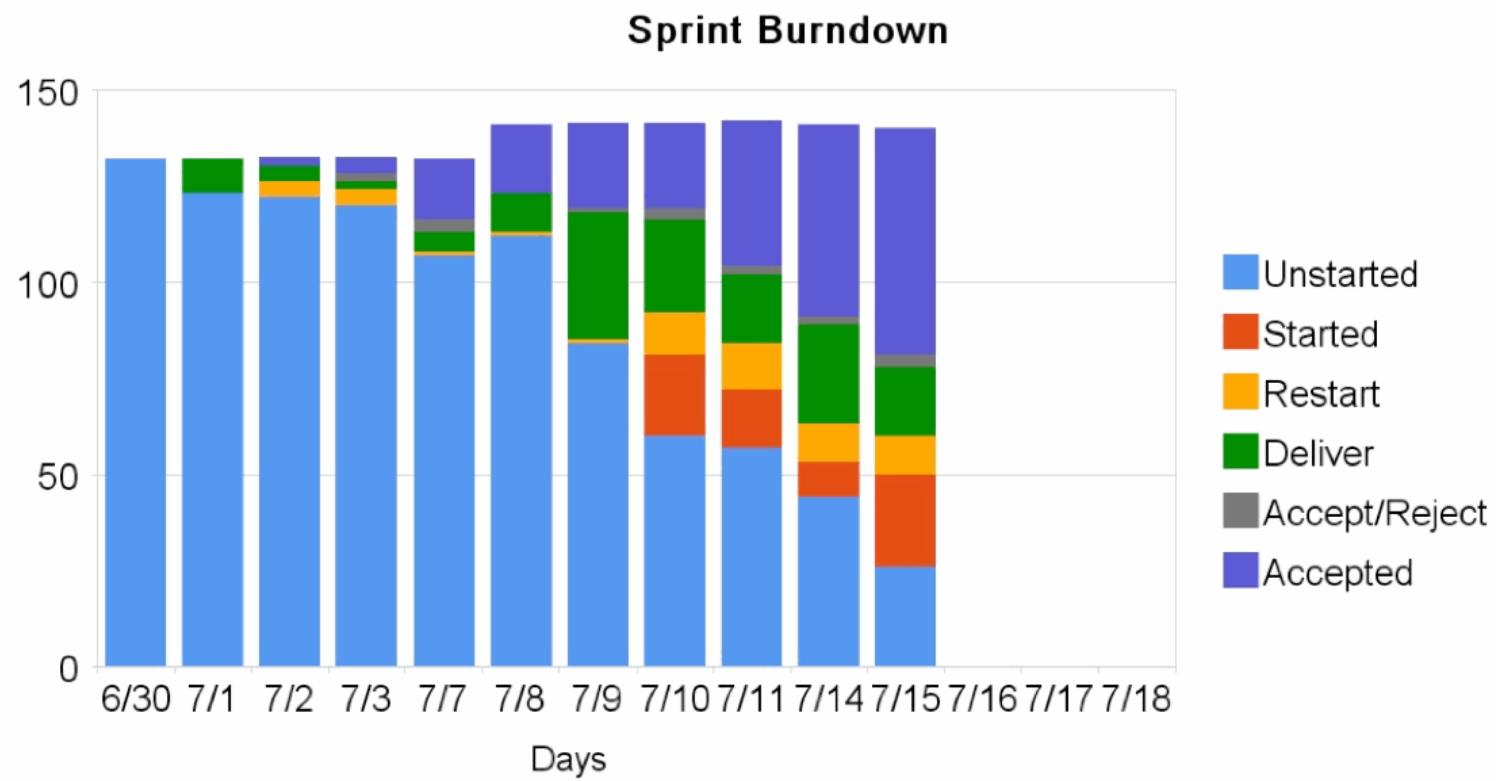


Agile Workflow with Scrum



Burndown Charts & Scrum Boards

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



Burndown Charts & Scrum Boards

CURRENT

- 94 | 1 Oct - Current Pts: 0 of 22 %
- ▶ 🍑 💬 CC order should be deleted when changing the payment but then clicking cancel (JK)
- ▶ 🍑 💬 credit cards Handle authorization replay attacks (SB)
- ▶ 🔎 Sprint 18 Below Finish
- ▶ ⭐ 2 💬 PB2719: Specify Processing Charge (SB) Accept Reject
- ▶ ⭐ 4 💬 PB2726: CC Configuration (DT) Deliver
- ▶ 🍑 💬 customer, feature bug Check case of same day move in / move out with Ursula GPR - Inconsistency with Occupancy Period (DT) Deliver
- ▶ 🍑 💬 JournalEntriesTest#test_add_journal_entry__and_reverse is flaky (PK) Finish
- ▶ 🍑 💬 customer, feature bug, production error Tenant Transfer exception for user with restricted posting permissions when there is a security deposit transfer. (ST) Finish
- ▶ ⭐ 8 💬 PB2739: Credit Card Receipt (SB) Finish
- ▶ ⭐ 8 💬 PB2740: Convert ACH Selenium 1 tests to Selenium 2 (JK) Finish
- ▶ 🌐 PB2740: Update rental application selenium tests (JK) Finish
- ▶ 🌐 PB2740: Update receipt selenium tests (DT) Finish
- ▶ 🌐 credit cards Credit Card: Look into tying out prod Gross Proceeds account (OG) Finish
- ▶ 🍑 💬 customer, feature bug Accrual Reporting issue with Tenant Transfer SD Start
- ▶ 🌐 💬 Consolidate PO search fields (DT) Start
- ▶ 🌐 PB2740: Update/add ACH related selenium tests Start
- ▶ 🌐 💬 Look into why some of the selenium 2 test case times have doubled Start
- ▶ 🌐 PB2725: Remove newly-unused columns/tables from apply and cota databases Start
- ▶ 🔎 Sprint 18 Above Finish

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

APM Bundle Trunk			
1.) Listings Commit & Assets	Idle	Run ...	x
1.) Property Assets	1 running	Run ...	x
1.) Property Commit	1 running	Run ...	x
1.) Tportal Commit & Assets	Idle	Run ...	x
2.) Bundle Commit	Pending (1)	2 queued	Run ...
3.) Bundle FAT - Integration Tests	Pending (1)	1 running	Run ...
3.) Listings FAT - Javascript, Plugins, Selenium	Idle	Run ...	x
3.) Property Appearance	Pending (1)	1 queued	Run ...
3.) Property FAT - End to End Imports	Pending (1)	1 queued	Run ...
3.) Property FAT - Import, DB Merge	Pending (1)	1 queued	Run ...
3.) Property FAT - PDF Comparison	Pending (1)	1 queued	Run ...
3.) Property FAT - Reports, Plugins & Javascript	Pending (1)	1 queued	Run ...
3.) Property FAT - Saved Reports	Pending (1)	1 queued	Run ...
3.) Property FAT - Selenium	Pending (1)	1 queued	Run ...
3.) Property FAT - Selenium Flaky	Pending (1)	1 queued	Run ...
3.) Property FAT - Selenium WebDriver	Pending (1)	1 running	Run ...
3.) Tportal FAT - Javascript, Plugins, Selenium	Idle	Run ...	x
4.) System FAT - End-to-end	Failed to start build #816 on Agent ci5 details »	Pending (3)	1 queued
5.) Bundle RC	Failed to start build #56551 on Agent ci5 details »	Pending (5)	2 queued
9.) Listings NFAT - Security Fuzzer and Static Analysis	Idle	Run ...	x
9.) Property NFAT - CSS Best Practices	Idle	Run ...	x
9.) Property NFAT - Rails Best Practices	Idle	Run ...	x
9.) Property NFAT - Security Fuzzer (runs nightly)	Pending (6)	Idle	Run ...
9.) Property NFAT - Security Static Analysis	Pending (6)	Idle	Run ...
9.) Tportal NFAT - Security Fuzzer and Static Analysis	Idle	Run ...	x
APM Bundle Trunk - Load Testing	Pending (6)	Idle	Run ...

Continuous Integration



Pair Programming



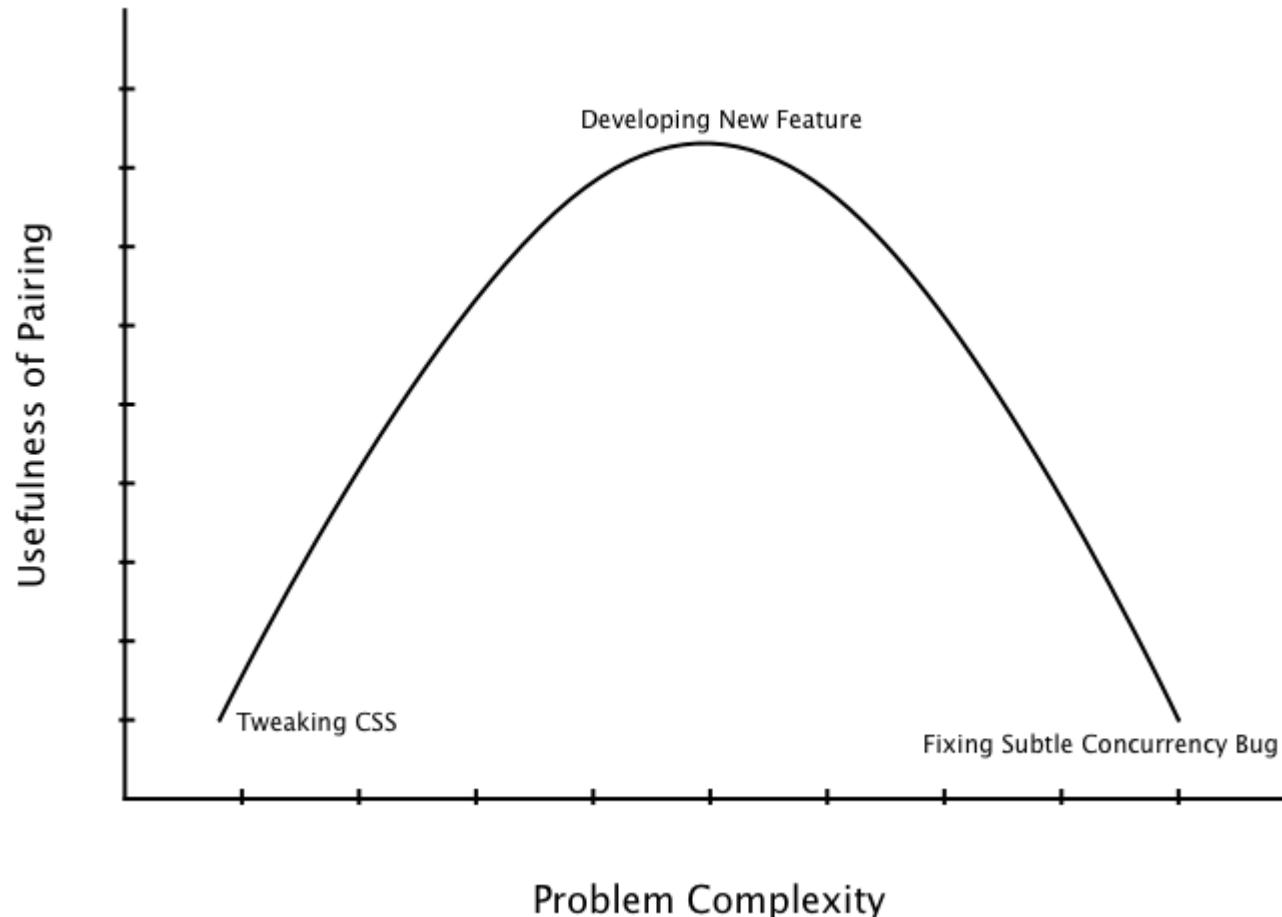
Pair programming: two developers share one computer and discuss all code that is being written.

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

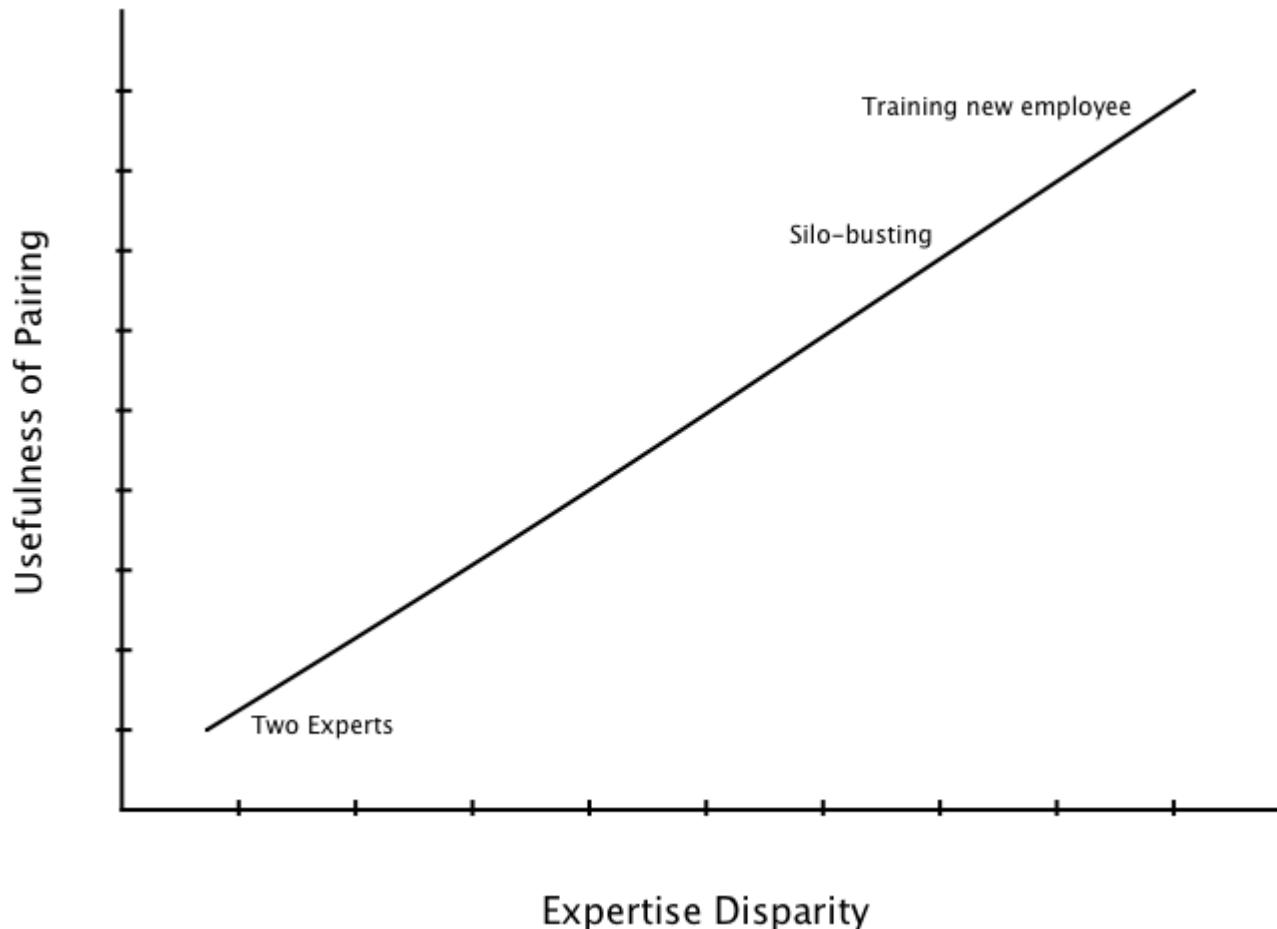
Pair Programming

Like any tool, it has strengths and weaknesses.



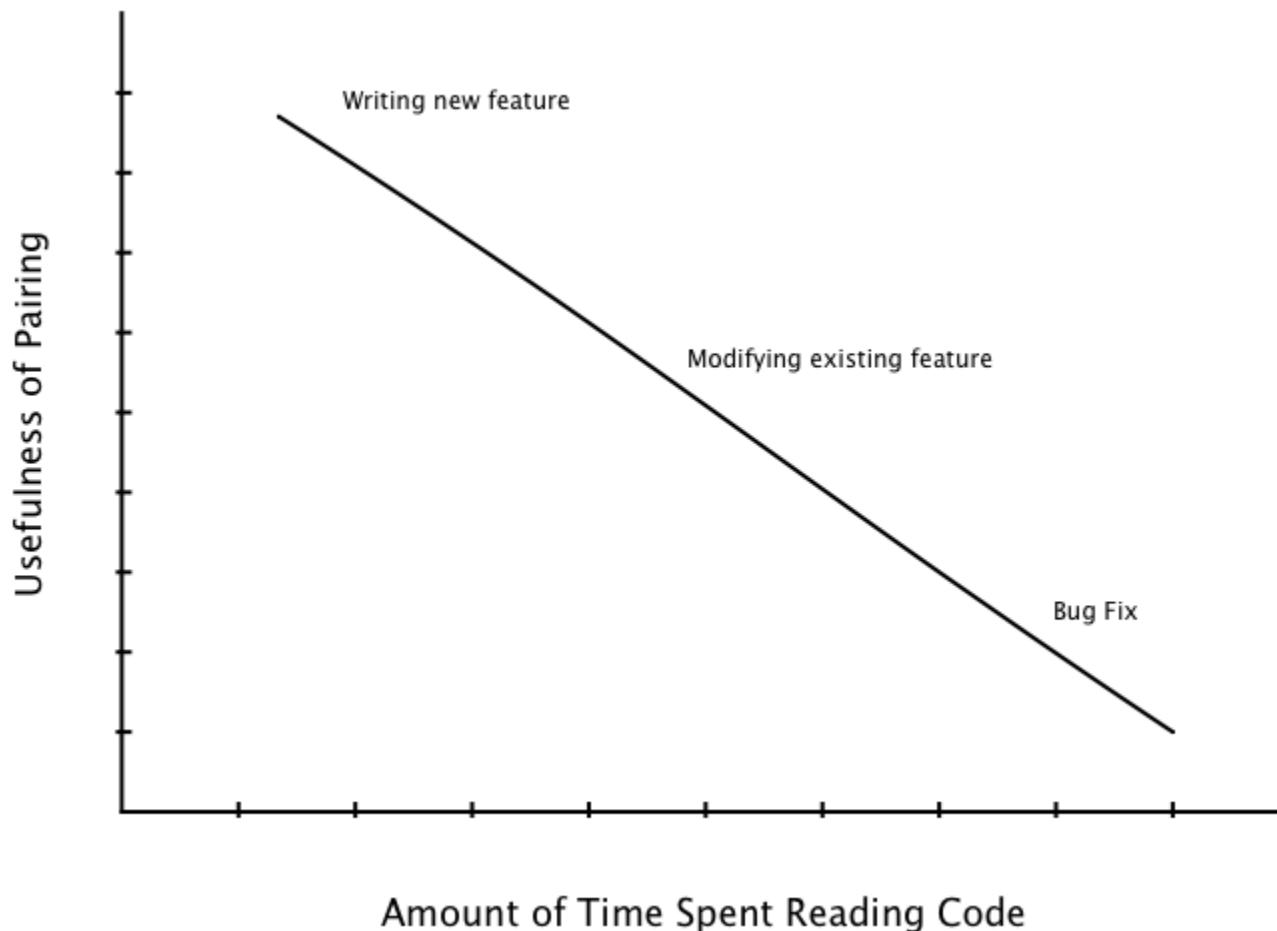
Pair Programming

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.