Scrum Team Seminar

Mark C. Paulk, Ph.D.

April 4, 2012
Carnegie Mellon MSE

The Agile Manifesto

“We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

• individuals and interactions over processes and tools
• working software over comprehensive documentation
• customer collaboration over contract negotiation
• responding to change over following a plan

That is, while there is value on the items on the right, we value the items on the left more.”

http://www.agilealliance.org
Principles Behind the Agile Manifesto -1

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Principles Behind the Agile Manifesto -2

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
Principles Behind the Agile Manifesto

Continuous attention to technical excellence and good design enhances agility.

Simplicity—the art of maximizing the amount of work not done—is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

---

Six Sweet Spots

Dedicated developers

Experienced developers

Small co-located team

Automated regression tests

Easy access to users

Short increments and frequent delivery to real users
Scrum – Inspect & Adapt (Cohn 2010)

Scrum’s 3 + 3 + 3

Three roles
• Product Owner
• ScrumMaster
• Development Team

Three ceremonies
• Sprint Planning Meeting
• Daily Scrum Meeting
• Sprint Review Meeting

Three artifacts
• Product Backlog
• Sprint Backlog
• Burndown Chart
**Product Owner (James 2010)**

- Single person responsible for maximizing the return on investment (ROI) of the development effort
- Responsible for product vision
- Constantly re-prioritizes the Product Backlog, adjusting any long-term expectations such as release plans
- Final arbiter of requirements questions
- Accepts or rejects each product increment
- Decides whether to ship
- Decides whether to continue development
- Considers stakeholder interests
- May contribute as a team member
- Has a leadership role

**Prioritization Factors (Cohn 2006)**

- Financial value
- Cost of developing
- Learning and new knowledge – both product and project knowledge
- Risk
  - high value, high risk
  - high value, low risk
  - low value, low risk
  - avoid low value, high risk
**ScrumMaster** *(James 2010)*

- Facilitates the Scrum process
- Helps resolve impediments
- Creates an environment conducive to team self-organization
- Captures empirical data to adjust forecasts
- Shields the team from external interference and distractions to keep it in group flow (a.k.a. the zone)
- Enforces timeboxes
- Keeps Scrum artifacts visible
- Promotes improved engineering practices
- Has no management authority over the team
  - anyone with authority over the team is by definition not its ScrumMaster
- Has a leadership role

**Scrum Development Team** *(James 2010)*

- Cross-functional (e.g., includes members with testing skills, and often others not traditionally called developers: business analysts, domain experts, etc.)
- Self-organizing / self-managing, without externally assigned roles
- Negotiates commitments with the Product Owner, one Sprint at a time
- Has autonomy regarding how to reach commitments
- Intensely collaborative
- Most successful when located in one team room, particularly for the first few Sprints
- Most successful with long-term, full-time membership.
  - Scrum moves work to a flexible learning team and avoids moving people or splitting them between teams.
- 7 ± 2 members
- Has a leadership role
Product Backlog

Lists the requirements for the product being developed

The master list of all functionality desired in the product

Each item in the Product Backlog has a description, a priority, and an estimate of the effort needed to complete it.

What’s a User Story?

A promise to have a discussion; not every detail needs to be included.

Describes functionality that will be valuable to either a user or purchaser of a system.

• Card – written description of the story used for planning and as a reminder
• Conversation – about the story that serve to flesh out the details of the story
• Confirmation – details that can be used to determine when a story is complete

User Stories Are A Beginning

“As a <type of user>, I want <capability> so that <business value>.”

Short, lightweight, imprecise...

A just-in-time approach...

Accompanied by as many conversations between the developers and the Product Owner as needed.

• Focus shifts from written to verbal communication


Epics and Themes (Cohn 2005)

In general we want to estimate user stories that are within one order of magnitude.

For features we’re not sure we want or that are not near-term, a “large” user story will minimize our effort.

Large user stories are called “epics”.

A set of related user stories that may be treated as a single entity for estimating or releasing planning is called a “theme”.

• Epics are often themes.
**Splitting Stories** *(Cohn 2006)*

By the data that will be supported

On operational boundaries – Create, Read, Update, Delete

Removing cross-cutting concerns (security, exception handling, error handling, logging, …)

Separating functional and non-functional aspects (performance)

Stories of mixed priority (e.g., login, invalid 3 times, email if denied access)

---

**Stories and Tasks** *(Kniberg 2007)*

Time estimates are usually easier to do (and more accurate) if a story is broken down into tasks.

- Stories are deliverable stuff that the Product Owner cares about.
- Tasks are non-deliverable stuff, or stuff that the Product Owner doesn’t care about.

The task breakdown is usually quite volatile.

The Product Owner doesn’t need to be involved in this level of detail.
**Tech Stories** *(Kniberg 2007)*

Stuff that needs to be done but that is not deliverable, not directly related to any specific stories, and not of direct value to the Product Owner

- For example… install continuous build server… write a system design overview… refactor the DAO layer… upgrade Jira (bug tracker)

Try to avoid tech stories

- transform a tech story into a normal story with measurable business value
- see if the work could be done as a task within another story
- or define it as a tech story, and keep a separate list of such stories

**Sprints**

A development iteration of a month or less in duration.

Sprint duration is fixed throughout the development effort.

Only the Product Owner has the authority to cancel the Sprint.

Sutherland and Vodde suggest that Sprints may be 2-6 weeks long.

**Sprint Planning Meeting**

When the iteration (Sprint) is planned

Time boxed to eight hours (for a one month Sprint)

Two parts (4+4 hours):
- determining what will be done in the Sprint
- how the Team is going to build the product increment during the Sprint

**Sprint Backlog**

An output of the Sprint Planning Meeting

Consists of the tasks for the Sprint derived from the Product Backlog.
Division of Concerns

During a Sprint Planning Meeting, scope, importance, and estimates are tuned continuously through face-to-face dialog between the Development Team and the Product Owner.

Scope and importance are set by the Product Owner.

Estimates are set by the Development Team.

Story Points

People are better at estimating “this is like that” than they are at absolute values.

Story point estimates do not change because of experience.

When estimates relative to estimates for other features, it does not matter whether they are correct, high, or low, so long as they are consistent... velocity is the equalizer.
Planning Poker

Everyone on the Team is involved in estimating every story – builds common understanding, collective ownership.

Deck of cards: 0, ½, 1, 2, 3, 5, 8, 13, 20, 40, 100, ?

Simultaneous estimation by facing card with time estimate in story points

Anchor with typical story being 5

Building the Foundation

![Effort vs. Sprint Chart](chart.png)

- Green line: Architecture & Infrastructure
- Blue line: Business Value
Sprint Zero

Not officially part of Scrum

Controversial – see LinkedIn / Certified ScrumMasters

Not really a Sprint
• Open ended
• Time when project initiation is decided
• May be when project charter is drafted
• May be when initial Product Backlog is built

Can devolve into specification writing!

Taskboard

Wallboard with columns for
• Not checked out stories and tasks
• Checked out stories and tasks (with name of person)
• Done stories and tasks

Plus Sprint Goal, Burndown Chart, unplanned items

Update during the Daily Scrum
Release Plan(ning)

Describes the goal of the release, the highest priority items in the Product Backlog, the major risks, and the overall features and functionality that the release will contain.

Establishes a probable delivery date and cost, assuming that nothing changes.

*Vodde:* Scrum only talks about Release Planning (not a Release Plan). A Release Plan is usually just a prioritized, roughly estimated Product Backlog, with stories roughly assigned to the next few iterations.

Daily Scrum Meeting

A time-boxed, 15-minute meeting used to inspect progress toward the Sprint goal and to make adaptations that optimize the value of the next workday.

Three questions:
- What did you do since the last Scrum?
- What got in your way?
- What are you going to do before the next Scrum?
“Done”

“Done” defines what the Team means when they commit to “doing” a Product Backlog item in a Sprint.

Completely “done” includes all of the analysis, design, refactoring, programming, documentation and testing.

Can have a checklist... or make the tester the “signoff guy.”

---

*Story Points and Velocity (Cohn 2005)*

Story points - a relative size measure

Velocity – a measure of a team’s rate of progress in story points per iteration

A key tenet of agile estimating and planning is that we estimate size but derive duration.

Planning errors are self-correcting because of the application of velocity

Estimating in story points separates estimation of effort from estimation of duration
A Burndown Chart (Kniberg 2007)

A Burndown Chart (Mason 2009)
Test Driven Development

1 – write an automated test
2 – write enough code to pass the test
3 – refactor to improve readability and remove duplication
4 – repeat

Test-driven development (TDD) is more important than both Scrum and XP.

Tester – someone whose primary skill is testing

The tester is the “signoff” guy.

Sprint Review Meeting

A four-hour time-boxed meeting (for one-month Sprints) that is held at the end of a Sprint where the Team presents the functionality done in the iteration to the Product Owner and other stakeholders.

The Team demonstrates and discusses the work done in the Sprint.
Sprint Retrospective

A three hour, time-boxed meeting (for one-month Sprints) held after the Sprint Review and prior to the next Sprint Planning meeting where the Team discusses what went well in the last Sprint and what can be improved for the next Sprint.

I’d say the retrospective is the second most important event in Scrum (the first being the Sprint Planning Meeting).


Barriers to the Success of Agile

A customer who insists on the big specification...

A culture that requires long hours to prove commitment...

Projects that are too big (more than about ten programmers)...

An environment with a long time to gain feedback (e.g., realistically test the software)...

The wrong physical environment (e.g., team members on different floors, not co-located)...

We do “agile”, just not most / any of the practices…
Work Environment

Seat the Development Team together
• Audibility
• Visibility
• Isolation

Product Owner is near, but not with the Team.

Managers should leave the Development Team alone – self managed.
• Read DeMarco and Lister’s Peopleware

No Silver Bullet

Scrum will not solve your problems.

Scrum will make your problems visible.

You will have to solve your problems.
Scrum Alliance

A not-for-profit professional membership organization created to share the Scrum framework and transform the world of work.

• Increase awareness and understanding of Scrum
• Provide resources to individuals and organizations using Scrum
• Promote the iterative improvement necessary to succeed with Scrum
• Hosts Scrum Gatherings and supports Scrum User Groups

Web: http://www.scrumalliance.org/

Questions
Contact Information

Dr. Mark C. Paulk  
Institute for Software Research  
Carnegie Mellon University  
Wean Hall 5101  
5000 Forbes Avenue  
Pittsburgh, PA 15213 USA

Email: mcp@cs.cmu.edu  
or  Mark.Paulk@ieee.org  
Web: http://www.cs.cmu.edu/~mcp/