Scaling Scrum
Learning Objectives for Conforming Courses

November 2014

by the Scrum Alliance Scaling Scrum Learning Objectives Committee

Introduction

Purpose

The Scrum Alliance intends to allow independent courses delivered by authorized teachers (such as CSTs) to be qualified as "a conforming Scaling Scrum course." Such courses may then be listed as "Additional Qualification" courses on the Scrum Alliance website, and graduates will receive an "Additional Qualification" badge associated with their existing CSM/CSPO certification.

These Learning Objectives (LOs) are the basis for evaluating and confirming candidate courses.

Scope of Scaling Scrum Approaches

The scope/mandate from the Scrum Alliance for these LOs is scaling Scrum. Other systems, frameworks, and methods may be elements of "conforming Scaling Scrum courses", while it is assumed and expected that scaling Scrum is the overarching framework. Hence, where applicable, the LOs assume conformance to Scrum rules and principles.

Scaling Scrum Learning Objectives versus Scaling Scrum Frameworks

These LOs do not themselves define an explicit framework for scaling Scrum. Different and numerous scaling Scrum frameworks may be taught in qualifying courses, as long as the course achieves these LOs. However, since the subject is scaling Scrum, the LOs inevitably assume or constrain scaling frameworks to some degree; for example, a Product Owner, Product Backlog, and a Sprint are expected.

Definitions

LO - Learning Objective.

Scrum - The standard version of Scrum communicated in a CSM course and consistent with the Scrum Guide, but that — crucially — is focused on a one-Team product group rather than scaling.

Large group, product group - The organizational unit that works together on one shippable product, including Development Teams, Product Owner(s), ScrumMaster(s), supporting managers, etc.

One Sprint - All the teams together in the same Sprint that starts and ends together and creates one common potentially shippable product increment.

Product - The term "product" is used to include what are normally called external shippable products, and "internal products" used within an enterprise that are often spoken of as *applications* or *systems*.

Scaling Scrum, scaled Scrum - Applying Scrum in the context of at least two Development Teams working together on one common product. It could be 100 Teams.

Team, team - A Development Team.

Prerequisites

Course participants have successfully completed a CSM or CSPO course.

Scope & Organization of Candidate Courses

Conforming courses may contain *more* subjects than coverage of the LOs, but the LOs are a minimum requirement. For example, a course may also cover Scrum applied to non-software-product domains, other non-Scrum-related frameworks, and so on.

Conforming courses can be organized in any way; we do *not* expect courses to have to be organized according to these LOs or major sections. And one LO may be addressed via topics dispersed within different course areas.

Submissions for a candidate course must include mapping information between the LOs and course content.

Introduction to the Learning Objectives

Categories - The LOs are organized in the following broad categories:

1. Change & Related Principles
2. Sprint Events & Coordination
3. Structure
4. Product Backlog
5. Roles
6. Multi-site Development

Comments & Suggestions - Some LOs have an associated comments section for Information that we felt is useful to provide clarification, context, or suggestions, though these are not rules or required LOs, and can be 'ignored'.

LO Category: Change & Related Principles

Introduction: Scaling Scrum has a major change impact on most organizations, and so a course needs to clearly address the topic. Some of the principles in one-team Scrum are also particularly significant to highlight when scaling Scrum, as they can and should influence the adoption itself. For example, *empirical process control*. Beyond that, there are many change models, principles, and techniques that are noteworthy but widely varied.

As an overarching comment and recommendation, we encourage that participants know that it *must be possible for true one-team Scrum to be workable before scaling Scrum can possibly succeed*, though this point is not itself a learning objective.

Summary: Wide-scope empirical process control and continuous improvement

LO: (1) Define empirical process control and continuous improvement. (2) Explain why the scope of these expand to the entire organization when scaling Scrum.

Comments & Suggestions: Empirical process control and continuous improvement are central to one-team Scrum, but the scope is the local team and its relatively simple environment. In scaling Scrum a much broader scope of these principles is important, in areas that previously did not adopt these principles or consider themselves part of a change. For example, within senior management of an R&D or Product Management organization. Also, empirical process control and continuous improvement need to be applied to the organizational design itself (structures, policies, processes, roles, etc.) of the large product group that is scaling Scrum; that scope is beyond what a one-team Scrum group traditionally considers.

Summary: Whole-product focus

LO: Explain the benefits of creating a shippable whole-product increment for the end of each Sprint, and organizing the multiple teams and processes towards this goal.

Summary: Systems/Models supporting large-scale organizational change

LO: Know, at a simple introductory level, at least two 'systems' or models related to large-scale organizational change or improvement. For example, Systems Thinking, Lean Thinking, Organizational/Scrum Patterns, Kotter Model, Integral Model.

Comments & Suggestions: The changes implied by scaling Scrum are not superficial; they touch on changing roles, positions, policies, groups, processes, and power structures. A participant leaving this course should appreciate that successfully introducing scaling Scrum will involve successfully dealing with organizational change. There are *many* large-scale organizational change systems; a participant should have some exposure to and awareness of at least some of these (and not assume there is only one approach), and consider applying their insights when introducing scaling Scrum. A change model can be a large topic; the scope of learning in this course is expected to be a brief, high-level, cursory overview/introduction just sufficient to raise awareness and know where to look for more information.

Summary: Techniques supporting large-scale organizational change

LO: Know how to apply at least two techniques for large-scale organizational change or improvement. For example, group-level Retrospective, kaizen, Open Space, Go See, PDSA, improvement backlog.

Comments & Suggestions: Driven by the same motivation as the LO for *Systems/Models Supporting Organizational Change*. We suggest a bias towards techniques common in the Scrum or agile community, so that the learner will be aware of common advice from others involved in scaling Scrum.

LO Category: Sprint Events & Coordination

Summary: Sprint Planning

LO: When there are multiple teams working together in one Sprint, know at least one technique for doing Sprint Planning, while also knowing that the technique can change.

Summary: Areas of inter-team coordination

LO: (1) Know the areas and motivations of inter-team coordination, including: working on related items, integration, shared components, skills development, agreements and standards, and evolving architecture. (2) Know why coordination by the self-organizing teams is preferred.

Comments & Suggestions: All of these areas exist when there is only one team, but are greatly heightened issues when there are many teams. In a large traditional organization, managers

(apparently) handle most inter-team coordination issues, so there is a large and unfamiliar set of coordination issues and skills needed for *self-organizing between teams*. The skills development starts with *knowing* the many areas of coordination, and the motivation for it. Also, the need for coordination is sometimes a side-effect of unnecessary or artificial dependencies, such as private rather than collective code ownership, or due to delayed integration, or "mini waterfalls". Therefore, the course participant should ideally be able to identify "self-inflicted coordination needs" and know that coordination often can and should be reduced by changing the underlying organization design.

Summary: Techniques of inter-team coordination

LO: (1) Know several techniques and structures for multiple teams to coordinate within different areas within one Sprint.

Comments & Suggestions: See the comments on the *Areas of inter-team coordination* LO. Example structures/techniques to teach include Getting Together and Talking, Communities of Practice, Open Space, Continuous Integration, multi-team Design Workshops, and Scrum of Scrums.

Summary: Product-Owner Team coordination

LO: If there is both an overall Product Owner and supporting ones (a "Product Owner Team"), know the issues related to their coordination, and at least one coordination approach.

Comments & Suggestions: In the *Roles* category, see also *Product Owner* role.Summary: Product Backlog refinement

LO: Know several techniques for multiple teams to participate in Product Backlog refinement. Identify when and why some form of overall or multi-team Product Backlog refinement is needed, and when not. Explain the tradeoffs and implications of separate teams doing separate Product Backlog refinement versus some form of overall or multi-team product backlog refinement.

Summary: Sprint Review

LO: Know several approaches to holding an overall Sprint Review, with special attention to the context of needing to inspect many Product Backlog Items (from multiple teams) within a short duration. Explain the motivations and benefits of holding an overall Sprint Review.

Comments & Suggestions: The meaning of *overall* in this context is one meeting for all teams, though not necessarily the participation of all team members. This does not necessarily replace also having a single-team Sprint Review.

Summary: Sprint Retrospective

LO: Know at least one approach to holding an overall Sprint Retrospective. Explain the motivations and benefits of doing so.

Comments & Suggestions: The meaning of *overall* in this context is one meeting for all teams, though not necessarily the participation of all team members. This does not replace a single-team Sprint Retrospective.

Summary: Initial Product Backlog creation

LO: Explain issues and techniques of initially creating a single Product Backlog in a large group when there are not only many teams but *many* stakeholders and perhaps even many sites, all in the context of also starting adoption of scaling Scrum (i.e., moving from a large traditional organization).

Comments & Suggestions: The distinctive point of this LO is not initial Product Backlog creation techniques per se; it is its creation in the context of *many* teams, *many* stakeholders, and perhaps many sites, all transitioning from a big traditionally-structured organization. For example, suppose a telecomm product group that made a 3G product then next decides to develop LTE, and there are 100 teams at five sites that will start the work; how to approach initial creation of a single Product Backlog for the whole product?

Summary: Release planning

LO: Know techniques and the motivations for large-group release planning.

Comments & Suggestions: Because large-scale development is often done with unnecessarily long release cycles, emphasize planning for and moving towards a classic Scrum model of actually shipping every Sprint, or reducing the duration between releases.

This LO is closely related to the *Initial Product Backlog creation* LO, in that the situations are similar, and some of the issues and techniques are identical. However, initial Product Backlog creation does not *require* planning for a release beyond the next Sprint; that depends on the situation. The special point of large-scale release planning is a release *more than one Sprint in the future*, combined with *many teams* (and perhaps sites) involved, probably believing they need to

estimate a gigantic set of vague "not ready" items on which they have *very limited information*, perhaps combined with needing to plan for *difficult-to-eliminate Undone work* not handled each Sprint (such as stability testing in a special hardware lab), and usually in an *ongoing rolling wave*. For example, how to do ongoing release planning in that context?

We recommend raising the following awareness for participants: Traditional large-scale development usually has a strong habit of very long-term and big-batch release planning, even sometimes with a group of people dedicated to this activity. After adopting a scaling Scrum approach, there is often an *habituated belief supported by a lingering organizational system* that the same degree and techniques of release planning remain necessary, when in fact they may not.

Summary: Software design/architecture activities

LO: Know several approaches and techniques for creating and maintaining a good software design/architecture when there are multiple teams coordinating within the Sprint, and also when there is collective code ownership.

Comments & Suggestions: A *software*-intensive product is being created; paying attention to the design and technical excellence is especially critical when there are multiple teams. Although Scrum does not (of course) mandate specific technical activities, people leaving a scaling Scrum course should be aware of some techniques/approaches in this crucial area, such as multi-team design workshops or an architecture community of practice.

Summary: Technical practices

LO: Know at an introductory level technical practices that strongly support scaling.

Comments & Suggestions: For example, multi-stage continuous integration. These can be complex topics and the participant need only have a brief cursory overview, and information on where to learn more.

LO Category: Structure

Summary: One Sprint

LO: Explain why a whole-product focus requires all teams to work in one Sprint that ends together and delivers one common potentially shippable product increment.

Comments & Suggestions: Explain how "asynchronous Sprints" or "own-team Sprints" cause problems in integration and synchronization between teams.

Summary: One integrated product

LO: Explain why scaling Scrum and a whole-product focus requires one integrated product at the end of every Sprint and how this increases transparency and control while reducing delay and risk.

Comments & Suggestions: Avoid cross-team integration that is delayed until near the end of a Sprint. Suggest ways to do this (continuous integration, etc.) .

Summary: Team organization

LO: Explain the advantages of teams organized as cross-functional, cross-component, and co-located resulting in each team delivering Done items that are part of one potentially shippable product increment.

Comments & Suggestions: Correct and working one-team Scrum is a necessary precondition for scaling Scrum. This LO underlines that point. But especially when scaling, this team organization has an impact on changing the organizational structures and assumptions in major ways with large and political implications, such as the elimination of existing single-function groups and related manager positions. So it is important that the participant be able to explain the motivation for and advantages of real Teams, as this will challenge the existing structures of most traditional large-scale organizations.

Summary: Team organization: Feature teams

LO: Define feature teams. Explain the value of structuring teams this way, and how scaling Scrum naturally leads to a feature team organization.

Comments & Suggestions: See comments above.

Summary: Team organization: Component teams

LO: Define component teams. Explain the drawbacks of structuring teams this way and how traditional development with its narrow-technology-specialization and late integration leads to component team organization. Explain how component teams lead to a sequential life-cycle and single-function groups with handoff. Explain situations in which having some component teams are strongly motivated.

Comments & Suggestions: This refers to architectural components. Consider teaching about transitional organizational structures that start with component teams and move to feature teams, when the existing organization is based on component teams. And consider teaching about situations that involve a mixture of both. In addition, introduce *scientific management* and how its lingering influence and ideas strongly relate to component teams and single-specialization.

Summary: Organizational structure

LO: Explain how organizational structure is impacted when scaling Scrum; how functional- or component-based organizations cause conflict between customer-focused teams and how organizational structure changes to have more customer-focus.

Comments & Suggestions: Share specific examples of organizational change that was triggered by adopting Scrum at scale.

Summary: Organizational policies

LO: Explain how organizational policies are impacted when scaling Scrum; how the focus on individual accountability and individual performance measures harms the shared responsibility of teams and how introducing competition between teams reduces the whole-product focus.

Comments & Suggestions: Share specific positive examples of changes in policy such as organizations abandoning performance appraisals or negative examples of what happens to teams when the organizational policy leads to a "my-team"-focus.

Summary: Product focus over project/program focus

LO: (1) Explain why and how Scrum normally has a long-term product-focus rather than a project/program-focus and how especially in large groups this changes organizational structure (such as the removal of a Project/Program Management Office) and practices (such as change in budgeting). (2) Identify situations that are valid exceptions, such as "fixed duration" *projects* done by software outsourcers, and applying Scrum outside of software products in *projects* such as a marketing initiative.

Comments & Suggestions: Explain how this change in focus causes a de-emphasis on project/program management and emphasizes product management.

Summary: Role of managers

LO: Know the role of managers in a scaled Scrum organization. Organizations are likely to still have managers but their management style changes from controlling the work and product to supporting the teams.

LO Category: Product Backlog

Summary: Single Product Backlog

LO: Explain the value of a single Product Backlog for a large group.

Comments & Suggestions: Explain the problems caused by separate team "product" backlogs.

Summary: Customer-centric Items

LO: Know how and why the Product Backlog should contain customer-centric Product Backlog Items rather than tasks.

Comments & Suggestions: Customer-centric items are relevant to groups small or large, but the violation of this principle is especially common in large groups, usually due to prior

single-function teams and sequential processes, so the participant should know why the principle is violated (by adding *tasks* rather than customer-centric items) in large groups new to scaling, and what to do about it.

Summary: Whole item to a team

LO: Explain why a ready item in the Product Backlog is not shared across teams but instead is implemented by exactly one team.

Summary: Gigantic requirements can be split small

LO: Explain that gigantic Product Backlog Items can be split smaller, while still keeping the items customer-centric. Be able to refer to at least one example.

Comments & Suggestions: "Our requirements are too big and therefore we can't adopt Scrum" is a common excuse or belief. It is so common especially when scaling that it is useful to address this concern by demonstrating that it is not true, with at least one concrete example based on a gigantic initial item.

Summary: Single Definition of Done

LO: Explain the motivations for a single common Definition of Done for all teams in the product group, including visibility and coordination. Know that different teams can expand it but not do less than the common Definition of Done, and know why.

Comments & Suggestions: Explore the relationship between the Definition of Done and which functional groups (UX, UI design, analysis, system design/architecture, test, …) are absorbed into teams. Explore how expanding the Definition of Done leads to more groups and the "Undone Organization" being eliminated and absorbed.

Summary: Undone work

LO: Explain how Undone work is created by having a difference between shippable and the Definition of Done (an imperfect Definition of Done) and how Undone work causes a lack of transparency, increased risk, and delay.

Comments & Suggestions: Also cover the common ways of dealing with Undone work, including: (1) Release Sprints (or HIP Sprints), or (2) an Undone Organization. Explore the advantages and

disadvantages of each. Explore how an Undone Organization is eliminated over time as the Definition of Done expands/improves.

Summary: Continuous improvement and adoption using Definition of Done

LO: Know how the expansion of the Definition of Done over time relates to continuous improvement of the organizational capability and how the Definition of Done can be used for gradual adoption in an organization.

LO Category: Roles

Summary: Product Owner role

LO: (1) Given the context of a single Product Backlog for a large group, know at least the common solution of one overall Product Owner to provide overall product strategy, coordination, and guidance. (2) If there are also other supporting Product Owners, explain their role in contrast to the overall Product Owner. (3) Explain at least one mechanism for their coordination.

Comments & Suggestions: (1) Although one overall Product Owner (PO) is not the only possible scaling Scrum solution, it is the model used in virtually all existing scaling models, so needs to be covered in a scaling Scrum course. (2) Other alternatives can also be explored, but prefer they are in use and directly known to the teacher rather than speculative. (3) Explore situations when one overall PO can be the direct PO for several or all teams of a product, and when not. For example, in a 2-team product group it is more likely. And when not (such as a 20-team product and/or many sites), and thus there are supporting POs. In the latter case, explore when one supporting PO can serve several teams or not. As a corollary, ensure the learner knows that both one-to-many and one-to-one PO to Team relationships are possible when scaling, and the trade-offs in these alternatives.

Summary: ScrumMaster role

LO: Know the expanded role and sphere of ScrumMasters in a large group, including teaching about and supporting: (1) large-scale organizational change, (2) Sprint events and coordination when many teams in one Sprint, (3) a Product Owner team, (4) managing a large single Product Backlog, (5) the issues of a Definition of Done and Undone work when scaling, and (6) partnering with managers. Also, explain the trade-offs of a ScrumMaster serving one versus several teams, and the advantages of dedicated ScrumMasters especially when scaling.

Comments & Suggestions: Probably maximum of 3 teams served by one ScrumMaster.

Summary: Developer role and "Team role"

LO: Know the expanded role and sphere of Developers and Teams in a large group, including (1) how self-organizing within a team is different than between teams, (2) areas and techniques for inter-team coordination, (3) why coordination by the self-organizing teams is preferred, and (4) new-skills development to become a flexible Developer.

Comments & Suggestions: (1) In a traditional organization, managers (apparently) handle most coordination issues, so there is a large and unfamiliar set of coordination skills needed for self-organizing between teams. See the related coordination "areas and techniques" LOs in the *Sprint Events and Coordination* section. (2) New-skills development is an issue even in a one-team Scrum product group, but is a much-heightened issue when transitioning from a large traditional group. Why? Because in a traditional group there is strong siloing of functional skills, where each person does only one kind of task. So when transitioning from a large traditional group there is a greater "multi-skill deficit", and a conditioned local-optimization mindset of "single specialization is best."

LO Category: Multi-site Development

Introduction: This section is not for the case of one team dispersed/distributed to multiple sites, or the case of a one-team product group with a remote Product Owner. It is for the case of large-scale development with many teams and many sites working together on one product. This is a large topic, and this course is not expected to take a deep dive into it. However,

participants leaving this course should have some introductory familiarity with this common subject.

Summary: Multi-site issues when scaling

LO: Identify common issues in scaling Scrum with many teams and sites supporting one product.

Comments & Suggestions: In addition to 'obvious' issues (e.g., time zones, language, second-class sites, coordination), note culture issues. Culture is a huge topic and this course is not expected to (and probably should not) explore it in detail, but the learner should probably be aware of the topic, and that there is a body of knowledge to help.

Summary: Multi-site development versus dispersed teams

LO: Define, compare, and contrast multi-site development with co-located teams, versus individual teams whose members are dispersed at multiple sites, when many teams and sites for one product.

Comments & Suggestions: Explain the advantages of multi-site development with co-located teams over dispersed teams, given many teams and sites for one product.

Summary: Multi-site solutions and techniques

LO: Explain some solutions and techniques to improve multi-site development with many teams and sites supporting one product.

Comments & Suggestions: Suggestions: on-site ScrumMasters, co-located teams, video calls, regular visits from a remote PO, teams visit other sites. Topics: how to conduct multi-site Scrum meetings, working with a remote PO, coordinating between teams at different sites.