

CSM Practicing Certification Renewal Assessment

Name: Lowell Lindstrom, lowelllindstrom@wideopenwest.com , 857-732-9330

Date: November 2005

Scrum depends on the inspect and adapt mechanisms of process control to manage the complexity of projects. For inspection to work, everyone must know what is being made visible. To implement the Scrum process, such regulating mechanisms as defined roles, involvement versus commitment, time-boxes, and regular cycles are used.

1. Describe one project on which you have used Scrum over the last twelve months.

I have been coaching a number of Scrum projects over the last twelve months. In most cases, project managers had been through the CSM training and were applying what they had learned. From this vantage point, I am able to observe the struggles and successes that individuals encounter when applying scrum in their project work. This report will describe one of these experiences.

Failure is, in a sense, the highway to success, inasmuch as every discovery of what is false leads us to seek earnestly after what is true, and every fresh experience points out some form of error which we shall afterwards carefully avoid.

John Keats, English lyric poet (1795 - 1821)

The project that I will describe had successes and failures, but ultimately cannot be considered a Scrum success. The organization was having success with Scrum on a different project and attempting to expand its use to all projects. The goal of the project was the porting of an application from a mainframe environment to a java-based architecture while adding new features to satisfy a regulatory requirement.

My involvement of the project began after the project had been labeled “in jeopardy” and the teams were resizing the remaining work and requesting a significant increase in the staff involved.

• Purpose - what business goal was the project intended to deliver?

Legislation created a hard deadline for the company to enhance this product. This was layered on top of an existing goal to port the application from the mainframe, introducing an enterprise J2EE architecture.

• Length - what was the duration of the project?

The project was scheduled for two years. Development ramped up with roughly sixteen months remaining on the project.

- **Cost - what were the budgeted and actual costs?**

The exact budget details are not available, but a resizing just prior to my involvement suggested that the project would cost double the original estimate.

- **Value - what were the projected benefits and actual (if measured) actual benefits?**

The project was driven by a legislated need; therefore the value was not articulated other than in absolute terms, i.e. if we don't do it, we don't operate.

Unfortunately, as part of porting the application, the existing functionality and the new requirements were treated as having equal and mandatory priority and value. This blocked the discussion of tradeoffs required to accomplish either goal of deploying new technology or meeting the new functionality requirement.

- **Size - how many people were on the project team(s), how were they organized into teams?**

The project was matrix and included 35 to 45 people. The teams were organized around the functional areas of the product, which also reflected the architecture of the system.

- **Teams - were the teams cross-functional and self-organizing? Were the teams collocated in an open space? Were the teams physically separated within one location, or located in more than one physical location?**

The team's were collocated and cross-functional. In fact, at first glance, the project was following Scrum closely to the book. However, Scrum is much more than a set of practices, steps, and daily meetings. This project did not embrace some of Scrum's principles, such as empowered and responsible teams, centralizing all backlog decisions through a single product owner, and an unwavering commitment by the team to make achievable Sprint Goals and to meet them.

A specific challenge in the team's structure was that the product owner and analysts worked in a different building on the same campus. Various techniques were attempted to close the feedback loop between the team and the product owner, however this continued to be a challenge. Whereas the analysts represented the Product Owner to the different Scrum teams, they were reluctant to make decisions based on a history of being overridden by the product owner after work had been completed.

The management structure of the teams was more complex than necessary and did not change to reflect the use of Scrum. Each team had a development manager and a delivery manager. The Scrummaster reported to the delivery manager and was typically a project manager, often without technical experience in the J2EE. Some teams were able to work beyond the org chart and distribute the work effectively. Other teams fell into a trap of too many chefs (managers) giving the team conflicting direction or no direction in the cases where consensus could not be reached among the management team. Ultimately, the key to any organizational structure is to succeed in spite of it, since there

is no one structure that can effectively facilitate a large group of people responding to a changing market.

• Initiation - how was the project initiated? How was the team trained to use the Scrum process?

The team started as a small 4 person team evaluating the technical feasibility of moving the application to J2EE. Although successfully validating the architecture, the lack of visible business value to the Product Owner left the organization perceiving that the project was significantly behind. This led to a tripling of the team's size over the course of one month. It was the classic "throw bodies at the problem" reaction. Due to the perceived time pressure, no team initiation or forming activity was held. Recommendations were made to create a learning/team forming event, but this was not approved. Many of the new team members were new to Scrum and agile. This bogged the team down as each person gave their impression of why agile would or would not work.

The Scrummaster was a CSM, but no other training was provided. We joined the team as mentors after the team swelled. We used lunch 'n' learn meetings and did targeted module training where possible to get the team to understand the programming practices that we were deploying.

• Reporting - how did you report progress to management and the customers?

The backlog was communicated using a spreadsheet including all of the required features. The product owner managed this through each of the resource managers of the teams. The entire product backlog was slotted to the next six sprints based on the teams' estimates. This provided some validation that the teams would be able to meet the mandated delivery date. The teams that had the trust of management to deliver were able to adjust their backlog as required to reflect changes. The teams that were perceived as being the most at risk were not allowed the same flexibility. In their case, the project was managed with a traditional conformance to plan mindset. For these teams, any change to the original definition of all of the sprints resulted in a stressful escalation up the management chain, usually via emails from the Product Owner to the executive team.

Burn down charts were used, but rarely did they align to what the team said, For example, the burn down chart would show that the team was behind in progress towards the Sprint goal, but the team would still state that they would make it. Sometimes they did, but other times they did not.

As part of our involvement, we introduced acceptance testing as the primary indicator of progress. By using *Fitness*^{*}, the ambiguity around when stories were "done" started to clear. As teams saw the effectiveness of automated acceptance tests, they worked hard to get testing in place. It fell short of becoming a report sent to management, but was headed in that direction.

* www.fitnessse.org

• Change - what difficulties were surfaced by Scrum that had to be resolved? How were these resolved?

Transitions to agile methods will typically erase any façade that hides deficiencies in the functioning of the software development organization. In this organization, the success of the first Scrum project masked the challenge that a transition to Scrum can cause. The primary difficulties involved the management structure and the programming practices.

Some of the management structural problems are described above. Above the team level, different executives were deploying different and conflicting improvement initiatives to try to save the project. The organization was simultaneously deploying a Scrum initiative out of Development, a PMI initiative out of the Project Management Office, and a process metrics initiative out of the delivery organization. None of these silo focused initiatives could be successful as localized improvements. All required organizational support and buy-in. Kaboom!

To resolve these problems, we began meeting with participants with each of the initiatives in attempt to integrate the different initiatives. This was effective to a point, but in the end the conformance to plan bias of PMI and metrics did not accommodate the inspect and adapt mechanisms of Scrum. Since the senior management team had not engaged in the Scrum initiative, these conflicting mechanism were not resolved and to yielded only to political influence.

• Management - what was the previous role of the ScrumMaster? Who took on the role of Product Owner? To what degree were they successful in fulfilling their roles?

The SMs were project managers prior to Scrum. The Product Owner was from the business and was empowered to make decisions about the functionality of the system. Different teams had different success. In one case, the SM adapted to his new role, focusing on removing obstacles from the team and experimenting with new tools for tracking the project. This team had a strong technical lead with whom the ScrumMaster collaborated well. The Product Owner praised this team for their predictability and responsive delivery.

The other team was less successful. This team had a more complex problem to solve. The core of the team, which had been together for a year, was comfortable working independently from each other, and funneling all project status through the project manager. This was sufficient when the team was small and focused on technical feasibility; a phase of the project which was acknowledged as being less predictable and was thought to require less customer involvement. The ScrumMaster struggled to find a valuable way to contribute, resorting to orchestrating the daily sprint meetings, focusing on the data required to update the burndown chart. Tasks and estimates were assigned rather than developed by the team. The same Product Owner routinely communicated concern and risk about this team's progress.

These contrasting examples from two Scrum teams on the same project with the same Product Owner highlight some of the realities and challenges of Scrum on larger projects. The Scrummasters involved were both CSMs. Despite well intended efforts on both parts, one team was effective in a collaborative setting, the other was not. The Product Owner worked well with one team, but was frustrated by the other. The Product Owner trusted that the effective team would adapt to changing circumstances. With the ineffective team, there was less faith that the team could adapt, thus the Product Owner and other managers focused on having a plan to measure against.

• Engineering - what software engineering practices or environment had to be changed?

The team was urged to collaborate and communicate more. Although this may not seem like an engineering practice, it is critical for continuous integration to work. Automated testing was introduced via Fitnesse to remove ambiguity from the completion status of the Sprint goals. Organizationally, the build process was managed by a centralized group, which limited the ability of the team to ensure successful nightly builds. TDD was introduced.

• Stabilization - for how long did the software have to be stabilized before it could be released? How did you structure this stabilization process?

This metric is not available. They were not integrating after each Sprint making it impossible to assess the true stability of the software. With the use of automated acceptance testing, each team started to get a sense of the stability of their part of the software.

• Success - to what degree was the project successful? To what degree was the Scrum process instrumental in the success of the project?

The project was restructured as data revealed that the original estimates did not reflect the capability of the team that was available to do the work. The revised estimates showed increased cost and risk. As a result, a new project was formed to satisfy the near term functional requirement on the existing software infrastructure. Although not successful in helping the original project meet its goal, Scrum was instrumental in the flushing out the true project status early enough to adapt to a less risky approach.

• Scrum Process - to what degree was the Scrum process implemented "out of the box?" To what degree did you have to modify the Scrum process for this project? For each modification, how did you formulate the modification so that the basic inspect/adapt mechanisms continued to function? What parts of Scrum couldn't be implemented, or failed, and why?

This project exhibited many of the common challenges facing individuals, teams and organizations when transitioning to Scrum. Although most of these challenges are not surprising, we have yet to as a community to provide a sufficient learning infrastructure to achieve a higher rate of success with actual application.

The project was successful with:

<i>Elements of the scrum implemented</i>	<i>Opportunities to improve the use of scrum</i>
Use the monthly sprint rhythm to establish time-boxed deliverables	Address backlog problems earlier and more aggressively.
Collocating the team to enhance communication	Use team forming techniques to encourage greater collaboration amongst team members
Daily sprint meetings	Keep the focus on what is needed to achieve the sprint goals.
Regular interaction with the product owner	Confront and resolve conflicts in the product backlog and in how the team and product owner interact. Use more successful teams as a model
Maintaining a product backlog	Given the delivery date requirement, reduce the size of product backlog items to ensure that accurate estimates can be established

On the team that was not meeting their Sprint goals, we implemented a mid-Sprint check to help the team better assess their progress towards the goal. This had the effect of creating two 2-week iterations, increasing the opportunities for inspection and adaptation.

2. How do you cause the accuracy of Product Backlog estimates to improve? To what degree does their accuracy matter?

- Larger backlog items are more difficult to estimate accurately than smaller ones. Breaking the backlog items into smaller elements of functionality will facilitate better estimates.
- Use team estimating techniques such as wide-band Delphi to reduce the error that an individual might make.
- Three point estimates (optimistic, expected, pessimistic) can express the uncertainty associated with an estimate.
- During the sprint review, discuss the estimates that were inaccurate and determine specific actions to take during future estimation sessions
- Do not estimate backlog items that are not sufficiently understood. Add a spike (bounded experiment) to the current sprint to increase the understanding of the backlog item

Accuracy in the estimates of product backlog only matters to the extent that we must predict the delivery date, scope, and cost of the project (quality being a byproduct of these). At one extreme, some projects can tolerate fixing the cost (usually team size) and delivery date, but accept whatever the team delivers. In this case, the team can focus more energy on delivery more (productivity) rather than estimating and predicting. On other projects, the market may create the undesirable and risky situation where the three

parameters are fixed. In that case, part of the risk reduction strategy for the project will need to be greater accuracy in the estimation of the product backlog.

3. How do you cause the accuracy of what a team commits to for a Sprint to what the team actually delivers?

As a Scrummaster, one does not “cause” the accuracy of what the team commits to versus what it delivers. We can, however, be very effective at facilitating the successful achieving of Sprint goals. This starts with being realistic and disciplined about the goals we commit to in the Sprint. The most common problem that prevents teams from achieving Sprint goals is over committing in the first place. The Scrummaster must help the team consider all of the activities required to satisfy the goal. This can be particularly challenging for the team during an agile transition, since the team members may not be familiar with the new practices being used. I normally see the predictability for the Sprints start to settle in roughly 2-3 Sprints if the team is using the mid-sprint checks.

Social pressures can cause over optimism on the part of the team members. Often individuals believe (and in some cases, rightly) that giving aggressive, risky estimates is part of what it takes to get ahead. Challenging ourselves to do more than we thought possible is an excellent motivator. However, in the absence of reliable inspection, this can develop into bravado and reinforce a culture that values aggressive action over delivering. Sprint Retrospectives will highlight this problem when performed diligently and honestly.

One of the often overlooked opportunities to achieve the Sprint goal is to allow the clarification of the exact implementation to happen in the Sprint. This gives the team and Product Owner a knob to turn together to meet the goal with the appropriate elegance in the solution. If we are inspecting throughout the Sprint, knowing where we are relative the goal, then we can engage the Product Owner in evaluating different implementations that allow the most value to be delivered in the Sprint.

Other important things to reinforce include:

- Preventing pressure from product owners and other chickens to over commit
- Helping the team stay focused on the Sprint goal
- Using a Mid-Sprint check, favoring having 50% of the goal done at the midpoint.

4. What metrics do you use to track the development process? Which metrics have been changed, removed, or newly implemented as a result of using Scrum?

The primary metrics that I encourage teams to measure, chart, and post in the work area are Velocity, Passing Acceptance Tests, Percent Acceptance Tests Automated, and any metrics that reflect the business goal of the project (e.g. Customer complaints, rate of

installations, efficiency of customers workflow, etc.). For acceptance tests, I am advocating the Acceptance Test Burndown Chart which burns down (or up) the passing acceptance tests as the Sprint progresses. Overtime, I expect acceptance testing data to be more valued as an indicator of progress than task data.

The team also should have transitory metrics based on current problems or improvement goals. A common example is to chart the length of the Daily Sprint meeting if it is taking too long. Once the discipline of that meeting is reestablished, this metric won't be required.[†]

5. What type of training, resources, or tools would best help you successfully employ Scrum in the future?

Most of my work with Scrum has been in organizations that come away from the CSM course with a greatly over-simplified expectation of what it will take their organization to transition to Scrum. A course or workshop that brings the reality of organizational change to the transitioning organization would help set this expectation better.

A course on the importance of and techniques for Open Workspace and team collaboration would help sensitive team members to the interpersonal dynamics required to succeed in team environment.

A course which provides techniques for removing the ambiguity around done would help establish the value of acceptance testing.

Although offerings are cropping up for the Product Owner, these are just scratching the surface of how to change Product Management to leverage the capabilities of Scrum.

[†] Shoji Shiba et al., *Four Practical Revolutions in Management*, Productivity Press, Portland, Oregon, 2001 gives excellent treatment of using metrics to drive continuous improvement, breakthrough improvement, and process standardization (Chapter 6).

1. (Optional) Scrum and Extreme Programming are sometimes used together. What must be considered when this is done?

The following table is from a paper (in draft) entitled, “An XPers Guide to Scrum.” The goal of this paper is help XP teams understand and leverage the elements of Scrum that can help them improve their delivery.

Scrum/XP Definitions

Scrum Terminology [‡]	XP Equivalent or Approximate	Notes
Burndown Graph	Big Visible Chart (BVC)	The burndown graph is a specific BVC that is used to track the remaining work in a Sprint. XP suggests a number of charts/metrics at the teams discretion. [§]
Chicken	None	Stakeholder and Member of the Project Community are terms that most equate to this role.
Daily Scrum meeting	Daily Standup Meeting	Scrum time-boxes this meeting. XP does not specify the time-box, but would support a time-box as a corrective action to elongating meetings.
Done	Done	In XP, this is usually referring to the completion of a User Story and the standard for completion is the passing of one or more automated acceptance tests
Estimated work remaining	Tracking data	The estimated work remaining and the associated Burndown graph are articulated specifically in Scrum. In XP, this is one viable approach to tracking.
Increment	No formal equivalent. “The Iteration” or “Iteration X” are often used.	The functionality delivered by an XP team during an iteration is a set of stories. XP does not define a name for that set of stories.
Increment of potentially shippable product functionality	Last Iteration Prior to Release	
Iteration	Iteration	Iteration and Sprint are synonymous in Scrum
Pig	None	Members of the Whole Team would be considered Pigs in XP
Product Backlog	Release Plan	In XP, these are simply the stories that are not in the current iteration. “Backlog” and “Future Stories” are often used.

[‡] Agile Project Management with Scrum, Ken Schwaber, Microsoft Press, 2004

[§] See Jeffries, et al, *Extreme Programming Installed*, Addison Wesley, Boston, 2001

Scrum Terminology[†]	XP Equivalent or Approximate	Notes
Product Backlog items	User Stories	XP Planning focuses on items that result in work that creates software functionality. These items are called User Stories. Some XP teams write “User Stories” for non-functional requirements in a way that approximates Scrum. In Scrum, planning focuses on any item of work required to deliver value to the business. (see also Sprint Backlog Task)
Product Owner	Customer	XP has embraced the notion that this is a team activity. In Scrum, the Product Owner will often be leading team that is performing the work required to own the product. Accountability and authority remain with a single Product Owner.
Scrum	Extreme Programming	The name of the set of the practices.
ScrumMaster	Coach and Project Manager combined	The distinctions here are subtle and will depend on the style of the individual in the role. XP softens the role of the Coach, whereas the ScrumMaster is a critical role within Scrum.
Sprint	Iteration	In Scrum, these last 30 days. There is little flexibility in the duration. In XP, the most popular iteration length is two weeks, with some teams preferring one week and others preferring three weeks. Some XP teams will adjust the iteration length (prior to the iteration) based on the uncertainty of the contents of the iteration.
Sprint Backlog	Iteration Plan	
Sprint Backlog task	Task	In Scrum, all elements of work are tasks. In XP, functionality is expressed as User Stories. The work required to implement the User Stories is expressed as tasks.
Sprint Planning Meeting	Iteration Planning Meeting	
Sprint retrospective meeting	Retrospective	As with most Agile methods, the Retrospective has become the standard form of feedback and team self-assessment.
Sprint review meeting	No formal equivalent, but most teams have a name for it.	“Iteration Closeout”, “End of Iteration” are common names. In XP, passing acceptance tests are used to validate that the iteration is done. As such, a formal review of all of the Product Backlog is often replaced by a review of the results of the automated tests from the last build.
Stakeholder	Stakeholder	Member of the Project Community is also becoming a popular term for the stakeholder.
Team	Whole Team	
Time-box	Time-box	In XP, the iteration is time-boxed. Scrum time-boxes all of its specified activities.