

CSM Practicing Certification Renewal Assessment

Name: Stacia Heimgartner
30, 2005

email:

date: November

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.
Describe:

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

The purpose of the project was to integrate Primavera's web base product (myPrimavera) with a collaboration tool (Interwoven) to provide lightweight integrated project management and collaboration capability for project teams.

Scrum was implemented approximately four months into the project, as our VP observed that we were stuck in design paralysis. Teams of designers, business analysts and product owners had engaged in approximately 3-4 months of design; no development work had started as the design was far too complex to 'nail down' in order for actual development to begin [so we thought]. Stuck in the traditional waterfall mentality, nobody could imagine beginning developing a product that the design wasn't 'complete' for.

With Scrum giving us a 'can-do' attitude, we began our first sprint with the goal of working functionality – albeit skeletal at best – to prove to management that the Primavera and Interwoven databases could communicate and that workflow/document management solutions weren't pipe dreams, but certain realities. From there, following Scrum principles, the teams were able to deliver the integrated web product, plus an upgraded client/server product, all within nine months' timeframe. The timing advantages of Scrum were leveraged in that we could 'call' the release whenever necessary, thus allowing us to show new and exciting [and working!] functionality to customers at the Annual Conference – three months before General Availability.

- **Length - what was the duration of the project?**

The duration of the project was approximately nine months.

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

We were a cost center and projects were not budgeted in a traditional method. The project started and finished with the same number of resources always used for development purposes.

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

Immediate value of over \$14m in revenue from one particular customer. Other value derived from creating a footprint in the industry with the first lightweight project management/collaboration tool, providing significant competitive advantage over other companies increasing their foothold in the project management space.

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

I was scrum master of three teams, of seven total project teams. My teams were tasked with 1) creating an entirely new 'portal'-based UI [The Portal Team], 2) integrating Primavera database with that of Interwoven's for workflow creation [The Project Initiation Team], and 3) creating a document management system [The Document Management Team]. The other teams were focused on the client/server and other modules of the application. The teams were mostly split up into Feature teams; some into components.

- **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?**

A very exciting story here! The teams, at first, lived in their cubes, communicating via email and defect system. My Portal team, however, was the very first to collocate at Primavera. It took lots of persuasion and a "c'mon, just try it" attitude. They almost backed out at one point, but most of the team decided to go ahead with the move by taking over a very large conference room, everyone working around a big conference table. Team members brought in their desktops, laptops, a server or two, bowls of candy, and even their favorite plants. After a week of this new, exciting and morale-boosting work, several more team members moved in, until finally, the entire team was collocated. Productivity shot up, the room was abuzz with energy and development, and this team couldn't have been happier. Daily scrums became a mechanism for chickens from other teams to catch up since the Portal team was driving most of the new functionality – the team itself stayed synched up by nature of working together. Osmotic communication happened all of the time, and several team members described in retrospectives just how beneficial this was.

Next thing you know, my other two teams moved into smaller conference rooms together, and then all of the 5.0 teams eventually followed suit. Budget was set aside for floor renovation! Down with the cubes! Most of the cubicles were reworked in order to create large team spaces; however, small offices were kept here and there so that employees could have personal time for phone calls or sanity. Out of approximately 85 developers, only two or three never really got used to the collocation idea, but went along and were able to cope with the change.

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

The project was initiated in January of 2003 by use of the traditional project management approach (waterfall). The team was trained to use Scrum in the April '03 timeframe by Ken Schwaber. Scrum masters were trained and certified, and Ken met with teams monthly to ensure we were following the scrum disciplines. We called him the Hurricane because when he would visit, it was like a Category 4 swept through. Usually we were caught slipping back into what we knew – our comfort zone – but Ken would help us realize for ourselves that we were straying off track, and then help us get back of track.

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

Our early reporting attempts during the scrum transition were a mess! Reporting was a nightmare to start, since we were switching the project management paradigm on executives who basically invented PMI and wrote the book on earned value management. We began reporting to management by use of the product and sprint backlogs; however, after one lengthy meeting with the COO and CEO, we realized that the reports were too narrow-focused. Showing each team's sprint backlog was way too deep in the weeds for executives. I remember one quote as "so what you're telling me is that we're going to get more, for less?" During this switch from old to new, we had to incrementally improve on our reporting. We learned to leverage sprint demos as a way to

see 'where we were'. Seeing it at the demo was a checkpoint of working functionality. Reports then became easier once the management and executive team had seen this working functionality. We learned to use the product backlog to baseline and then predict the 'end point' every sprint. We also learned how to use the Primavera tool as a way to track sprint and product backlogs. There was also beginning development of a scrum module to integrate with the existing toolset, but by the time I had left, it was not in production.

What we did learn was that no previous project ever had accurate reporting; we basically 'winged' it. For the first time in Development, we knew where we stood.

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

The biggest changes, often immeasurable, were cultural. Empowerment brought fear and resistance.

One example of this was when our VP ruled out overtime. It was not allowed any longer. I will never forget the look of astonishment and disappointment on team members' faces as they were told that they needed to spend time with their families as their lives outside of the office were far more important than the work occurring inside the office. During the ensuing sprint, it was discovered that two team members worked overtime to deliver a great piece of functionality. To their surprise, this was not met with a pat to the back and an 'atta boy', but instead very harsh words that if they ever did it again, they would be heavily reprimanded. It took a strong VP to help institute this cultural change and to help people understand the concept of 'sustainable pace.' With sustainable pace, and with people who are energized and happy outside of work, teams become efficient at meeting realistic goals. They become credible and reliable.

- **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?**

My previous role to that of ScrumMaster was project manager. The marketing folks took on the role of product owner. I feel that we were very successful in fulfilling our roles.

As a previous manager, the move to ScrumMaster was very difficult. I had a tough time losing that competitive tendency and realizing that leadership was much more important and effective in helping teams self-manage. At Primavera, which is based in traditional project management, there was a bit of ego associated with being a project manager there. 'Losing' the PM title for that of ScrumMaster challenged me to grow in ways that I had not before. I had to become the team champion, the shepherd, the leader. I had to learn to influence without having power. I took advantage of our VP's leadership program, taking one idea per month and using that to help me become a stronger leader and facilitator. I also had a leadership buddy who helped me grow. It was interesting to note that I was not the only new 'leader' struggling with these new challenges. We all were; it shook us up. But the good news was that we had each other to rely on.

The marketing folks struggled at first with their new roles. Before Scrum, there was a strong sense of 'do as I say' from Marketing to Development. Basically, the teams were expected to code everything and deliver on a date that Marketing expected. Scrum allowed the development teams to push back; it gave us the framework of a backlog to promote a sense of fairness and trade. What could never be stated before – that we simply cannot develop everything they wanted every time they wanted – was finally able to be said and heard. The product owners became more of a part of the team than they ever had. Although they weren't daily team members, they certainly made themselves

available for the team for questions and clarification. They almost always attended demos.

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

After 7 or 8 sprints, it was determined that the legacy client/server code must be refactored and partitioned correctly as a foundation for test driven development. The developers knew that TDD using Fitnesse was certainly the best way to go. It was a natural progression once teams collocated. The code was eight years old, and it was time for an overhaul. This movement began right as I was leaving to start my services engagement, but I believe that most has been completed and parts of it are still underway. Teams are using Fitnesse today.

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

Stabilization occurred in two parts. After sprint 5 [approximately] we determined that there were over 200 defects recorded in the web product [some of which were there prior to the start of the project]. Luckily the testing coverage was pretty high [around 80%] and we had been running full regression testing during each sprint on newly delivered functionality. However, there was still much new functionality to be delivered, and we realized and admitted that we were not an iteration away from shippable. Taking this into account, the teams and management decided to take a stabilization sprint to gain control before continuing with new functionality. After reducing the defect count to less than 50 [approx 7-10 defects per team], the teams felt as though adding new functionality would certainly be less risky. They continued with sprint 7 adding new functionality, but also had a new goal of less than 5 defects at the end of the sprint. The combined approach of reducing risk by reducing defects plus selectively adding high priority backlog items allowed us to guide the release in with the right combination of functionality and quality.

- **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 a huge success. It achieved the business goal of securing a footprint in the lightweight PM collaboration tool space. It also introduced new and exciting functionality for our existing user base, allowing teams to collaborate on projects in new ways. It also was the start of bridging the gap from high end, costly client server technology, to less expensive web-based technology, allowing new customers to employ PM software without such a high cost. We were also able to 'call' the release in time to show major functionality at the annual conference, three months before release to market. This flexibility, afforded by Scrum, allowed us to sign business before the release had shipped.

The project was also successful in that it turned around a team of 85 professionals who had previously been drug through the mud on projects. There were no more death marches, no more Gantt charts that so 'accurately' told us where we were. There was a huge sense of accomplishment and camaraderie amongst the teams. It was a remarkable effort and a wonderful project to work on. We all learned so much about ourselves.

- **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?**

The Scrum process was pretty much implemented 'out of the box.' We really did not modify the process as we found that it worked the way it was. We did eventually use the concept of 'rotating ScrumMasters' so that each person on the team could understand how to remove obstacles, as well as engage product management, etc. This interchanging of roles helped to cement everyone's knowledge of the process and what each role entails. There was no part of Scrum that wasn't implemented; I think we realized that once part depends on the next, and that the basic disciplines are intertwined and integral to success [at least for us]. When we came to questions of what was right or wrong, we employed the Scrum principle of 'it's about common sense.' This helped us out of tricky situations.

One thing we had to watch out for was the mentality that 'it's Scrum. If we don't get to [that piece of functionality], then we just don't get to it. We'll pick it up next time.' It was important to reinforce to the teams that they should strive for 100% goal attainment, and that Scrum provided the framework and the tools to do that. Teams learned that through Scrum, they could become functioning business units, and engage with parts of the organization that they had never been able to, or been allowed to, before.

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

During release planning, the teams can decompose the product backlog items far enough to understand the underlying components and estimates. However, I believe that it is reasonable for these estimates to not be so reliable at first, but to improve upon them with each ensuing sprint. The closer teams get to implementing the functionality, the more they understand the product backlog and what the backlog entails. It is important to communicate the estimates to the business as just that – estimates.

One way to improve the product backlog estimates is to have an engaged product owner working with the teams to consistently communicate the vision of the product backlog requirement.

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

*Proper **team** planning and decomposition increases the accuracy of backlog estimates. Some teams want to rush through this step; I do not let them! The degree of accuracy matters greatly when trying to help a team meet its goals. If a team does not plan out the sprint appropriately, they don't really have a concept of what the performance challenge is. When a team understands all the parts and pieces, and signs up for that work together, it is more likely to be successful in attaining the goals of the sprint. A properly planned sprint backlog, planned by the team, allows the team to be successful.*

It is important to help a team stay on track for what was promised. Clearing obstacles out of the way, promoting communication with the product owner in times of uncertainty, as well as up-front decomposition are tools that I use to help increase goal attainment. When a team 'wins' with a high acceptance rate of sprint deliverables, the resulting energy is unstoppable.

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 Product and Sprint backlog and burndown chart are wonderful tools for self-management and for improving accuracy in commitments. Just as important is educating the team about how to use these tools, and reminding them that they are responsible for its creation and upkeep. A team cannot go anywhere if it first does not understand where it is. One tactic I use with new teams is when I see they're struggling, usually I notice their backlogs represent the struggle. At the end of a daily scrum, I will bring up the fact that there are no obstacles reported, yet the trendline is flat, or on the upswing, or whatever the case is. I then ask the team simply 'why is that?' Usually, the underlying issues begin to surface.

I LOVE seeing the 'percent complete' statuses go away. I love it when the language of "we're on track" is replaced with functioning code. I try and educate my teams/clients that there is no better metric than working code. I have been lucky in that my clients have not asked me to translate scrum metrics into waterfall metrics, but I do understand this happens in other places. During release planning and with each subsequent sprint planning sessions, I do like to have teams identify cross-team dependencies and post these to a wiki for everyone to observe. I also encourage mid-sprint health checks, especially for new teams, to help them understand how they are tracking against their commitments. I strongly recommend a strong scrum of scrums team to help facilitate and resolve cross-team issues and dependencies.

The more mature the scrum implementation, the less emphasis on bug arrival/fix rate metrics; however, for new scrum teams, this metric can be valuable, depending upon the quality of code introduced into the code base. What a team does 'now' certainly pays off later!! Helping them understand that all of this work constitutes the 'done' picture helps them in estimating and increasing accuracy in those estimates.

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

A leadership course!! A course for project managers who are 'crossing the chasm' to ScrumMaster-land. Keeping active in the Agile community also helps, and I will seek Facilitator certification to increase my level of facilitation skills.

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

The inspect-and-adapt cycle. Scrum can work with any engineering practice, but its disciplines of backlog/demo/review/feedback cycle must be upheld.