

# CSM Practicing Certification Renewal Assessment

Name: Tobias Fors

email: [tobias.fors@citerus.se](mailto:tobias.fors@citerus.se)

date: 2005-05-17

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?
  - A new image management system, needed to support print and online publication efforts.
- Length - what was the duration of the project?
  - Around 12 months.
- Cost - what were the budgeted and actual costs?
  - Confidential. Project came in slightly over original budget.
- Value - what were the projected benefits and actual (if measured) actual benefits?
  - The projected benefits of the new system where: 1) increased operational stability (current systems were highly unstable), and 2) that an improved system would make it possible for the organization to expand its online business, something that was held back in part by the current systems in use.
- Size - how many people were on the project team(s), how were they organized into teams?
  - The project started out with a couple of business analysts doing requirements work. After I entered the project, one developer was asked to help out, later on one more was brought in, and finally a third one, plus one person that focused on testing.
  - Surrounding the team were the usual suspects from a big IT organization, whose help we tried to use as best we could, while attempting not to get caught up in big bureaucracy.
- 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?
  - Initially, we were split up over several floors in the same building, but over time, we managed to first get collocated on one floor, and then get a dedicated project room, which resulted in an amazing communications boost.
- Initiation - how was the project initiated? How was the team trained to use the Scrum process?
  - When I was called in to help with the project, a couple of business analysts had been working on a bunch of specification documents for a few months, and the problem that needed solving had been identified about a year before.
  - I proceeded pretty much by the book (not knowing many other ways to proceed...), and gathered all that had been involved so far and facilitated the creation of a product backlog, explaining that we would refine the details as we went along. I presented the basics of iterative work. This concept was very well received by the product owner.
  - A candidate product that seemed possible to adapt to the organization's needs had been identified, but after this, not much had happened. I suggested we recruit a skilled developer that could help us investigate the third-party product the client was interested in.
  - I soon discovered that the entire organization was wrapped up in an attempt to implement RUP and an additional project management framework. I stepped on a few toes arguing that we needed to focus more on learning during the project than on trying to create perfect specifications up front (and yes, I know that's not

what RUP tells you to do, but for some reason, it seems that's what people think RUP tells you to do). I made the decision that it would be most efficient to try to explain what we were trying to achieve in terms of the methodologies the organization was trying to learn. For this reason, I avoided excessive use of Scrum terminology. Instead, I tried to find familiar terms for the different important concepts of Scrum, so that we could benefit from Scrum without getting slapped for trying an "illegal methodology". The Art of the Possible in Action...

- I spent considerable time during the first weeks talking with "process specialists" that were sent to investigate our project. Only after several such encounters did I realize that I was actually actively shielding the other project members from having to deal with bureaucracy. In the end, I was called to a "project review" meeting with representatives from the different functional areas of the organization. My first instinct was to go to the meeting alone, in order to allow my colleagues to keep working on the solution, but I changed my mind and decided to extend the "invitation" to the developer and the business analyst I worked with at that time. At that meeting, rather than just me as project manager trying to explain the project, we all chipped in – and each and every one of us could explain where we were heading, why, and how. That felt extremely good, and after this, the number of investigations into our project seemed to fade away, and our communication with the different functional groups seemed much improved.
- Reporting - how did you report progress to management and the customers?
  - We held demonstrations at the end of every sprint. Aside from this, budgeting and budget follow-up was handled by me and a person permanently employed by the organization. Normally, project managers would submit budgets that would then be reviewed. We chose to do "pair-budgeting", gathering before one computer and working with the numbers – me explaining my view of things, and my colleague explaining the intricacies of the budgeting system.
- Change - what difficulties were surfaced by Scrum that had to be resolved? How were these resolved?
  - The major difficulty we ran into was a consequence of us having underestimated the need for qualification of the third-party product we employed. Documentation was scant, and the product seemed less than stable. In the end, one of the developers established a direct contact with one of the lead developers on the third-party product's development team. This resolved many, but not all, problems.
- 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?
  - I was first retained as a project manager-for-hire, and told that I was to use the in-house methodologies of my client. However, I felt that we could never succeed under those conditions only, so I dubbed myself Scrum Master. The person who had requested the project to begin with turned out to be an excellent product owner.
- Engineering - what software engineering practices or environment had to be changed?
  - While recruiting, I specifically tried to gauge the developers' familiarity with engineering practices such as the use of unit tests and refactoring. During the interviews everything seemed fine, but I soon learned that things were different in practice. I ended up having one-on-one conversations with the developers, trying to explain my view of why programmer testing and design improvements are both important, and asking them to tell me their view on the issue. Things improved after this, but we never reached a state where we could confidently say that our code had an extensive test suite.
- Stabilization - for how long did the software have to be stabilized before it could be released? How did you structure this stabilization process?
  - After twelve months, my contract with the client was set to end. A person had been permanently retained to take responsibility of the system. We worked in

tandem for the last couple of months, during which the system became gradually more stable.

- Success - to what degree was the project successful? To what degree was the Scrum process instrumental in the success of the project?
    - Even though we tried, the system was never successfully released to production during the project. We tried several times, but each time discovered a new impediment. After the twelve months however, the system went live, and soon replaced the most unstable of the existing systems – the web publication system. However, our new system was also intended to simplify work with the print production. This has not yet happened, and I would say that a major reason is that we failed to deliver incremental releases of the product to the client. Had we been able to this, the client would probably have been able to get a better grip on how to integrate the new capabilities he had asked for into his workflow.
    - I would say that the Scrum process was very much a success factor, and in retrospect, people from the client company have said the same thing: that without the demonstrations of actual software every month, we would probably not have accomplished anything at all, due to the complexity of the workflows we tried to support. Even with the constant feedback, we still failed to reach one of our goals, but the learning that occurred during the project has, I think, been substantial, both as regards to how projects can be run, and as regards to the problem domain.
  - 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?
    - As mentioned above: this time I found it most suitable to not wave the Scrum flag too vividly. Instead, I tried to use terminology familiar with the client organization, while maintaining focus on the important concepts of self-organization, working with visible commitments and continuous feedback. There is merit to this way of working, I learned. A lot of people react negatively when a new methodology is mentioned – this could largely be avoided this time; even though many new concepts were introduced, we did it gradually, and the good habits kind of "sneaked up on us" as the group developed both itself and the software.
    - One difficult issue was that of organization. In this company, people were used to pretty well-defined roles (at least on paper). For example, a business analyst had been working on the problems related to this project long before I entered the scene. I noticed that having an analyst on the team was both beneficial and detrimental. On the good side, she possessed a lot of important knowledge. On the bad side, her presence tended to absolve the developers from the responsibility of understanding the problem domain themselves. What I did was to use a kind of soft-hard attitude here; soft in the sense that I let the team organize like they wanted, but hard in the sense that I required each team member participate in the interaction with the product owner during planning meetings and demonstrations.
    - Another difficulty was what I guess is a "classic" – that of handling part-time involvement in a project. I never managed to staff the team with "full-timers" only, so we had to rely on some help from people belonging to different functional groups from time to time. It was obvious how much more "expensive" the communication with these persons were. We were just too far away from each other, both spatially and as regards to the commitment to the project, to work together efficiently.
2. How do you cause the accuracy of Product Backlog estimates to improve? To what degree does their accuracy matter?
- I found that relating different backlog items to each other often worked wonders in trying to

get a grip of the complexity of different items. “Ok, so you say this would take 3 days to accomplish, I guess that means it’s about as complex as this one then? – No, no, it’s much harder, I guess I’d have to change my estimate to four days.”

- The accuracy of the backlog items really felt important only for the sake of 1) getting some kind of very very rough estimate of the size of the whole project effort (note, very very rough), and, more importantly 2) for the team to be able to understand what kind of commitments could be made at the start of an iteration.
- A nice result of having to split work into iterations is that the team has to learn *how to split things*, which means that they have to learn how to discuss work in terms of pieces that are valuable to the product owner. In other words, the time-box effectively helps you out of the problem were you end up “having to” do a whole lot of things in order to accomplish one tiny bit of functionality that the product owner wants to see.

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

- One: dialogue. People need to get together and learn how to cooperate in creating a short-term goal that all members of the team can commit to, and plan how this goal could be realized. Talk, talk, talk. Retrospectives: look back at what happened and ask what could have been different. That proved difficult indeed – for me as a ScrumMaster to ask what I perceived was perceived as difficult questions (!), and for us all to open up to each other. Once done talking – making sure we commit to actually improving the things we said needed improvement.
- Two: technical prowess. You need to *be able to* build a system in increments. I was so happy when a team member came to me and said: “You know Tobias, in the beginning, I thought you were nuts saying that we should build working software in one month. Nothing can be accomplished in a month. But now I’ve realized that it’s all about what we commit to do. We don’t have to do everything in a month - *we have to try our best to do what we commit to do.*”

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?

- I’m a little bit unsure of how to define metrics here, but let’s go with “everything used to measure progress”. Here’s what we used:
  - NUMBER ONE: A visible goal prioritization dialogue, with all parties involved. This leads to what I like to call “rapid negotiation”. Everybody’s gathered in the same room, suggestions and explanations shoot back and forth, and commitments for a known time period are made. At the end of the period: visible demonstration of whether goals were reached or not.
  - Sprint burndowns – very effective in giving the team some visual feedback on progress and chance of meeting commitments. Difficulty: we used an excel sheet, which tended to be a bit too “hidden” to be effective.
  - For some sprints later in the project, we exchanged the excel sprint burndown for a huge whiteboard, where we posted commitments. This was more visual, but still often not up to date. One stated reason was that since we were all in the same room, and such a small team, we knew where we were going anyway. I think that a more realistic reason for the charts not being updated was that people were simply not used to talking about progress much. I feel that this is often a result of people having been subjected to quite formal and rigorous reporting systems, which tend to be overwhelming, so the result is that people learn to avoid all kinds of reporting systems.
  - To express the backlog in a format recognizable for people in the organization, I tried to plot it on a simple timeline. I felt that some people, when shown the Excel sheet, simply thought that a spreadsheet representation of a project plan could not be a serious plan.
- In retrospect, here’s what I would have found useful, but that we never had:
  - A one-page report to management with an outline of the last iterations results, impediments and so on, like the one you (Ken) suggested when you were here with us in Uppsala. Not sure why I didn’t do this as a simple routine; it would not

have been much work (once a month).

- Some kind of developer test coverage measurements, to give us some kind of visible feedback on how well-exercised our code was
- As a complement to the value-driven prioritization: even better shared understanding of the value of the different features we worked on. Not sure today what form this would take.

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

- Personally, I would like to get better at facilitating everyday dialogues and encounters in an effective, non-threatening, manner. I guess this one is one of those life-long endeavors.
- Also, I plan to try to learn more about the different methods companies use to try to understand and calculate the value of their investments. This is so tightly connected to what iterative projects are about: on the financial side, I think iterative work in software development is all about handling investments wisely. A venture capitalist that once tutored us here at Citerus used the term “tube financing” to explain how he liked to handle investments in his companies: supply some funds, let people get to work using those funds, then inspect, and if it still seems wise, dispense some more funds “from the tube”. Quite a perfect analogy, I would say.
- Generally, I think the software community, or actually companies in general, would benefit most from a more pragmatic view on projects. Obviously, this is happening right now for us in software, which is excellent, but I think the same problems and misunderstandings plague other business as well, and for this which we call agile today to stick, the concepts need to spread to spread even more widely.

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

- I have no personal experience of an XP project, but I would think it wise to talk about at what rate XP’s engineering practices need to be adopted. To get the team started, my current recommendation would be to get a commitment for a first sprint and try live up to it, then reflect on what’s missing and gradually introduce the engineering practices as the project is rolling. As opposed to trying to engineer the “perfect environment” before committing to any “real work”.