Mature Agile with a twist of CMMI

Carsten Ruseng Jakobsen
Systematic Software Engineering
crj@systematic.dk

Kent Aaron Johnson
AgileDigm, Incorporated
kent.johnson@agiledigm.com

Abstract

Systematic is an agile company working at CMMI level 5, where the default way of working is based on Scrum and story based early testing development. Solid experiences in combining CMMI with Scrum and story based development, has shown that the mix provides strong synergies [2] and insights into what CMMI practices fit and amplify the execution of Scrum and story based early testing development

This paper presents specifically how agile methods like Scrum are successfully combined with CMMI. CMMI provides solid support for what disciplines to consider. When applied the disciplines create a focus on important aspects of agile methods that perhaps are not normally elaborated, for example how to ensure a proper quality of a product backlog or how to ensure a proper "production line" for the project. This guidance may not be needed for small agile projects, but as the agile movement continues to grow, and is used for larger and more complex projects, agile projects will need to address these issues related to increased size and complexity.

The experiences from combining CMMI and Scrum have led Systematic to identify examples of explicit guidance from CMMI that help to execute normal Scrum activities even better.

These activities can be implemented in the spirit of the agile manifesto and principles and by doing so agile methods can be augmented and matured to ensure that even larger and more complex projects in the future can and will benefit from agile - with a twist of CMMI.

1. Introduction

This paper presents the experiences on how CMMI amplifies Agile and recommends a subset of activities an agile project could adopt from CMMI to improve performance. Agile purists and small agile projects may find these activities non-agile or counter-productive; however in larger and/or distributed projects these activities will prove to be invaluable.

The paper does not describe how to mature an organization from CMMI level 1 to CMMI level 2, but the activities described could be part of such a change. Rather this paper highlights a set of activities that could be considered the glue between agile only projects based on Scrum and the disciplines expected from projects and organizations working toward CMMI level 2 or 3.

This paper presents how "a twist of CMMI" can help establish a more solid framework for agile projects to support even more complex projects by adopting some of the practices from the CMMI.

2. Context for the experiences

Systematic was established in 1985 and employs more than 450 people worldwide with offices in Denmark, Finland, USA and the UK. It is an independent software and systems company focusing on complex and critical IT solutions within information and communication systems. Often these systems are mission critical with high demands on reliability, safety, accuracy and usability.

Customers are typically professional IT-departments in public institutions and large companies with longstanding experience in acquiring complex software and systems. Solutions developed by Systematic are used by tens of thousands of people in the defense, healthcare, manufacturing, and service industries. Systematic was appraised 11 November 2005 using the SCAMPI[SM] method and found to be CMMI level 5 compliant. During 2006 Systematic adopted Scrum and a story based early testing approach to software development.

CMMI provides insight into what processes are needed to maintain a **disciplined** mature organization capable of predicting and improving performance of the organization and projects. Scrum provides guidance for efficient management of projects in a way that allows for high **flexibility** and **adaptability**. When mixing the two, a magic potion emerges, where the mindset from Scrum ensures that processes are implemented efficiently while embracing change, and CMMI ensures that all relevant processes are considered with proper discipline.

Individually CMMI and Scrum has proven benefits, but also pitfalls. An agile company may implement Scrum

[SM] Capability Maturity Model Integration, and SCAMPI are service marks of Carnegie Mellon University

correctly, but fail to obtain real benefits due to lack of consistent and sufficient execution of engineering or management processes. CMMI can help agile companies to institutionalize agile methods more consistently and understand what processes to address.

A company can comply with CMMI, but fail to reach optimal performance due to inadequate implementation of processes. Scrum and other agile methods can guide such companies towards more efficient implementation of CMMI process requirements.

Systematic has gained valuable experiences in combining Scrum and CMMI that are relevant both for projects in a CMMI and agile context.

3. Experiences from mixing

The first major experience from working with Scrum in a CMMI context is that CMMI embraces Scrum. CMMI has more practices and support for initial project planning and for final delivery and project closure. In Scrum terms, CMMI suggests activities before and after sprints are executed on the product backlog.

From a CMMI perspective, the initial Scrum product backlog is created during project planning and during project execution; sprints are executed and the product backlog is updated. This logically splits planning into two parts: overall CMMI project planning and detailed agile planning through Scrum.

This separation has led to overall planning where work is planned in sufficient detail as opposed to a complete decomposition. The overall planning produces a set of overall project plans and a Scrum product backlog where a complete list of prioritized features or work for the project is managed.

The primary change to project execution processes, was to integrate Scrum as the method for completing small iterations (sprints), on a selected subset of the work with highest priority.

Systematic experience indicates that this mix of CMMI, Scrum, and agile is beneficial, because

- CMMI planning can be considered a kind of disciplined sprint zero, where it is ensured that an optimal framework for the project is established, including a high quality product backlog, a production line definition, and well known targets and vision for the project as a whole.
- CMMI risk management proactively addresses possible impediments before they are encountered by the team.
- CMMI quality planning specifies more accurately and efficiently the quality targets of the project and helps developers to a better interpretation of completion criteria and sprint goals.
- CMMI will ensure that the project is tracked as a whole allowing the Scrum Team to concentrate on current sprint, knowing that they periodically are informed of overall project status.
- Scrum requires discipline regarding automatic test, a nightly build, and integration. CMMI supports this need for discipline and has led some projects at Systematic to monitor "fix-time after failed builds" for more than a year. The measure has proven to be cheap to establish, easy to understand, and therefore facilitating good habits.
- CMMI expects the project to seek objective measures of performance of the project's processes. In Scrum progress (of sprints) is primarily measured through the sprint burn down chart and the sprint review meeting. The project manager tracks the project as a whole based on selected measures within key areas like, product size, earned value, schedule, and quality. CMMI project planning provides good overall plans for the complete project where each completed sprint is very valuable input.
- CMMI ensures that agile methods are institutionalized, including
    - Consistent implementation throughout the organization and continuous improvement, e.g. Systematic Scrum Guidelines, story inspection checklist.
    - Role based training of all roles, e.g. Scrum Master and Product Owner.

When Systematic adopted Scrum, the roles Scrum Master, Product Owner, and Team were introduced. Most projects in Systematic have a person who communicates and understands the customer. In Systematic this person is appointed the role Product Owner. In our experience the Product Owner is often also the Project Manager, but could also be Software Architect, or User Experience Engineer. The Scrum Masters are in most cases equivalent with the team leader roles in Systematic. Projects in Systematic are staffed with people working full time on the project and the team is co-located.

3.1. How to establish a better initial product backlog

Experience. Project Planning in CMMI is a disciplined and comprehensive Sprint Zero.

In order to run a good Scrum, it is vital to have good product backlog. When Systematic adopted Scrum, the project planning process was updated to produce an initial product backlog. Expected CMMI practices include decomposition of work into manageable pieces that are estimated and analyzed for dependencies, planning of stakeholder involvement, and total project risk assessment.

In addition to the product backlog a set of overall project plans are established. Agile teams talk about a sprint zero to establish the foundation for the team to do efficient sprinting. Project Planning in CMMI can be perceived as a sprint zero to produce a coherent set of plans, that will help improve execution of the product backlog. Such plans cover topics like, stakeholder management, milestone and delivery schedules, cost estimates, and quality.

The initial version of these plans are typically established within few weeks after project initiation and will focus on the most certain elements of the projects plan, leaving more uncertain parts to be managed on the product backlog.

Figure 1 shows the main activities performed to establish the project plan. The sequence shown in the figure is advisory. The same activity may be performed multiple times, and often many activities are performed simultaneously. Each of the activities are supported by short step-wise descriptions of how the activity normally is performed, what inputs are required and what output are produced.

The descriptions are based on lessons learned and best practices from all projects within Systematic, and are organized with short outlined descriptions at the top, and detailed guidelines and templates at the bottom.

This provides the team with solid support for establishing the projects plans. The amount of time spent to establish project plans varies from project to project, but for most projects, initial project planning can be completed within weeks. One of the reasons for this is the handling of uncertain scope through Scrum combined with proper risk management from CMMI.

Concurrent to the initial planning, the overall solution architecture and product quality objectives are elaborated and documented by software architects and lead developers.

Before the project moves from planning (sprint zero) to project execution (sprinting) the project manager validates that the overall project objectives and plans are achievable and realistic, and that planning has reduced project risks sufficiently.

Many of the activities in figure 1 are in line with the intensions of Scrum, the main difference is that in CMMI these activities are elaborated and documented.

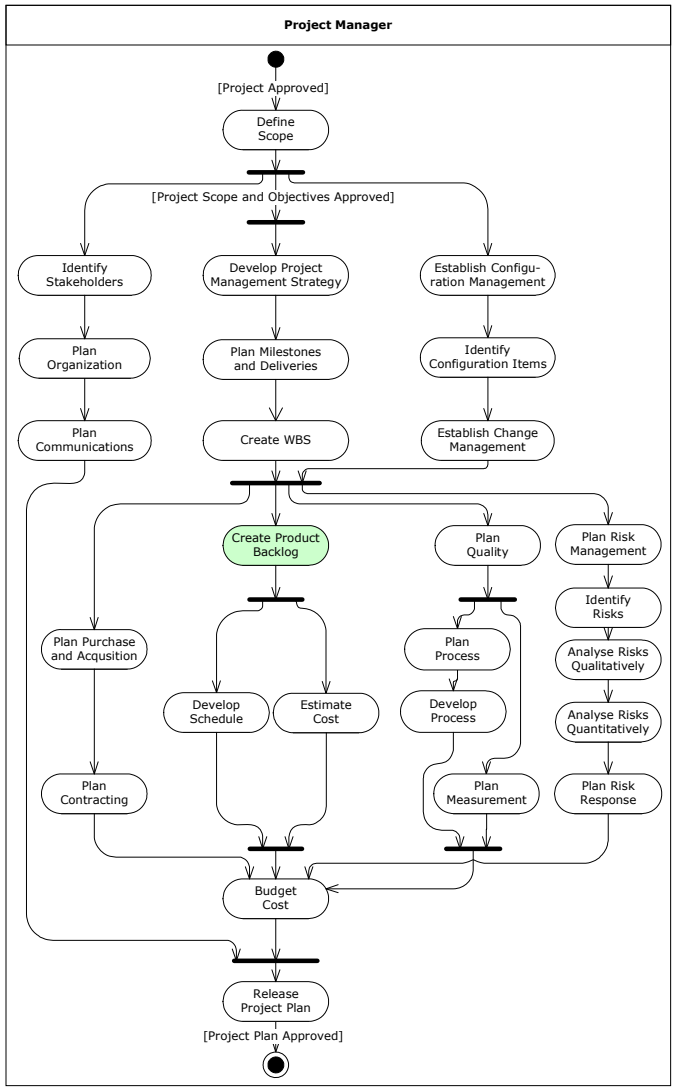


Figure 1 Overview of CMMI planning activities

Experience. Two levels of planning and tracking: Project as a whole and each sprint.

Sprinting on the product backlog is started, when the project plans and solution architecture are approved by senior management.

From this point tracking of the project is done at two different levels concurrently. The project manager tracks the projects overall plans and team(s) tracks progress of active sprint(s).

The plans established with planning activities provides a better context for defining sprint visions and goals, compared to projects only using a Sprint backlog. They also allow the team to focus on the sprint, because they can rely on the project manager tracking the overall progress.

3.2. Risks and impediments

Experience: Planning and risk management activities reduces risk of product backlog

The first draft product backlog is assessed for risk by the team. Asking the team to estimate the products backlog using 3-point estimates for effort will reveal the most uncertain parts of the work in the Product Backlog. Conducting Risk Identification meetings will identify other important risks, and allow proactive mitigation to be initiated.

Experience: Risk management can proactively prevent impediments

Scrum has a strong focus on removing impediments as soon as they are identified, however CMMI risk management activities focus on proactively identify some of these impediments as risks, and through mitigation eliminate them before they occur as an impediment in the future.

The distinction between risk and impediment is that risk describe a problem that may occur, whereas an impediment is problem that has occurred and is impacting planned progress.

Risk management activities are easily integrated with Scrum activities. During project planning the project plans and solution architecture are inputs for initial identification of risks. During project execution, bi-weekly meetings of 10-15 minutes are arranged, where the status of known risks is reported and new risks are identified.

It is our experience that these risk management meetings should be kept outside the daily scrum meeting. New risks may be reported on the daily scrum, in which case the risk manager will just take a note.

3.3. How to ensure high quality

Scrum is designed to produce high quality in terms of perceived and conceptual integrity where short iterations and sprint review customers are main drivers for high quality.

Experience: Explicit quality plans improves helps the team to build the right quality in

One of the results of planning is a quality assurance schedule (QAS), where it is outlined what quality activities will be used to ensure the quality objectives are achieved. The QAS may specify

- What stories are subject to inspection
- What code is subject to review
- What documents are subject to what types of review
- What unit test and automatic test is produced
- What is included in the acceptance test

A typical QAS document is only a few pages long, but the above descriptions can help a scrum team to elaborate and understand the definition of done.

Experiece: Use checklist to ensure quality of stories

One of the important aspects of Systematic story based development method was to ensure focus on early test.

Story Completion Checklist

Story:	
Feature:	
Developers:	
Inspected by:	

Activity	Work Product(s)	Completed	Inspected	# Defects
Story scope and estimate reconsidered		☐	☐	/
Development environment established		☐	☐	/
Requirements drafted		☐	☐	/
User interface drafted		☐	☐	/
Technical design drafted		☐	☐	/
User documentation drafted		☐	☐	/
Test design drafted		☐	☐	/
Requirements complete		☐	☐	__
Existing manual tests identified		☐	☐	/
Tests drafted (and coordinated with the Test Designer)		☐	☐	/
Code drafted		☐	☐	/
Manual tests complete		☐	☐	__
Automated tests complete		☐	☐	__
Test design complete		☐	☐	__
Code complete. (User interface inspected by UE)		☐	☐	__
Manual tests executed and passed		☐	☐	__
Technical design documentation complete		☐	☐	__
User interface documentation complete		☐	☐	__
User documentation complete		☐	☐	__
Story integrated (and integration tests passed)		☐	☐	__

Story complete		
Date	Developers	Inspector

- At start of story, cross out non-applicable activities and add extra activities.
- The "Work Product(s)" column is optional.
- Fill out checkboxes with ✓ or +. Do not pass a line until all activities before it are completed and inspected. During inspections, note number of defects found and mitigated in each completed activity.
- When all checkboxes are filled out with ✓or +, sign the checklist and give it to the Team Leader.

Figure 2 Story inspection checklist

The quality of stories are generally ensured by focus on early specification of test and by getting somebody else to look at the work done.

Developing a story includes many different activities, that need to be structured to some degree.

The Story Completion Checklist accomplishes these goals, by structuring activities and defining when work must be inspected by an inspector.

The activities in the checklist all have a short 5-10 line description in a procedure called "Execute story", that clarifies why and how the activity is performed.

The inspector role is often appointed to a lead developer, and this way the inspection also serves as an opportunity for knowledge sharing between experienced and less experienced developers.

The Story Completion Checklist ensures quality at the story level and makes it easier for the developer at the same time.

3.4. Test, integration, release and configurations management

Scrum promotes short iterations, e.g. one sprint per month, and this in turn drives the need for efficient configuration management, test, integration and release.

CMMI helps with:

- Establish standards for production line, including standard setups for build- and test servers
- Establish discipline on criteria for integration
- Measures to objectively evaluate performance
- Disciplines to maintain integrity of configuration management system, builds, and releases.

Experience: Automated test is a must in order to do one month sprints

When the sprint duration is one month, all tests must be automated to the extent possible. It is an integrated part of developing a story, to also implement the automated test verifying the story. Automated test are used on the teams shared repository and run every time a developer commits code to the shared build server.

Experience: Automated test and integration must be supported by a standard production line

Every project needs this infrastructure and therefore we have established standard production line setups, allowing projects to get started faster.

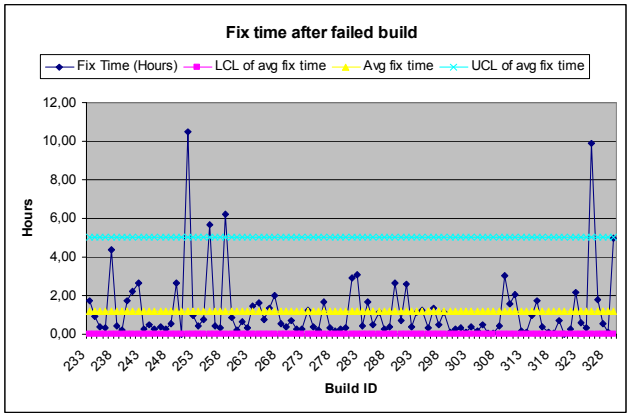
Experience: Continuous integration must be supported with discipline for check in

In order to avoid chaos when developers continuously integrate with each other, we have defined the following criteria for check in of code to the integration repository: The test must run smoothly in the developers sandbox and the code must comply with the code standard checked with FxCop (a static code analysis tool).

Experience: Focus on "fix-time after failed build" drives good discipline in project

Using standard infrastructure setups allows for efficient data collection and analysis. In particular Systematic has been inspired from Lean thoughts on flow and jidoka (stopping the production line when an error is detected).

We want our projects to be able to deliver on a daily basis, and hence that unresolved failed builds are fixed within a working day. We use CruiseControl (a build management tool) to signal all developers when a build fails. We also monitor the objective by analyzing data from the build servers using control charts like the one shown below.



Many projects have achieved this one work day objective, merely by the focus on the measure. The objective is easy to understand, and presenting the information in CruiseControl and control charts has established a good habit of fixing broken builds immediately when they fail.

Experience: Periodic audit of Configuration Management system builds good habits

As part of every sprint delivery a work product evaluation (WPE) is conducted and for every delivery to the customer a functional configuration audit (FCA) is conducted. The purpose is to ensure that the build product is correct and complete.

WPE and FCA are executed and documented by filling out check-lists that helps to ensure that

configuration management activities has been executed correctly for the build or release. Usually these activities can be accomplished within one or two hours.

The experience is that this checking of the configuration management system on a monthly basis, builds good habits on the team to remember configuration management disciplines, and as a result builds and releases are complete and correct, and may be re-created in the future should the need arise.

4. Agile with a twist of CMMI

In [2], [3] we described how the generic practices from CMMI can be used to institutionalize Scrum in your organization. In the following we present our recommendations to activities an agile project could consider adopting. These activities are inspired by the mandatory goals and expected practices from a subset of CMMI process areas.

We recommend the following activities to agile projects:

1. Establish your own sprint zero, and include activities in item 2-6 below in it.
2. Use Risk Management to proactively address risks before they are identified as impediments
3. Decompose requirements into features on the product backlog. Prepare the product backlog by decomposing the highest prioritized features to stories allowing for efficient sprint planning. (This defines what you are really going to do.)
4. Use 3-point effort estimates on elements of the product backlog during initial planning.
5. Analyze dependencies, stakeholders, risk on elements of product backlog.
6. Establish milestone and delivery plan and their initial relationship to product backlog.
7. Use Story Completion Checklist to maintain high quality of stories produced.
8. Decide and communicate quality objectives including, what code and documentation to formally review to elaborate definition of done.
9. Establish standards for project “production line” including development, build servers, and test servers.
10. Automate test and nightly build, and measure performance.
11. Establish criteria for committing of code to integration.
12. Maintain integrity of configuration management, by using a checklist for Work Product Evaluation and execute it by the end of each sprint.

5. Conclusion

This paper presented a few practical advices for agile projects on additional activities to adopt particularly in larger or distributed projects.

Our recommendation to the Agile community is to extend agile methods inspired from an understanding of the mandatory goals and expected practices for CMMI level 2 and 3. These practices make good sense, and you could argue that it has always inherently been expected as part of your agile method. In general the CMMI model provides a good understanding what practices to consider – but you will have to adopt it to your context, and find agile implementations for the practices.

When projects grow, we believe you need more discipline. We have described how a more disciplined sprint zero, risk management, and various checklists with minimal effort can bring you slightly more discipline into your project – and we believe that doing so will bring success to larger or distributed agile projects.

6. References

[1] M. Poppendieck and T. Poppendieck, *Lean Software Development: An Implementation Guide*: Addison-Wesley, 2006.

[2] J. Sutherland, C.R. Jakobsen and K.A. Johnson, "CMMI and Scrum - a magic potion for code warriors" in proceedings for Agile 2007

[3] M. K. Kulpa and K. A. Johnson, *Interpreting the CMMI: A Process Improvement Approach, Second Edition*. Boca Raton: Auerbach Publications, 2008