Scrum and CMMI Level 5: The Magic Potion for Code Warriors

Jeff Sutherland, Ph.D.
Patientkeeper Inc.
jeff.sutherland@computer.org

Carsten Ruseng Jakobsen
Systematic Software Engineering
crj@systematic.dk

Kent Johnson
AgileDigm Inc.
kent.johnson@agiledigm.com

Abstract

Projects combining agile methods with CMMI[1] are more successful in producing higher quality software that more effectively meets customer needs at a faster pace. Systematic Software Engineering works at CMMI level 5 and uses Lean product development as a driver for optimizing software processes. Valuable experience has been gained by combining Agile practices from Scrum with CMMI.

Early pilot projects at Systematic showed productivity on Scrum teams almost twice that of traditional teams. Other projects demonstrated a story based test driven approach to software development reduced defects found during final test by 40%.

We assert that Scrum and CMMI together bring a more powerful combination of adaptability and predictability to the marketplace than either one alone and suggest how other companies can combine them.

Introduction

One of the trends in the software industry is software projects are more comprehensive and complex while customers at the same time request faster delivery and more flexibility. Successful software development is challenged by the supplier's ability to manage complexity, technology innovation, and requirements change. Customers continually requests solutions faster, better and more cost-effective. Agile and CMMI methods both address these challenges but have very different approach and perspective in methods applied.

Management of complexity requires process discipline, and management of increased speed of change requires adaptability. CMMI primarily provides process discipline and Scrum enhances adaptability. This leads to the question, whether or not it is possible to integrate CMMI and agile practices like Scrum to achieve the benefits from both – or even more?

This paper provides an analysis of the effect of introducing Agile practices like Scrum and story based test driven software development and knowledge gained on what is required to be CMMI compliant, while running an Agile company.

[1] ® Capability Maturity Model, CMM and CMMI are registered in the U.S. Patent and Trademark Office

CMMI

The Capability Maturity Model (CMM) has existed since 1991, as a model based on best practices for software development. It describes an evolutionary method for improving an organization from one that is ad hoc and immature to one that is disciplined and mature [71]. The CMM is internationally recognized and was developed by the Software Engineering Institute at Carnegie Mellon University, Pittsburgh, USA.

In 2002, a new and significantly extended version called CMMI was announced, where the 'I' stands for 'Integration' [72]. This model integrates software engineering, systems engineering disciplines, and software acquisition practices into one maturity model. CMMI defines 25 process areas to implement. For each process area required goals, expected practices and recommended sub-practices are defined. In addition a set of generic practices must be applied for all processes.

The past 15 years of experience with CMM and CMMI, demonstrates that organizations appraised to higher levels of CMM or CMMI improve the ability to deliver on schedule, cost, and agreed quality. Increasingly, the industry requires suppliers to be appraised to CMM or CMMI level 3 or higher [73]. A number of governmental organizations worldwide, have established CMMI maturity requirements. Recently the Danish Minister of Science proposed regulations to require public organizations to request documentation of their supplier's maturity [74].

Scrum

Scrum for software development teams began at Easel Corporation in 1993 [21] and emerged as a formal method at OOPSLA'95 [22]. A development process was needed to support enterprise teams where visualization of design immediately generated working code. Fundamental problems inherent in software development influenced the introduction of Scrum:

- Uncertainty is inherent and inevitable in software development processes and products - Ziv's Uncertainty Principle [53]
- For a new software system the requirements will not be completely known until after the users have used it - Humphrey's Requirements Uncertainty Principle [58]
- It is not possible to completely specify an interactive system – Wegner's Lemma [54]
- Ambiguous and changing requirements, combined with evolving tools and technologies make implementation strategies unpredictable.

"All-at-Once" models of software development uniquely fit object-oriented implementation of software and help resolve these challenges. They assume the creation of software involves simultaneous work on requirements, analysis, design, coding, and testing, then delivering the entire system all at once [31].

Sutherland and Schwaber, co-creators of Scrum joined forces with creators of other Agile processes in 2001 to write the Agile Manifesto [57]. A common focus on working

software, team interactions, customer collaboration, and adapting to change were agreed upon as central principles essential to optimizing software productivity and quality.

CMMI and Agile methods

Soon after publication of the Agile Manifesto in 2001, Mark Paulk principal contributor and editor of Capability Maturity Model Version 1.0 [71], observed that Agile practices are intended to maximize the benefits of good practice [75, 76]. *"The SW-CMM tells what to do in general terms, but does not say how to do it; agile methodologies provide a set of best practices that contain fairly specific how-to information – an implementation model – for a particular kind of environment."* However, Paulk noted that aligning the implementation of agile methods with the interests of the customer and other stakeholders in a government contracting environment for software acquisition might be an impossible task, where high customer interaction is difficult.
Surdu [77] and McMahon [78] reported positive experiences in 2006 using agile processes on government contracts while noting the need for process discipline, good system engineering practices, and development of self-motivated teams. Collaboration with customers was achieved through agile education and negotiation. These studies provide practical confirmation of Paulk's analysis of the applicability of agile practices in a CMM environment.

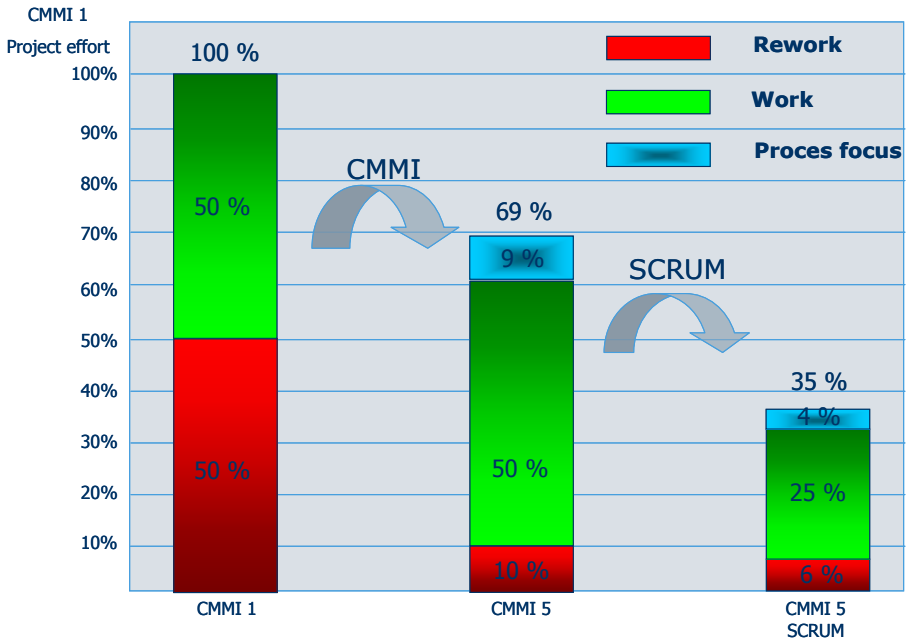
Paulk [76] points out that *"When rationally implemented in an appropriate environment, agile methodologies address many CMM level 2 and level 3 practices."* Similarly Kane and Ornburn present a mapping of Scrum and XP to CMMI [79] demonstrating that a majority of the CMMI process areas related to Project Management can be addressed with Scrum and the majority of process areas related to software engineering can be addressed with XP. CMMI expects that processes are optimized and perhaps replaced over time and states: *"Optimizing processes that are agile and innovative depends on the participation of an empowered workforce aligned with the business values and objectives of the organization."* [72] (page 49).

We agree with the authors above that Agile methodologies advocate good engineering practices that can be integrated in the CMMI framework, and consider the largest drawback of most Agile methodologies is a limited concept of institution-wide deployment. Institutionalization is key to implementation of all processes in CMMI, and is strongly supported by a set of Generic Practices. It is our belief that these practices could be used to ensure that Agile methodologies are institutionalized in any organization.

Agile methods like Scrum and XP are practical methods that can support different parts of CMMI. Combining Scrum and CMMI practices can produce a more powerful result than either alone and can be done in way where CMMI compliance is maintained. A more detailed analysis of a full implementation of the Scrum development process along with some XP engineering practices used at Systematic shows quantitative results of introducing good agile practices and how to maintain CMMI compliance in an Agile company.

Scrum and CMMI: a magic potion

Systematic was established in 1985 and employs 371 people worldwide with offices in Denmark, 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[SM2] method and found to be CMMI level 5 compliant. Working at CMMI level 5 brings many advantages. Systematic has first hand experience of reduction in rework by 38% to 42% over earlier levels, estimation precision deviation less than 10%, and 92% of all milestones delivered early or on time. At the same time, extra work on projects has been significantly reduced.
More importantly, Systematic has transformed over twenty years of experience into a unified set of processes used by all software projects. Historical data are systematically collected and analyzed to continuously provide insight into the capability and performance of the organization.
The use of a shared common process makes it easier for people to move from project to project and share experiences and lessons learned between projects. Insight into the capability and performance of processes makes it possible to evaluate performance of new processes to performance of existing processes. And this forms the foundation for continuous improvement.



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

Figure 1: CMMI and Scrum Productivity Gains

In short, Systematic is able to deliver what the customer has ordered on schedule, cost and quality using 69% effort compared to a CMMI Level 1 company [80, 81]. This benefit comes at the minimal cost of 9% process focus in project management and engineering. CMMI Level 5 is increasingly a requirement from customers and key to obtaining large contracts, especially within defence and healthcare. Customers recognize that CMMI Level 5 gives high predictability and better-engineered product for scalability, maintainability, adaptability, and reliability.
Early results indicate that when CMMI traditional processes are optimized using Scrum, the productivity for large projects is doubled and the amount of rework is reduced an additional 40% over that of CMMI Level 5 companies. It is important to note that the optimized process is a mixed process, using traditional CMMI processes to establish a project baseline expressed as a product backlog combined with Scrum as the preferred way to implement the project in iterations of one month Sprints. The combination of the CMMI and Scrum into the optimized CMMI Scrum process includes the proper activities to establish sufficient planning needed by both customer and supplier, and at the same time the flexibility and adaptability provided by Scrum. This combined process is treated similarly to any other process in CMMI.

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.

Individually CMMI and Scrum has proven benefits but also pitfalls. An Agile company may implement Scrum correctly but fail due to lack of institutionalization, (see section 0) or inconsistent or insufficient 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 methodologies can guide such companies towards more efficient implementation of CMMI process requirements.

How Systematic adopted Scrum

Here we describe the generic steps of the process Systematic executed that resulted in the adoption of Scrum, early testing, and story based development.

Identify Business Objectives and Needs. *CMMI states [72] (page 55) that "successful process-improvement initiatives must be driven by the business objectives of the*

organization". Business objectives and needs are addressed by the strategy of the organization.

Systematic made a strategic decision to use Lean as the dominant paradigm for future improvements. Lean has demonstrated notable results for many years in domains such as auto manufacturing, and due to it's popularity, has been adapted to other domains, including product and software development. It was expected that adoption of a Lean mindset would facilitate a more efficient implementation of CMMI.

The strategic decision to use Lean as a dominant tool for optimization of processes, is input to CMMI Organizational Process Focus (OPF) and driven by an organizational shared function for process improvements.

Analysis. *Different Lean dialects were evaluated and Lean Software Development [8] was identified as the dialect most relevant to Systematic. Lean Software Development is an agile toolkit. A careful interpretation of the Agile Manifesto shows that this is not necessarily in conflict with CMMI Level 5.*

The Agile Manifesto recognizes that processes, tools, comprehensive documentation, contract negotiation and following a plan have value, but emphasizes people, interactions, working software, customer collaboration and responding to change to have more value. *"The agile methodology movement is not anti-methodology; in fact, many of us want to restore credibility to the word. We also want to restore a balance: We embrace modeling, but not merely to file some diagram in a dusty corporate repository. We embrace documentation, but not to waste reams of paper in never-maintained and rarely used tomes. We plan, but recognize the limits of planning in a turbulent environment." [57]*

Successful application of Lean Software Development - an agile toolkit, depends on the adoption of an agile mindset to supplement the process focus. Systematic values are consistent with the Agile Manifesto, and special focus was placed on the following aspects for new improvements:

Individuals and interactions. Empowerment: the person executing the process is also responsible for updating the process.

Working software over documentation. Critical evaluation of what parts of the documentation or process can be removed or refined to increase the customers perceived value of the activities is essential.

Responding to change. Determining how the process could be improved to support reduced cycle time drove customer value.

Lean competences were established, through handout of books, formal and informal training, and walk-the-talk activities. Project Managers were trained in Lean Software Development, and Mary Poppendieck [8] visited Systematic for a management seminar on Lean Software Development.

This seminar established an understanding of the Agile and Lean mindset. Based on this training, causal dependencies between the principles and tools in Lean Software Development were analyzed, and as a result test practices and reduced cycle time were identified as good candidates for initial activities. They represented a good starting point for implementing Lean and at the same time focused on processes where improvements would have significant effect on efficiency.

Pilot. *Lean advocates that the people performing a process should be responsible and empowered to maintain that process. In the introduction to the CMMI OPF process area CMMI says the same thing.*

An analysis of the causal dependencies in Lean Software Development led to the decision to seek improvements based on the principles of Amplify Learning, Deliver Fast, and Build Integrity In.

Selected projects were asked if they would like to pilot improved processes related to test and reduced cycle time respectively. Project staff were trained in the Lean mindset and asked to suggest how to adopt Lean into their processes. The result was a selection of Scrum and early testing based on story-based development.

The result of the pilots were two-fold: it confirmed the general idea of using Lean mindset as source for identification of new improvements, and secondly it provided two specific successful improvements showing how agile methods can be adopted while maintaining CMMI compliance.

Implementation. *It was decided to adopt Scrum and story based software development in the organization. Process Action Teams (PATs) were formed to integrate the experience and knowledge gained from the pilots, into the processes shared by all projects in the organization. The PATs were staffed with people that would be expected to execute the new process when released.*

The largest change to project planning is that features and work are planned in sufficient detail as opposed to a complete initial detailed analysis. The result is a Scrum Product Backlog with a complete prioritized list of features/work for the project. All features have a qualified estimate, established with a documented process and through the use of historical data, but the granularity of the features increase as the priority falls. The uncertainty that remains is handled through risk management activities.

The primary change to project execution processes, was to integrate Scrum as method for completing small iterations (Sprints), on a selected subset of the work with highest priority.
This work verified that Scrum could be adopted in project management while maintaining CMMI compliance. This is important because, one of the first steps to embrace change is to ensure that project management processes support and allow agility. In addition the people executing the process were trained [redacted]

by Jeff Sutherland [82], who also did a management seminar on Scrum at Systematic. Concurrent to the above pilots, Lean was considered by all projects and shared functions as one of several ways to identify possible improvements.

Result. *The first step for Systematic in adapting a Lean mindset resulted in the adoption of Scrum and story based development as the recommended way of working. Systematic provides a default implementation of a Projects Defined Process (PDP) known as PDP Common. The PDP Common has been updated to integrate Scrum and story based development into the relevant processes.*

The apparent result of adopting agile methods into existing CMMI compliant processes, has led to integration of processes or process areas that initially were implemented separately. The new processes are more efficient, and the changes have improved quality, customer and employee satisfaction.

Risk. *Some of the risks of applying Agile mindset to a CMMI organization include*:

- Degrading CMMI compliant processes to non-compliance.
- Local optimizations increasing project efficiency at the expense of inefficiency at the organizational level, e.g. due to lack of organizational coordination.

These risks were handled by a central process team that kept the organization on track with respect to the risks and change management. The process team was responsible for:

- Build and share competencies on Lean, Agile and Scrum with the organization.
- Define and communicate vision, constraints and measures for adoption of a Lean mindset.
- Encourage and empower different parts of the organization to challenge current process implementations with a Lean mindset, in search of improvement opportunities.
- Collect experiences from the organization and consolidate improvements at the organizational level.

The dominant risk for failure in adapting Lean is insufficient understanding or adoption of Lean or Agile mindset. Systematic has addressed this risk by inviting Jeff Sutherland and Mary Poppendieck to Systematic to establish a good understanding of Lean, Scrum and Agile.

Systematic experience from pilots

In a period of approximately 4 months, two small projects piloted Scrum and early testing in story based development.

Scrum. *The first pilot was initiated on a request for proposal, where Systematic inspired by Lean principles suggested a delivery plan with bi-weekly deliveries and stated explicit expectations to customer involvement and feedback.*

One of the main reasons that Systematic was awarded the contract was the commitment to deliver working code bi-weekly and thereby providing a very transparent process to the customer. During project execution, a high communication bandwidth was kept between the team, the customer and users. This was identified as one of the main reasons for achieving high customer satisfaction.

The delivery plan and customer involvement resulted in early detection of technological issues. Had a traditional approach been used these issues would have been identified much later with negative impacts on cost and schedule performance.

However, productivity of this small project was at the expected level compared to the productivity performance baseline for small projects. Another small project using Scrum shows a similar productivity and the same indications on high quality and customer satisfaction.

At Systematic, productivity for a project is defined as the total number of lines of code produced divided by the total project effort spent in hours. These numbers are gathered from the configuration and version control system. Data are attributed with information related to programming language, type of code: new, reuse or test. This definition of productivity has been chosen because it provides sufficient insight and is very simple and efficient to collect.
Systematic has established and maintains a productivity performance baseline (PPB) for productivity compared to project size estimated in hours, from data collected on completed projects [83]. The data shows that productivity is high on small projects and declines with the size of the project with traditional CMMI Level 5. The productivity performance baseline in Systematic is divided into two groups: small projects less than 4000 hours and large projects above 4000 hours. Productivity of small projects is 181% the productivity of large projects.
When comparing the projects using Scrum to the current productivity baseline it is seen that productivity for small projects is insignificantly changed, but the productivity for large projects shows 201% increase in productivity. As mentioned above, the large projects did additional improvements, and it is therefore not possible to attribute the benefit solely to Scrum. However the people involved all agree that Scrum was a significant part of this improvement.

There is a strong indication that large projects in Systematic using Scrum will double productivity going forward. Small projects in Systematic already show a high productivity. We believe that this is because small projects in Systematic always have been managed in a way similar to Scrum. However quality and customer satisfaction seems to be improved and we believe this is because Scrum has facilitated a better understanding of how small projects are managed efficiently.

Early testing. *Two different approaches were used to facilitate early test. One large project decided to use a story based approach to software development and another project decided to focus on comprehensive testing during development.*

The idea of story-based development was to subdivide features of work, typically estimated to hundreds of hours of work into smaller stories of 20-40 hours of work. The implementation of a story followed a new procedure, where the first activity would be to decide how the story could be tested before any code was written. This test could then be used as the exit criteria for implementation of the story.

In order to ensure that the new procedure was followed, the procedure included a few checkpoints where an inspector would inspect the work produced, and decide whether or not the developer could proceed to the next activity in the procedure. These inspections are lightweight, and could typically be done in less than 5 minutes.

Many benefits from story-based development were immediately apparent. The combination of a good definition of when a story was complete, and early incremental testing of the features, provided a very precise overview of status and progress for both team and other stakeholders.
Developing a series of small stories rather than parts of a big feature is more satisfactory, and creates a better focus on completing a feature until it fulfills all the criteria for being "done".
This project finished early, and reduced the number of coding defects in final test by 38% compared to previous processes.

The project using comprehensive testing placed test specialists together with the developers. As in the story based approach, this caused discussion and reflection between testers, developers, user experience engineers and software architects, before or very early in the development of new functionality. As a consequence the amount of remaining coding defects in final test were reduced by 42%.
Based on these two projects test activities should be an integrated activity through out the projects lifetime. Scrum inherently supports this, through cross-functional teams and frequent deliveries to the customer.

Real needs. *A customer sent a request for proposal on a fixed set of requirements. When Systematic responded, we expressed our concern that the scope and contents expressed in the requirements were beyond the customer's real needs.*

Systematic decided to openly share the internal estimation of the requirements with the customer, for the purpose of narrowing scope by removing requirements not needed or too expensive compared to the customers budget. The customer agreed to re-evaluate the requirement specification, and the result was that requirements and price were reduced by 50%.

This experience supports results in a Standish Group Study reported at XP2002 by chairman Jim Johnson, showing that 64% of features in a fixed price contract are never or rarely used by end-users.

We believe that this illustrates how important it is to have a high communication bandwidth with the customer, in order to find out what the real needs are. Success is not

achieved by doing the largest project, but by doing the project that provides the most value for the customer, leaving time for software developers to work with other customers with real needs. It gives motivation to developers to provide solutions to real need, which in turn benefits dedication and productivity.
Even though this experience is related to activities before the project is started, the challenge of maintaining close communication with the customer, to ensure that the project delivers the most value within the agreed constraints, continues and is strongly supported by Scrum.

Guide for mixing CMMI and Agile

The previous section has described how Systematic, an organization appraised to CMMI Level 5, has adopted agile methods. This section presents our advice to the agile organizations on how to adopt the concept of institutionalization.

How CMMI can improve Agile

Our focus is on using CMMI to help an organization institutionalize Agile Methods. We have all heard Agile Methods described by some as just another disguise for undisciplined hacking and of some individuals who claim to be Agile just because they "don't document." We believe the value from Agile Methods can only be obtained through disciplined use. CMMI has a concept of *Institutionalization* that can help establish this needed discipline.

Institutionalization is defined in CMMI as "the ingrained way of doing business that an organization follows routinely as part of its corporate culture." Others have described institutionalization as simply "this is the way we do things around here." Note that institutionalization is an organizational-level concept that supports multiple projects.

CMMI supports institutionalization through the Generic Practices (GP) associated with all process areas. For the purposes of our discussion, we will look at the 12 generic practices associated with maturity levels 2 and 3 in the CMMI [72] pp. 39-44 and how they might help an organization use Agile Methods. We have paraphrased the generic practices (**shown in bold text below**) to match our recommended usage with Agile Methods. In CMMI terms, the projects in an organization would be *expected* to perform an activity that accomplished each of these generic practices. We have used Scrum as the example Agile Method to describe some of the activities that relate to these practices.

Establish and maintain an organizational policy for planning and performing Agile Methods (GP 2.1). The first step toward institutionalization of Agile Methods is to establish how and when they will be used in the organization. An organization might determine that Agile Methods will be used on all projects or some subset of projects based on size, type of product, technology, or other factors. This policy is a way to clearly communicate the organization's intent regarding Agile Methods. In keeping with the Agile Principle of face-to-face conversions at "all hands meeting" or a visit by a senior manager during a project's kick off could be used to communicate the policy.

Establish and maintain the plan for performing Agile Methods (GP2.2). This practice can help ensure that Agile Methods do not degrade into undisciplined hacking. The

expectation is that Agile Methods are planned and that a defined process exists and is followed. The defined process should include a sequence of steps capturing the minimum essential information needed to describe what a project really does. The plan would also capture the essential aspects of how the other 10 generic practices are to be implemented in the project. In Scrum, some of this planning is likely to be captured in a product backlog and/or sprint backlog, most likely within a tool as opposed to a document.

Provide adequate resources for performing Agile Methods (GP2.3). Every project wants, needs, and expects competent professionals, adequate funding, and appropriate facilities and tools. Implementing an activity to explicitly manage these wants and needs has proved useful. In Scrum, for example, these needs may be reviewed and addressed at the Sprint Planning Meeting and reconsidered when significant changes occur.

Assign responsibility and authority for performing Agile Methods (GP 2.4). For a project to be successful, clear responsibility and authority need to be defined. Usually this includes a combination of role descriptions and assignments. The definitions of these roles identify a level of responsibility and authority. For example, a Scrum Project would assign an individual or individuals to the roles of Product Owner, Scrum Master, and Team. Furthermore, the roles in the Team are likely to include a mix of domain experts, system engineers, software engineers, architects, programmers, analysts, QA experts, testers, UI designers, etc. Expertise in the Team is likely to include a mix of domain experts, system engineers, software engineers, architects, programmers, analysts, QA experts, testers, UI designers, etc. Scrum assigns the team as a whole the responsibility for delivering working software. The Product Owner is responsible for specifying and prioritizing the work. The Scrum Master is responsible for assuring the Scrum process is followed. Management is responsible for providing the right expertise to the team.

Train the people performing Agile Methods (GP 2.5). The right training can increase the performance of competent professionals and supports introducing new methods into an organization. People need to receive consistent training in the Agile Method being used in order to ensure institutionalization. This practice includes determining the individuals to train, defining the exact training to provide, and performing the needed training. Training can be provided using many different approaches, including programmed instruction, formalized on-the-job training, mentoring, and formal and classroom training. It is important that a mechanism be defined to ensure that training has occurred and is beneficial.

Place designated work products under appropriate level of configuration management (GP 2.6). The purpose of a project is to produce a deliverable product (or products). This product is often a collection of a number of intermediate or supporting work products (code, manuals, software systems, build files, etc.). Each of these work products has a value and often goes through a series of steps that increase their value. The concept of configuration management is intended to protect these valuable work products by defining the level of control, for example, version control or baseline control and perhaps multiple levels of baseline control to use within the project.

Identify and involve the relevant stakeholders as planned (GP 2.7). Involving the customer as a relevant stakeholder is a strength of Agile Methods. This practice further identifies the need to ensure that the expected level of stakeholder involvement occurs. For example, if the project depends on customer feedback with each increment, build, or sprint, and involvement falls short of expectations it is then necessary to communicate to the appropriate level, individual, or group in the organization to allow for corrective action. This is because corrective action may be beyond the scope of the project team. In advanced Scrum implementations, this is often formalized as a MetaScrum [40] where stakeholders serve as a board of directors for the Product Owner.

Monitor and control Agile Methods against the plan and take appropriate corrective action (GP 2.8). This practice involves measuring actual performance against the project's plan and taking corrective action. The direct day-to-day monitoring is a strong feature of the Daily Scrum Meeting. Further, examples of this can be seen in Scrum with the use of the Product Burndown Chart showing how much work is left to do at the beginning of each Sprint and the Sprint Burndown Chart showing the total task hours remaining per day. Scrum enhances the effectiveness of the plan by allowing the Product Owner to inspect and adapt to maximize ROI, rather than merely assuring plan accuracy.

Objectively evaluate adherence to the Agile Methods and address noncompliance (GP2.9). This practice is based on having someone not directly responsible for managing or performing project activities evaluate the actual activities of the project. Some organizations implement this practice as both an assurance activity and coaching activity. The coaching concept matches many Agile Methods. The Scrum Master has primary responsibility for adherence to Scrum practices, tracking progress, removing impediments, resolving personnel problems, and is usually not engaged in implementation of project tasks. The Product Owner has primary responsibility for assuring software meets requirements and is high quality.

Review the activities, status, and results of the Agile Methods with higher-level management and resolve issues (GP2.10). The purpose of this practice is to ensure that higher-level management has appropriate visibility into the project activities. Different managers have different needs for information. Agile Methods have a high level of interaction, for example, Scrum has a Sprint Planning Meeting, Daily Scrum Meetings, a Sprint Review Meeting, and a Sprint Retrospective Meeting. Management needs are supported by transparency of status data produced by the Scrum Burndown Chart. This, in combination with defect data can be used to produce a customized management dashboard for project status. Management responsibilities are to (1) provide strategic vision, business strategy, and resources, (2) remove impediments surfaced by Scrum teams that the teams cannot remove themselves, (3) ensure growth and career path of staff, and (4) challenge the Scrum teams to move beyond mediocrity. The list of impediments generated by the Scrum teams is transparent to management and it is their responsibility to assure they are removed in order to improve organizational performance.

Establish and maintain the description of Agile Methods (GP 3.1). This practice is a refinement of GP2.2 above. The only real difference is that description of Agile Methods in this practice is expected to be organization-wide and not unique to a project. The result is that variability in how Agile Methods are performed would be reduced across the organization; and therefore more exchange between projects of people, tools, information and products can be supported.

Collect the results from using Agile Methods to support future use and improvement the organization's approach to Agile Methods (GP 3.2). This practice supports the goal of learning across projects by collecting the results from individual projects. The Scrum Sprint Retrospective Meeting could be used as the mechanism for this practice.

All of these generic practices have been useful in organizations implementing other processes. We have seen that a number of these generic practices have at least partial support in Scrum or other Agile Methods. We believe that implementing these practices can help establish needed discipline to any Agile Method.

Critiques of CMM

In research funded by the Danish government, Rose et. al. surveyed the literature on critiques of CMM [84]. They observed that the chief criticism of CMM is not the process itself, but the effects of focus on process orientation. While side effects of process focus may be viewed as simply poor CMM implementation, organizations with heavyweight processes are highly prone to poor execution.

As with any other model, good and bad implementations of CMM exist. We believe that bad implementations are one of the main reasons for the existence of many negative criticisms of CMM. Such implementations are often characterized as in the table below, whereas many good CMM implementations address most of the criticism.

One way to enhance chances for a good CMM or CMMI implementation is to use Scrum. Applying Scrum and agile mindset while implementing CMMI will help to recognize that CMMI addresses people and technology aspects, in a way fully supportive of an agile mindset.
More importantly, the work in this paper has shown that the mix of CMMI and Scrum blends a magic potion for software development that is even better than the sum of the two alone.
We acknowledge that the CMM criticism listed in the table below exist, but from our knowledge of CMMI we consider it to be incorrect. But a bad implementation of CMMI may be perceived this way. Even though good CMMI implementations can be done without agile methods, the table shows that Scrum will contribute with a beneficial focus on issues stemming from "bad" CMMI implementation.

CMM criticism	**Scrum support**

CMM reveres process but ignores people.	Scrum is the first development process to treat people issues the same as other project management issues [85].
Does not focus on underlying organizational problems that should be solved.	A primary responsibility of the Scrum Master is to maintain and resolve an impediment list that contains organizational issues, personal issues, and technical problems.
Ignores quality in the software product assuming an unproven link between quality in the process and quality in the resulting product. Differing project and organizational circumstances may mean that a process that delivers a good product in one context delivers a poor product in another context.	The Scrum Product Owner is responsible for continuously reprioritizing the Product Backlog to maximize business value in current context.
Lack of business orientation	The primary focus of Scrum is on delivering business value.
Poor awareness of organizational context.	Creation and prioritization of features, tasks, and impediments is always done in organizational context by inspected and adapting.
Ignores technical and organizational infrastructures.	Daily inspection and adaptation in Scrum meetings focuses on technical and organizational issues.
Encourages an internal efficiency focus and thus market and competition blindness.	Focus is on delivering business value. Type C Scrum allows an entire company to dominate a market segment through inspecting and adapting in real time to competition [40].

Conclusions

This paper shows that CMMI and Scrum can be successfully mixed. The mix results in significantly improved performance while maintaining compliance to CMMI Level 5 as compared to performance with either CMMI or Scrum alone.

Scrum pilot projects showed significant gains in productivity and quality over traditional methods. These results led to an ROI based decision to more widely introduce Scrum and consider other Agile practices in Systematic. Scrum now reduces every category of work (defects, rework, total work required, and process overhead) by almost 50%.

This paper has outlined how Systematic adopted Scrum and story based development into its CMMI processes inspired from a strategic focus on Lean. For Agile companies the article has presented how Generic Practices from CMMI can be used to institutionalize

agile practices. Furthermore the article has presented Lean Software Development [8] as an operational tool to identify improvement opportunities in a CMMI 5 company.

We think companies in defense, aerospace, and other industries that require high maturity of processes, should carefully consider introducing Agile practices into the workplace and all software companies should consider introducing CMMI practices into their environment.
Our recommendation to the Agile community is to use the CMMI generic practices from CMMI Level 3 to amplify the benefits from Agile methods. The efficiencies of agile practice can lower the cost of CMMI process improvements making the benefits more widely available. Our recommendation to the CMMI community is that Agile methods can fit into your CMMI framework and will provide exciting improvements to your organization.