Innovations Syst Softw Eng (2008) 4:17–29
DOI 10.1007/s11334-007-0040-1

ORIGINAL PAPER

Blending Scrum practices and CMMI project management process areas

Ana Sofia C. Marçal · Bruno Celso C. de Freitas · Felipe S. Furtado Soares · Maria Elizabeth S. Furtado · Teresa M. Maciel · Arnaldo D. Belchior

Received: 13 December 2007 / Accepted: 19 December 2007 / Published online: 11 January 2008

Abstract Software development organizations that have been employing capability maturity models, such as SW-CMM or CMMI for improving their processes are now increasingly interested in the possibility of adopting agile development methods. In the context of project management, what can we say about Scrum's alignment with CMMI? The aim of our paper is to present the mapping between CMMI and the agile method Scrum, showing major gaps between them and identifying how organizations are adopting complementary practices in their projects to make these two approaches more compliant. This is useful for organizations that have a plan-driven process based on the CMMI model and are planning to improve the agility of processes or to help organizations to define a new project management framework based on both CMMI and Scrum practices.

Keywords Scrum · Agile development methods · CMMI · Agile project management

1 Motivation

According to Software Engineering Institute (SEI), during the last several years, organizations have been increasingly motivated to adopt quality models focused on maturity of software process, such as the capability maturity model for software (SW-CMM) and capability maturity model integration (CMMI) [23]. One of the possible reasons for this motivation is related to the fact these organizations have discovered that improvements in software quality are broadly associated with adequacy and adherence of their processes to the highest levels of these models, providing benefits related to project performance, quality of products and services as well as an increase in client satisfaction [3].

As Boehm [7] stated, so far, this decade has seen a continuation in the trend toward rapid application development and an acceleration of the pace of change in information technology, in organizations, in competitive countermeasures, and also in the environment. This rapid change of pace has caused increasing frustration to the heavyweight plans, specifications, and other documentation imposed by contractual inertia and maturity model compliance criteria. He continues saying that the late 1990s saw the emergence of a number of agile methods such as Adaptive Software Development, Crystal, Dynamic Systems Development, eXtreme Programming (XP), Feature Driven Development, and Scrum. All of these methods employ agile principles, such as iterative cycles, early delivery of working software and simplicity as defined in Agile Manifesto [5] published in 2001.

The Manifesto for Agile Software Development brought changes in the software development community, generating

A. D. Belchior "in memorian".

A. S. C. Marçal · M. E. S. Furtado · A. D. Belchior
Department of Applied Computer Science, University of Fortaleza, Av. Washington Soares 1321, Fortaleza, CE 60811-341, Brazil

M. E. S. Furtado
e-mail: elizabet@unifor.br

A. S. C. Marçal (✉) · B. C. C. de Freitas · F. S. F. Soares · T. M. Maciel
C.E.S.A.R - Recife Center of Advanced Systems and Studies, Rua Bione, no. 220, Cais do Apolo 50.0303-90, Recife, PE, Brazil
e-mail: ana.sofia@cesar.org.br

B. C. C. de Freitas
e-mail: bruno.freitas@cesar.org.br

F. S. F. Soares
e-mail: felipe.furtado@cesar.org.br

T. M. Maciel
e-mail: teresa@cesar.org.br

contradictory opinions and discussions in several segments of manufacturing, civil and aerospace construction as well as in project management. The Manifesto essentially defines a new focus on the software development based on agility, flexibility, communication abilities and capacity for offering new products and services with high value to the marketplace in shorter periods of time [15].

From the point of view of project management, APM (agile project management) comes together with the Agile Manifesto and represents a set of values, principles and practices, which help the project team to deliver high-value products and services in a challenged environment [15]. The main values of APM approach fulfils two purposes: the necessity to build agile and flexible products and to build agile and flexible teams.

In this context, Scrum [1] has attracted significant attention amongst software practitioners during last five years. Whereas the Extreme Programming method, which has been widely accepted as one of the most important agile approaches, has a definite programming flavor, Scrum concentrates on managing software projects. Scrum was developed by Ken Schwaber and Jeff Sutherland in 1996. They considered that software development is complex and unpredictable, being applicable to volatile environments. Scrum includes monitoring and feedback activities, in general, as well as daily and quick meetings with everyone of the project team. It aims at identifying and planning corrective actions to possible issues or impediments to the development process [24].

SW-CMM or CMMI and other agile methods have been compared in several studies, including practice mappings and comparisons between both approaches [6,19,20,25]. For example, Paulk [20] suggests for example that XP's use of stories, on site customer and continuous integration fulfill the SW-CMM requirement management goals. On the other hand, Turner and Jain [25] found in their study that several of the CMMI components and agile methods were in conflict, most of them being those addressing organizational processes. Pikkarainen and Mäntyniemi [21] conclude that many of them were also found to be supportive or neutral to each other, especially those focusing on project management.

In the project management context, what can we say about adopt Scrum and CMMI together? Can they co-exist? How agile project management used with Scrum is compliant with the CMMI goals and practices?

This paper presents an analysis of the CMMI and Scrum practices identifying how Scrum practices address the project management process areas of CMMI, presenting major gaps between them. Additionally, it approaches how organizations are adopting complementary practices in their projects to turn these two approaches more compliant. This is useful for organizations that have their plan-driven process based on the CMMI model and are planning to improve their process toward agility or to help organizations to define a new project management framework based on both CMMI and Scrum practices. It is important to point out that we will not describe a proposal for the extension of Scrum here, but this proposal is already underdevelopment and was briefly presented on [18].

The paper is organized as follows: Sect. 2 presents the background overview of CMMI and the Scrum. Section 3 focuses on describing the methodology used to do the mapping between CMMI project management process areas and Scrum practices, showing the rating, gaps and the strengths between them and the results from the overall mapping. Section 4 presents a study case showing how the organizations has been adopted agile practices in their projects, mainly about estimates, risk management and issues management. The last section concludes the paper with final remarks.

2 Background overview

2.1 Project management and the Capability Maturity Model Integration

According to SEI, CMMI (capability maturity model integration) is a process improvement maturity model for the development of products and services. It consists of best practices that address development and maintenance activities that cover the product lifecycle from conception through delivery and maintenance [23].

CMMI is available in two representations: staged or continuous. Each representation organizes process areas differently. These two representations are really just different views of the same content. A staged representation may be said to focus on the organization's processes as a whole, to provide a road map for process improvement with proven predefined groupings of process areas, and to provide an easy migration path from the SW-CMM. A continuous representation may be said to focus on improvement to individual process areas chosen to align with specific organizational needs and to provide an easy migration path from Electronic Industries Alliance Interim Standard (EIA/IS) 731 [17].

CMMI model components are grouped into three categories, reflecting how to interpret them: required (specific and generic goals), expected (specific and generic practices), and informative [23]. Sub-practices, typical work products, amplifications, generic practice elaborations, goal and practice titles, goal and practice notes, and references are examples of informative model components.

CMMI for Development (CMMI–DEV), Version 1.2, describes 22 process areas [23]. A process area is a group of related activities performed collectively to achieve a set of goals. In the context of these models, processes refer to "what to do" rather than "how to do it". A process area specifies goals that describe the result of successful application and

Table 1 CMMI project management process areas

Level	Process area
Level 2	Project planning (PP)
	Project monitoring and control (PMC)
	Supplier agreement management (SAM)
Level 3	Integrated project management + IPPD (IPM + IPPD)
	Risk management (RSKM)
Level 4	Quantitative project management (QPM)

practices that describe required (and expected) activities to achieve those goals. Some goals and practices are specific to the process area; others are generic and apply across all process areas.

Process areas can be grouped into four categories: process management, project management, engineering and support. The project management category, focus of this paper, encompasses management activities, like planning, monitoring and control. Table 1 illustrates this category' process areas, which are organized by level following a staged representation.

2.2 Scrum

Ken Schwaber first described Scrum in 1996 [1] as a process that accepts that the development process is unpredictable, formalizing the "do what it takes" mentality, and has found success with numerous independent software vendors. The term is borrowed from rugby: "[A] scrum occurs when players from each team huddle closely together... in an attempt to advance down the playing field" [14].

According to Schwaber [24], Scrum starts with the premise that software development is too complex and unpredictable to be planned exactly in advance. Instead, empirical process control must be applied to ensure visibility, inspection, and adaptation. The different environmental and technical variables (such as time frame, quality, requirements, resources, implementation technologies and tools, and even development methods) must be controlled constantly in order to be able to adapt to changes flexibly. This is achieved through an iterative and incremental development process.

Scrum implements an iterative, incremental skeleton through three roles [24]:

- The product owner represents the interests of everyone with a stake in the project and its resulting system. Furthermore he maintains the product backlog, i.e., a prioritized list of project requirements with estimated times to turn them into completed product functionality.
- The team is responsible for developing functionality. Teams are self-managing, self-organizing, and cross-functional, and they are responsible for figuring out how to turn product backlog into an increment of functionality within one iteration and managing their own work to do so. Team members are collectively responsible for the success of each iteration and of the project as a whole.
- The Scrum master is responsible for managing the Scrum process, i.e., for teaching Scrum to everyone involved in the project, for implementing Scrum so that it fits within an organization's culture and still delivers the expected benefits, and for ensuring that everyone follows Scrum rules and practices.

A detailed Scrum flow is shown in Fig. 1. According to Schwaber [24], a Scrum-based project starts from a high-level vision of the system to be developed. After that, product backlog is created containing a list of known requirements. So, the items of the product backlog are prioritized and divided into small time-boxed iterations (called sprints).

Every task in Scrum is carried out through sprints. A sprint is a 30-day period of development time. Schwaber [24] explains that each sprint is initiated with a sprint planning meeting, where the product owner and team get together to collaborate about what will be done for the next sprint. Selecting from the highest priority product backlog, the product owner tells the team what is desired, and the team tells the product owner how much of what is desired can be turned into functionality over the next sprint. In the first sprints, most of architecture and infra-structure work are done, so less functionalities are released.

After deciding what has to be done in the next sprint, the team develops the sprint backlog, i.e., a list of tasks that must be performed to deliver a completed increment of potentially shippable product functionality by the end of the sprint. The sub-tasks in the list emerge as the sprint evolves and should be divided so that each takes roughly 4–16 h to finish.

During the execution of each sprint, the team meets daily in 15-min meetings to track work progress and schedule other meetings, if necessary. At the daily Scrum, each team member answers three questions: What have you done on this project since the last daily Scrum meeting? What will you do before the next meeting? Do you have any obstacles?

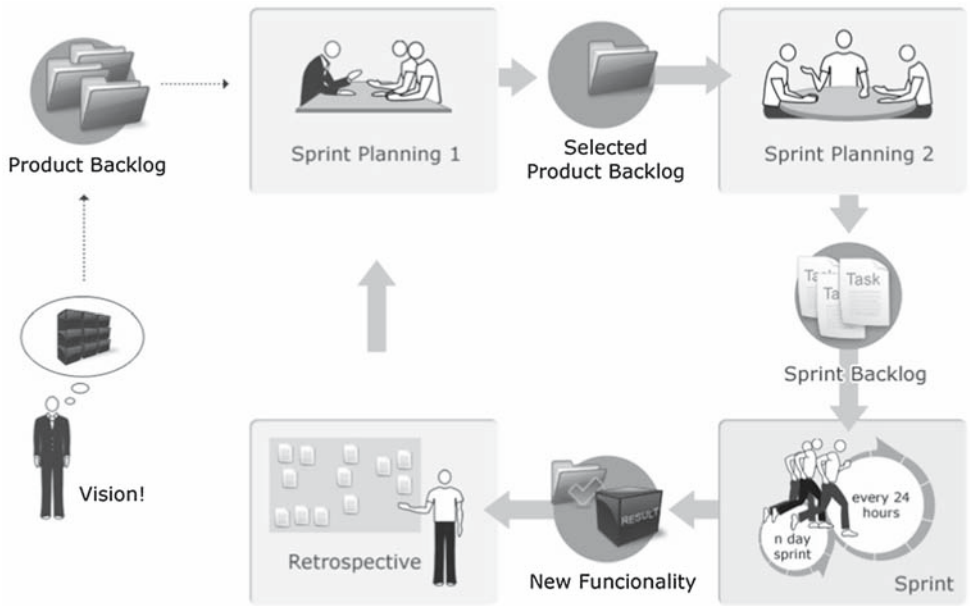


Fig. 1 Scrum process overview [13]

Ate the end of each sprint, the team presents its results to the stakeholders. This presentation will guide the inspection of developed functionalities and possible adjustments to the project.

At the end of the sprint, a sprint review meeting is held at which the team presents what was developed during the sprint to the product owner and any stakeholders who wish to attend. After the sprint review and prior to the next sprint planning meeting, the Scrum master also holds a sprint retrospective meeting in order to encourage the team to revise, within the Scrum process framework, its development process to make it more effective and enjoyable for the next sprint.

Schwaber [24] concludes saying that together, the sprint planning meeting, the Daily Scrum, the sprint review, and the sprint retrospective constitute the empirical inspection and adaptation practices of Scrum.

In Scrum, the monitoring of project progress is carried out through two main graphs: *Product Burndown* and *sprint Burndown*. These graphs show, in a timeline, how much work remains to be done, providing an excellent mechanism to view correlations between quantity of work to be done and the how much work the project team can "burn".

3 Scrum practices versus CMMI project management areas

The mapping between CMMI process areas and Scrum practices considers the staged representation of CMMI–DEV model, version 1.2, released in August 2006 [23]. The assessment only considers the project management process areas as this was its focus.

For each process area, a mapping between its specific practices and the Scrum practices was carried out. Several considerations were identified in order to establish this mapping, identifying gaps and strengths. After that, a coverage rating for each practice was established considering the following criteria in Table 2.

After the rating phase, a coverage percentage for each process area was calculated based on the total quantity of specific practices. Afterwards, the results were grouped and a complete view of the CMMI project management process areas coverage by Scrum practices was generated. Each mapping and the general results are described in next sub-sections.

3.1 Mapping the process area project planning

According to CMMI–DEV [23], the purpose of project planning (PP) is to establish and maintain plans that define project activities. PP has three specific goals (SG 1—establish estimates, SG 2—develop a project plan, SG 3—obtain commitment to the plan), enclosing 14 specific practices. The mapping of all specific practices related to this process area is presented below.

SP 1.1 Estimate the scope of the project

Its purpose is to establish a top-level work breakdown structure (WBS) to estimate the scope of the project. In Scrum, the initial definition of the project's scope occurs during the pregame planning phase, when the stakeholders can contribute to product backlog creation. In this case, the WBS is composed of the product backlog and the set of all pre-defined sprints, providing the necessary resources to estimate the project's scope. Detailed estimates are carried out at the beginning of

Table 2 Criteria for Scrum practices rating

Rating		Criteria
U	Unsatisfied	The practice is not addressed by Scrum
PS	Partially satisfied	There is some evidence of the practice being addressed by Scrum; however, the practice is not fully addressed.
S	Satisfied	The practice is fully addressed.

each sprint, in the second part of the sprint planning meeting. So, this practice is *satisfied*.

SP 1.2 Establish estimates of work product and task attributes

Its purpose is to define what is necessary to establish and maintain estimates of the attributes of the work products and tasks. There are no explicit orientations in Scrum to establish, for instance, size and/or complexity of items of product backlog and sprint backlog. Beyond that, no method is explicitly mentioned to guide the estimates, such as wideband delphi [26], function point analysis [12], use case points [4] or story points [10]. In this way, this practice is *unsatisfied*.

SP 1.3 Define project lifecycle

Its purpose is to define a project lifecycle on which to scope the planning effort. This practice is fully addressed by Scrum because it defines a lifecycle as showed by Larman [16], composed of four phases:

- Planning establishes a project vision and the stakeholders' expectations, beyond assuring funding/budgeting for project execution.
- Staging identifies and prioritizes the requirements (at least, for the next sprint). Break the product backlog in sprints, according to previous prioritization, considering the team productivity.
- Development implements the system in a set of 30-day iterations (sprints), when, at the end of each sprint, a product increment is presented to the stakeholders.
- Release covers system deployment.

So, this practice is *satisfied*.

SP 1.4 Determine estimates of effort and cost

Its purpose is to estimate the project effort and cost for the work products and tasks based on estimation rationale. In Scrum, estimates are carried out on two levels: product backlog and sprint backlog. Product backlog estimates are high level estimates, so less accurate, providing a visibility of each requirement size (effort). sprint backlog estimates are more accurate than the first ones. Team estimation is calculated by performance on previous sprints, capacity for the forthcoming sprint and the relative complexity of the tasks required to deliver the sprint goal [9]. However, these effort estimates don't follow a formal method nor are they derived from size or complexity as required by the CMMI model. Scrum doesn't mention the importance of using a historical base. Cost is not explicitly mentioned in Scrum, just effort, but it is necessary for the product owner to calculate the project's budget and funding. In this way, this practice is classified *partially satisfied*.

SP 2.1 Establish the budget and schedule

Its purpose is to establish and maintain the project's budget and schedule. In Scrum, the project's budget and schedule are obtained from the product backlog and directly derived from the estimated effort. Product backlog is prioritized and subdivided in sprints, considering team allocation and its workload. The schedule is composed of a set of 30-day sprints. On the other hand, Scrum doesn't provide orientations about establishing budget. Considering this gap, this practice was rated *partially satisfied*.

SP 2.2 Identify project risks

Its purpose is to identify and analyze project risks. Scrum considers a risk as a possible impediment for the project. The risks identification occurs in an iterative way, during daily meetings and registered on white-boards, flip charts or impediments list. But, this risk identification doesn't occur in a systematic and parameterized manner, using, for instance, risk categories and sources. So, this practice is *partially satisfied*.

SP 2.3 Plan for data management

Its purpose is to plan for the management of project data. The practices and rules defined in Scrum contribute for good communication and promote collaboration between team and stakeholders, beyond providing visibility of project progress. According to Schwaber [24], any data generated by the project must be stored in a public folder, available to everyone. Much project information is communicated through meetings or documents. However, there is no formal procedure to collect, consolidate and publish this information and data.

Data privacy is another weakness, so this practice is *unsatisfied*.

SP 2.4 Plan for project resources

This practice plans the necessary resources for the project to run. In Scrum, team allocation and infrastructure availability are carried out at the beginning of the project, during the *staging* phase [2]. In the product backlog are included necessary resources to development, such as machines, tools and other necessary investments for configuring the project's work environment. During its execution, the Scrum master is responsible for providing new resources when the actual resources are not enough or new impediments related to insufficient resources are reported in the daily meetings. This practice is *satisfied*.

SP 2.5 Plan for needed knowledge and skills

This practice focuses on the planning for knowledge and skills needed to perform the project. In Scrum, teams are multi-functional groups, self-managed, and made up of seven skilled people implementing the sprint backlog items. The team is composed of analysts, designers, QA, developers, data administrators, and architects amongst others. Senior members must mentor, monitor and guide other members. The project team should, if possible, be made up with high skilled people (considering technical and business skills), able to implement the sprint backlog. Otherwise, training and mentoring can be included in the product backlog [2]. So, this practice is rated *satisfied*.

SP 2.6 Plan stakeholder involvement

Its purpose is to involve identified stakeholders. Scrum defines how stakeholders will be involved during the project's execution. This involvement is monitored by the Scrum master and registered in a communication plan. In this way, this practice is considered *satisfied*.

SP 2.7 Establish the project plan

Its purpose is to establish and maintain the overall project plan content. According to Schwaber [24], the minimum plan necessary to start a Scrum project consists of a vision and a product backlog. The vision describes why the project is being undertaken and what the desired end state is. The product backlog defines the functional and non-functional requirements that the system should meet to deliver the vision, prioritized and estimated. Vision document and product backlog create a base for elaborating a high-level project plan. So, this practice is rated *satisfied*.

SP 3.1 Review plans that affect the project

Its purpose is to review all plans that affect the project to understand project commitments. In Scrum, plans are revised at the beginning of each sprint and possible adaptations are carried out in accordance with change requirements and technologies. The CMMI model doesn't explain which plans need to be revised, such as the QA plan, the CM plan nor the test plan amongst others. Therefore, this practice is considered *satisfied*.

SP 3.2 Reconcile work and resource levels

Its purpose is to reconcile the project plan to reflect available and estimated resources. This practice is *satisfied*, because this work reconciliation occurs during the sprint planning meeting. The team, product owner and Scrum master define the functionalities to be developed in the sprint.

SP 3.3 Obtain plan commitment

This practice addresses the commitment from relevant stakeholders responsible for performing and supporting plan execution. The plan commitment occurs continuously at the beginning of each sprint, during the sprint planning meeting. The product owner, the Scrum master and the team define the priorities of product backlog for each sprint and which items are to be developed in the next sprint. During the sprint execution, if the team workload is not enough to develop all the agreed items, the product owner can decide which sprint backlog item could be removed. On the other hand, if the team workload is greater than the effort necessary to implement the sprint backlog items, product owner can allocate other items from product backlog. Therefore, this practice is *satisfied*.

The general rating for this process area is shown in Fig. 2.

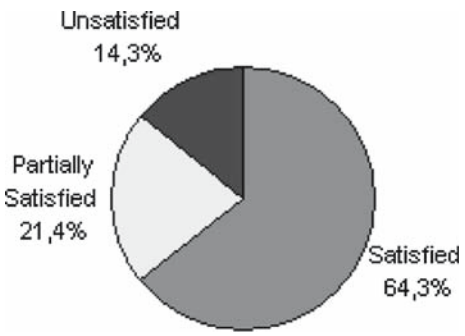


Fig. 2 General coverage for project planning process area

3.2 Mapping the process area project monitoring and control

The purpose of project monitoring and control (PMC) is to provide an understanding of the project's progress so that appropriate corrective action is taken when the project's performance deviates significantly from the plan [23]. PMC encloses ten specific practices grouped in two specific goals (SG 1—monitor project against plans—and SG 2—manage corrective action to closure). The mapping of all specific practices related to this process area is presented below.

SP 1.1 Monitor project planning parameters

Its purpose is to monitor the actual values of the project planning parameters against the project plan. In Scrum, project tracking occurs through burndown graphs and project meetings previously mentioned. Product burndown shows the speed of releasing product backlog items by the team. It is analyzed at the end of each sprint. It helps to monitor the planning of functionalities releasing, providing visibility if the sprint goal will be successfully achieved or if re-planning the sprint's scope is required to achieve the planned date. Sprint burndown daily shows the speed of the team and the progress of its activities in a sprint. Sprint burndown provides a supporting tool for planning necessary corrective actions.

Tracking meetings, mainly daily meetings, allows a day-by-day tracking of the team's progress and assesses the current difficulties of carrying out the planned activities. These difficulties must be quickly overcome by the Scrum master so that the team does not lose its focus or sprint goal.

In spite of that, like cost, size and effort estimates are not carried out in a systematic manner, there is not a formal tracking of them as required by the CMMI model. So, training tracking is also informal, due to the fact that there is no formal planning. In this way, this practice is rated *partially satisfied*.

SP 1.2 Monitor commitments

Its purpose is to monitor commitments against those identified in the project plan. In Scrum, commitments of each sprint are established during the sprint planning meeting and monitored through sprint burndown and daily meetings and, finally, reviewed in the sprint retrospective meeting. During a sprint, the team cannot receive any additional work from stakeholders or product owner. Just the team itself can update sprint backlog in order to maintain uninterrupted focus on sprint activities. So, this practice is rated *satisfied*.

SP 1.3 Monitor project risks

Its purpose is to monitor risks against those identified in the project plan. In Scrum, the daily meeting can help to identify impediments (issues, dependencies, risks etc), so risks are identified but they are not analyzed properly. Risks are registered on white boards, flip charts or impediment lists and monitored by the Scrum master, so they are tracked in an informal way. In other words, this practice is considered *partially satisfied*.

SP 1.4 Monitor data management

Its purpose is to monitor the management of project data against the project plan. This practice is *unsatisfied* because Scrum does not adopt any procedure for planning and tracking data management, as required by CMMI model.

SP 1.5 Monitor stakeholder involvement

Its purpose is to monitor stakeholder involvement against the project plan. In Scrum, stakeholders' involvement tracking is carried out during project meetings by the Scrum master, who is responsible for assuring that all stakeholders understand and respect the rules and practices defined in Scrum. In spite of there not being any register of this monitoring, this practice is *satisfied* because it is possible to find indirect evidence such as impediment list updated, product backlog and sprint backlog updated amongst others.

SP 1.6 Conduct progress reviews

Its purpose is periodically reviewing the project's progress, performance, and issues. In Scrum, the project control is carried out by frequent inspections and progress review meetings (the daily meeting and sprint review meeting). In this way, this practice is fully *satisfied*.

SP 1.7 Conduct milestone reviews

Its purpose is to review the accomplishments and results of the project at selected project milestones. As commented in SP 1.6, milestone reviews occur at the end of each sprint. In sprint review meetings, the project progress is inspected, providing visibility of the accomplishment of commitments. Therefore, this practice is considered *satisfied*.

SP 2.1 Analyze issues

Its purpose is to collect and analyze the issues and determine the corrective actions necessary to address the issues. During the Daily Scrum meetings, the team reports all impediments against expected quality or performance levels. Impediments are registered on a white board, flip chart or impediment list and they are erased when overcome. Beyond that, the Scrum master is in charge of resolving the impediments as soon

Fig. 3 General coverage for project monitoring and control process area

as possible, taking appropriate corrective actions. So, this practice is *satisfied*.

SP 2.2 Take corrective action

Its purpose is to take corrective action on identified issues. As mentioned before, corrective actions are taken for the impediments found. However, there is not any register of how these actions are planned and monitored. So, this practice is *partially satisfied*.

SP 2.3 Manage corrective action

Its purpose is to manage corrective actions to closure. As mentioned before, impediments registered on the white board, flip chart or impediment list are erased when resolved. So, all corrective actions are monitored to closure. However, the results of these actions are not analyzed to determine its effectiveness. Therefore, this practice is *partially satisfied*.

The general rating for this process area is shown in Fig. 3.

3.3 Mapping the process area supplier agreement management

The purpose of supplier agreement management (SAM) is to manage the acquisition of products from suppliers [23]. In Scrum, there is not any practice addressing the acquisition of products and product components that are delivered to the project's customer. So, all of its specific practices are *unsatisfied*.

3.4 Mapping the process area integrated project management + IPPD

According to CMMI–DEV [23], the purpose of integrated project management (IPM) is to establish and manage the project and the involvement of the relevant stakeholders according to an integrated and defined process that is tailored from the organization's set of standard processes. For integrated product and process development (IPPD), IPM + IPPD also cover the establishment of a shared vision for the project and the establishment of integrated teams that will carry out objectives of the project.

IPM + IPPD are composed of three specific goals (SG 1 use the project's defined process, SG 2 coordinate and collaborate with relevant stakeholders and SG 3 apply IPPD principles) enclosing 14 specific practices.

In this process area, all of the six specific practices related to SG1 (the project is conducted using a defined process that is tailored from the organization's set of standard processes) are assessed *unsatisfied*. Because Scrum does not define a set of organizational standard processes, it just establishes a set of practices and rules defined for the project. In other words, the project's defined process is not derived from a set of organizational processes.

The mapping of other specific practices related to this process area is presented below:

SP 2.1 Manage stakeholder involvement

Its purpose is to manage the involvement of the relevant stakeholders in the project. As commented before, in PMC SP 1.5, Scrum practices and rules implicitly define how stakeholders will be involved in the project tracking. This involvement is monitored by the Scrum master. So, in this way, this practice is *satisfied*.

SP 2.2 Manage dependencies

Its purpose is to participate with relevant stakeholders to identify, negotiate, and track critical dependencies. In Scrum, dependencies and risks can be managed as impediments, being identified through Scrum daily meetings. The Scrum master is in charge of resolving any identified problem as soon as possible, including dependencies. However, there are no registers of negotiation, meeting minutes or agreed dates to remove such dependencies. Beyond that, there is no planning of tracking strategies or verification of the dependencies. Therefore, this practice is *partially satisfied*.

SP 2.3 Resolve coordination issues

Its purpose is to resolve issues with relevant stakeholders. This practice is *partially satisfied*, for the same reasons presented for SP 2.2.

SP 3.1 Establish the project's shared vision

Its purpose is to establish and maintain a shared vision for the project. The Scrum planning process sets the stakeholders' expectations. The plan is a way of synchronizing stakeholders' expectations with the team's expectations [24]. It is also important that the whole team understands the essence

of what the project or product is trying to achieve. This is where the project vision comes in. It should be as short and as accessible as possible but communicates the substance and character of the undertaking [9]. Therefore, this practice is *satisfied*.

SP 3.2 Establish the integrated team structure

Its purpose is to establish and maintain the integrated team structure for the project. In Scrum, when several teams work in a collaborative environment, this project is called scaled project and the mechanisms used to coordinate their activities are called scaling mechanisms [24]. When scaling Scrum to larger projects some guidelines must be followed:

- Attempt to keep the team size to eight.
- Do not engage all the teams until the infrastructure is built. This may require the first few sprints being done by only a single team and primarily non-functional requirements being built.
- Divide the work into teams that logically represent the work and minimize the need for multiple assignments of people.
- Create a daily Scrum meeting of representatives from each Scrum, held after the daily Scrum.

So, these guides help to establish an integrated team structure and this practice is *satisfied*.

SP 3.3 Allocate requirements to integrated teams

Its purpose is to allocate requirements, responsibilities, tasks, and interfaces to teams in the integrated team structure. In Scrum, some practices are critical for the scaled project success, such as [24]:

- Build a scalable infra-structure before scaling the project. This structure must support just one team initially and grows during successive sprints. Non functional requirements to build the scaling infra-structure must to be added to the product backlog and prioritized jointly with other business functionalities in the Scrum staging phase, before the first sprint.
- Always release business value in the sprints carried out for building the infrastructure
- Optimize the initial team capabilities and prepare additional teams. Additional teams must be composed of, at least, one member of the initial team, playing an infra-structure and architect expert role.

All of these practices establish the necessary requirements of integrating teams. So, this practice is *satisfied*.

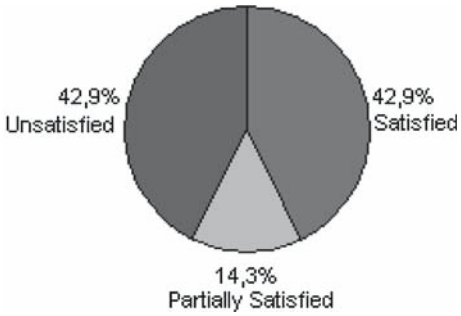


Fig. 4 General coverage for IPM + IPPD process area

SP 3.4 Establish integrated teams

Its purpose is to establish and maintain integrated teams in the structure. This practice is *satisfied*, for the same reasons presented for SP 3.2 e SP3.3.

SP 3.5 Ensure collaboration among interfacing teams

This practice ensures collaboration among interfacing teams. This practice is *satisfied*, due to the same reasons presented for SP 3.2 e SP3.3.

The general rating for this process area is shown in Fig. 4.

3.5 Mapping the process area risk management

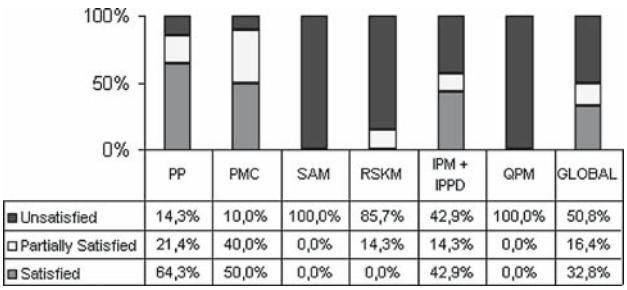
The purpose of risk management (RSKM) is to identify potential problems before they occur so that risk-handling activities can be planned and invoked as needed across the life of the product or project to mitigate adverse impacts on achieving objectives [23]. As previously mentioned, in Scrum, risks are identified, but it does not mention practices to define sources, parameters or categories to analyze and control the risk management effort. In Scrum, there are no strategies for risk response or a mitigation plan for the critical risks based on historical bases or similar. In this way, the assessment, categorization and prioritization of these risks occur in an informal manner. Therefore, all of the specific practices of RSKM are *unsatisfied*, except SP 2.1 identify risks, because it is *partially satisfied*.

The general rating for this process area is shown in Fig. 5.

3.6 Mapping the process area quantitative project management

The purpose of quantitative project management (QPM) is to quantitatively manage the project's defined process to achieve the project's established quality and process-performance objectives [23]. According to CMMI–DEV, in this process area, the quality and process-performance objectives, measures, and baselines identified are developed as described in

Fig. 5 General coverage for risk management process area



	PP	PMC	SAM	RSKM	IPM + IPPD	QPM	GLOBAL
Unsatisfied	14,3%	10,0%	100,0%	85,7%	42,9%	100,0%	50,8%
Partially Satisfied	21,4%	40,0%	0,0%	14,3%	14,3%	0,0%	16,4%
Satisfied	64,3%	50,0%	0,0%	0,0%	42,9%	0,0%	32,8%

Fig. 6 CMMI project management process areas covered by Scrum

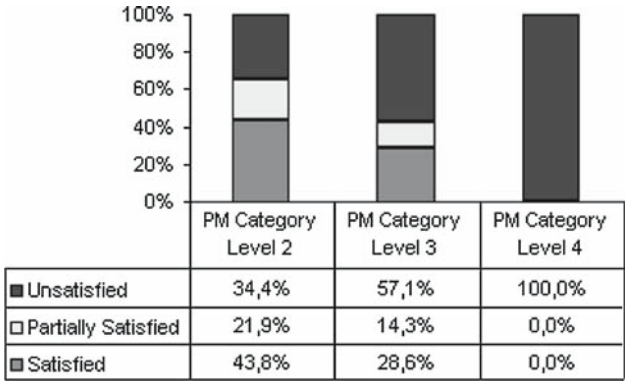
the organizational process performance process area. Subsequently, the results of performing the processes associated with the quantitative project management process area (e.g., measurement definitions and measurement data) become part of the organizational process assets referred to in the organizational process performance process area. In Scrum, there is not any practice addressing this process area. Therefore, all of its practices are *unsatisfied*.

3.7 Overall results from the mapping

General results of the mapping are shown in Fig. 6, a consolidated view of the coverage of CMMI project management process areas by Scrum practices.

This result shows that 32.8% of specific practices of CMMI project management process areas are satisfied, 16,4% are partially satisfied and 50.8% are unsatisfied. In other words, Scrum is not fully complaint with CMMI project management process areas, mainly related to SAM, RSKM and QPM process areas. If SAM and QPM are not considered, 44.5% of the specific practices of CMMI project management process areas are satisfied, 22.2% are partially satisfied and 33.3% are unsatisfied.

Considering each CMMI project management process area according to its maturity level, another analysis can be made as shown in Fig. 7.



	PM Category Level 2	PM Category Level 3	PM Category Level 4
Unsatisfied	34,4%	57,1%	100,0%
Partially Satisfied	21,9%	14,3%	0,0%
Satisfied	43,8%	28,6%	0,0%

Fig. 7 CMMI project management process areas covered by Scrum. rating grouped by maturity level, considering only the project management process area category

In this way, process areas related to maturity level 2 (PP, PMC and SAM) have 43.8% of its specific practices satisfied by Scrum, 21.9% are partially satisfied and 34.4% are unsatisfied. If SAM is not applicable to the organization context, Scrum becomes more attractive (58.3% satisfied, 29.2% partially satisfied and 12.5% unsatisfied).

But things don't sound so good when considering process areas related to maturity level 3 (IPM + IPPD and RSKM). These process areas have 28.6% of its specific practices satisfied by Scrum, 14.3% are partially satisfied and 57.1% are unsatisfied, due to a weak adherence of Scrum to RSKM practices, lack of organizational process and systematic use of historical bases, as required by IPM process area. Finally, considering process areas related to maturity level 4, these specific practices are unsatisfied by Scrum.

We can realize that the major gaps between Scrum and PP, PMC, IPM+IPPD and RSKM process areas are mainly concentrated on:

- Lack of alternative practices or techniques to estimate project's efforts or costs. It affects directly PP and PMC practices.
- Lack of practices for risk management, affecting RSKM, PP e PMC practices.
- Lack of additional practices for issues and dependencies management, affecting practices related to PMC specific goal 2 and some practices of IPM.
- Lack of planning and controlling of project's budget, compromising PP and PMC practices.
- Lack of planning and controlling of project's data, impacting the adherence to PP and PMC practices.
- Lack of integrated project management due to absence of an integrated and defined process, tailored from the organization's standard processes, according to IPM + IPPD specific goal 1.
- Absence of a historic base, affecting the evolution of higher maturity levels.

Beyond that, part of these gaps is related to absence of documentation (written evidences) during activities execution. It is caused by one of Agile Manifesto values "Working software over comprehensive documentation", meaning that the project team should document just what they consider significant for the project, independent of the organizational knowledge.

4 Investigation to blend practices of Scrum and CMMI for project management improvement

The results described in the previously sections allowed us to start our blending strategy by accomplishing two parallel tasks. First, we defined a proposal for the extension of Scrum, that is based on resolving major gaps of Scrum related to estimates, risks and issues management practices required by CMMI. It doesn't approach every gap found, but try to cover the main ones in order to make Scrum almost compliant with CMMI maturity level 2. SAM and QPM weren't considered because these process areas not always are applicable to the organizations and they are not so important in the context of agile project management. Second, we investigated how the organizations which have been adopting agile methods in their process (or not adopted yet, but have interest to) are both carrying out estimates and applying risk and issues management practices. Spaces do not permit a full description of these tasks, and we will just present here the results of such investigation. Interested readers can get the proposal in [18].

4.1 Study case

The investigation was performed through an online research [11] targeting Brazilian organizations interested to improve their project management processes based on CMMI and agile methods.

The research was sent to several organizations through discussion lists related to the main theme of this work. After the deadline for collecting responses, 73 valid responses were collected from different organizations. The sample was quite diversified and organizations came from all the Brazilian regions participated of this study.

The research was published through Survey Monkey tool (http://www.surveymonkey.com). The questionnaire had nine questions about the organization, addressing information about its structure (such as its location, its size, its age, its software development process adopted and its interest on process improvement), as well as information about how its professionals estimate and manage risks and issues.

4.1.1 Organizations profile

The responses collected were consolidated and some graphs were generated. Figure 8 shows that 44% of the organizations have no more than 10 years, 37% are have between 11 and 40 years and just 19% have more than 40 years. Considering organizations that have no more than 40 years, we have 81% of the sampling, in other words, most of them are young organizations.

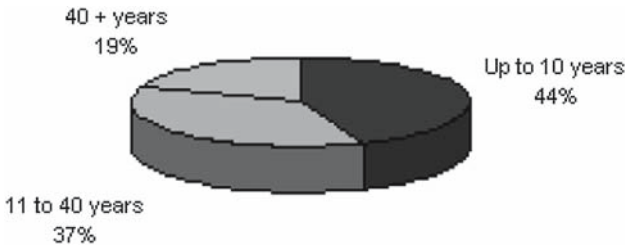


Fig. 8 Organizations age

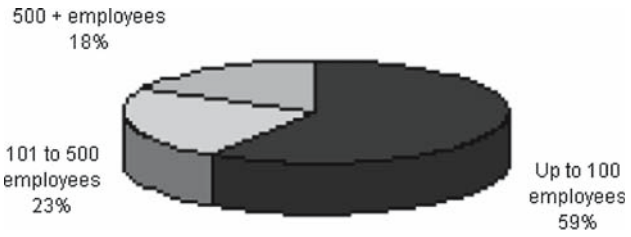


Fig. 9 Size (number of employees) of the organizations

In relation to their size (considering how many employees are involved with project management and development activities), Fig. 9 shows that 59% have no more than 100 professionals, 23% have between 101 and 500 people working in projects and just 18% have more than 500 employees. The results show that most of them are small in size.

4.1.2 Results presentation

The analyzed results helped to understand how the organizations are applying the three suggested practices for extending Scrum, according to [18]:

- Practice 1—Complexity estimate
- Practice 2—Risk management
- Practice 3—Issues management

The analyses were based on the answers for three questions:

1. How do you estimate the project efforts in your organization? Do you use any specific technique? If yes, what technique?
2. What is the organization experience related to risk management? Is it used any procedure/tool for helping it? Do you follow any model (as PMBOK or CMMI, for example)?

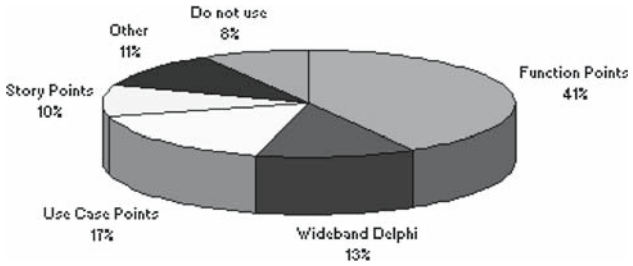


Fig. 10 Estimate methods used frequently

3. What is the organization experience related to issues management? Is it used any procedure/tool for supporting it?

Just 56 organizations (76.7%) answered the questions above. Considering this universe, seven have no interest on using agile methods, 19 apply agile methods in their processes and 30 do not use agile method but have inclination to do. In other words, 87.5% of the analyzed organizations apply or have interest to apply agile methods.

Analyze of results: *estimates*

Figure 10 shows the distribution of estimate methods mentioned by the 56 organizations. The result considers one or more methods in the same organization.

In relation to estimate methods used by the organizations, 41% use function points [12], 17% use case points [4], 13% wideband Delphi [26] and 10% story points [10].

The results also show that, from the organizations that use Scrum, 50% use story points for estimating, as well. It is similar to the practice of complexity estimate mentioned on [18].

Analyze of results: *risk management*

In relation to risk management experience, 23.2% of organizations have few or no experience. From the ones that have it, Fig. 11 shows that 34% based their processes on PMBOK [22], 21% on CMMI, 24% do not use any model e 21% do not manage risks.

One of them mentions risk management through impediments, as Scrum suggests. In other word, most of them follow a more formal method for risk management, based on CMMI or PMBOK. However, it is necessary a more agile alternative to manage risks in the Scrum's flow of activities.

Analyze of results: *issues management*

In relation to issues management, 45% have few or no experience (Fig. 12). From the organizations with experience in it, most of them mention tools or procedures to support it but

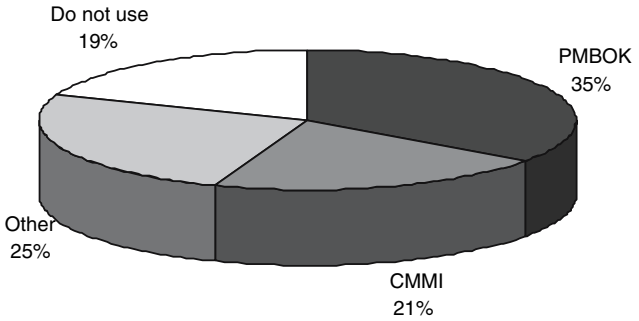


Fig. 11 Reference models for risk management

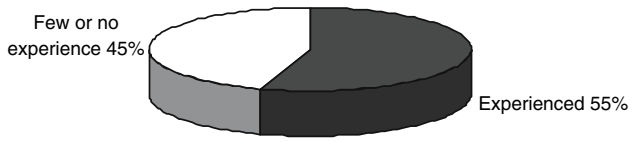


Fig. 12 Experience of the organizations with issues management

not mention reference models. Just one of them mentions Scrum retrospective as a moment for identifying and analyzing corrective actions to be incorporated in the next iteration (or sprint).

Similarly to the analysis for risk management, it is necessary agile practices for managing issues in the Scrum's flow activities. According to Chin [8], identify and monitor project issues and its corrective actions systematically help to resolve critical problems efficiently and it is essential for a well succeed project management.

5 Conclusions and future work

According to the analysis above, Scrum is a recommended starting point for organizations with small teams and without defined processes. Most of necessary fundamentals for institutionalizing CMMI project management process areas related to maturity level 2 are created without compromising the agility needed by these organizations. However, organizations searching higher maturity levels are not fully attended by Scrum practices. Other alternative practices are necessary to complement Scrum in order to address CMMI requirements.

The results of the study case show that the adoption of agile practices in the software development process has been a trend in both CMMI-based organizations and organizations that want to reach any CMMI maturity level

Also, it can be concluded that few adaptations on Scrum, mainly related to agile risk management, issues management and estimates methods, make it much more compliant (or fully compliant) with CMMI project management process areas.

In this way, the definition of an agile project management approach, based on Scrum and CMMI, is significantly useful for organizations that have been trying to improve their processes using a combination between agility and maturity models.

Therefore, the following directions present interesting points to continue the work:

- Complement the initial proposal of extension of Scrum with new alternative practices, making it more compliant with CMMI, without losing its agility.
- Apply this approach in real software development projects.
- Refine the extension of Scrum using the feedback collected through the real experience of the proposed methodology.
- Define an integrated tool to support agile project management based on Scrum and adherent to CMMI.

References

1. Advanced Development Methods (1996) Controlled chaos: living on the edge, http://www.controlchaos.com/old-site/ap.htm
2. Advanced Development Methods (2003) Scrum methodology—incremental, iterative software development from agile processes, Revision 0.9
3. Alleman G (2004) Blending agile development methods with CMMI, Cutter IT J 17(6):5–15
4. Banerjee G (2001) Use case points—an estimation approach. http://www2.fiit.stuba.sk/~bielik/courses/msi-slov/reporty/use_case_points.pdf
5. Beck K. et al (2001) Manifesto for agile Software Development, http://agilemanifesto.org/
6. Boehm B, Demarco T (2002) The agile Methods Fray, IEEE Computer Science, pp 90–91
7. Boehm B (2006) A view of 20th and 21st century software engineering, ICSE 2006
8. Chin G (2004) Agile project management: how to succeed in the face of changing project requirements, Amacon
9. Cochango (2006) Scrum for team systems. http://www.Scrumforteamsystem.com
10. Cohn M (2006) Agile estimating and planning, Prentice Hall, Englewood Cliffs, p 330
11. Fink A (1995) The survey handbook. The Survey kit, vol 1. Thousand Oaks, Sage p 129
12. Garmus D, Herron D (2001) Function point analysis: measurement practices for successful software projects. Addison-Wesley, Boston
13. Gloger B (2007) The Zen of Scrum, http://www.glogerconsulting.de
14. Highsmith J (2002) Agile software development ecosystems. Addison-Wesley, Boston
15. Highsmith J (2004) Agile project management—creating innovative products. Addison-Wesley, Boston
16. Larman C (2004) Agile & iterative development, a manager's guide. Addison-Wesley, Boston
17. Menezes W (2002) To CMMI or not to CMMI: issues to think about. Crosstalk 15(2): 9–11
18. Marçal A, Freitas B, Soares F, Maciel T, Belchior A (2007) Estendendo o Scrum segundo as Áreas de Processo de Gerenciamento de Projetos do CMMI, CLEI 2007: XXXIII Conferencia Latinoamericana de Informática, San Jose, Costa Rica, 9–12 October
19. Orr K (2002) CMM versus agile development: religious wars and software development, Cutter Consortium. Executive Report, vol 3 No 7
20. Paulk M (2001) Extreme programming from a CMM perspective. IEEE Softw 18(6): 19–26
21. Pikkarainen M, Mäntyniemi A (2006) An approach for using CMMI in agile software development assessments: experiences from three case studies. In: SPICE 2006 conference, Luxemburg, 4–5 May
22. Project Management Institute (2004) A guide to the project management body of knowledge, 3rd edn. Project Management Institute, Nashnua
23. Software Engineering Institute (2006) CMMI–DEV: CMMI for development, V1.2 model, CMU/SEI-2006-TR-008, http://www.sei.cmu.edu/cmmi/general/
24. Schwaber K (2004) Agile project management with Scrum, Microsoft Press, Redmond
25. Turner R, Jains A (2002) Agile meets CMMI: culture clash or common cause. XP/Agile Universe. pp 153–165
26. Wiegers K (2007) Practical project initiation: a handbook with tools, Microsoft Press, Redmond