VI SINGEP
Simpósio Internacional de Gestão de Projetos, Inovação e Sustentabilidade
International Symposium on Project Management, Innovation and Sustainability
ISSN: 2317-8302
V ELBE
Encontro Luso-Brasileiro de Estratégia
Iberoamerican Meeting on Strategic Management

# CMMI e Metodologias Ágeis no Desenvolvimento de Softwares

**PAULO ROBERTO DA SILVA JUNIOR**
SENAC
paulo.rsilva@gmail.com

**MARIO ROBERTO DOS SANTOS**
UNINOVE – Universidade Nove de Julho
mario.rsantos@terra.com.br

**FÁBIO YTOSHI SHIBAO**
UNINOVE – Universidade Nove de Julho
fabio.shibao@gmail.com

# CMMI E METODOLOGIAS ÁGEIS NO DESENVOLVIMENTO DE *SOFTWARES*

**Resumo**

O objetivo desta pesquisa foi avaliar as boas práticas de Processos CMM/CMMI e suas possibilidades de integração com os métodos de desenvolvimento e gerenciamento ágeis *Scrum* e *Extreme Programming* (XP). Foram realizadas pesquisas teóricas, com base em livros e artigos científicos de periódicos nacionais e internacionais, assim como *sites* especializados no assunto. Foram também utilizadas como fonte de pesquisa, a análise de discussões e análise de profissionais de tecnologia da informação (TI) em blogs, fóruns e matérias de revistas. Para complementar a análise do tema, foi realizada uma pesquisa, por meio de um questionário, no período de 09 à 22 de janeiro/2017, respondida por 39 profissionais de TI. O resultado desta pesquisa mostrou que 87,2% dos profissionais consultados acreditam que é possível aplicar simultaneamente Metodologias Ágeis e CMMI em projetos de desenvolvimento de *software*.
**Palavras-chave**: CMMI; *Extreme programming* (XP); Métodos ágeis; *Scrum.*

**Abstract**

The purpose of this research was to study and analysis the good uses of CMM and CMMI process and theirs possibilities of integration with agile methods Scrum e Extreme Programming (XP). Theoretical search was made throw books and scientific articles nationals and internationals one as specialized websites on software development. Another search source was discussions analyses and information technology professionals' analyses on blogs, forums and magazines. In additional, an on-line search was made whit a quiz, between 09th and 22th January on 2017 and it received 39 answers by IT professionals. The quiz's results shows that 87.2% of these professionals think that is possible apply simultaneal Agile Methods and CMMI on projects of software development.

**Key words**: Agile methods; CMMI; Extreme programming (XP); Scrum.

## 1 Introdução

Um dos desafios das empresas no século XXI é conseguir se adequar às mudanças tecnológicas e trazer a melhor experiência para os seus clientes, conquistando e fidelizando novos consumidores e investidores. Esses fatos geram a necessidade de as empresas oferecerem soluções adaptadas as expectativas e aos desejos dos seus clientes, atualizando-se de forma rápida e eficiente pois, "[...] o mercado exige cada vez mais produtos inovadores e disponíveis em um rápido espaço de tempo" (Michels & Ferreira (2013, p. 52).

Dentro da expectativa de atender bem os clientes conjugado com o uso eficiente de recursos e de tempo, disponibilizando produtos e serviços de qualidade, as empresas estão adotando a gestão de projetos (Ferreira, Almeida, Leão, & Silva, 2013).

As empresas buscam métodos e processos de gerenciamento de projetos mais eficientes pois faz-se necessário ganhos de produtividade e melhoria de desempenho nessa gestão (Barboza, Vaz, Antunes, & Salume, 2016), porque falhas nesses processos poderão representar grandes prejuízos para essas empresas.

Dentro da área de desenvolvimento de *software* não é diferente, novos padrões, metodologias e processos de gestão de projetos são desenvolvidos e discutidos com o objetivo de inovar as formas de construção e manutenção das soluções de tecnologia, desde os *devices*, desenvolvimento de *softwares* incluindo a construção de datacenters e a respectiva infraestrutura. Segundo Vargas (2016, p. 50) "[...] um dos desafios na área de desenvolvimento de *software* está na complexidade da compreensão das expectativas do cliente através da análise dos problemas e da criação de soluções que efetivamente atendam aos anseios e agreguem valor."

O uso de produtos de *software* vem crescendo e isso acarretou o aumento da exigência por qualidade nos produtos obrigando os desenvolvedores a seguir alguns modelos, como, por exemplo o *Capability Maturity Model Integration* [CMMI] (Garcia, Oliveira, Salviano, & Vasconcelos, 2015), pois as metodologias tradicionais para gerenciamento de projetos já não atendem plenamente as necessidades gerenciais (Rovai, 2013).

Essas exigências por qualidade e novas funções estão em constantemente evolução, levando a uma volatilidade de requisitos elevados, o que exige também que as empresas de *software* sejam altamente flexíveis o que as levou a adotarem métodos incrementais e ágeis (Petersen & Wohlin, 2010).

Existem trabalhos na literatura que comparam e sugerem a adoção da metodologia CMMI e métodos ágeis, mas não necessariamente sua integração. Os processos CMMI têm como foco a entrega da documentação e os métodos de gerenciamento ágeis na entrega rápida dos produtos para os clientes.

O objetivo desta pesquisa foi avaliar as boas práticas de Processos CMMI e as possibilidades para integração com os Métodos de Gerenciamento Ágeis *Scrum* e *Extreme Programming* (XP), visando assim integrar dois processos distintos, minimizando possíveis riscos e falhas nas entregas de projetos, diminuindo custos operacionais, alinhamento de recursos e entrega com qualidade ao cliente final. Dessa forma, pretende-se responder a seguinte questão de pesquisa:

**É possível integrar as práticas de processos CMMI com Métodos de Gerenciamento Ágeis?**

Para avaliar a experiência e conhecimento desses *frameworks* foi realizada uma pesquisa com profissionais da área por meio de um questionário disponibilizado no *site* do *SurveyMonkey*.

Este trabalho está estruturado da seguinte forma: após esta introdução, a seção dois apresenta o referencial teórico; a seção três, o método de pesquisa empregado; na seção

quatro, o resultado e a análise dos dados; na seção cinco, as considerações finais e, posteriormente, as referências utilizadas.

## 2 Referencial Teórico

Nesta seção será realizada uma breve descrição do CMMI e dos métodos ágeis *Scrum* e *Extreme Programming* (XP). Além desses dois métodos, ressalte-se, porém, que na literatura apresentam-se vários outros métodos ágeis de desenvolvimento de *software*, entre os quais podem ser citados: *Agile Modeling*, *Agile Unified Process* (AUP), *Agile Data Method*, *Dynamic Systems Development Method* (DSDM), *Essential Unified Process* (EssUP), *Feature Driven Development* (FDD), *Getting Real*, *OpenUP* (Open Unified Process) (Carvalho & Mello, 2009).

### 2.1 *Capability Maturity Model Integration* (CMMI)

Uma das primeiras publicações que tratavam o conceito de Modelo de Maturidade, foi desenvolvida por Watts Humphrey no livro "*Managing the Software Process*" publicado em 1989, onde apresentou os princípios e conceitos básicos nos quais muitos dos modelos de maturidade e de capacidade estão baseados, conforme o *Software Engineering Institute* (SEI, 2006).

O termo "modelo de maturidade" e os cinco níveis que o acompanham, foram inspirados pelo trabalho de Philip Crosby: *Manufacturing Maturity Model* (Paulk, Curtis, Chrissis, & Weber, 1993).

O modelo original de *Capability Maturity Model* (CMM), *Software Capability Maturity Model* (SW-CMM), foi publicado em 1991 pelo SEI (Paulk, 2009), e desenvolvido a partir de uma demanda do Departamento de Defesa dos Estados Unidos da América (EUA), dada a necessidade de compreender, remediar e evitar falhas e gastos excessivos em projetos de desenvolvimento de *softwares* em larga escala para o governo (Anderson, 2012).

O *Capability Maturity Model Integration* (CMMI) ou Integração do Modelo de Maturidade de Capacitação, não é uma metodologia, não orienta como fazer ou quem deve fazer, mas o que deve ser feito. É um modelo sobre práticas maduras e consolidadas para desenvolvimento e manutenção de produtos e serviços, cobrindo todo o ciclo de vida, desde a concepção até a entrega do *software* (SEI, 2010). É um modelo resultado da união dos modelos CMM, baseados nas melhores práticas das indústrias em engenharia, aquisição de *software* e gerenciamento das equipes de trabalho, visando reduzir os custos e a complexidade de se implantar múltiplos modelos (SEI, 2010).

O modelo da SEI segundo Mello (2011, p. 13):

> [...] oferece uma estrutura e elementos chave para um processo de software eficaz, abrangendo todo o ciclo de produção, desde a concepção até a entrega e manutenção do software, representando ainda um caminho evolutivo para a organização em busca de um processo maduro e disciplinado.

A avaliação do estágio de desenvolvimento de uma empresa em relação a aplicação do modelo CMMI é realizado por intermédio de um avaliador credenciado que define qual o estágio que ela se encontra e identificará qual o próximo estágio a ser alcançado e quais competências devem ser adquiridas (SEI, 2010).

Diversos conceitos e modelos compõem o CMMI, entre os quais podem ser destacados (SEI, 2010):

a) *Software Engineering Capability Maturity Model* (SW-CMM);
b) *Systems Engineering Capability Maturity Model* (SE-CMM);
c) *Software Development Capability Evaluation* (SDCE);

d) Normas da *International Organization for Standardization* (ISO) da série 9000 (ISO 9000);
e) Metodologia Seis Sigma;
f) *Organization Project Management Maturity Model* (OPM3 – PMI).

Todos os modelos CMMI refletem os níveis de maturidade em seu *design* e conteúdo. Um nível de maturidade é composto por práticas específicas e genéricas relacionadas a um conjunto predefinido de áreas de processos que melhoram o desempenho global da organização. O nível de maturidade de uma organização é uma indicação do desempenho da organização em uma determinada disciplina ou conjunto de disciplinas (SEI, 2006, p.37).

O modelo CMMI apresenta cinco níveis de maturidade organizacional e cada um representa a base para a melhoria contínua dos processos: (1) Inicial; (2) Gerenciado; (3) Definido; (4) Gerenciado Quantitativamente; e (5) Em Otimização (SEI, 2006) e são descritos como segue:

1) Nível 1 Inicial: os processos são caóticos e dependem da competência das pessoas para obterem sucesso. Normalmente, os produtos e serviços funcionam, mas extrapolam os orçamentos e não cumprem o prazos. Tendência de se comprometerem além da sua capacidade, abandonam o processo em um momento de crise, e são incapazes de repetir os próprios sucessos.

2) Nível 2 Gerenciado: os projetos têm a garantia de que os processos são planejados e executados de acordo com uma política; empregam pessoas experientes que possuem recursos adequados para produzir saídas controladas; envolvem partes interessadas relevantes; são monitorados, controlados e revisados; e são avaliados para verificar sua aderência em relação à descrição de processo. O status dos produtos e a entrega dos serviços estão visíveis para a gestão em pontos definidos. Os produtos de trabalho e serviços satisfazem às descrições de processo, aos padrões e procedimentos especificados.

3) Nível 3 Definido: os processos são bem caracterizados e entendidos, e são descritos em padrões, procedimentos, ferramentas e métodos. O conjunto de processos-padrão da organização é estabelecido e melhorado ao longo do tempo. Esses processos-padrão são utilizados para estabelecer uniformidade no contexto da organização. Os processos são gerenciados com base na compreensão de como as atividades de processo relacionam-se e nas medidas detalhadas do processo, seus produtos de trabalho e serviços.

4) Nível 4 Gerenciado Quantitativamente: a organização e os projetos estabelecem objetivos quantitativos para a qualidade e o desempenho de processo, utilizando-os como critérios na gestão de processos. Objetivos quantitativos baseiam-se nas necessidades dos clientes, dos usuários finais, da organização e dos responsáveis pela implementação de processos. A qualidade e o desempenho do processo são entendidos em termos estatísticos e gerenciados ao longo da vida dos processos. Identificam-se as causas de variação de processo e as fontes dessas causas são corrigidas para prevenir sua recorrência.

5) Nível 5 em otimização: a organização melhora continuamente seus processos com base no entendimento quantitativo das causas comuns de variação inerentes ao processo. Tem foco na melhoria contínua do desempenho de processo por meio de melhorias incrementais e inovadoras de processo e de tecnologia. A organização trata as causas comuns de variação de processo e promove as mudanças a fim de melhorar o desempenho e satisfazer aos objetivos quantitativos de melhoria de processo.

As áreas de processos são organizadas em quatro categorias: Gestão de Processo, Gestão de Projeto, Engenharia e Suporte. Essas categorias enfatizam como as áreas de processos existentes se relacionam. A Figura 1, a seguir, mostra as categorias, as áreas de processos com os quais tem interação e os respectivos níveis de maturidade.

| Categoria | Área de Processo | Nível de Maturidade |
|---|---|---|
| Engenharia | Integração de Produto | 3 |
| | Desenvovimento de Requisitos | 3 |
| | Gestão de Requisitos | 2 |
| | Solução Técnica | 3 |
| | Validação | 3 |
| | Verificação | 3 |
| Gestão de Processo | Implantação de Inovações na Organização | 5 |
| | Definição dos Processos da Organização | 3 |
| | Foco nos Processos da Organização | 3 |
| | Desempenho dos Processos da Organização | 4 |
| | Treinamento na Organização | 3 |
| Gestão de Projeto | Gestão Integrada de Projeto | 3 |
| | Monitoramento e Controle de Projeto | 2 |
| | Planejamento de Projeto | 2 |
| | Gestão Quantitativa de Projeto | 4 |
| | Gestão de Riscos | 3 |
| | Gestão de Contrato com Fornecedores | 2 |
| Suporte | Análise e Resolução de Causas | 5 |
| | Gestão de Configuração | 2 |
| | Análise e Tomada de Decisões | 3 |
| | Medição e Análise | 2 |
| | Garantia da Qualidade de Processo e Produto | 2 |

**Figura 1** – Categorias e áreas de processos do CMMI
**Fonte**: Adaptado de SEI (2006, p.44).

As organizações podem alcançar melhorias progressivas em sua maturidade organizacional, conseguindo primeiro o controle no âmbito do projeto até chegar à melhoria contínua de processo no contexto da organização, utilizando tanto dados quantitativos quanto dados qualitativos para a tomada de decisão. (SEI, 2006, p. 40)

A utilização do CMMI propõe uma redução de custos com o aprimoramento dos processos nos seguintes itens: previsão de custos e tempo mais efetivos; aumento de produtividade; maior qualidade dos produtos e atendimento da necessidade do cliente; aumento de retorno de investimento; não ter inconsistências e diminuição de duplicações (SEI, 2006).

Os primeiros adeptos do CMMI eram desenvolvedores de sistemas de grande escala, com aversão ao risco e de missão crítica, muitas vezes com altos níveis de supervisão de gerenciamento e governança hierárquica (Glazer, Dalton, Anderson, Konrad, & Shrum, 2008).

## 2.2 Metodologias ágeis

No início de 2001, um grupo de profissionais independentes com uma forte ligação com a indústria de *software* fundou o que foi mais tarde chamado de movimento ágil. O manifesto ágil foi divulgado em uma reunião de 17 especialistas em *software* entre 11 a 13 de fevereiro em Utah, EUA (Melo et al. 2013) sendo uma das características do manifesto a identificação do que era considerado ou não ágil (Bernardo, 2014).

O objetivo do manifesto foi chamar a atenção para a ideia de que para produzir *software* de alta qualidade e valioso, as equipes de desenvolvimento devem se concentrar em valores e princípios, tais como (1) indivíduos e interações, (2) *software* de trabalho, (3) colaboração com clientes e (4) responder à mudança (Melo et al. 2013). Ainda em 2001 criou-

se a *Agile Alliance* com o objetivo de manter as discussões dos métodos ágeis existentes e disseminar o conhecimento. (Bernardo, 2014)

O manifesto ágil foi o princípio para discussão de outras formas de implantação de projetos se diferenciando dos métodos tradicionais. De acordo com os princípios ágeis enunciados nesse manifesto, desenvolvedores de *software* - contando com excelência técnica e desenhos simples - criam valor para o negócio, fornecendo *software* para os usuários em curtos intervalos regulares. Esses princípios têm gerado uma série de práticas que são acreditados para entregar maior valor aos clientes (Dingsoyr, Nerur, Balijepally, & Moe 2012).

Os doze princípios do manifesto ágil para desenvolvimento de *softwares* são descritos como (Beck et al., 2001):

1) A maior prioridade é satisfazer o cliente por meio da entrega contínua e adiantada de *software* com valor agregado.
2) Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças para que o cliente possa tirar vantagens competitivas.
3) Entregar frequentemente *software* funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.
4) Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.
5) Construir projetos em torno de indivíduos motivados, dando a eles o ambiente e o suporte necessário, e confiando neles para fazer o trabalho.
6) O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é por intermédio de conversa face a face.
7) *Software* funcionando é a medida primária de progresso.
8) Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
9) Atenção contínua a excelência técnica e bom *design* aumentam a agilidade.
10) Simplicidade: a arte de maximizar a quantidade de trabalho não realizado é essencial.
11) As melhores arquiteturas, requisitos e *designs* emergem de times auto-organizáveis
12) Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Dyba e Dingsoyr (2009) citaram em sua pesquisa sobre aplicação de métodos ágeis, que, embora o estudo tenha identificado sérias limitações, tais como a permanência do cliente no local por longos períodos e a dificuldade de introduzir métodos ágeis em projetos grandes e complexos, os resultados sugerem que os métodos ágeis podem melhorar a satisfação no trabalho, a produtividade e a satisfação do cliente. Os autores citaram também que os estudos de equipes ágeis maduras sugerem que é necessário ter foco em fatores humanos e sociais para ter sucesso. Também um alto nível de autonomia individual deve ser equilibrado com um alto nível de autonomia de equipe e responsabilidade corporativa (Dyba & Dingsoyr, 2009).

Métodos ágeis de desenvolvimento de *software* têm sido cada vez mais adotados e tornaram-se uma das principais abordagens de desenvolvimento de *software* (Melo et al., 2013). “A proposta deste tipo de metodologia consiste em dividir o desenvolvimento de *software* em diversas iterações de ciclos, que duram poucas semanas, em que o cliente recebe ao final de cada ciclo uma aplicação que agregue valor ao negócio” (Barboza et al., 2016, p.4).

As metodologias ágeis oferecem oportunidades de liderança empresarial porque são notavelmente diferentes das metodologias tradicionais de desenvolvimento de *software* (Qureshi, 2017). As metodologias tradicionais são bem definidas e por esse motivo, muitas vezes, não conseguem responder suficientemente rápido em um ambiente em mudança,

podendo não ser viável sua utilização em todos os casos. Agilidade significa eliminar essa morosidade associada às metodologias tradicionais de desenvolvimento de *software* e promover uma resposta rápida a ambientes em mudança, seja pelas mudanças nos requisitos pelos usuários ou nos prazos acelerados dos projetos (Erickson, Lyytinen, & Siau, 2005).

Conforto, Amaral, Silva, Di Felippo e Kamikawachi (2016) definiram em sua pesquisa que agilidade é a capacidade da equipe do projeto de mudar rapidamente o plano do projeto como uma resposta às demandas de clientes ou interessados, mercado ou tecnologia para alcançar um melhor desempenho do projeto e do produto em um ambiente de projeto inovador e dinâmico. Os autores fizeram essa proposição, pois até aquela data não haviam encontrado na literatura uma definição robusta com um conjunto adequado de variáveis desenvolvidas para auxiliar na avaliação da agilidade que considerasse toda a teoria da gestão de projetos (Conforto et al. 2016).

Alguns métodos ágeis podem ser destacados, como o *Scrum* e o *Extreme Programming* (XP), que serão a seguir relatados.

***Scrum***

O *Scrum* foi importante no desenvolvimento dos métodos ágeis e tornou-se amplamente conhecido por volta de 2006, quando a *Scrum Alliance* (uma organização sem fins lucrativos) se tornou uma entidade corporativa e iniciou o processo de certificação para reunir profissionais que atendessem aos seus critérios (Melo et al., 2013).

A definição de *Scrum*, conforme os criadores do Guia do *Scrum*, Schwaber e Sutherland (2016, p.3), é "[...]um *framework* dentro do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível. [...] é leve, simples de entender, extremamente difícil de dominar". Está fundamentado nas teorias empíricas de controle de processo e é "[...] uma abordagem enxuta de desenvolvimento de produtos" (Carvalho & Mello, 2009, p.1).

O *Scrum* não é um processo ou uma técnica para construir produtos, é uma estrutura na qual o desenvolvedor poderá empregar vários processos ou técnicas podendo melhorar a eficácia das práticas de gerenciamento e desenvolvimento de produtos (Schwaber & Sutherland, 2016). "O método não requer ou fornece qualquer técnica específica para a fase de desenvolvimento, apenas estabelece conjuntos de regras e práticas gerenciais que devem ser adotadas para o sucesso de um projeto" (Carvalho & Mello, 2009, p.3).

A Figura 2 mostra o *Scrum framework*.

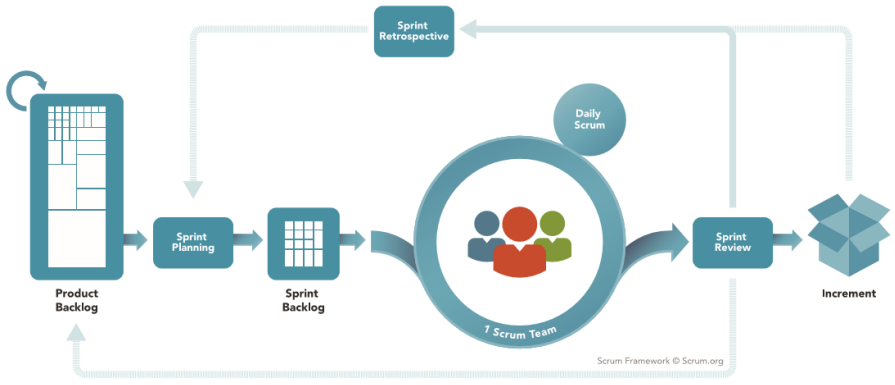


**Figura 2** – *Scrum framework*
**Fonte**: https://www.scrum.org/Resources/What-is-Scrum.

O *product backlog* (ou *backlog* do produto) é uma lista ordenada de tudo que deve ser necessário no produto. O *sprint backlog* é um conjunto de itens do *backlog* do produto selecionados para a *sprint*, juntamente com o plano para entregar o incremento do produto e

atingir o objetivo da *sprint*. O *backlog* da *sprint* é a previsão do Time de Desenvolvimento sobre qual funcionalidade estará no próximo incremento e sobre o trabalho necessário para entregar essa funcionalidade em um incremento "Pronto". O *backlog* da *sprint* torna visível todo o trabalho que o Time de Desenvolvimento identifica como necessário para atingir o objetivo da *sprint*. Pronto significa que o trabalho está completo e é usado para assegurar quando o trabalho está completado no incremento do produto (Schwaber & Sutherland, 2016).

*Sprint* é um *time box* de um mês ou menos, durante o qual um "Pronto", versão incremental potencialmente utilizável do produto, é criado. São compostas por uma reunião de planejamento da *sprint*, reuniões diárias, o trabalho de desenvolvimento, uma revisão da *sprint* e a retrospectiva da *sprint*. Cada *sprint* tem a definição do que é para ser construído, um plano projetado e flexível que irá guiar a construção, o trabalho e o resultado do produto. Cada *sprint* pode ser considerado um projeto com horizonte não maior que um mês (Schwaber & Sutherland, 2016).

O guia apresenta algumas sugestões para o uso da metodologia, entre as quais podem ser citadas (Schwaber & Sutherland, 2016):

a) Grupos de trabalho (*Scrum team*) entre três a nove pessoas;

b) Reuniões diárias (*Daily Scrum*) para acompanhamento das evoluções. Utilização de quadros para que a equipe visualize o que precisa fazer, o que está sendo feito e o que será feito;

c) Utilização de prazos curtos e pré-definidos (*time box*) e as entregas são concluídas e comunicadas para a equipe: o *sprint* é um exemplo de *time box*, que trata de ciclos de entrega.

O *Scrum* fornece um conjunto de melhores práticas destinadas a entrega rápida e de valor para o cliente e a sua adoção pelos desenvolvedores de *softwares* está crescendo com o objetivo de aumentar a taxa de sucesso dos respectivos projetos (Tavares, Silva, & Souza, 2016).

### ***Extreme Programming* (XP)**

O *Extreme Programming* (XP) começou na Chrysler Corporation em 1996 e tornou-se um dos métodos mais reconhecidos da família Agile. Antes do final da década de 1990, ficou evidente para muitas empresas e desenvolvedores de *software* que em muitas configurações, as comunicações presenciais, a interação rigorosa com o cliente, com pequenas equipes e rápidas, com entregas frequentes, produziram *softwares* de grande qualidade (Glazer et al., 2008). Essa metodologia tem como objetivo ter agilidade e garantir a satisfação do cliente.

As práticas do XP descritas por Medeiros (2013) são:

a) Planejamento - definindo o que será e o que não será feito.

b) Programação em pares – dois desenvolvedores dividindo um computador (um codifica e o outro crítica).

c) Pequenas versões – entregas pequenas e frequentes.

d) Propriedade coletiva – todos responsáveis pelo *software*.

e) Metáforas – não utilização de termos técnicos, trazendo uma comunicação transparente para o cliente.

f) Integração contínua – integração diária com a realização de todos os testes.

g) Projeto simples – atender os requisitos atuais e não se preocupar com os requisitos futuros.

h) Semana de 40 horas – não realização de horas extras.

i) Testes – validação durante o desenvolvimento.

j) Cliente presente – definindo as prioridades e os testes de aceitação.

k) Refatoração – aprimorar o código sem alterações na funcionalidade.

l) Padronização de código – a equipe define o padrão de código no início do projeto.

m) Reunião diária – proveniente do *Scrum*.

O dinamismo do método é caracterizado por meio de quatro valores (Paulk, 2001): (i) comunicação contínua com o cliente e dentro da equipe; (ii) simplicidade, alcançada por um foco constante em soluções minimalistas; (iii) *feedback* rápido por intermédio de mecanismos como testes unitários e funcionais; e (iv) coragem de lidar de forma proativa com os problemas. Esse dinamismo funciona unindo toda a equipe por meio de práticas simples, com *feedback* suficiente para permitir que a equipe saiba em qual estágio o processo de desenvolvimento está e sintonize as práticas com a situação (Jeffries, 2001, como citado em Erickson et al., 2005). É focada no aumento das relações como chave para o sucesso, incentiva o trabalho em equipe, se preocupa com a aprendizagem dos desenvolvedores e promove um bom ambiente de trabalho (Letelier & Penadés, 2006).

Uma das principais características do XP é que o processo de desenvolvimento é a codificação do que o cliente especifica. Nenhuma ferramenta ou funcionalidade é projetada antecipadamente sem necessidade porque o XP está orientado a desenvolver um produto em tempo hábil, pois, se os recursos forem necessários mais tarde, o cliente notificará a equipe de desenvolvimento. Isso representa uma diferença com o processo normal de desenvolvimento de *software*, em que todos os requisitos devem ser especificados antes do início do desenvolvimento em si. Assim, os requisitos do usuário podem ser vistos como dinâmicos e em vez de estáticos e configurados (Erickson et al., 2005). É adequada para projetos com requisitos imprecisos e mutáveis com alto risco técnico (Letelier & Penadés, 2006).

## 3 Procedimentos metodológicos

Para a realização deste trabalho foram realizadas pesquisas teóricas, com base em livros e artigos científicos de periódicos nacionais e internacionais, assim como *sites* especializados no assunto.

Foram utilizadas como fonte de pesquisa também a análise de discussões e análise de profissionais de TI em blogs, fóruns e matérias de revistas.

Para complementar a análise do tema, foi realizada uma pesquisa, por meio de um questionário, no período de 09 à 22 de janeiro/2017 disponibilizado no site https://pt.surveymonkey.com. Foram encaminhados e-mails para 62 profissionais da área de TI, solicitando a participação na pesquisa, sendo respondida por 39.

A pesquisa procurou identificar o perfil profissional dos respondentes, inquirindo sobre a empresa onde trabalha (localização física, porte da empresa), faixa etária do respondente e nível de escolaridade. Indagou, também, sobre a área de atuação, experiência com desenvolvimento de *software*, conhecimento sobre métodos e padrões de gestão de projetos, experiência pessoal no uso desses métodos e padrões, práticas de gestão de projetos adotados pelas respectivas empresas. A última pergunta, referente ao tema desta pesquisa, procurou verificar a opinião dos profissionais se entendem que seria possível aplicar simultaneamente Metodologias Ágeis e CMMI em projetos de desenvolvimento de *software*.

## 4 Análise dos resultados

Nesta seção, serão apresentadas as comparações entre as metodologia ágeis e o modelo CMMI e a análise e resultado da pesquisa realizada.

### 4.1 Comparação das metodologias ágeis com o Modelo CMMI

A Figura 3, mostra um resumo do resultado da comparação entre os principais itens dos métodos ágeis e CMMI encontrados na literatura.

| **Métodos Ágeis** | **CMMI** |
|---|---|
| Metodologia | Não é metodologia é um modelo |
| Define papéis | Não define papéis |
| Como fazer | O que fazer |
| Recomenda o ciclo de vida iterativo e incremental | Não estabelece ordem de execução dos processos |
| Agilidade nos projetos | Melhoria de processos |
| Objetivo: desenvolvimento de *software* (não manutenção) | Objetivo: desenvolvimento e manutenção de *software* |
| Preparado para mudanças | Mais resistente à mudanças |
| Menos controlado, com poucos princípios | Maior controle, com políticas e normas |
| Cliente parte da equipe | Cliente atua mais nas reuniões |
| Grupos pequenos | Grupos grandes |
| Poucos artefatos | Muitos artefatos |
| Poucas regras | Mais regras |

**Figura 3** – Comparação entre os métodos ágeis e CMMI
**Fonte**: Elaborado pelos autores.

O ponto de destaque do Figura 3 é o formato mais burocrático do CMMI, por ter muitos artefatos e regras, é um modelo "o que fazer" que possui maior controle do processo por meio de políticas e normas.

Por outro lado, os Métodos Ágeis são preparados para mudanças com poucas regras, poucos artefatos, foco na agilidade do processo e o cliente é entendido como parte da equipe, o que pode ser mais interessante para a entrega de determinados projetos.

## 4.2 Resultado e análise da pesquisa

O perfil profissional dos 39 respondentes apresentou as seguintes características:
a) 95% dos entrevistados trabalham no Estado de São Paulo (capital e grande São Paulo);
b) 97,4 % têm mais de 25 anos;
c) 66,7% está cursando ou já concluiu a pós-graduação;
d) 69,2% trabalham em grandes empresas (500 ou mais funcionários);
e) 84,6% possuem experiência em desenvolvimento de *software*;

A atuação profissional dos pesquisados revelou que a maioria é gerente de projetos ou *scrum master* (33%); ou desenvolvedor (31%); na sequência têm-se governança de TI (15%); arquiteto de infraestrutura (2%); cliente (5%); infraestrutura (3%); *product owner* (8%); analista de segurança (3%). Destaca-se que os profissionais respondentes fazem parte da equipe de projetos, o que contribuiu para o resultado da pesquisa.

O conhecimento dos participantes sobre o tema é caracterizado, conforme apresentado na Tabela 1.

**Tabela 1** – Conhecimento sobre o tema

| Metodologias | Nenhum | Básico | Intermediário/ Avançado | Participou de curso | Total |
|---|---|---|---|---|---|
| Scrum | 20,5 | 35,9 | 33,3 | 10,3 | 100 |
| Extreme Programming (XP) | 43,6 | 28,2 | 23,1 | 5,1 | 100 |
| CMMI | 28,9 | 39,5 | 31,6 | 0 | 100 |
| Lean Development | 59 | 28,2 | 10,2 | 2,6 | 100 |
| Kanban | 33,4 | 23,1 | 38,4 | 5,1 | 100 |
| RUP | 61,6 | 12,8 | 25,6 | 0 | 100 |
| PMBOK | 7,7 | 28,2 | 56,4 | 7,7 | 100 |
| PRINCE2 | 71,8 | 17,9 | 10,3 | 0 | 100 |

**Fonte**: Dados da pesquisa

Os profissionais que contribuíram com a pesquisa possuem, em sua maioria, conhecimentos intermediário ou avançado em metodologias tradicionais baseadas no PMBOK (56,4%), *Kanban* (38,4%), *Scrum* (33,6%) e CMMI (31,6%). Considerando-se o objetivo da pesquisa, nota-se que, considerando-se os métodos ágeis *Scrum* (33,3%) e XP (23,1%), 56,4% dos respondentes têm experiência nesses métodos. Observa-se também que é o mesmo percentual (56,4%) dos profissionais que têm conhecimento sobre o PMBOK.

A seguir, a Tabela 2, apresenta a experiência dos profissionais pesquisados com as metodologias.

**Tabela 2** – Experiência dos profissionais pesquisados

| Metodologias | Nenhuma | Inferior à 1 ano | Acima de 1 ano | Total |
|---|---|---|---|---|
| Scrum | 23,1 | 30,8 | 46,1 | 100 |
| Extreme Programming (XP) | 56,4 | 7,7 | 35,9 | 100 |
| CMMI | 47,4 | 21 | 31,6 | 100 |
| Lean Development | 69,2 | 12,8 | 18 | 100 |
| Kanban | 38,5 | 20,5 | 41 | 100 |
| RUP | 68,4 | 5,3 | 26,3 | 100 |
| PMBOK | 20,5 | 12,8 | 66,7 | 100 |
| PRINCE2 | 87,2 | 5,1 | 7,7 | 100 |

**Fonte**: Dados da pesquisa

Similar a tabela relacionada com o conhecimento, a experiência dos profissionais é maior com metodologias tradicionais baseadas no PMBOK (66,7%) e, em seguida, com *Scrum* (46,1%), *Kanban* (41%) e CMMI (31,6%). Considerando-se a soma dos percentuais, acima de um ano de experiência, *Scrum* (46,1%) e XP (35,9%) têm-se um valor relevante (82,0%) de experiência profissional nos métodos ágeis. O uso de metodologias tradicionais baseadas no PMBOK (66,7%) é também relevante, pois, segundo Melo et al. (2013), ainda é muito forte nas universidades e empresas brasileiras a cultura e a tradição de planejar, desenvolver e documentar a avaliação da evolução dos projetos.

O que se notou também na pesquisa é que as empresas, em sua maioria, praticam esses *frameworks*, conforme mostra a Figura 4.


**Figura 4** – Práticas de gestão de projetos nas empresas
**Fonte**: Dados da pesquisa

Uma das prováveis explicações para o uso do *Scrum* (66,7%) pelas empresas, é porque uma das qualidades das metodologias ágeis, segundo Letelier e Penadés (2006), é a simplicidade, tanto na aprendizagem quanto na aplicação, reduzindo, dessa forma, os custos de implementação em uma equipe de desenvolvimento e por esse motivo levou a um interesse crescente nessas metodologias.

Como última pergunta, questionou-se os profissionais da possibilidade de se aplicar simultaneamente Metodologias Ágeis e CMMI em projetos de desenvolvimento de *software* e 87,2% afirmou que é possível.

Os participantes tiveram a oportunidade de justificar a resposta e indicaram a combinação do CMMI e métodos ágeis (utilizando o que cada um pode apresentar como melhor e pontos não conflitantes), sinalizando que a maturidade do grupo viabiliza mais essa prática. E, ainda, seguir os princípios das Metodologias Ágeis quanto às documentações e processos do CMMI, utilizando esse como método de apoio.

Os respondentes (12,8%) que acreditam não ser possível essa aplicação simultânea destacou a exigência de alto nível de maturidade e documentação, regras e padrões o que não se enquadra com metodologias ágeis. Também foi comentado que o CMMI tem como princípio maior a redução de variações diferentemente dos métodos ágeis que incentiva essas variações.

Os resultados aqui apresentados, têm alguns valores em consonância com a pesquisa de Melo et al. (2012) sobre o uso de métodos ágeis no Brasil, com 466 respondentes realizada entre maio e agosto de 2011: (i) 60,7% das organizações adotam projetos ágeis em 50% ou mais situações; e (ii) 64,8% das organizações têm mais de um ano de experiência com métodos ágeis; (iii) método mais aplicado pelos pesquisados é o *Scrum* (51,1%).

A pesquisa de Melo et al (2012) revelou também: (a) razão mais importante para uso de métodos ágeis é aumentar a produtividade (91%); (b) maior preocupação das empresas é com falta de documentação (50,6%); (c) principal barreira para adoção mais ampla é a capacidade de mudança da cultura organizacional (51,3%); (d) principal benefício obtido foi aumento de produtividade (69%); (e) principal causa de falha foi a falta de experiência com métodos ágeis.

## 5 Considerações finais

Esta pesquisa avaliou os conceitos que envolvem os *frameworks* CMMI e Metodologias ágeis e notou-se que a utilização dos dois *frameworks* possibilita uma melhor performance em desenvolvimento de *software*.

Destaca-se que o CMMI possui processos mais rigorosos complementando as metodologias ágeis e que traz como valor agregado ser mais importante o *software* desenvolvido do que documentação abrangente, ou seja, a utilização dos dois *frameworks* de forma complementar poderá ser uma estratégia na área de desenvolvimento de *software*.

O resultado da pesquisa com 39 profissionais mostrou que 87,2% acreditam que é possível aplicar simultaneamente Metodologias Ágeis e CMMI em projetos de desenvolvimento de *software*.

Para a questão de pesquisa estipulada neste trabalho:

**É possível integrar as práticas de Processos CMMI com Métodos de Gerenciamento Ágeis?**

Dentro das limitações da pesquisa, mas em razão do que foi observado na literatura e o resultado da consulta com os profissionais, verificou-se que o CMMI e os métodos ágeis podem conflitar em algumas disciplinas, porém há áreas em que ambos podem existir e liderar aprimorando a qualidade e velocidade de entrega de *software*. Os métodos ágeis aparentam ser menos rigorosos, principalmente em comparação a metodologias tradicionais de gerenciamento de projetos, mas exige maior disciplina da equipe do projeto e processos para garantir alto desenvolvimento de velocidade e qualidade. Apresenta a disciplina e processos por meio de documentação leve em áreas chaves no CMMI, portanto, o desenvolvimento Ágil pode ser utilizado integrado com o CMMI.

Segundo Glazer et al. (2008), os métodos de desenvolvimento ágil e as práticas CMMI são muitas vezes percebidas como incompatíveis entre si mas sugerem que a discórdia não precisa existir e propõe que as empresas trabalhem para tirar o máximo de proveito de usar ambas as sinergias e explorar as potencialidades para melhorar o desempenho do negócio.

Sheshasaayee e Vijaykumar (2015) advertiram que cada método de desenvolvimento tem limitações e a identificação de tais fraquezas permite que as organizações ou os desenvolvedores possam obter mais benefícios nos seus desenvolvimentos e o método Ágil não é uma exceção e tem gargalos que reduzem a sua produtividade. Spundak (2014) mencionou que tanto as abordagens tradicionais quanto as ágeis têm suas vantagens e desvantagens, quando comparadas às diferentes características do projeto. Dybá e Dingsoyr (2009) recomendaram que os profissionais estudem cuidadosamente as características de seus projetos e os comparem com as características necessárias dos métodos ágeis relevantes.

Não há uma abordagem universal para enfrentar com sucesso qualquer projeto de desenvolvimento de *software* e por esse motivo todos os métodos devem ser adaptados ao contexto do projeto (Letelier & Penadés, 2006). Há, portanto, muitos métodos que possibilitam aos desenvolvedores de *software*, tenham referências para conduzir um projeto de forma adequada e alcançar os objetivos propostos. O melhor método é aquele que os profissionais consigam conciliar as necessidades dos projetos e das organizações associados à qualidade nas entregas; capacidade de controle de custos prazos, riscos entre outras (Santos & Santos, 2016).

Entre as limitações desta pesquisa está a quantidade de respondentes (39), a ausência de tratamento matemático dos resultados e também a quantidade de perguntas realizadas, dessa forma, não será possível estender os resultados aqui encontrados.

Uma sugestão de pesquisa futura será ampliar o número de respondentes e de perguntas e, posteriormente, submeter as respostas a um tratamento matemático adequado o que, provavelmente, possibilitará estender os achados.

## Referências

Anderson, D. (2012). *Princípios e Valores CMMI*. Recuperado em 8 janeiro, 2017 de https://msdn.microsoft.com/pt-br/library/hh765978(v=vs.120).aspx.

Barboza, L. F., Vaz, A. C. F., Antunes, T. G. P., & Salume, P. K. (2016). Análise comparativa entre as abordagens ágil e tradicional de gestão de projetos: Um estudo de caso no setor industrial. *Anais do Simpósio Internacional de Gestão de Projetos, Inovação e Sustentabilidade*, São Paulo, SP, 5.

Beck, K., Beedle, M., Bennekum, A, van, Cockburn, A., Cunningham, W., Fowler, M. et al. (2001) *Manifesto for agile software development*. Recuperado em 6 janeiro, 2017 de http://agilemanifesto.org/.

Bernardo, K. (2014). *Manifesto Ágil, como tudo começou*. Recuperado em 6 janeiro, 2017 de http://www.culturaagil.com.br/manifesto-agil-como-tudo-comecou/.

Carvalho, B. V., & Mello, C. H. P. (2009). Revisão, análise e classificação da literatura sobre o método de desenvolvimento de produtos ágil Scrum. *Anais do Simpósio de Administração da Produção, Logística e Operações Internacionais*, São Paulo, SP, 12.

Conforto, E. C., Amaral, D. C., Silva, S. L., Di Felippo, A., Kamikawachi, D. S. L. (2016). The agility construct on project management theory. *International Journal of Project Management, 34*(4), 660-674.

Dyba, T., & Dingsoyr, T. (2009). What do we know about Agile Software development? *IEEE Software, 6*(5), 6-9.

Dingsoyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: towards explaining agile software development. *The Journal of Systems and Software, 85*(6), 1213-1221.

Erickson, J., Lyytinen, K., & Siau, K. (2005). Agile Modeling, Agile Software Development, and Extreme Programming: the state of research. *Journal of Database Management, 16*(4), 88-100.

Ferreira, B. A. A., Almeida, J. O. R., Leão, P. R. C., & Silva, N. P. G. (2013). Gestão de riscos em projetos: uma análise comparativa da Norma ISO 31000 e o Guia PMBOK®, 2012. *Revista Gestão e Projetos – GEP, 4*(3), 46-72.

Garcia, F. W. S., Oliveira, S. R. B., Salviano, C. F., & Vasconcelos, A. M. L. (2015). Uma abordagem para a implementação multi-modelos de qualidade de software adotando a CERTICS e o CMMI-DEV. *Revista de Sistemas de Informação da FSMA, 2015*(16), 26-40.

Glazer, H., Dalton, J., Anderson, D. J., Konrad, M., & Shrum, S. (2008). *CMMI or Agile: why not embrace both!* Software Engineering Institute, Carnegie Mellon University. Recuperado em 3 junho, 2017 de http://cmmiinstitute.com/sites/default/files/resource_asset/08tn003.pdf.

Letelier, P., & Penadés, M. C. (2006). Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *Técnica Administrativa, 5*(26). Recuperado em 13 julho, 2017 de http://www.cyta.com.ar/ta0502/b_v5n2a1.htm.

Medeiros, H. (2013). *Práticas em XP: extreme programming*. Recuperado em 8 janeiro, 2017 de http://www.devmedia.com.br/praticas-em-xp-extreme-programming/29330.

Mello, M. S. (2011). *Melhoria de processos de software multi-modelos baseada nos modelos MPS e CMMI-DEV*. Dissertação de mestrado, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ.

Melo, C. O., Santos, V., Corbucci, H., Katayama, E., Prikladnicki, R., Goldman, A., & Kon, F. (2012). *Métodos ágeis no Brasil: estado da prática em times e organizações*. Relatório Técnico RT-MAC-2012-03, Departamento de Ciência da Computação, IMEiUSP, 1-9.

Recuperado em 12 maio, 2017 de http://ccsl.ime.usp.br/agilcoop/files/metodos_ageis_brasil_estado_da_pratica_em_times_e_organizacoes.pdf.

Melo, C. O., Santos, V., Katayama, E., Corbucci, H., Prikladnicki, R., Goldman, A., & Kon, F. (2013). The evolution of agile software development in Brazil. *Journal of the Brazilian Computer Society, 9*(4), 523-552.

Michels, E., & Ferreira, M. G. G. (2013). Gerenciamento ágil no processo de desenvolvimento de produtos inovadores: uma análise bibliográfica sistemática. *Revista de Gestão e Projetos – GeP, 4*(1), 52-76.

Paulk, M. C. (2001). Extreme Programming from a CMM perspective. *IEEE Software, 18*(6), 19-26.

Paulk, M. C. (2009). *A History of the capability maturity model for software*. Recuperado em 12 fevereiro, 2017 de https://pdfs.semanticscholar.org/6fb0/c324e08698a9e364693151605a74982b487a.pdf.

Paulk, M. C, Curtis, B., Chrissis, M. B., & Weber, C. V. (1993). *Capability maturity model for software, version 1.1*. Recuperado em 23 março, 2017 de https://www.sei.cmu.edu/reports/93tr024.pdf.

Petersen, K., & Wohlin, C. (2010). The effect of moving from a plan-driven to an incremental software development approach with agile practices: an industrial case study. *Empirical Software Engineering, 15*(6), 654-693.

Qureshi, M. R. J. (2017). Adaptive framework to manage multiple teams using agile methodologies. *International Journal of Modern Education and Computer Science, 9*(1), 52-59.

Rovai, R. L. (2013). Metodologias inovadoras para gestão de projetos: modelo referencial para implantação da ITILV3 através da metodologia PRINCE2: estudo de caso. *Revista de Gestão de Projetos – GeP, 4*(2), 252-270.

Santos, P. R., & Santos, M. R. (2016). Comparação entre os padrões de gerenciamento de projetos PMBOK, ICB e PRINCE2. *Anais do Simpósio Internacional de Gestão de Projetos, Inovação e Sustentabilidade*, São Paulo, SP, 5.

Schwaber, K., & Sutherland, J. (2016). *Guia do Scrum*. Traduzido por Cruz, Fabio & Sucena, Eduardo Rodrigues. Recuperado em 5 maio, 2017 de http://www.scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-Portuguese Brazilian.pdf.

Sheshasaayee A. & Vijaykumar, H. (2015). Identifying bottlenecks in agile software development using Theory of Constraints Principles. *Indian Journal of Science and Technology, 8*(9), 1-10.

Software Engeneering Institute - SEI. *CMMI para Desenvolvimento – Versão 1.2*. Pittsburgh: Carnegie Mellon, 2006.

Software Engeneering Institute - SEI. *CMMI para Desenvolvimento – Versão 1.3*. Pittsburgh: Carnegie Mellon, 2010.

Spundak, M. (2014). Mixed agile/traditional project management methodology – reality or illusion? *Procedia - Social and Behavioral Sciences, 119*, 939-948.

Tavares, B. G., Silva, C. E. S., & Souza, A. D. (2016). Analysis of Scrum practices for risk treatment. *Product: Management & Development, 14*(1), 38-46. http://dx.doi.org/10.4322/pmd.2016.006

Vargas, L. M. (2016). Gerenciamento ágil de projetos em desenvolvimento de software: um estudo comparativo sobre a aplicabilidade do SCRUM em conjunto com o PMBOK e/ou PRICE2. *Revista de Gestão e Projetos – GeP*, *7*(3), 48-60.