Nesta seção você encontra artigos voltados para as práticas e métodos ágeis.

Extensão do Scrum segundo o MPS.BR nível G

Implementando agilidade e maturidade em uma fábrica de software

Fernando Szimanski
fernando@criativadsi.com.br
Possui graduação em Processamento de Dados (UNOPAR, 2002), Pós-graduação em Melhoria em Processo de Software (UFLA, 2006) e Mestrado em Engenharia de Sofware (CESAR.EDU, 2009). Atualmente é professor e coordenador de estágio do Centro Universitário UNIRG e Gerente de Projetos da Criativa Tecnologia. Tem experiência na área de Engenharia de Software, com ênfase em Modelos e Padrões de Melhoria no Processo de Software, Metodologias Ágeis e Gestão de Projetos.

Jones Oliveira de Albuquerque
jones.albuquerque@gmail.com
Possui graduação em Ciência da Computação (UFPE, 1994), mestrado em Ciências da Computação (UFPE, 1997) e doutorado em Ciências da Computação (UFMG, 2002). Atualmente é Professor Adjunto da Universidade Federal Rural de Pernambuco e colaborador ad hoc do C.E.S.A.R. – Centro de Estudos e Sistemas Avançados do Recife. Tem experiência na área de Ciência da Computação, foi diretor de empresa e atua na área de Engenharia de Software e Modelagem Matemática, atuando principalmente nas seguintes linhas: engenharia de software, processo de software, qualidade de software, matemática computacional, epidemiologia computacional e modelagem de sistemas.

Felipe Furtado
felipe.furtado@cesar.org.br
É graduado em Engenharia Mecânica pela UFPE, especialista em Tecnologias da Informação pelo Centro de Informática da UFPE e mestre em Ciência da Computação na área de produtividade de software também pela UFPE. Com 14 anos na área de desenvolvimento de software, atualmente é doutorando e coordenador da garantia da qualidade de software e do SEPG (Software Engineering Process Group) no C.E.S.A.R. É Scrum Master desde 2007 e possui experiência na definição e implantação de processos aderentes ao CMMI e em avaliações SCAMPI, com participação em avaliações Classe A CMMI níveis 2 e 3.

De que se trata o artigo?
Este artigo apresenta a extensão da metodologia ágil Scrum para as áreas de processo do MPS.BR nível G aplicada em uma pequena fábrica de software.

Para que serve?
A extensão realizada neste trabalho pode contribuir de forma relevante para organizações que têm o propósito de adotar práticas ágeis no seu processo de software mantendo a compatibilidade com o MPS.BR nível G.

Em que situação o tema é útil?
Introduzir conceitos ágeis em processos de software buscando um equilíbrio entre agilidade e maturidade é uma alternativa capaz de promover a melhoria da qualidade dos produtos e, consequentemente, aumento da competitividade no mercado.

Ao final dos anos 90, como reação às formas tradicionais de desenvolvimento de software, que baseado em estudos realizados pela indústria e academia [12], foi apontada como a principal responsável por falhas encontradas em projetos de software, acompanhamos o surgimento de várias metodologias ágeis, como: *Adaptive Software Development*, *Crystal*, *Dynamic Systems Development*, *eXtreme Programming* (XP), *Feature Driven Development* (FDD) e *Scrum* [4].

As metodologias ágeis são uma coleção de diferentes técnicas (ou práticas), que utilizam os mesmos valores e princípios básicos, tais como ciclos iterativos, entrega rápida de software funcionando e simplicidade, conforme está definido no Manifesto para Desenvolvimento Ágil [2].

Neste sentido, organizações que procuram melhoria em seus processos de

produção através de modelos e *frameworks* como *Capability Model Integration* (CMMI), *Control Objectives for Information and related Technology* (COBIT), *Information Technology Infrastructure Library* (ITIL), entre outros, agora também acreditam que introduzir conceitos ágeis em seus processos de desenvolvimento, buscando um equilíbrio entre agilidade e maturidade, é uma alternativa capaz de promover a melhoria da qualidade dos seus produtos e, consequentemente, aumento da competitividade no mercado [9].

Segundo o [Softex 2007], alcançar competitividade pela qualidade para as empresas de software implica tanto na melhoria da qualidade dos produtos de software e serviços correlatos, como dos processos de produção e distribuição de software. Desta forma, assim como para outros setores, qualidade é fator crítico de sucesso para a indústria de *software*.

O desenvolvimento ágil e os modelos e padrões de qualidade de software tradicionais são vistos frequentemente como contraditórios, pois se tem o raciocínio que os modelos são muito burocráticos, enquanto que o desenvolvimento ágil é *ad-hoc* [6]. Na verdade existem desafios na integração entre os dois, embora o esforço possa valer a pena, pois ao final pode-se obter qualidade no produto através da união de maturidade e agilidade.

Nessa direção, [Chin 2004] afirma que, se por um lado o excesso de formalidade e estruturação tende a inibir e limitar as equipes, por outro, a liberalidade caótica e informal, desprovida de processos, pode fazer com que os objetivos do projeto nunca sejam atingidos.

Inserido neste contexto, este trabalho tem o objetivo de propor uma estratégia para extensão do *Scrum* segundo as áreas de processo do guia MPS.BR nível G. Este estudo se inicia com o mapeamento entre o *Scrum* e o MPS.BR através de uma avaliação dos resultados esperados do guia segundo as práticas do Scrum. A partir deste mapeamento, uma extensão do Scrum é realizada através da adição de práticas complementares para satisfazer o guia. Ao final, é gerado um novo processo de desenvolvimento para uma fábrica de software.

O MPS.BR

O MPS.BR tem como objetivo definir um modelo de melhoria e avaliação de processo de software, preferencialmente para as micro, pequenas e médias empresas, para satisfazer as suas necessidades de negócio e também ser reconhecido nacional e internacionalmente como um modelo aplicável à indústria de software [11].

O Modelo de Referência MR-MPS define sete níveis de maturidade, que são uma combinação entre processos e capacidade de processos, tais como: A (Em Otimização); B (Gerenciado Quantitativamente); C (Definido); D (Largamente Definido); E (Parcialmente Definido); F (Gerenciado); G (Parcialmente Gerenciado). Para cada um destes sete níveis de maturidade foi atribuído um perfil de processos e de capacidade de processos que indicam onde a organização tem que concentrar o esforço para melhoria de forma a atender os objetivos de negócio [11].

O nível G de maturidade do MPS.BR (Parcialmente Gerenciado) é composto pelos processos de Gerência de Projetos e Gerência de Requisitos satisfazendo os atributos de processo AP 1.1 e AP 2.1.

O processo de Gestão de Projetos (GPR) é composto por dezessete resultados esperados e tem o propósito de estabelecer e manter planos que definem as atividades, recursos e responsabilidades do projeto, bem como prover informações sobre o andamento do projeto que permitam a realização de correções quando houver desvios significativos no desempenho do projeto [11].

O processo de Gerência de Requisitos (GRE) é composto por cinco resultados esperados. O seu propósito é gerenciar os requisitos do produto e dos componentes do produto do projeto e identificar inconsistências entre os requisitos, os planos do projeto e os produtos de trabalho do projeto [11].

O Scrum

A metodologia ágil *Scrum* foi criada em 1996 por Ken Schwaber e Jeff Sutherland, e destaca-se das demais metodologias ágeis pela maior ênfase dada ao gerenciamento do projeto [10].

Trata-se de uma abordagem empírica focada nas pessoas para ambientes em que os requisitos surgem e mudam rapidamente, resultando em uma abordagem que reintroduz as ideias de flexibilidade, adaptabilidade e produtividade [3]. Ela se baseia em princípios como: equipes pequenas, requisitos pouco estáveis ou desconhecidos e iterações curtas. As iterações são divididas em intervalos de tempos de, no máximo 30 dias, também chamadas de *Sprints*.

Papéis e responsabilidades

O Scrum define para sua estrutura iterativa incremental três papéis principais [10]:

- *Scrum Master:* gerencia o processo, ensinando o *Scrum* a todos os envolvidos no projeto e implementando o *Scrum* de modo que o mesmo esteja adequado à cultura da organização; deve garantir que todos sigam as regras e práticas do *Scrum*; e é responsável por remover os impedimentos do projeto [10]. Ele é o líder e facilitador entre o *Team* e o *Product Owner*;
- *Product Owner:* é o responsável pelo retorno sobre o investimento (ROI), ou seja, seu foco é na parte comercial do produto. Ele é o representante de todos os *stakeholders* e os representa na priorização do *Backlog* e questões de requisitos. Esta pessoa deve estar à disposição da equipe em qualquer momento, mas sobretudo durante o *Sprint Planning Meeting* e *Sprint Review Meeting* [13];
- *Team*: é um grupo de pessoas com diferentes habilidades necessárias para transformar requisitos em um incremento potencialmente "*entregável*". Para tanto, geralmente são uma mescla de analistas, designers, gerentes de qualidade, desenvolvedores, etc. A equipe tem a autoridade de decidir, quando necessário, as ações que serão realizadas e priorizá-las organizando-as em *Sprints*. *Effort estimation*, *Sprint Backlog*, revisões de *product Backlog List* e sugestões de impedimentos para serem removidos do projeto também são atividades do time [13].

As fases do Scrum

O ciclo de desenvolvimento do Scrum é baseado em três fases principais [1], conforme ilustra a **Figura 1**:

• PRE-GAME: dividida em duas subfases, Planejamento e *Staging*. A subfase Planejamento tem por objetivo estabelecer a visão do projeto e expectativas, garantindo recursos para a sua execução. A subfase *Staging* tem por objetivo avaliar as várias dimensões do projeto criando itens adicionais ao *Product Backlog* relacionados com o tipo do sistema, time, ambiente de desenvolvimento e tipo de aplicação;

• GAME: consiste de múltiplas *Sprints* para o desenvolvimento dos incrementos de funcionalidades do produto. Onde o time analisa a situação atual do produto, avaliando quais mudanças devem ser implementadas, procedendo ao final com o desenvolvimento através de etapas de análise, projeto, implementação, testes e documentação de mudanças;

• POST-GAME: fase de fechamento, que tem por objetivo preparar o produto para o seu lançamento. Isto inclui atividades de integração, análise, documentação do usuário, treinamento e preparação do material de marketing.

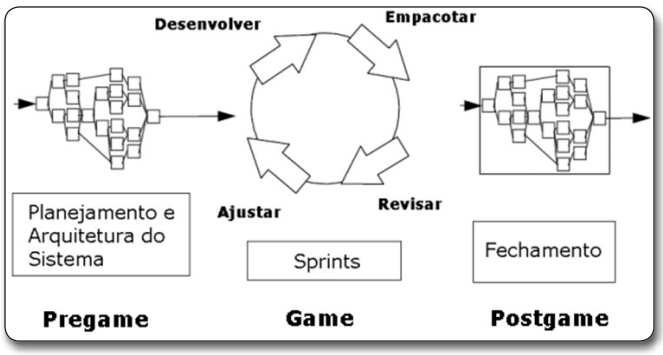


Figura 1. Fases do Scrum

Fluxo do Scrum

Um projeto se inicia com a visão do produto que será desenvolvido [10], com a lista das características do produto estabelecidas pelo cliente, além de algumas premissas e restrições. Em seguida, o *Product Backlog* é criado contendo a lista de todos os requisitos funcionais e não funcionais. O *Product Backlog* é então priorizado pelo *Product Owner* e dividido em *Releases*.

No *Scrum*, são realizadas iterações chamadas de *Sprints*. Segundo [Schwaber 2008], cada *Sprint* inicia-se com uma reunião de planejamento (*Sprint Planning Meeting*), na qual o *Product Owner* e o Time decidem em conjunto o que deverá ser implementado, ou seja, quais unidades de trabalho serão incluídas nas iterações de *Sprint* (*Selected Backlog*). A reunião é dividida em duas partes. Na primeira parte (*Sprint Planning 1*), o *Product Owner* apresenta os requisitos de maior valor e prioriza aqueles que devem ser implementados. O time então define colaborativamente o que poderá entrar no desenvolvimento da próxima *Sprint*, considerando sua capacidade de produção. Na segunda parte (*Sprint Planning 2*), o time planeja seu trabalho, definindo o *Sprint Backlog*, que são as tarefas necessárias para implementar as funcionalidades selecionadas no *Product Backlog*. Nas primeiras *Sprints*, é realizada a maioria dos trabalhos de arquitetura e de infraestrutura. A lista de tarefas pode ser modificada ao longo da *Sprint* pelo time e as tarefas podem variar entre 4 a 16 horas para a sua conclusão.

Na execução das *Sprints*, diariamente o time faz uma reunião de 15 minutos para acompanhar o progresso do trabalho e agendar outras reuniões necessárias. Ao final da *Sprint*, é realizada a reunião de revisão (*Sprint Review Meeting*) para que o time apresente o resultado alcançado na iteração ao *Product Owner*. Neste momento as funcionalidades são inspecionadas e adaptações do projeto podem ser realizadas. Em seguida o *Scrum Master* conduz a reunião de retrospectiva (*Sprint Retrospective Meeting*), com o objetivo de melhorar o processo/time e/ou produto para a próxima *Sprint*.

O monitoramento do progresso do projeto é realizado através de gráficos de acompanhamento (*Burndown Charts*). Eles são um excelente mecanismo para visualizar a correlação entre a quantidade de trabalho a ser realizada (em qualquer ponto) e o progresso do time do projeto para reduzir este trabalho [8].

A **Figura 2** demonstra detalhadamente o fluxo de desenvolvimento do Scrum.

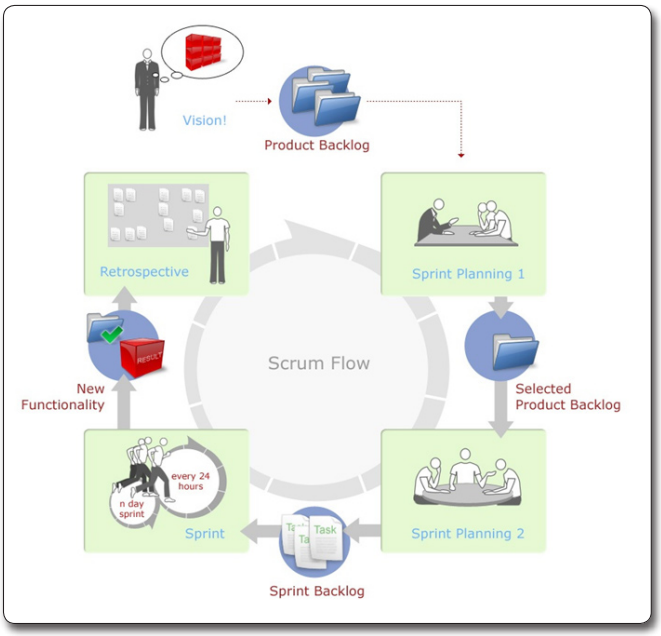


Figura 2. Fluxo do Scrum (Fonte: Adaptação de [7])

Mapeando o Scrum para as áreas de processo do MPS.BR nível G

Para cada área de processo do MPS.BR nível G foi realizada uma análise detalhada entre os resultados esperados do MPS.BR e as práticas encontradas no Scrum, classificando cada resultado esperado de acordo com os critérios estabelecidos na **Tabela 1**.

Para avaliar o processo segundo o método de avaliação do MPS.BR (MA-MPS), o processo de desenvolvimento da organização é avaliado baseado em evidências (diretas e indiretas) de um conjunto de atividades e tarefas para atingir este propósito. Portanto, estes aspectos não foram considerados no escopo deste trabalho, pois o mapeamento foi realizado com

o objetivo somente de tornar a organização aderente ao guia e não para fins de certificação.

Classificação		Critério
NS	Não Satisfeito	Não há evidências da prática na metodologia
PS	Parcialmente Satisfeito	Há evidências da prática na metodologia, embora a prática não esteja plenamente atendida.
S	Satisfeito	A prática está totalmente atendida na Metodologia.

Tabela 1. Critérios para classificação das áreas de processO

Após a realização da classificação, conforme os critérios acima estabelecidos, foram calculados os percentuais de satisfação de cada área de processo entre os critérios definidos, tomando como base o número total de resultados esperados de cada área.

Para o processo de desenvolvimento de software de uma organização estar totalmente aderente ao nível G do MPS.BR, é necessário também atender aos atributos de processo AP 1.1 e AP 2.1 no que se refere aos resultados esperados RAP1 a RAP8. Por essa razão, também foi realizado o mapeamento entre os atributos de processo acima relacionados e o novo processo de software da empresa em questão.

O mapeamento detalhado apresentando os pontos que o Scrum satisfaz ou não os resultados esperados do MPS.BR nível G (Gestão de projetos e Gerenciamento de requisitos) pode ser encontrado em [15].

Os resumos das análises para os processos de Gestão de Projetos (GPR) Gerenciamento de Requisitos (GRE) são apresentados nas **Tabelas 2** e **3**, respectivamente.

MPS.BR – Nível G			SCRUM	
Área de Processo	**Abrev.**	**Objetivos**	**Classificação**	**Resumo das Práticas**
Gestão de Projetos (GPR)	GPR1	Escopo do trabalho	Satisfeito	Elaboração da Visão do Produto e Definição dos Itens de Backlog (IBL)
	GPR2	Dimensionamento de tarefas e produtos de trabalho	Parcialmente Satisfeito	Estimativas através de Story Points
	GPR3	Modelo e as fases do ciclo de vida do projeto	Satisfeito	Iterativo incremental. Fases: Pregame, Game e Postgame
	GPR4	Estimativa de esforço e o custo baseado em dados históricos ou referências técnicas	Parcialmente Satisfeito	Retorno sobre o Investimento (ROI), Estimativas através de Story Points e divisão de IBLs em tarefas
	GPR5	O orçamento e o cronograma do projeto	Parcialmente Satisfeito	Planejamento de Sprints e Estimativas através de Story Points
	GPR6	Riscos do projeto	Satisfeito	Daily Meetings, Impediment Backlog e responsabilidades do Scrum Master
	GPR7	Planejamento de recursos humanos	Satisfeito	Definição dos Recursos humanos (Time, Product Owner, Scrum Master) baseado no perfil
	GPR8	Planejamento das tarefas, os recursos e o ambiente de trabalho	Satisfeito	Itens adicionais ao Product Backlog e Tarefas da Sprint (Task)
	GPR9	Planejamento de dados relevantes do projeto (armazenamento, privacidade e segurança)	Não Satisfeito	Prática não mencionada no Scrum
	GPR10	Planos para a execução do projeto	Satisfeito	Visão do Produto, Product Backlog, Sprint Backlog, Sprint planning 1 e 2 e Daily meeting
	GPR11	Avaliação de viabilidade de atingir as metas do projeto	Satisfeito	Sprint Planning (1 e 2) e Definição do objetivo da iteração na elaboração do Sprint Backlog
	GPR12	Revisão e compromisso com o Plano do Projeto	Satisfeito	Sprint Planning (1 e 2) e Daily meeting
	GPR13	Monitoramento do progresso do projeto	Satisfeito	Product Burndown, Sprint Burndown, Daily Review e Retrospective Meeting
	GPR14	Gerenciamento do envolvimento das partes interessadas no projeto	Satisfeito	Daily Scrum Meeting, Sprint Review Meeting e Sprint Retrospective Meeting
	GPR15	Revisões são realizadas em marcos do projeto e conforme estabelecido no planejamento	Satisfeito	Sprint Review Meeting
	GPR16	Identificação e registros de problemas	Satisfeito	Daily Scrum Meeting e Impediment Backlog
	GPR17	Ações para corrigir e prevenir desvios em relação ao planejado	Satisfeito	Daily Scrum Meeting, Impediment Backlog e Retrospective Meeting

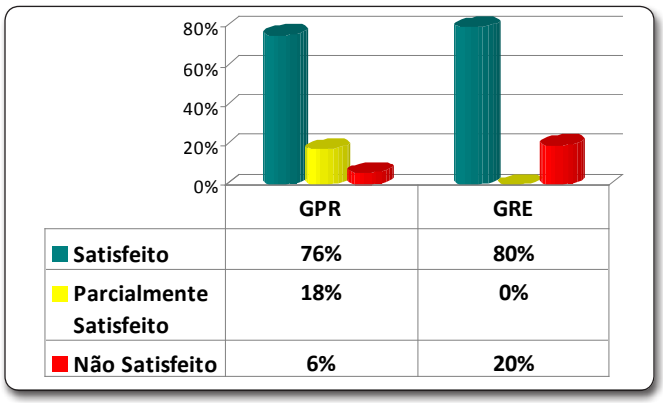
Tabela 2. Resumo do mapeamento entre Scrum e a área de GPR

MPS.BR - Nível G			SCRUM	
Área de Processo	**Abrev.**	**Objetivos**	**Classificação**	**Resumo das Práticas**
Gerenciamento de Requisitos (GRE)	GRE1	Entendimento dos requisitos	Satisfeito	Visão do produto, Definição de itens de Backlog e Elaboração das tarefas da Sprint
	GRE2	Aprovação de requisitos	Satisfeito	Aprovação de requisitos através do Business Value (BV), Priorização de requisitos (ROI) e Sprint Planning (1 e 2)
	GRE3	A rastreabilidade bidirecional entre os requisitos e os produtos de trabalho	Não Satisfeito	Prática não mencionada no Scrum
	GRE4	Revisões em planos e produtos de trabalho do projeto	Satisfeito	Daily Scrum Meeting e Sprint Review Meeting
	GRE5	Gerenciamento de mudanças nos requisitos	Satisfeito	Sprint Planning (1 e 2), Daily Scrum Meeting e Sprint Review Meeting

Tabela 3. Resumo do mapeamento entre Scrum e a área GRE

Resultado do mapeamento

Este comparativo entre o *Scrum* e as áreas de processo do MPS.BR nível G teve por objetivo mapear o alinhamento e a conformidade entre eles. A partir deste mapeamento descobrimos que o *Scrum* não satisfaz totalmente o nível G do MPS.BR, conforme apresentado na **Figura 3**.



	GPR	GRE
Satisfeito	76%	80%
Parcialmente Satisfeito	18%	0%
Não Satisfeito	6%	20%

Figura 3. Resultado Geral da Avaliação

Este resultado se deve a alguns fatores, tais como:

- Lacunas na definição de um método apropriado de complexidade ou tamanho para estimativas, impactando diretamente no resultado esperado GPR2 e indiretamente no GPR5;
- Ausência de utilização de dados históricos organizacionais ou referências técnicas para definição de esforço e custo de um projeto, afetando o resultado esperado GPR4;
- Ausência de definição de privacidade e segurança no acesso às informações do projeto, comprometendo totalmente o GPR9;
- Ausência de rastreabilidade entre os requisitos e produtos de trabalho, afetando diretamente o GRE3.

Portanto, para implementação de um processo aderente às práticas definidas nas áreas de processo do nível G do MPS.BR, foi necessário complementar o *Scrum* com práticas adicionais conforme visto na próxima seção.

Estendendo o Scrum para o MPS.BR nível G

Baseado no resultado geral do comparativo entre o *Scrum* e as áreas de processo do MPS.BR nível G, para as lacunas encontradas (itens "**Parcialmente Satisfeito**" e "**Não satisfeito**"), soluções complementares foram adicionadas ao processo, conforme descrição abaixo.

GAP1 – Lacunas na definição de um método apropriado de complexidade ou tamanho para as tarefas e os produtos de trabalho do projeto, impactando diretamente o GPR2.

Para realização da estimativa das tarefas foi indicado o uso da técnica *Story Points* com *Planning Poker.*

A ideia do *Planning Poker* [Mountain Goat Software 2008] é pontuar os itens de *Backlog*, onde cada membro da equipe possui um conjunto de cartas numeradas com a sequência de *Fibonacci*. Inicialmente escolhe-se a funcionalidade de menor complexidade e atribui a ela a pontuação 2 (dois) para servir como referência. A partir deste item de backlog são realizadas estimativas relativas para os demais itens. Caso haja divergências entre as cartas mostradas, as pessoas que atribuíram o menor e o maior valor explicam o motivo que os levaram a tal estimativa. É importante que o *Scrum Master* esteja envolvido como mediador para gerenciar conflitos.

GAP2 – Ausência de estimativa do esforço e o custo para a execução das tarefas e dos produtos de trabalho com base em dados históricos ou referências técnicas.

O *Scrum* não menciona a utilização de dados históricos organizacionais, ou seja, ele somente define a utilização de dados de *Sprints* anteriores para o mesmo projeto. Portanto, para definição dos custos e esforço serão analisados dados históricos organizacionais mantidos através de um documento que contempla informações de projetos anteriores.

GAP3 – Ausência de definição do orçamento.

Para definição do orçamento será utilizada a política de negociação da empresa em questão. Este documento fornece orientações sobre os procedimentos a serem observados na negociação dos projetos de software da empresa no tocante às regras em relação às pessoas, condutas, vedações, divulgação de informações e violações.

GAP4 – Ausência de privacidade e segurança no acesso às informações.

A proposta é armazenar as informações coletadas no projeto utilizando a política de segurança, acesso e armazenamento das informações de projetos. Este documento define a estratégia para implementação nos projetos da empresa no que se refere à infraestrutura (confiabilidade, segurança e estabilidade) e nível operacional (níveis de usuários, responsabilidades, limites e controle de versões).

GAP5 – Ausência de rastreabilidade entre os requisitos e produtos de trabalho.

Para a rastreabilidade vertical cada item de *Backlog* (IBL) será inicialmente identificado, então, na criação do *Sprint Backlog* cada tarefa estará associada a um IBL e os artefatos de engenharia estarão associados ao IBL correspondente. Portanto, a rastreabilidade vertical pode ser distribuída em cada artefato gerado, como por exemplo, o código implementado.

Adaptando a extensão no processo de uma Fábrica de Software

Para validação deste trabalho, foi implementada a extensão proposta na CRIATIVA Tecnologia, que é uma empresa de tecnologia que cria produtos e serviços utilizando Tecnologia da Informação (TI). Atualmente a empresa conta com cerca de trinta colaboradores, atendendo diversos clientes e atuando em três áreas de negócio: fábrica de software, treinamentos e prestação de serviços técnicos em equipamentos eletrônicos.

Histórico de melhoria de processos da CRIATIVA

Os trabalhos em melhoria de processo de software na CRIATIVA iniciaram cerca de três anos após a sua fundação (2003), onde após um crescimento significativo de sua base de clientes, a empresa começou a apresentar diversos problemas relacionados ao processo de produção de software. Baseado neste contexto, foi realizada a modelagem de um processo de desenvolvimento baseado no guia MPS.BR nível G [14], onde no período inicial pode-se notar melhorias. Entretanto, após alguns meses de utilização, alguns fatores impactaram negativamente na empresa em relação ao processo que foi definido, tais como: dificuldade em seguir o processo, produção de artefatos desnecessários, desmotivação dos colaboradores, entre outros.

Em seguida foi proposto o desenvolvimento de um novo processo de *software* que proporcionasse maturidade para a organização e que ao mesmo tempo tivesse seu foco nos princípios do manifesto ágil.

O novo processo de software da CRIATIVA Tecnologia

A partir de uma perspectiva de gerenciamento e baseado no ciclo de desenvolvimento iterativo e incremental definido por [10] para o framework *Scrum*, chegou-se a um processo composto por três fases: *Pregame*, *Game* e *Postgame*, apoiadas por atividades de Monitoramento e Controle, conforme ilustrado na **Figura 4**.

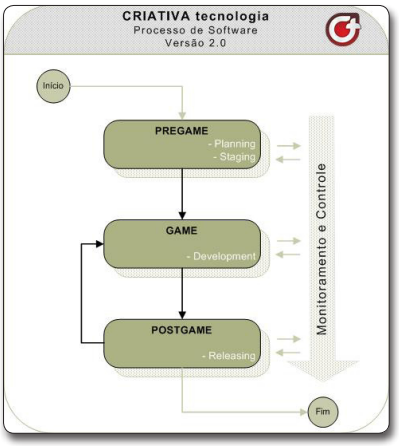


Figura 4. Processo de Software Macro da CRIATIVA Tecnologia

Com o objetivo de proporcionar o entendimento para os colaboradores da empresa em questão, para todas as fases deste novo processo foram descritas suas etapas, objetivos, conceitos chave e quadros de estruturação de cada atividade com seus respectivos papéis, entradas, saídas e artefatos.

Fase PREGAME: Esta fase tem como objetivos delimitar o escopo do projeto, verificar a viabilidade econômica e eliminar riscos a partir de uma arquitetura estável. A mesma foi dividida em duas subfases: *Planning* e *Staging*, o seu fluxo de atividades é apresentado na **Figura 5**.

Conceitos chave da subfase *Planning*

Esta é uma fase inicial que tem seu início através da criação da Visão do Produto, acessível por todos e de responsabilidade do *Product Owner*.

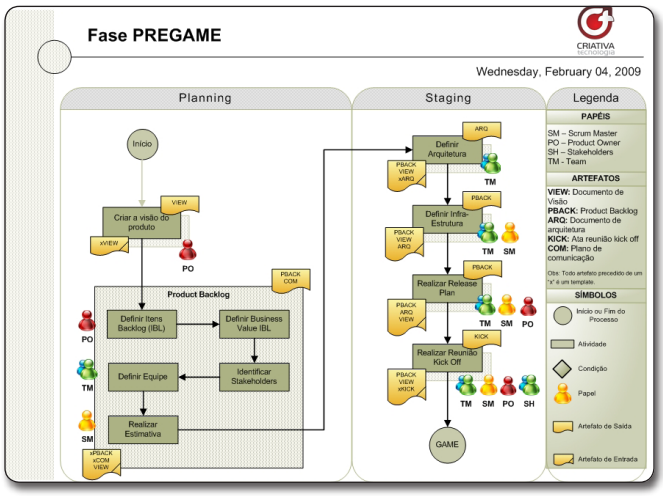


Figura 5. Fase PREGAME

O *Product Backlog* é outro artefato de responsabilidade do *Product Owner* (PO), mas também é atualizado pelo time, em comum acordo com o PO. Pode conter requisitos funcionais e não funcionais.

Conceitos chave da subfase *Staging*

Esta subfase é uma continuação da subfase inicial *Planning*, mas com o foco voltado à definição de algumas atividades, como: organização da infraestrutura para desenvolvimento do projeto, definição da arquitetura, definição do plano de comunicação, elaboração do *Release Plan* e realização da reunião de *kick-off*.

Fase GAME: esta fase (**Figura 6**) tem como objetivo criar releases do produto podendo obter versões funcionais do software. Nesta direção, o time realiza algumas atividades como: dividir os *Itens de Backlog* (IBL) que foram selecionados para a *Sprint* em tarefas, realizar estimativas, priorizar e analisar a viabilidade dos IBLs. Durante as reuniões diárias, as tarefas são selecionadas por cada membro do time. Desta forma, espera-se que o compromisso da equipe seja obtido e todos sigam para o desenvolvimento de uma parte do produto com base na arquitetura definida, enfatizando o gerenciamento de custos, recursos e qualidade.

Conceitos chave da fase *Game*

Esta fase se inicia com as reuniões da *Sprint* que são divididas em dois níveis, *Sprint Planning 1* e *Sprint Planning 2*. A *Sprint Planning 1* é uma reunião que tem por objetivo a verificação de dados históricos organizacionais, priorização de IBL da *Sprint*, seleção dos IBLs que irão compor a *Sprint* e elaboração da *Sprint Backlog*. A *Sprint Planning 2* é a segunda reunião de planejamento com participação do time e do *Scrum Master* para realização de atividades como: definição de tarefas (*Task*) necessárias para realização da *Sprint*, divisão das tarefas da *Sprint* entre a equipe, análise da capacidade produtiva da equipe e atualização do *Task Board*.

Ao final das *Sprint Planning 1* e *2*, a obtenção de compromisso e revisões no planejamento foram realizadas por todos os envolvidos. A partir disso é executada a *Sprint*, que consiste no desenvolvimento das tarefas definidas para cada IBL, com o objetivo de obter um produto potencialmente entregável.

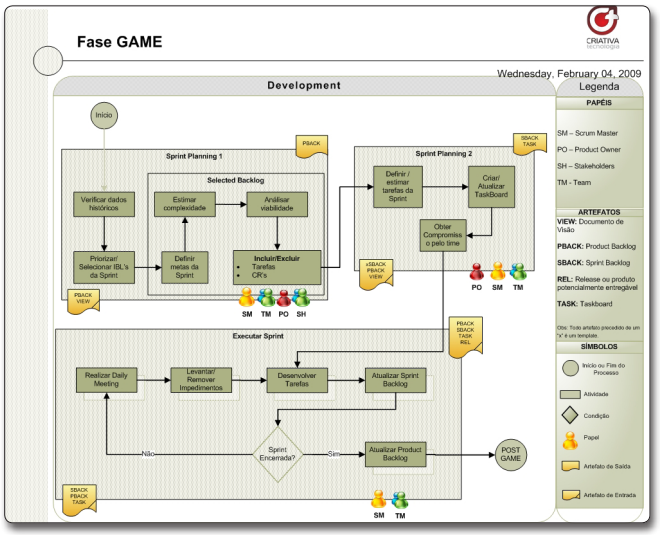


Figura 6. Fase GAME

Durante a execução da *Sprint* é realizada a atividade *Daily Scrum*, que é uma reunião diária onde o time monitora o progresso da execução das atividades e procura identificar os impedimentos encontrados.

Posteriormente, o *Scrum Master* trabalha no sentido de elaborar ações para resolver tais impedimentos e disseminar a solução para o time. Atualizações na *Sprint Backlog* e *Product Backlog* são realizadas para controlar o progresso das *Sprints*.

Fase POSTGAME: esta fase (**Figura 7**) tem como objetivos apresentar o produto ao cliente para validação, realizar uma reunião para melhoria do processo e avaliar a capacidade produtiva do time.

Conceitos chave da fase *POSTGAME*

A reunião de revisão da *Sprint*, chamada de *Sprint Review Meeting*, fornece algumas visões para todos os envolvidos, tais como: visibilidade se a meta da *Sprint* foi alcançada, apresentação de resultados, inspeção de funcionalidades para possíveis mudanças e adaptações, e a reunião de retrospectiva (*Sprint Retrospective Meeting*).

Ao final desta fase, caso haja nova *Sprint*, ou seja, caso ainda exista a necessidade de serem desenvolvidas outras funcionalidades, novos ciclos são realizados iniciando-se pela fase *GAME*.

Ao final, informações coletadas no projeto são armazenadas para utilização em projetos futuros formando a base de dados de histórico organizacional.

Estudo de Caso

Visando avaliar os efeitos da aplicação da extensão proposta neste artigo, foi realizado o estudo de caso na empresa em questão em projetos de *software* selecionados com base na semelhança das características de domínio, escopo, arquitetura, tempo, custo e equipe.

A definição das métricas para realização de um comparativo entre o processo de software anterior e o novo processo foi baseada nos fatores que mais estavam impactando negativamente nos projetos, como entregas no prazo, satisfação do cliente e quantidade de *bugs*.

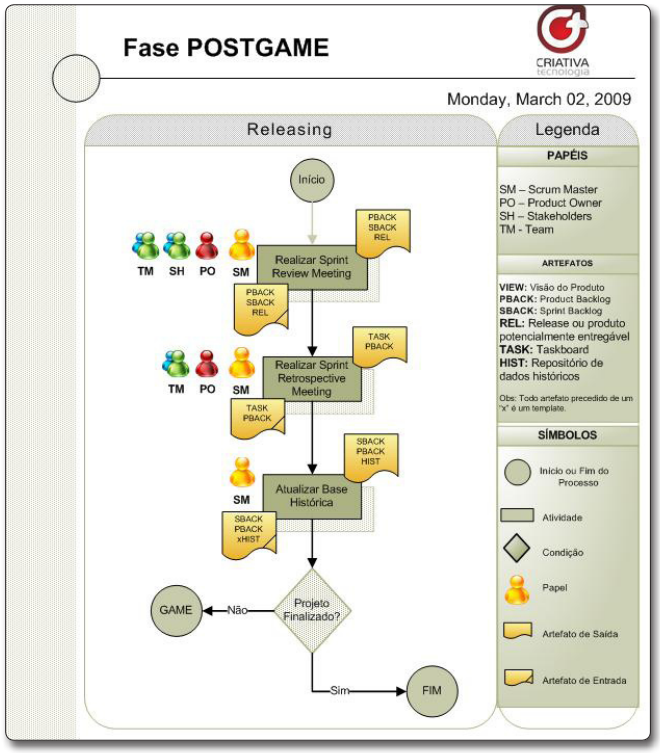


Figura 7. Fase POSTGAME

Como resultado da avaliação da métrica **entregas no prazo**, foi possível observar que, apesar do time não ter conseguido completar 100% das tarefas que foram atribuídas à iteração, pode-se observar um aumento de 42,5% de entregas no prazo.

Para medir a **satisfação dos clientes**, foi aplicado um questionário em relação a diversos aspectos, entre eles: comunicação, qualidade do produto, gerenciamento de mudanças, entregas no prazo, dentre outros. Para avaliar o processo foram utilizados três indicadores ("**Atende Totalmente**", "**Atende Parcialmente**" e "**Não atende"**), sendo que para cada questão, o questionário disponibilizou orientações de como ela deveria ser avaliada. Foi possível observar que o projeto que utilizou o novo processo de desenvolvimento obteve aumento de 57,14% das respostas "atendem totalmente" no questionário, proporcionando uma grande redução nos valores para as respostas que atendem parcialmente (46,39%) e manutenção da ausência de respostas que não atendem o cliente.

Para medir a **quantidade de *Bugs***, foram coletados os dados na ferramenta *Bug Tracker* da empresa, e pôde-se observar que o Projeto que utilizou o novo processo teve um número menor de *Bugs* que o processo anterior (equivalente a 75%).

Através das métricas apresentadas anteriormente, pode-se observar que há indícios de sucesso no uso conjunto do modelo de maturidade MPS.BR com a metodologia ágil Scrum.

Alguns aspectos nos projetos que utilizaram o novo processo podem ser ressaltados:

• O quadro de tarefas (*Taskboard*) proporcionou uma grande visibilidade do andamento das tarefas;
• A participação da equipe nas decisões do projeto proporcionou comprometimento e motivação na realização das tarefas diárias;
• Através das reuniões diárias pôde-se monitorar a produção do time, promover *feedback* e remover impedimentos que poderiam atrasar o bom andamento do desenvolvimento da *Sprint*.

A medição de entregas no prazo proporcionou a visibilidade da performance da empresa no cumprimento dos prazos estabelecidos, aumentando a eficiência em projetos futuros devido ao ajuste realizado na capacidade de produção do time (*Velocity*) a cada novo ciclo iterativo do novo processo (*Sprint*).

A satisfação dos clientes é uma forma de identificar a qualidade do software que foi produzido, que por sua vez é auxiliada por um processo de desenvolvimento eficaz.

Ao identificar a satisfação dos clientes da empresa em questão, observa-se que após a implementação deste novo processo, eles estão apresentando um nível mais alto de satisfação comparado ao processo anterior, o que significa que a CRIATIVA obteve efeitos positivos no seu novo processo de desenvolvimento. No entanto, podem-se observar situações onde a empresa deve melhorar ainda mais a qualidade dos serviços prestados, tais como:
• **Melhoria na qualidade do produto:** Apesar da diminuição na quantidade de *bugs* na fase de desenvolvimento, durante os testes de aceitação foram registradas solicitações de mudanças;
• **Cumprimento dos prazos:** Apesar de ter melhorado a taxa de entrega, ainda pode ser observado falha no cumprimento dos prazos acordados.

A medição da quantidade de *bugs* na ferramenta *Bug Tracker* proporcionou grande visibilidade e acompanhamento da solução dos mesmos ao final de cada iteração dos projetos. Devido à participação constante do *Scrum Master*, proximidade dos membros do time e colaboração do cliente, o número de *bugs* diminuiu consideravelmente.

Conclusão

Este trabalho apresentou a extensão do framework Scrum para as áreas de processo de Gestão de Projetos e Gerenciamento de Requisitos contempladas no MPS.BR nível G.

No mapeamento entre o *Scrum* e o guia MPS.BR nível G, descobriu-se que ele não cobre totalmente os resultados esperados do guia, mas pode ser um ponto de partida, pois proporciona uma base fundamental para empresas que estão iniciando a melhoria de processos, principalmente as que dispõem de poucos recursos como as micro e pequenas empresas.

Visto que após o mapeamento ainda restaram alguns resultados esperados do MPS.BR não cobertos plenamente pelas práticas do *Scrum*, outras alternativas foram pesquisadas e relacionadas na extensão proposta neste projeto para adaptação do processo de uma fábrica de software.

A extensão relacionada ao gerenciamento de requisitos através de abordagens ágeis foi uma forma alternativa e eficaz de gerenciar requisitos e solicitações de mudanças durante o desenvolvimento do projeto, evitando o retrabalho.

A extensão relacionada à gestão ágil de projetos foi uma alternativa capaz de gerar inovação, diminuição do tempo de entrega dos projetos, desenvolvimento de produtos de trabalho que agregam valor real para o cliente e resultados para a empresa.

O novo processo proposto demonstrou um avanço inicial no sucesso de um projeto de software desenvolvido pela empresa, mas como se encontra ainda em fase de implantação, o monitoramento se faz necessário para obtenção da melhoria contínua e obtenção de melhores resultados. ●

Referências

1. Adm, Advanced Development Methods. (2003). Scrum Methodology – Incremental, Iterative Software. Development from Agile Processes.

2. Beck, K., et al. (2001). "Manifesto for Agile Software Development." Manifesto for Agile Software Development. *http://www.agilemanifesto.org* (acesso em 22 de Março de 2008).

3. Beedle, M., and Schawaber, K. (2002). Agile Software Development With Scrum. New Jersey, Books.

4. Boehm, B. (2006). "A View of 20th and 21st Century Software Engineering." ICSE06. Shanghai, China, ACM.

5. Chin, G. (2004). Agile Project Management: How to Succeed in the Face of Changing Project Requirements. New York, Amacon.

6. Davis, C, et al. (2004). An Agile Approach to Achieving CMMI. *http://www.agiletek.com/images/AgileTek/pdf/an_agile_approach_to_achieving_cmmi.pdf* (acesso em 27 de Dezembro de 2008).

7. Gloger, B. (2008) "Scrum Glossary - Sprint It." Sprint It. http://sprint-runner.com (acesso em 15 de Junho de 2008).

8. Marçal, A. S. (2007). Estendendo o SCRUM segundo as Áreas de Processo de Gerenciamento de Projetos do CMMI. Recife.

9. Playfair, K (2008). "When 'General Agile' Isn't Enough - Why Scrum Wins in the Enterprise." Agile Journal. *http://www.agilejournal.com/content/view/808/111/* (acesso em 29 de Dezembro de 2008).

10. Schwaber, K. (2008). Agile Project Management With Scrum. Redmond, Microsoft Press.

11. Softex. MPS.BR (2007), Melhoria de Processo de Software Brasileiro - Guia Geral V1.2. Rio de Janeiro, Softex.

12. Standish, The Standish Group International. (2004).."The Chaos Report." Standish Group. secure.standishgroup.com/reports/reports.php?rid=500.

13. Szalvay, V. (2007). "Glossary of Scrum Terms." *http://www.scrumalliance.org/articles/39-glossary-of-scrum-terms#1121* (acesso em 25 de mAIO de 2008).

14. Szimanski, F. (2006). Implementando MPS.BR nível G na melhoria do processo de software em uma pequena empresa. Simpósio Mineiro de Sistemas de Informação. Lavras, UFLA.

15. Szimanski, F. (2009). Extensão do Scrum segundo as áreas de processo do MPS.BR nível G. Dissertação de mestrado. Recife, CESAR.