UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Camila Bastos

Fellipe Guilherme Rey de Souza

ENACTMENT DE PROCESSOS DE DESENVOLVIMENTO DE *SOFTWARE* MODELADOS COM EPF-*COMPOSER* NO MICROSOFT PROJECT

Alfenas, 3 de fevereiro de 2014.

UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ENACTMENT DE PROCESSOS DE DESENVOLVIMENTO DE *SOFTWARE* MODELADOS COM EPF-*COMPOSER* NO MICROSOFT PROJECT

Camila Bastos

Fellipe Guilherme Rey de Souza

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Alfenas como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador(a): Rodrigo Martins Pagliares

Alfenas, 3 de fevereiro de 2014.

Camila Bastos
Fellipe Guilherme Rey de Souza

ENACTMENT DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE MODELADOS COM EPF-COMPOSER NO MICROSOFT PROJECT

A Banca examinadora abaixo-assinada aprova a monografia apresentada como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal de Alfenas.

Prof. Douglas Donizeti de Castilho Braz

Universidade Federal de Alfenas

Prof. Eduardo Gomes Salgado

Universidade Federal de Alfenas

Prof. Rodrigo Martins Pagliares (Orientador)

Universidade Federal de Alfenas

Alfenas, 3 de fevereiro de 2014.

Dedico o meu trabalho a toda minha família, em especial, aos meus pais que foram meus grandes parceiros nessa longa caminhada.

Camila Bastos

Dedico esta obra aos meus pais, que me ofereceram seus incessantes cuidados durante toda a minha vida, sempre me guiando em minhas dificuldades e, principalmente, me dando forças para seguir em frente mesmo quando não tinha vontade ou motivação.Esta obra foi feita pensando inteiramente em vocês e seus esforços para que minha graduação se tornasse possível.

Fellipe Guilherme Rey de Souza

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me dado saúde, força, coragem e fé para enfrentar essa longa caminhada e também por me dar pessoas que sempre me apoiaram e enfrentaram essa jornada de quatro anos ao meu lado.

Agradeço aos meus pais, por sempre me apoiarem. Minha irmã pela ajuda e pelos seus conselhos. Meu noivo por ser sempre compreensivo e por ter me dado apoio e carinho quando mais precisava. Agradeço também meus avós pela torcida e orações.

Aos meus professores deixo a gratidão por me trazer conhecimento e me guiar durante toda a graduação. Agradeço também ao meu orientador Rodrigo Martins Pagliares pelo apoio no desenvolvimento deste trabalho.

Camila Bastos

Agradeço à Deus, por ter me dado o dom da vida. Aos meus pais, que sempre estiveram ao meu lado durante os momentos de dificuldade. Aos meus amigos, que compartilharam comigo os quatro melhores anos de minha vida. À universidade, principalmente ao corpo docente do curso de Bacharelado em Ciência da Computação, agradeço-os pela oportunidade de aprender, e muito, com vocês. Ao professor Rodrigo Martins Pagliares, agradeço pela dedicação, confiança e por todo o aprendizado durante o período em que fui orientado.

Fellipe Guilherme Rey de Souza

RESUMO

A demanda pela praticidade e agilidade proporcionada pelos *software*s aumenta significativamente a cada ano. O *software* está sendo introduzido nas mais diversas áreas de atuação, como por exemplo, a agricultura, pecuária, medicina, educação e muitas outras. Independente do ramo de atuação, as pessoas buscam nos *software*s um auxilio para executar suas tarefas de maneira rápida e fácil e, muitas vezes, tornam-se dependentes das praticidades oferecidas pelos mesmos. Como consequência da intensa utilização de *software*s, houve um aumento na demanda por produção de programas de computador que atendam as necessidades e expectativas da sociedade. A complexidade de alguns programas é elevada, exigindo que a equipe de desenvolvimento seja organizada e objetiva e ao mesmo tempo, ágil, capaz de cumprir prazos e atingir metas. Para adquirir tais características, muitas vezes as equipes recorrem a métodos e processos que as auxiliem. Os processos de desenvolvimento de *software* auxiliam na condução do trabalho para o produto final. O resultado de um processo é um produto que reflete a maneira ao qual ele foi modelado. Um processo modelado necessita de ser executado para que possa ser transformado em um produto. A ferramenta EPF-*Composer* (*Eclipse Process Framework Composer*) é utilizada para modelagem de processos e baseia-se nos conceitos do metamodelo SPEM (*Software Process Enginnering Metamodel*). De acordo com a especificação do SPEM, um processo modelado nesta ferramenta pode ser exportado para outras ferramentas, como o MS-*Project* (*Microsoft Project*), para realização do *enactment.* Outros autores afirmam que esta abordagem possui algumas deficiências. Utilizando a ferramenta EPF-*Composer* para modelar o processo *Scrum*, foi realizado uma pesquisa-ação, onde foi investigada a realização do *enactment* de processos com a ferramenta MS-*Project* em um ambiente de desenvolvimento de sistemas.

Palavras-Chave: *Enactment*, EPF-*Composer*, MS-Project, Scrum, SPEM.

ABSTRACT

The demand for convenience and flexibility provided by Software increase significantly each year. The Software is being introduced in several areas, such as agriculture, livestock, medicine, education and many others. Regardless of the field of expertise, people seek an aid in Software to perform tasks quickly and easily and often become dependent on conveniences offered by them. As a result of intensive use of Software, there was an increase in demand for production of computer programs that meet the needs and expectations of society. The complexity of some programs is high, requiring the development team being organized and objective and at the same time, agile, able to meet deadlines and achieve goals. To acquire such characteristics often teams resort to methods and processes that help them. The processes of Software development help in the execution of the work for the final product. The result of a process is a product that reflects the manner to which it was modeled. A modelled process needs to be enacted so that it can be turned into a product. The EPF-Composer (Eclipse Process Framework Composer) tool is used for process modeling and is based on the concepts of the metamodel SPEM (Software Process Metamodel Enginnering). According to the SPEM specification, a process modeled in this tool can be exported to other tools, such as MS- Project (Microsoft Project), for enactment. Other authors argue that this approach has some shortcomings. Using the EPF-Composer tool for modeling the Scrum process, a action research, where we investigated the realization of the enactment of processes with MS-Project tool in an environment of systems development was conducted.

Keywords: Enactment, EPF *Composer*, MS-Project, SPEM.

LISTA DE FIGURAS

LISTA DE ABREVIAÇÕES

EAP	Estrutura Analítica de Projetos
EPF-*Composer*	*Eclipse Process Framework Composer*
HTML	*HyperText Markup Language*
MS-Project	*Microsoft Project*
PML	*Process Modeling Languague*
PRPPG	Pró-Reitoria de Pesquisa e Pós-Graduação
SPEM	*Software Process Engineering Metamodel*
UNIFAL-MG	Universidade Federal de Alfenas
UML	*Unified Modeling Language*
WBS	*Work Breakdown Structure*
XML	*eXtensive Markup Languague*

SUMÁRIO

1
Introdução

Um fator importante para a implementação de sistemas de *software* com agilidade e qualidade é a familiarização dos desenvolvedores com as tarefas básicas para o desenvolvimento, tais como a maneira de adquirir e gerenciar os requisitos, como executar testes e como implementar com qualidade. (OMG, 2008*)*. Tais familiaridades são adquiridas com um bom planejamento de todas as atividades que devem ser desenvolvidas para o alcance do objetivo do projeto.

Os métodos de engenharia de *software* fornecem técnicas para construção de *software*. Tais métodos abrangem uma ampla gama de tarefas que incluem a análise de requisitos, *design*, construção de programas, teste e suporte. Esses métodos dependem de um conjunto de princípios básicos que regem cada área da tecnologia e incluem atividades de modelagem e outras técnicas descritivas. (PRESSMAN, 2010)

Um fluxo de atividades que pode ser citado para o desenvolvimento de um projeto é, primeiramente, a modelagem dos processos, mapeamento dos processos para um projeto, em seguida complementá-los para atender as necessidades específicas e por último, efetuar a sua execução e *enactment*.

Para modelagem dos processos, podem ser utilizados os conceitos do SPEM, que é um conjunto de construtores e regras para criação de modelos que podem ser utilizados para descrever e modelar um processo concreto ou uma família de processos de desenvolvimento de *software* relacionados. O SPEM utiliza mecanismos de extensão da semântica padrão da UML (*Unified Modeling Language*) e não possui a execução do processo em seu escopo. (ABDALA, 2006)

O EPF-*Composer* pode ser utilizado no processo de modelagem fornecendo uma estrutura de ferramentas baseadas nos conceitos do SPEM para definição, modelagem e gerenciamento de processos de desenvolvimento de *software* (OMG, 2008).

Depois de modelados, os processos necessitam de serem complementados com as informações e necessidades específicas para alcançar um objetivo, para em seguida, serem executados. Tais complementações como inserção de tarefas e alocação de recursos, podem ser efetuadas utilizando o MS-*Project* e quando concluídas, o projeto está pronto para ser executado.

1.1 Justificativa e Motivação

Um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo (PMBOK, 2013). Todo projeto é conduzido por pessoas dentro de parâmetros preestabelecidos de tempo, trabalho, custos e qualidade e utilizam processos para serem construídos (BUENO, 2011).

Os processos de desenvolvimento de *software* fornecem inúmeros benefícios para quem os utilizam. Através da definição ou especificação dos artefatos que devem ser desenvolvidos, os processos servem como uma orientação para as diversas atividades, recursos e elementos de uma organização.

O SPEM é um metamodelo bastante utilizado para a modelagem de processos. Entretanto, são necessários mais estudos e pesquisas em áreas relacionadas com o *enactment* dos mesmos (BENDRAOU *et al*, 2007).

O SPEM sugere duas abordagens para a realização do *enactment* de processos de *software* (OMG, 2008). Uma delas é o mapeamento de um WBS para um plano de projeto através de um mecanismo chamado instanciação. Com essa estratégia, podem ser utilizadas ferramentas como o MS-*Project* ou IBM *Rational Portfolio Manager* para efetuar a execução dos processos.

A segunda abordagem é a criação de um vínculo do modelo estático SPEM com modelos de comportamento externos através do pacote *Process Behavior* apresentado pelo SPEM.

1.2 Problematização

Apesar de o SPEM sugerir duas abordagens para a realização do *enactment* (descritas na seção 1.1), ambas possuem algumas deficiências e não cumprem todos os requisitos necessários para realização do mesmo (PORTELA *et al*, 2012). Alguns dos requisitos faltantes são: atualizações automáticas de associações entre tarefas e os seus papéis, controle automático do estado dos artefatos depois de cada atividade, entre outros.

1.3 Objetivos

1.3.1 Gerais

Considerando a primeira abordagem sugerida pelo SPEM, onde a realização do *enactment* dos processos pode ser feita utilizando a ferramenta MS-*Project* e que essa estratégia possui algumas deficiências, o objetivo principal deste estudo é investigar como realizar o *enactment* de processos de *software* modelados no EPF-*Composer* no MS-*Project*.

Será realizado uma pesquisa-ação na PRPPG (Pró-Reitoria de Pesquisa e Pós-Graduação) da UNIFAL-MG (Universidade Federal de Alfenas), onde será inserida a utilização da ferramenta MS-*Project* para o *enactment* dos processos de desenvolvimento de *software*.

1.3.2 Específicos

Os objetivos específicos são:

- Estudar modelagem de processos.
- Estudar a ferramenta EPF-*Composer*.
- Estudar a ferramenta MS-*Project*.

- Encontrar uma definição para *enactment* de processos de *software*.
- Realizar o *enactment* no MS-*Project*.
- Aplicar os conhecimentos adquiridos na PRPPG.

1.4 Organização da Monografia

O Capítulo 2 apresenta alguns fundamentos e teorias ao qual este trabalho foi embasado. O Capítulo 3 apresenta a descrição do local onde foi feito a pesquisa-ação bem como os métodos utilizados para atingir os objetivos desejados. O Capítulo 4 apresenta as conclusões obtidas após a realização da pesquisa-ação e, por último, o Capítulo 5 apresenta as revisões bibliográficas.

2
Fundamentação Teórica

Este capítulo apresenta as principais teorias que foram utilizadas para elaboração deste trabalho. Ele relata os principais conceitos e definições de cada assunto abordado.

2.1 Considerações Gerais

Enactment pode ser definido como execução dos processos de *software*, entretanto, a definição exata desse termo ainda é desconhecida na Engenharia de *Software*. Para Reis (2003, p.17), "*enactment* de processos significa que o mesmo não será totalmente automatizado como sugere o termo execução, mas sim, executado por pessoas e máquinas".

O EPF-*Composer* é uma ferramenta utilizada para modelagem de processos e permite criação, documentação e implantação de processos em diversos formatos (HAUMER, 2006). Essa ferramenta baseia-se no metamodelo SPEM e não possui o *enactment* em seu escopo, sendo necessária a utilização de outra ferramenta (OMG, 2008).

O MS-*Project* é um *software* gerenciador de projetos que permite a criação de cronogramas, gerenciamento de tarefas, custos, trabalho e recursos e o principal, controla e gerencia o projeto durante a sua execução, oferecendo assim suporte para *enactment* dos processos (BUENO, 2011).

Instanciação do processo consiste em gerar uma estrutura analítica de um projeto a partir da estrutura analítica do processo correspondente. Usando o EPF-*Composer*, a instanciação é parcialmente automatizada, gerando-se um arquivo XML que pode ser importado para uma ferramenta de gestão de projetos, como o MS-*Project*. Apenas as atividades, tarefas e marcos marcadas como planejados e não suprimidos são exportados. A instanciação é completada manualmente (PAULA FILHO, 2009):

- Atividades marcadas como repetíveis ou com múltiplas ocorrências são replicadas, adotando-se nomes que descrevem corretamente cada instância;
- Atividades marcadas como opcionais podem ser eliminadas, se não aplicáveis ao projeto específico;
- Atividades e tarefas não previstas no processo, mas específicas do projeto, podem ser acrescentadas.

2.2 Microsoft Project

O MS-*Project* é um *software* desenvolvido pela *Microsoft* para gerenciamento de projetos. Ele auxilia na organização do plano de trabalho do projeto, agendamento de metas que devem ser alcançadas, agendamento de tarefas nas sequencias corretas, alocação de recursos e custos, realização de uma sintonia no plano de trabalho satisfazendo o orçamento e preparação de relatórios explicativos para clientes, gerentes, trabalhadores e fornecedores (LÓPEZ, 2008).

A utilização do MS-*Project* cresceu enormemente com o passar dos anos em virtude de vários fatores, dentre os quais se destacam: fácil assimilação pelos usuários; presença de poderosas e avançadas ferramentas de planejamento; organização e controle de projetos; capacidade de gerenciamento de simples cronogramas até projetos múltiplos, complexos e extensos (BUENO, 2011).

2.2.1 História e evolução do MS-Project

Até meados da década de 1990, as primeiras versões do MS-*Project* eram focadas na produtividade individual dos gerentes de projetos, pois naquela época, nem todos os usuários estavam conectados eletronicamente. Todos os processos de planejamento, organização e controle eram realizados na estação de trabalho dos usuários de forma isolada e *offline*. Com o passar dos anos, a interação entre a equipe através do MS-*Project* foi aumentando de acordo com que suas versões foram evoluindo e o acesso à internet foi aumentando (BUENO, 2011).

A família de produtos do MS-Project inclui o MS-*Project Standard*, que é indicado para usuários que desejam gerenciar seus projetos sem utilizar as funcionalidades empresarias. Existe o MS-*Project Professional*, que possui todas as funcionalidades do *Standard* e é recomendado utilizá-lo juntamente com o MS-*Project Server* para obter a vantagem da colaboração (BUENO, 2011).

O MS-*Project Server* é o servidor da *Web* que provê os serviços de gestão empresarial dos projetos. Depois de criados no MS-*Project Professional*, os projetos são publicados no MS-*Project Server*. O MS-*Project Web Access* é uma interface da *Web* que permite o acesso às informações publicadas na versão *Server* (BUENO, 2011).

2.2.2 Características de um arquivo do MS-Project

Um arquivo de projeto gerado pelo MS-*Project* possui a extensão .mpp e contém diversos elementos. Dentre eles, podem ser citados:

- Campos: armazenam informações específicas sobre o projeto. Como exemplo, podem ser citados os campos Nome, Duração, Início e Término.
- Tabelas: mostram dados específicos do projeto no formato de linhas e colunas. Entrada, Custo e Trabalho são exemplos de tabelas do MS-*Project*.
- Modos de exibição: exibem os dados específicos do projeto por meio de gráficos, planilhas, formulários e campos. Um único modo de exibição pode conter, simultaneamente, um ou mais desses elementos.
- Formulários: detalham dados específicos do projeto, como campos que exibem informações sobre os recursos do projeto.
- Relatórios: agrupam e classificam informações de tarefas, recursos, atribuições de trabalho, custos, calendários etc. São exibidos no formato de visualização de impressão.

- Calendários: personalizam a agenda do projeto, determinando o período de trabalho dos recursos e quando as tarefas devem ser executadas.
- Filtros: exibem dados específicos do projeto, a partir de critérios predefinidos, ocultando as informações que não atendem a esses critérios.
- Grupos: tem a finalidade de categorizar e reportar dados do projeto. Dessa forma, eles reúnem e organizam tais dados a partir de critérios pré-definidos.
- Mapas: rastreiam campos do projeto para possibilitar a exportação e a importação em outros formatos.
- Módulos VBA: utilizam a linguagem de programação *Visual Basic for Applications* e automatizam rotinas e procedimentos do projeto.

As tarefas no MS-*Project* são criadas e organizadas de acordo com a EAP (Estrutura Analítica do Projeto) ou WBS, e assim, define-se hierarquias e dependências lógicas entre elas. É importante a definição de prazos e restrições para minimizar ou evitar atrasos no projeto (BUENO, 2011).

No MS-*Project*, é possível visualizar informações relacionadas ao cronograma em uma representação de Gráfico de *Gantt* (BUENO, 2011). Um Gráfico de *Gantt* é um gráfico de barras típico onde as atividades do cronograma ou componentes da estrutura analítica do trabalho são listadas verticalmente no lado esquerdo do gráfico, as datas são mostradas horizontalmente na parte superior e as durações das atividades são exibidas como barras horizontais posicionadas de acordo com as datas (PMBOK, 2013). A Figura 1 mostra um exemplo de estrutura analítica do trabalho juntamente com o gráfico de *Gantt* correspondente.

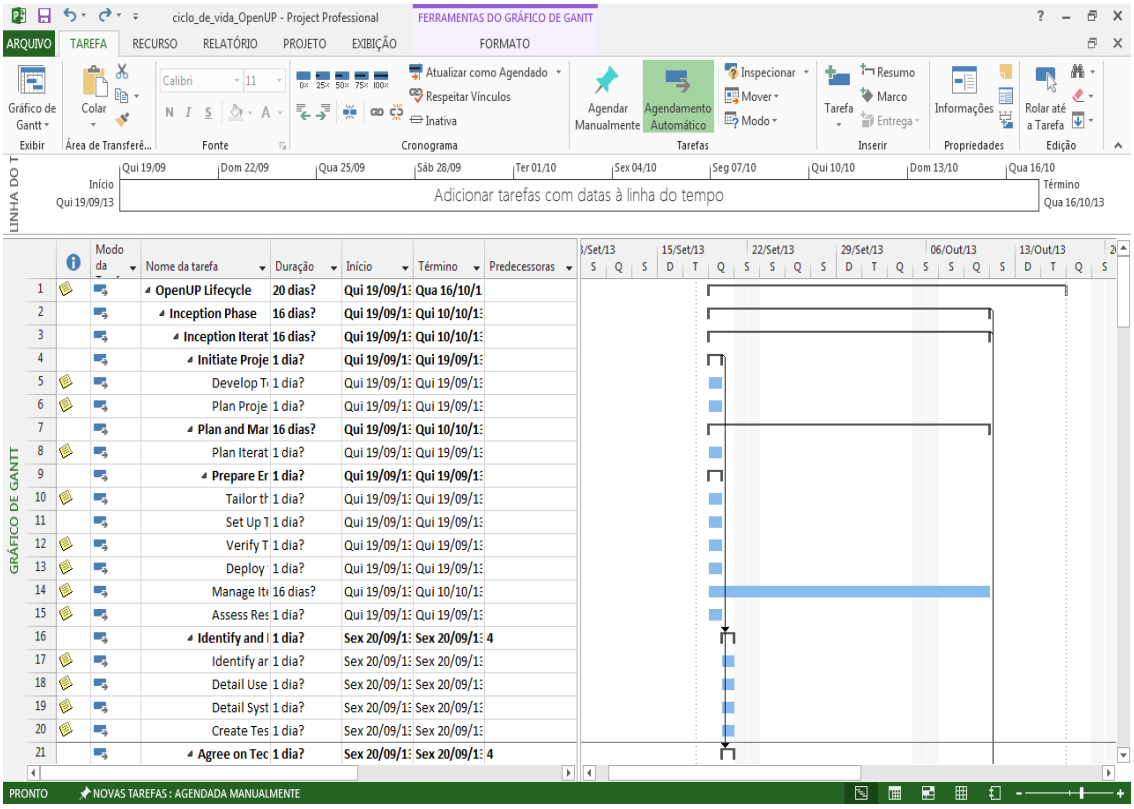
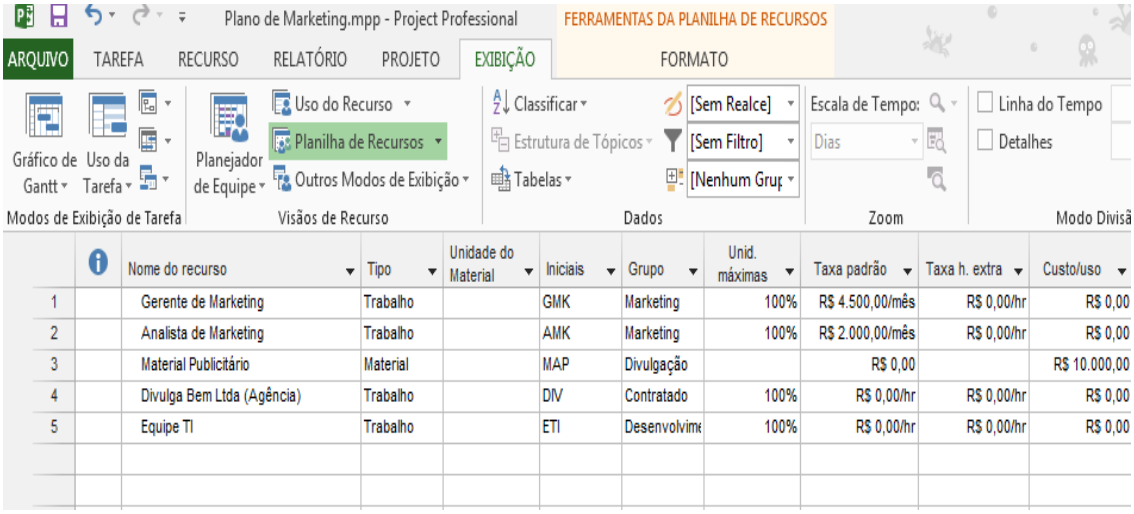


Figura 1 – Exemplo de WBS e Gráfico de *Gantt* no MS Project

Por meio do modo de exibição Planilha de Recursos, definem-se os recursos necessários para o projeto, levantando seus custos e disponibilidade. Depois de definidos todos os recursos, eles devem ser relacionados, cada um com a sua respectiva tarefa (BUENO, 2011). A Figura 2 mostra um exemplo de uma planilha de recursos.



	Nome do recurso	Tipo	Unidade do Material	Iniciais	Grupo	Unid. máximas	Taxa padrão	Taxa h. extra	Custo/uso
1	Gerente de Marketing	Trabalho		GMK	Marketing	100%	R$ 4.500,00/mês	R$ 0,00/hr	R$ 0,00
2	Analista de Marketing	Trabalho		AMK	Marketing	100%	R$ 2.000,00/mês	R$ 0,00/hr	R$ 0,00
3	Material Publicitário	Material		MAP	Divulgação		R$ 0,00		R$ 10.000,00
4	Divulga Bem Ltda (Agência)	Trabalho		DIV	Contratado	100%	R$ 0,00/hr	R$ 0,00/hr	R$ 0,00
5	Equipe TI	Trabalho		ETI	Desenvolvim	100%	R$ 0,00/hr	R$ 0,00/hr	R$ 0,00

Figura 2 – Planilha de Recursos no MS Project

Existe a possibilidade de gerenciamento e consolidação de múltiplos projetos em uma única janela. Para tal, pode-se definir um projeto com sendo mestre e os demais, como subprojetos. Para gerenciamento desses projetos, é recomendada a criação de um único arquivo com os recursos e que o mesmo, seja compartilhado entre todos os projetos. No MS-*Project*, esse arquivo é chamado de *pool* de recursos e permite o compartilhamento dos recursos por meio de agendamento de trabalho em diversos projetos, identificação de conflitos entre as atribuições e visualização de seu uso de maneira centralizada e consistente (BUENO, 2011).

O MS-*Project* oferece suporte para gerenciamento de recursos humanos. É possível fazer análise de carga de trabalho do projeto, abordando o conceito de superalocações de recursos, nivelamento de recursos humanos, análise gráfica de diversos itens do projeto, impressão e importação de relatórios de atribuição, planejador de equipes e inspetor de tarefas (BUENO, 2011).

2.3 SPEM

O SPEM é um metamodelo proposto pela OMG para descrição de um processo ou, uma família relacionada de processos de desenvolvimento de *software*.

Diferentemente da maioria das PMLs (Linguagens de Modelagem de Processos), o SPEM apareceu como uma proposta de unificação entre as diferentes metodologias existentes para modelagem de processos e utiliza a orientação a objeto para modelar uma família relacionada de processos de *software* utilizando a UML como notação. (GENVIGIR, 2003)

O SPEM é utilizado na definição de processos de desenvolvimento de *software* e seus componentes. Seu escopo é limitado somente aos elementos necessários para definição do processo sem adição de funcionalidades específicas. Seu objetivo é acomodar uma grande variedade de métodos e processos de diferentes estilos, origens e níveis de formalismo. A versão mais recente é o SPEM 2.0, que utiliza outras especificações da OMG e reutiliza a estrutura da UML 2. (OMG, 2008)

O SPEM 2.0 está estruturado em sete pacotes principais de meta modelos, onde o modelo é dividido em unidades lógicas. A Figura 3 mostra a estrutura dos pacotes do SPEM 2.0.

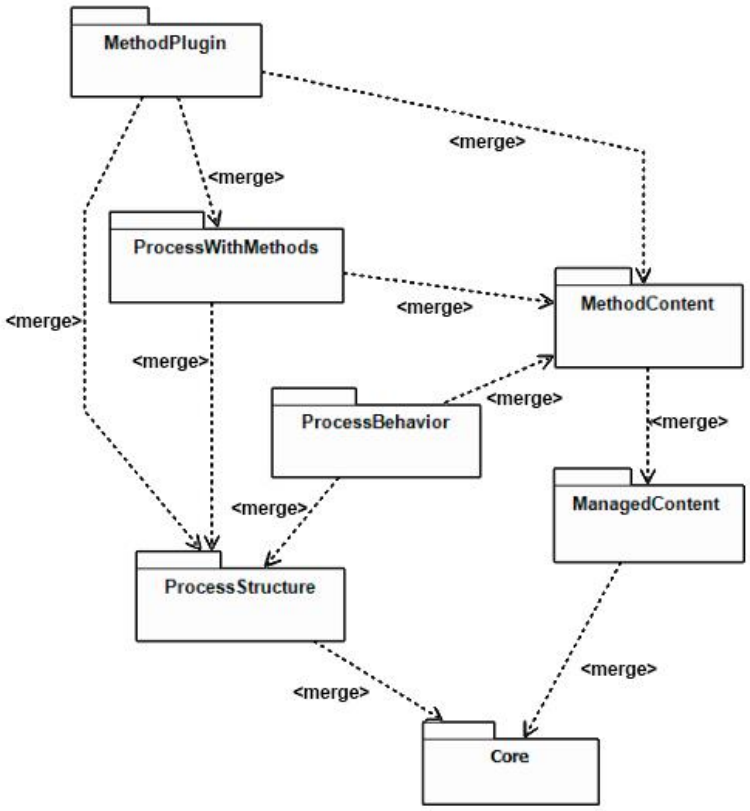


Figura 3 – Estrutura do SPEM 2.0 (OMG, 2008)

Os sete pacotes que formam o SPEM 2.0 possuem os seguintes recursos:

- *Core*: o pacote *Core* contém as classes de metamodelo e abstrações que constroem a base para as classes de todos os outros pacotes, ou seja, todas as classes comuns entre todos os níveis definidos no SPEM 2.0 foram colocadas no pacote *Core.*
- *Process Structure:* Este pacote define a base para todos os modelos de processo, que suporta a criação destes de forma simples e flexível. Ele provê mecanismos para o reuso de processos tais como a ligação de padrões de processos que permitem aos usuários montá-los como um conjunto de atividades dinamicamente encadeadas. Seus conceitos representam o processo como sendo uma estrutura de divisão estática, permitindo a definição de dependências entre atividades.
- *Process Behavior*: permite estender as estruturas do *Process Structure* com modelos comportamentais. Não define uma abordagem própria de comportamento, porém oferece *links* para modelos de comportamento externamente definidos, assim, permite a reutilização de abordagens a partir de outras especificações OMG ou de terceiros.
- *Managed Content*: muitas abordagens assumem que uma documentação descrita em uma linguagem natural de processos de desenvolvimento, são mais interessantes do que uma documentação formal. Atualmente, muitas equipes que utilizam o desenvolvimento ágil, assumem que os processos de desenvolvimento não podem ser formalizados com modelos, mas sim, documentados utilizando uma linguagem natural. O pacote *Managed Content,* introduziu conceitos para gerenciamento de do conteúdo das documentações em linguagens naturais. Tais conceitos podem ser usados de maneira independente ou em conjunto com outros conceitos da estrutura do processo.
- *Method Content*: fornece conceitos para construção de uma base de desenvolvimento que é independente de quaisquer processos de projeto específicos do SPEM 2.0. Acrescenta conceitos para definição do ciclo de vida e elementos que fornecem uma base de conhecimento

documentada de métodos de desenvolvimento de *software*, técnicas e melhores práticas. O conteúdo do *Method Content* compreende explicações textuais descrevendo como os objetivos específicos de desenvolvimento são alcançados. Um usuário SPEM 2.0 pode definir um *Method Content* como sendo uma orientação geral e construir uma base de métodos de conhecimento de desenvolvimento sem criar um processo, mas acrescentando uma estrutura ao seu conteúdo, assim como está previsto no *Managed Content.*

- *Process With Methods* : define e redefine estruturas para integração de processos definidos com *Process Structure* utilizando instancias do *Method Content.*
- *Method Plugin:* Introduz conceitos de *designing*, bibliotecas ou repositórios de processos e *method contents* e reusabilidade em grande escala. Estas definições permitem organizar diferentes partes de bibliotecas com base em diferentes camadas, bastante semelhante à arquitetura de *Software* em camadas.

2.3.1 Eclipse Process Framework *Composer*

O EPF-*Composer* é uma plataforma livre de código aberto que utiliza o metamodelo SPEM e permite a criação, adaptação, documentação, implantação de desenvolvimento de processos em vários formatos e permite que sejam feitas publicações no formato HTML (*HyperText Markup Language*) (HAUMER, 2006). Os dois principais objetivos do EPF-*Composer* (Figura 4) são:

- Fornecimento de uma representação de um sistema normalizado e bibliotecas de gerenciamento reutilizáveis que fornecem para os desenvolvedores entender métodos e principais práticas para desenvolvimento de *software*. O conteúdo gerenciado pode ser publicado no formato HTML e implantado em servidores *Web* para uso distribuído. (HAUMER, 2006)
- Fornecimento de suporte para gestão de processos de desenvolvimento. As equipes de desenvolvimento precisam definir as

melhores práticas ao longo do ciclo de vida de um projeto. Métodos de gestão devem ser aplicados durante as fases iniciais de um projeto, onde o foco é nas necessidades e requisitos das partes interessadas. Tais métodos devem ser realizados de maneira diferente durante as fases posteriores, onde o foco está na gestão de atualizações de requisitos, mudanças e os impactos causados por essas mudanças. O EPF-*Composer* auxilia os engenheiros e gerentes na aplicação dos métodos de gestão, tendo como resultado final, um catálogo de processos pré-definidos para situações típicas (HAUMER, 2006). A Figura 4 mostra um framework conceitual para o uso de uma implementação do SPEM.

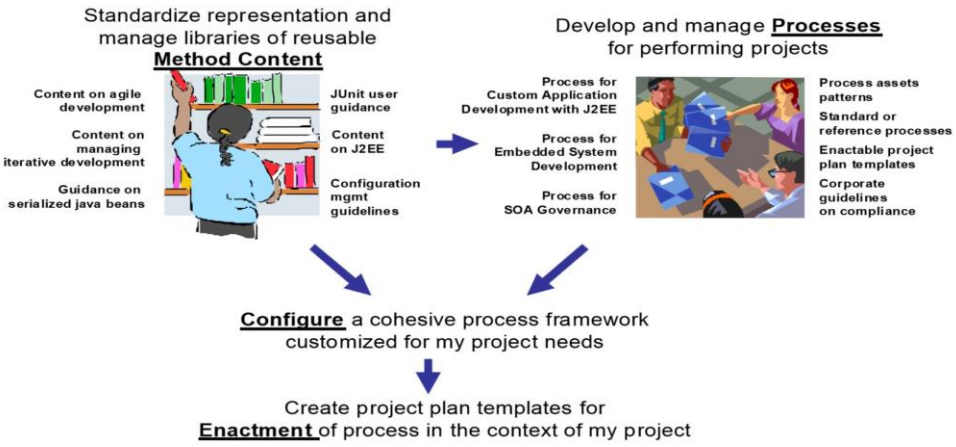


Figura 4 – EPFC fornece ferramentas para gerenciar bibliotecas de desenvolvimento de conteúdo e usá-los para montar processos para projetos específicos (OMG, 2008).

2.4 Processos de Desenvolvimento de *Software*

Um processo é um conjunto de ações e atividades inter-relacionadas realizadas para obter um conjunto pré-especificado de produtos, resultados ou serviços (PMBOK, 2013). O IEEE define processo como uma sequencia de passos executada para atingir um objetivo (IEEE, 2003).

Em outras palavras, um processo é uma receita que é seguida por um projeto. Um projeto concretiza uma abstração, que é o processo. Um processo é

definido quando existe uma documentação que o detalha: o que é feito (produto), quando (passos), por quem (agentes), as coisas que usa (insumos) e as coisas que produz (resultados) (PAULA FILHO, 2009).

Na engenharia de *Software*, os processos podem ser definidos para atividades como desenvolvimento, manutenção, aquisição e contratação de *Software*. Subprocessos podem ser definidos para cada um dos processos (PAULA FILHO, 2009). Ian Sommerville define um processo de *software* como um conjunto de atividades que leva à produção de um produto de *software* (SOMMERVILLE, 2007).

Existem vários processos de desenvolvimento de *software*, entretanto, algumas atividades fundamentais são comuns a todos (SOMMERVILLE, 2007).

- Especificação: define a funcionalidade do *software* e suas restrições.
- Projeto e implementação: o *software* que atenda a especificação deve ser produzido.
- Validação de *software*: o *software* deve ser validado para garantir que ela faça o que o cliente deseja.
- Evolução: o *software* deve evoluir para atender aos novos requisitos que naturalmente surgirão.

2.5 Métodos ágeis

Atualmente grandes empresas de desenvolvimento de *software* vêm adotando a metodologia de desenvolvimento ágil para gerenciamento de seus projetos. Tal metodologia visa um maior desempenho e mais agilidade na entrega de seus produtos.

Com o objetivo de definir uma melhor maneira para o desenvolvimento de sistemas, em 2011, um grupo com 17 metodologistas formaram uma Aliança para o Desenvolvimento Ágil que resultou em um manifesto contendo um conjunto de

princípios que definem critérios para os processos de desenvolvimento ágil. (FAGUNDES, 2008). Os princípios são (BECK, et. al. 2001):

- A prioridade é satisfazer ao cliente através de entregas contínuas e frequentes;
- Receber bem as mudanças de requisitos, mesmo em uma fase avançada do projeto;
- Entregas com frequência, sempre na menor escala de tempo.
- As equipes de negócio e de desenvolvimento devem trabalhar juntas diariamente;
- Manter uma equipe motivada fornecendo ambiente, apoio e confiança necessários;
- A maneira mais eficiente da informação circular através de uma conversa face a face;
- Ter o sistema funcionando é a melhor medida de progresso;
- Processos ágeis promovem o desenvolvimento sustentável;
- Atenção contínua a excelência técnica e a um bom projeto aumentam a agilidade;
- Simplicidade é essencial;
- As melhores arquiteturas, requisitos e projetos provêm de equipes organizadas;
- Em intervalos regulares, a equipe deve refletir sobre como se tornar mais eficaz.

2.5.1 SCRUM

É um método ágil iterativo e incremental para gerenciamento de um projeto, compatível com os padrões CMMI e PMBOK, concebido no ano de 1993 na Easel Corporation por Jeff McKenna, Jeff Sutherland e John Scumniotales e formalizado dois anos depois por Ken Schwaber. Assim como outros métodos ágeis, o Scrum já estava em estudo desde os anos 80.

Utilizando práticas iterativas e incrementais previstas no XP (*eXtreme Programming*), o *Scrum* utiliza a abordagem em espiral, fortemente focada em objetivos. A partir de uma lista de tarefas, serão desenvolvidas pequenas funcionalidades de tal forma que, ao chegar no final do *Sprint*, deverá ser entregue um produto potencialmente implantável.

As iterações presentes no *Scrum* são chamadas de *Sprints*. Cada uma possui duração de no mínimo uma semana e no máximo quatro, sendo que, obrigatoriamente, todos os *Sprints* possuirão o mesmo *time-box* (SCHWABER, 2004). Estas iterações são exercidas por alguns papeis e possuirão artefatos e reuniões que serão explicados nas seções seguintes. A Figura 5 ilustra as atividades realizadas durante uma *Sprint*.

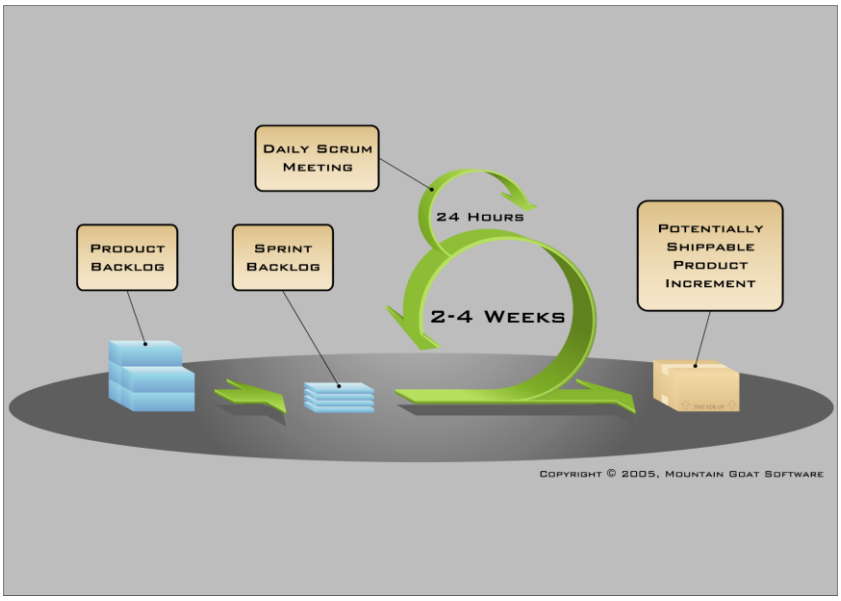


Figura 5 – *Sprint* (MOUNTAIN GOAT SOFTWARE, 2005)

2.5.1.1 Papéis

2.5.1.1.1 *Product Owner*

O *Product Owner* é o responsável pelo ROI (*Return On Investiment)* do projeto, o que significa que ele escolhe as funcionalidades do produto que serão desenvolvidas para resolução dos problemas críticos do negócio (SCHWABER, 2004).

Sendo o responsável por assegurar os interesses da empresa contratante e dos *stakeholders* envolvidos, o *Product Owner* tem como função gerenciar e priorizar o *Product Backlog*. Além disto, também define as funcionalidades do produto e aceita ou rejeita o resultado do trabalho da equipe.

2.5.1.1.2 *Scrum Master*

O *Scrum Master* é o responsável por garantir que as regras do *Scrum* estejam sendo cumpridas durante o desenvolvimento do projeto, com o intuito de garantir o bem-estar, boa produtividade da equipe e desenvolvimento de produtos de maior qualidade (SCHWABER, 2004).

Uma das principais funções deste papel é a de proteção do time, não permitindo que qualquer tipo de impedimento atrapalhe o desenvolvimento do produto e prejudique a equipe de finalizar o *Sprint*.

2.5.1.1.3 Time

A equipe é formada por um grupo de pessoas que variam de 5 a 9 membros, responsáveis por documentar, analisar, projetar, desenvolver e testar um produto que deverá ser entregue ao final da *Sprint* para o *Product Owner* e *stakeholders*.

São os responsáveis por transformarem o *Product Backlog* em incrementos e funcionalidades potencialmente entregáveis (SWCHABER, 2004). É recomendável que o time seja dividido em pares. Cada dupla poderá alocar para si uma tarefa que fora anteriormente detalhada no *Sprint Backlog*. O time do SCRUM deverá ser auto organizável e multifuncional e os pares deverão se esforçar para conseguirem atingir o objetivo do *Sprint*.

2.5.1.2 Artefatos

2.5.1.2.1 *Product Backlog*

Os requisitos do sistema ou produto a ser desenvolvido pelo projeto está listado no *Product Backlog* (SCHWABER, 2004). Este é uma lista de itens contendo todos os requisitos do sistema de maneira priorizada. É gerenciado pelo *Product Owner*, que também pode alterá-lo a qualquer momento, desde que as alterações não interfiram em alguma tarefa que foi detalhada no *Sprint Backlog*.

O *Product Backlog* é formado por itens, conhecidos como *features* (QUSEF, 2010). As *features* podem ser estórias de usuário ou casos de uso (SCHWABER, 2004). Os itens deste artefato devem ser detalhados o suficiente para que possam ser entendidos e estimados pela equipe. Não são aconselhadas *features* que sejam maiores que 2 semanas ou muito pequenas.

2.5.1.2.2 *Sprint Backlog*

Os itens do *Product Backlog* considerados pelo *Product Owner* como prioridade serão selecionados para que possam ser detalhados e desenvolvidos durante o *Sprint* que será realizado.

O número de itens a serem detalhados não é fixo, sendo dependente da quantidade de esforço de cada tarefa, da velocidade da equipe e do tamanho de cada iteração. O *Sprint Backlog* é uma figura muito evidente e em tempo real do trabalho que o time do Scrum deseja realizar durante o *Sprint* (SWCHABER, 2004).

2.5.1.2.3 *Burndown Chart*

É um gráfico utilizado para mostrar o andamento do projeto, onde o time poderá avaliar o seu progresso (SCHWABER, 2004). Possui dois eixos, sendo o das abcissas representando a duração da *Sprint* e o das ordenadas representando os pontos de história das funcionalidades do *Backlog*. O gráfico deve ser atualizado diariamente de acordo com o número de pontos de história vencidos pelo time no dia.

No gráfico, é traçada uma linha reta decrescente, tendo em um dos extremos o número máximo de *features* (primeiro dia) e no outro nenhum ponto de estória. Esta linha serve para guiar o time: Se o gráfico desenhado estiver abaixo da linha, quer dizer que a equipe está cumprindo o prazo com velocidade acima da planejada. Caso estiver acima da linha, o projeto está atrasado e a tendência é estourar o prazo. A Figura 6 ilustra um exemplo de um *Burndown Chart* no final da *Sprint*.

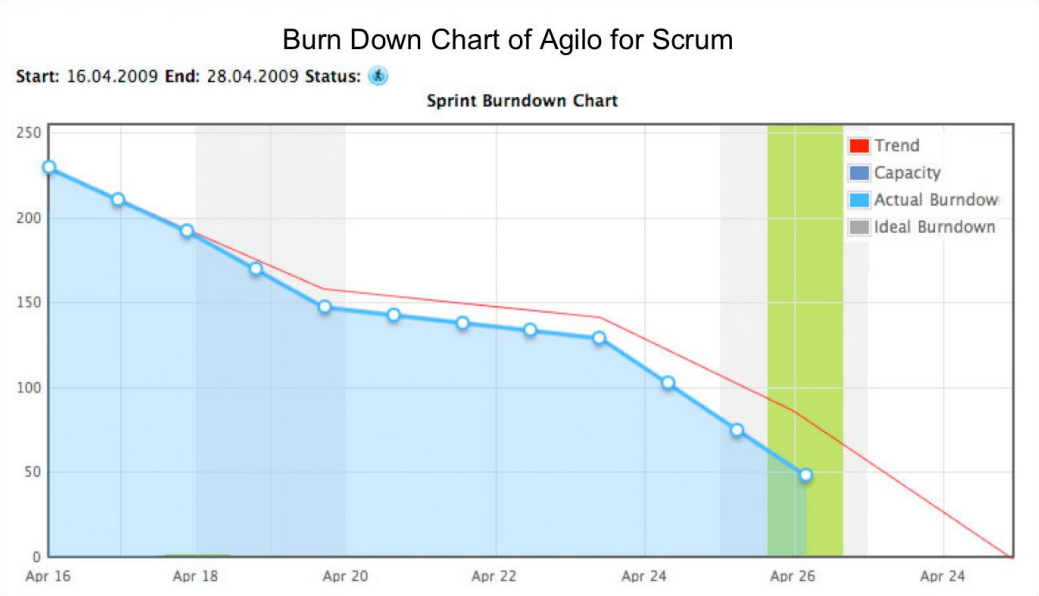


Figura 6 – *Burndown Chart* (EICKMANN, 2009)

2.5.1.3 Reuniões

2.5.1.3.1 Reunião de Planejamento do *Sprint*

A *Sprint* se inicia com um planejamento (*Sprint Planning Meeting*) que é dividido em duas partes, sendo a primeira a criação do *Product Backlog* priorizado pelo *Product Owner* e a segunda, a definição do *Sprint Backlog* pelo time e *Scrum Master*. Ambos artefatos foram anteriormente explicados nas seções 2.5.1.2.1 e 2.5.1.2.2.

Esta reunião serve para que o time planeje o que irão desenvolver no próximo *Sprint*. A estimativa dos itens do *Backlog* são feitas através de uma técnica chamada *planning poker*, onde os membros do time estimam o tempo em que trabalharão em uma tarefa usando cartas contendo os números de um a oito da sequencia de *Fibonacci*.

Para um *Sprint* de um mês, a Reunião de Planejamento é fixada para ocorrer em oito horas, porém, para intervalos mais curtos, é definido aproximadamente 5% do tamanho total da *Sprint* para a reunião (SCHWABER, 2004).

2.5.1.3.2 Reunião Diária

Uma reunião diária (*Daily Scrum Meeting*) deverá acontecer todos os dias, aconselhavelmente no mesmo horário. A reunião é realizada com a presença do *Scrum Master* e da Equipe, porém, qualquer pessoa pode participar da reunião.

Durante a reunião, deverão ser respondidas três perguntas básicas definidas por Ken Schwaber, sendo elas:

- O que fiz desde a última reunião diária?
- O que pretendo fazer hoje?
- Há algum problema impedindo meu trabalho?

É importante saber que a reunião não deverá ter duração superior a 15 minutos, para que não prejudique a produtividade da equipe.

2.5.1.3.3 Reunião de Revisão

Durante a reunião de revisão *(Sprint Review Meeting)*, todo o trabalho desenvolvido pela equipe deverá ser apresentado para o *Product Owner* para sua aprovação. Esta reunião ocorre no último dia da *Sprint* e o produto entregue pelo time deverá ser um *Software* potencialmente implantável.

A reunião possui duração de 4 horas ou não mais que aproximadamente 5% da duração da *Sprint* (SCHWABER, 2004) e cabe ao *Product Owner* aceitar ou

recusar o trabalho feito pela equipe. Trabalhos incompletos não deverão ser mostrados para validação.

2.5.1.3.4 Reunião de Retrospectiva

Com o intuito de melhorar a cada *Sprint*, a reunião de retrospectiva *(Sprint Retrospective)* tem como principal objetivo levar os membros do time a fazerem uma reflexão sobre a *Sprint* que se encerrou. Ela ocorre logo após a reunião de revisão.

Duas questões devem ser respondidas pela equipe:

- O que deu certo durante este *Sprint*?
- O que pode ser melhorado no próximo *Sprint*?

A duração desta reunião é fixa de 3 horas e cabe ao *Scrum Master* encorajar o time, revisar e identificar melhorias para serem implementadas na próxima *Sprint* (SCHWABER, 2004).

2.6 WBS para métodos ágeis

WBS é a decomposição hierárquica orientada à entrega do trabalho a ser executado pela equipe do projeto, para atingir os objetivos do projeto e criar as entregas necessárias (PMI, 2013). O WBS organiza as atividades necessárias para concluir o projeto e é utilizado para avaliação, programação e status do projeto.

WBS pode ser considerada semelhante a uma árvore genealógica, ou seja, possui uma estrutura de árvore que mostra a subdivisão dos componentes necessários para entregar um projeto (DENIS, 2010).

Um WBS ágil é organizado em torno de funcionalidades para o usuário final. Nele, os recursos são decompostos em *features*, as *features* em *epics* e os *epics* em funcionalidades que são implementadas dentro de uma iteração. *Features* podem ser adicionadas e removidas do WBS, desde que o somatório do trabalho ainda possa ser implementado dentro da *Sprint* (DENIS, 2010). A Figura 7 mostra a hierarquia de um WBS ágil.

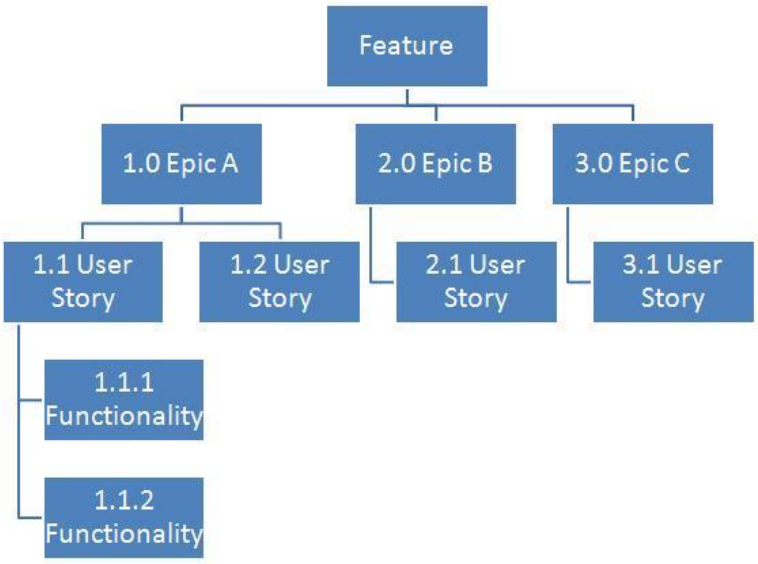


Figura 7 – WBS de Métodos Ágeis (DENIS, 2010)

A prática de definição, verificação de escopo e criação de WBS no método ágil é feita a cada iteração (VIANA, 2008). No gerenciamento tradicional existe uma preocupação em definir todo o escopo detalhadamente no início do projeto, formalizando-o com o WBS (VIANA, 2008).

3
Pesquisa-ação

Este capítulo descreve a maneira com a qual uma equipe de desenvolvimento de sistemas trabalhava com o Scrum antes da inserção das ferramentas propostas neste estudo bem como a metodologia utilizada para realizar o enactment de processos de Software.

Visando a grande demanda por equipes de desenvolvimento ágeis e que forneçam produtos de alta qualidade, este estudo buscou investigar a definição e realização do *enactment* no MS-Project praticando com uma equipe de desenvolvimento de *Software*. A equipe em questão utiliza o *Scrum* como metodologia de gerenciamento.

O método utilizado para alcance dos objetivos foi a realização de uma pesquisa-ação, que é uma maneira se fazer uma pesquisa em situações onde o pesquisador faz parte da realização da prática. Esse método procura unir a pesquisa à ação, isto é, desenvolver o conhecimento e a compreensão como parte da prática. (ENGEL, 2000)

Para realização da pesquisa-ação, foi proposta para equipe a introdução da ferramenta EPF-*Composer* para modelagem dos processos e a ferramenta MS-Project para realizar o *enactment* dos mesmos.

3.1 Visão Geral

A PRPPG é o órgão responsável pela coordenação e supervisão das atividades de pesquisa e pós-graduação da UNIFAL-MG. Com o aumento do conteúdo de informações para gerenciamento, houve um aumento considerável no trabalho realizado para execução das tarefas. Sendo assim, a PRPPG formou uma equipe de

desenvolvimento de *software* para implementação de sistemas que auxiliam na realização de suas tarefas.

Atualmente, a PRPPG possui uma equipe constituída de 6 estagiários para desenvolvimento e manutenção de seus sistemas. Os sistemas desenvolvidos pela equipe são:

- Perfil: sistema responsável por integrar todas as informações de pesquisadores e alunos que desenvolvem atividades de pesquisa ou colaborações com a UNIFAL-MG. Este sistema permite a identificação de usuários em todos os demais sistemas da PRPPG.
- PSP (Plataforma de Submissão de Projetos): foi o primeiro sistema a ser desenvolvido pela equipe e é responsável pela submissão e avaliação de projetos de iniciação científica da UNIFAL-MG. Esse sistema foi implantado em abril de 2011 e, desde então, vem sendo adaptado às necessidades da Instituição e de seus docentes, passando por atualizações periódicas.
- Linha Temática: este sistema visa o gerenciamento das atividades de pesquisa dos professores da UNIFAL-MG por períodos regulares.
- SGE (Sistema de Gerenciamento de Equipamentos): permite a realização de agendamentos por pesquisadores para a utilização de equipamentos multiusuários ou de solicitações de análise de amostras realizadas por equipamentos.

3.2 O *Scrum* na PRPPG

A equipe de desenvolvimento utiliza o *Scrum* como metodologia de gerenciamento dos projetos. Atualmente, não está sendo desenvolvido nenhum sistema novo e as principais tarefas feitas pelo *Scrum Team* são de manutenção e acréscimo de novas funcionalidades aos sistemas já existentes. A maneira com a qual a equipe trabalha com o *Scrum* será descrita nas próximas seções.

3.2.1 Planejamento do *Product Backlog*

A *Product Owner* é responsável por elaborar o *Product Backlog* de acordo com as suas necessidades. Em um único *Product Backlog* ela coloca as demandas de todos os sistemas que, na maioria das vezes, são escritas como histórias de usuário.

A equipe possui acesso ao *Product Backlog* através de uma planilha *online* que é compartilhada com todos os membros.

3.2.2 Reunião de Planejamento de *Sprint*

Cada *Sprint* realizada é iniciada com a reunião de *Sprint Planning*, que possui duração de quatro horas. Nessa reunião, o *Scrum Team* trabalha para entender o *Product Backlog* e, além disso, definir qual e como o trabalho deverá ser feito. A equipe de desenvolvimento escolhe a quantidade de itens do *Product Backlog* que entrarão na *Sprint* de acordo com a sua velocidade de trabalho.

No final da reunião, o artefato gerado é o *Sprint Backlog*. Este artefato contém os itens selecionados do *Product Backlog* divididos em tarefas e estimados. Cada tarefa é estimada em horas e o *Sprint Backlog* é compartilhado com todos os membros da equipe em uma planilha *online*. Com o *Sprint Backlog* definido, o time de desenvolvimento começa a trabalhar na implementação das tarefas.

3.2.3 Reunião Diária

Diariamente, às dez horas e cinquenta minutos da manhã, é realizada uma reunião com o objetivo de assegurar que toda a equipe esteja trabalhando de maneira a atingir os objetivos da *Sprint* e que nenhum fator esteja impedindo o seu trabalho. Caso exista algum empecilho, a *Scrum Master* trabalha para removê-lo o mais rápido possível.

3.2.4 Reunião de Revisão de *Sprint*

No final da *Sprint* é realizada uma reunião com duração média de três horas para que a *Product Owner* possa validar o que foi feito durante a *Sprint*. Com base no que

foi feito e aprovado, a *Product Owner* toma as decisões sobre o que deseja na próxima *Sprint* e atualiza o *Product Backlog*.

3.2.5 Reunião de Retrospectiva de *Sprint*

Após o término da reunião de revisão de *Sprint*, é realizada outra reunião para levantamento de aspectos ocorridos durante a *Sprint* que acarretaram em benefícios ou malefícios para a evolução do projeto.

3.3 Nova Metodologia de desenvolvimento da PRPPG

3.3.1 Criação da WBS no EPF-*Composer*

Através do EPF-*Composer* é possível criar um WBS que posteriormente será exportado para o MS-*Project*. Para criar o WBS do *Scrum*, não é necessário criar um processo específico, visto que o site do EPF-*Composer* fornece uma biblioteca prática com o processo modelado**.** Para começar a desenvolver o WBS, primeiramente é necessário a criação de um novo *Delivery Process*, que é feita conforme a Figura 8.

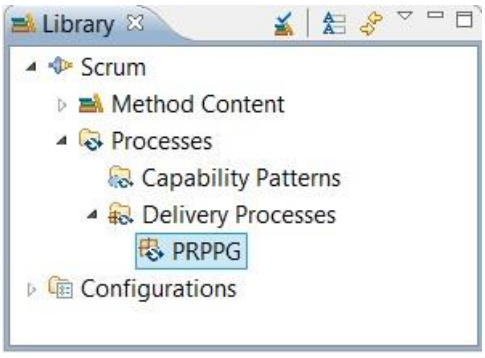


Figura 8 – Criação do *Delivery Process*

No WBS do *Delivery Process* poderão ser criados cinco diferentes tipos de *childs*, sendo eles *Phase*, *Iteration*, *Task Descriptor* e *Milestone*. Considerando que o *Scrum* é um método iterativo incremental, serão criados apenas *iterations* na WBS.

No processo de criação do WBS, primeiramente foi realizada uma reunião na PRPPG para estimar todo o *Product Backlog* e assim, descobrir a duração total do desenvolvimento. Após a estimativa, somaram-se todos os pontos de história resultando em um total de 2996. Através de *Sprints* anteriores, sabe-se que a velocidade da equipe é 47.

Como resultado da reunião, o time irá desenvolver todo o projeto em 63 *Sprints* (2996 pontos de história divididos pela velocidade da equipe, que é 47). Este valor será correspondente ao número de iterações que terá o *Delivery Process*, ou seja, o número total de *Sprints* do projeto (este número pode variar para cima ou baixo de acordo com o erro de estimativa da equipe). A Figura 9 ilustra o WBS com as *Sprints* criadas.

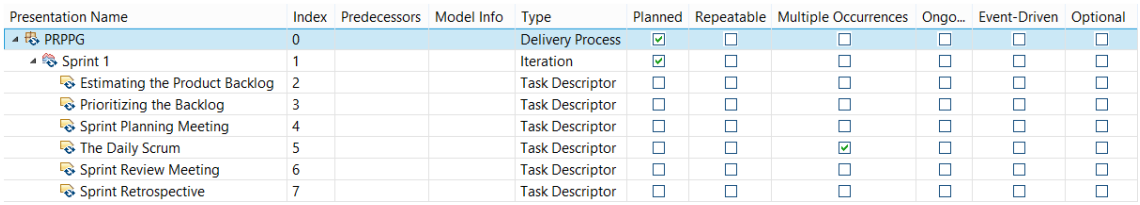
Presentation Na...	Index	Predecessors	Model Info	Type	Planned	Repeatable	Multiple Occurrences	Ongo...	Event-Driven	Optional
PRPPG	0			Delivery Process	☑	☐	☐	☐	☐	☐
Sprint 1	1			Iteration	☑	☐	☐	☐	☐	☐
Sprint 2	8			Iteration	☑	☐	☐	☐	☐	☐
Sprint 3	15			Iteration	☑	☐	☐	☐	☐	☐
Sprint 4	22			Iteration	☑	☐	☐	☐	☐	☐
Sprint 5	29			Iteration	☑	☐	☐	☐	☐	☐
Sprint 6	36			Iteration	☑	☐	☐	☐	☐	☐
Sprint 7	43			Iteration	☑	☐	☐	☐	☐	☐
Sprint 8	51			Iteration	☑	☐	☐	☐	☐	☐
Sprint 9	59			Iteration	☑	☐	☐	☐	☐	☐
Sprint 10	67			Iteration	☑	☐	☐	☐	☐	☐
Sprint 11	75			Iteration	☑	☐	☐	☐	☐	☐
Sprint 12	83			Iteration	☑	☐	☐	☐	☐	☐

Figura 9 – Criação das iterações no EPF-Composer

De acordo com a definição de Ken Schwaber em relação ao *Sprint*, todos possuem a mesma duração e seguem um mesmo ciclo. Portanto, todos irão possuir atividades básicas iguais, tais como: *Sprint Planning Meeting*, *Sprint Retrospective*, *Sprint Review Meeting*, *The Daily Scrum*, *Estimating the Product Backlog*, *Prioritizing the Backlog* e *Release Planning*.

Todas as atividades de uma *Sprint* estão presentes na biblioteca prática do *Scrum*. Outros artefatos presentes no método ágil, tais como os resultados obtidos no final de cada *Sprint*, um produto potencialmente entregue, um *Product Backlog* atualizado, dentre outros, apesar de estarem presentes na biblioteca prática não

serão incluídos na WBS criada, visto que o MS-*Project* não possui suporte para estes artefatos. As atividades que foram incluídas dentro de uma *Sprint* do *Delivery Process* possuem descritos quais são os papéis que cada membro do *Scrum* realiza, não necessitando nenhum esforço adicional. A Figura 10 mostra a estrutura de uma *Sprint* criada no WBS do EPF-Composer.



Presentation Name	Index	Predecessors	Model Info	Type	Planned	Repeatable	Multiple Occurrences	Ongo...	Event-Driven	Optional
PRPPG	0			Delivery Process	☑	☐	☐	☐	☐	☐
Sprint 1	1			Iteration	☑	☐	☐	☐	☐	☐
Estimating the Product Backlog	2			Task Descriptor	☐	☐	☐	☐	☐	☐
Prioritizing the Backlog	3			Task Descriptor	☐	☐	☐	☐	☐	☐
Sprint Planning Meeting	4			Task Descriptor	☐	☐	☐	☐	☐	☐
The Daily Scrum	5			Task Descriptor	☐	☐	☑	☐	☐	☐
Sprint Review Meeting	6			Task Descriptor	☐	☐	☐	☐	☐	☐
Sprint Retrospective	7			Task Descriptor	☐	☐	☐	☐	☐	☐

Figura 10 – *Sprint* presente na WBS criada

Para finalizar o processo de criação da WBS, é necessário marcar a opção Multiple Occurences para o *The Daily Scrum*. Isto indica que o *The Daily Scrum* será instanciado mais de uma vez durante a duração total da *Sprint*. Os atributos restantes não precisam ser selecionados, já que não irão interferir no resultado final da WBS.

3.3.2 Exportação dos Processos para o MS-Project

Adotando a primeira abordagem sugerida pelo SPEM (descrita na seção 1.1), onde o *enactment* pode ser realizado pelo MS-*Project*, o processo do *Scrum* foi exportado para esta ferramenta. Assim, através da aba *File* do EPF-*Composer* e posteriormente, acessando a opção *Export*, é possível selecionar um destino para o arquivo que será exportado e, neste caso, será o MS-*Project*.

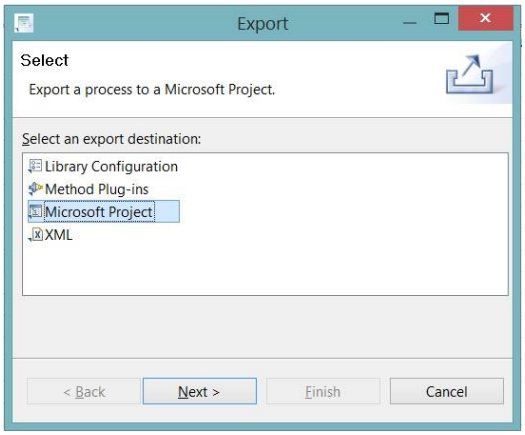


Figura 11 – Exportação do *Delivery Process* do EPF para o MS-Project.

Um guia genérico de exportação poderá ser encontrado na página 241 do Manual do Usuário do EPF-*Composer* (TUFT, 2010). Ao final do processo, será criado um arquivo no formado XML (*eXtensive Markup Languague*).

O arquivo XML gerado poderá ser importado no MS-*Project*. A importação é realizada ao abrir o arquivo XML na ferramenta. No final do processo de importação, um novo projeto será criado e complementado para atender os requisitos necessários da *Sprint*.

	Nome da tarefa	Duração	Início	Término	Predecessoras	Nomes dos recursos
1	**PRPPG**	**1 dia?**	**Qui 09/01/14**	**Sex 10/01/14**		
2	**Sprint 1**	**1 dia?**	**Qui 09/01/14**	**Sex 10/01/14**		
3	Estimating the Prod	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0]
4	Prioritizing the Back	1 dia?	Qui 09/01/14	Sex 10/01/14		
5	Sprint Planning Mee	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
6	The Daily Scrum	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Scr
7	Sprint Review Meet	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
8	Sprint Retrospective	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
9	**Sprint 2**	**1 dia?**	**Qui 09/01/14**	**Sex 10/01/14**		
10	Estimating the Prod	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0]
11	Prioritizing the Back	1 dia?	Qui 09/01/14	Sex 10/01/14		
12	Sprint Planning Mee	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
13	The Daily Scrum	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Scr
14	Sprint Review Meet	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
15	Sprint Retrospective	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
16	**Sprint 3**	**1 dia?**	**Qui 09/01/14**	**Sex 10/01/14**		
17	Estimating the Prod	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0]
18	Prioritizing the Back	1 dia?	Qui 09/01/14	Sex 10/01/14		
19	Sprint Planning Mee	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
20	The Daily Scrum	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Scr
21	Sprint Review Meet	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
22	Sprint Retrospective	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0];Prc
23	**Sprint 4**	**1 dia?**	**Qui 09/01/14**	**Sex 10/01/14**		
24	Estimating the Prod	0 dias?	Qui 09/01/14	Qui 09/01/14		Scrum Team[0]
25	Prioritizing the Back	1 dia?	Qui 09/01/14	Sex 10/01/14		

Figura 12 – Processo importado no MS-Project

Antes de utilizar esses processos na PRPPG, algumas adaptações foram realizadas. Os processos *Sprint Planning Meeting*, *Estimating the Product Backlog* e *Prioritizing the Backlog* foram adicionados no processo Planejamento de *Sprint*.

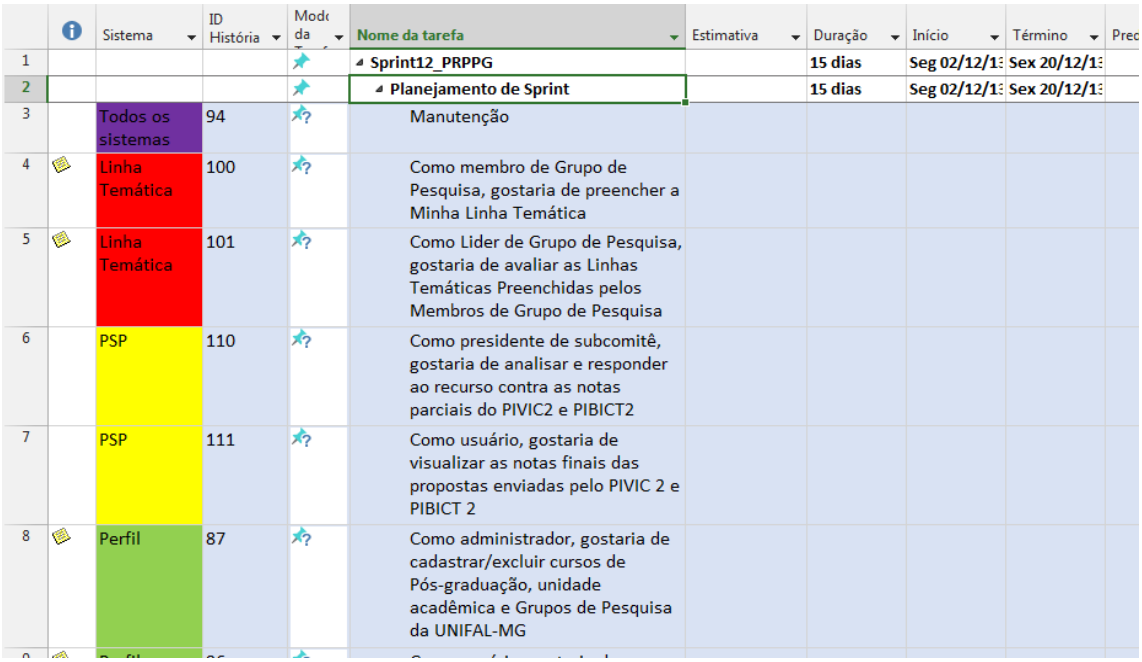
3.3.3 *Enactment* dos processos no MS-Project

Após a modelagem dos processos e exportação para o MS-*Project*, é necessário fazer o *enactment* dos mesmos para produzir o produto final. As próximas seções descrevem como esse procedimento foi realizado.

3.3.3.1 Planejamento do *Product Backlog* no MS-*Project*

A elaboração do *Product Backlog* passou a ser feita diretamente no MS-*Project*, dentro do tópico "Planejamento de *Sprint*" do WBS. A *Product Owner* escreve e prioriza os novos requisitos a serem implementados.

Algumas adaptações foram feitas para melhorar a organização do WBS. Foi criada uma coluna Sistema, onde foi colocado o nome do sistema referente ao item do WBS. Além disso, foi criada a coluna Estimativa onde foi armazenada a estimativa em pontos de história. A Figura 13 mostra uma parte do *Product Backlog* criado na primeira *Sprint*.



		Sistema	ID História	Mode da	Nome da tarefa	Estimativa	Duração	Início	Término	Prec
1					Sprint12_PRPPG		15 dias	Seg 02/12/1	Sex 20/12/1	
2					Planejamento de Sprint		15 dias	Seg 02/12/1	Sex 20/12/1	
3		Todos os sistemas	94		Manutenção					
4		Linha Temática	100		Como membro de Grupo de Pesquisa, gostaria de preencher a Minha Linha Temática					
5		Linha Temática	101		Como Lider de Grupo de Pesquisa, gostaria de avaliar as Linhas Temáticas Preenchidas pelos Membros de Grupo de Pesquisa					
6		PSP	110		Como presidente de subcomitê, gostaria de analisar e responder ao recurso contra as notas parciais do PIVIC2 e PIBICT2					
7		PSP	111		Como usuário, gostaria de visualizar as notas finais das propostas enviadas pelo PIVIC 2 e PIBICT 2					
8		Perfil	87		Como administrador, gostaria de cadastrar/excluir cursos de Pós-graduação, unidade acadêmica e Grupos de Pesquisa da UNIFAL-MG					

Figura 13 – Parte do *Product Backlog* da PRPPG

3.3.3.2 Estimando o *Product Backlog* no MS-*Project*

Com o *Product Backlog* pronto, na reunião de planejamento, a equipe começa a estimar os itens em pontos de história, dividir os itens em tarefas e em seguida, estimar as tarefas em horas. Esse processo resulta na *Sprint Backlog,* que está organizada em um WBS contendo as *features* que entraram na *Sprint* divididas em tarefas e estimadas. A Figura 14 mostra uma parte da *Sprint Backlog* gerada nessa reunião.

		Sistema	ID História	Modo da Tarefa	Nome da tarefa	Estimativa	Duração	Início	Término
4		**Linha Temática**	**100**		**Como membro de Grupo de Pesquisa, gostaria de preencher a Minha Linha Temática**	**1**			
5			100.1		Limitar todos os campos de texto de acordo com a quantidade de caracteres especificados acima do campo		1 hr		
6			100.2		Colocar uma menssagem de sucesso quando a linha tematica for ENVIADA OU SALVA		0,5 hrs		
7			100.3		Teste da história		0,5 hrs		
8		**Linha Temática**	**101**		**Como Lider de Grupo de Pesquisa, gostaria de avaliar as Linhas Temáticas Preenchidas pelos Membros de Grupo de Pesquisa**	**2**			
9			101.1		O título não deve exceder o espaço do layout.		1 hr		
10			101.2		Mostrar o botão enviar na tela de avaliação de uma linha temática		0,5 hrs		

Figura 14 – *Sprint Backlog* da PRPPG

Após a criação do *Sprint Backlog*, a implementação é iniciada. O controle de tarefas de *Sprint* também é feito no MS-*Project*. As próximas seções descrevem a maneira adotada para o controle das tarefas.

3.3.3.3 Alocação de tarefas pela equipe de desenvolvimento

Para controle das tarefas, é necessário inserir os membros da equipe na ferramenta. O MS-*Project* assume que para execução de um projeto, é necessária a alocação de recursos (BUENO, 2011). Com base nessa ideia, foi feito o controle da execução das tarefas, onde foi criado um recurso com o nome de cada integrante da equipe.

Quando uma pessoa fosse implementar uma tarefa, era necessário que ela alocasse o recurso, previamente cadastrado com o seu nome, para a tarefa em questão. Na WBS no MS-*Project* existe a coluna recurso, onde é possível visualizar quem está alocado para cada tarefa. A Figura 15 ilustra os recursos alocados para as tarefas da *Sprint*.

		Sistema	ID História	Mode da	Nome da tarefa	Estimativa	Duração	Nomes dos recursos	Início	Término
1					**Sprint12_PRPPG**		**15 dias**		**Seg 02/12/13**	**Sex 20/12/13**
2					**Planejamento de Sprint**		**15 dias**		**Seg 02/12/13**	**Sex 20/12/13**
3		Todos os sistemas	94		Manutenção	5				
4		**Linha Temática**	**100**		**Como membro de Grupo de Pesquisa, gostaria de preencher a Minha Linha Temática**	**1**				
5			100.1		Limitar todos os campos de texto de acordo com a quantidade de caracteres especificados acima do campo		1 hr	Nome do Desenvolvedor da Tarefa		
6			100.2		Colocar uma menssagem de sucesso quando a linha tematica for ENVIADA OU SALVA		0,5 hrs	Nome do Desenvolvedor da Tarefa		
7			100.3		Teste da história		0,5 hrs			

Figura 15 – Alocação de membros da equipe para as tarefas da *Sprint*.

3.3.3.4 Controle do status das tarefas

O MS-*Project* assume que o progresso de uma tarefa pode ser medido em porcentagens, assim uma tarefa poderá estar, por exemplo, 10% ou 60% pronta. Esse fato fere uma regra do *Scrum*, que assume que uma *feature* não está pronta ou está pronta totalmente, não existindo um meio termo.

Com isso, julgou-se necessário uma adaptação do controle de progresso de tarefas oferecido pelo MS-*Project*. Quando uma tarefa fosse alocada, ela ficaria no status de 0% pronta e, quando fosse totalmente finalizada, passaria para o 100%. Assim, a pessoa que alocasse a tarefa só alteraria seu status quando a mesma já fosse terminada. O status das tarefas podem assumir somente dois valores possíveis: 0% ou 100%.

3.3.3.5 Controle de duração e data de início/término das tarefas

O MS-*Project* apresenta em sua WBS três colunas que são: Duração, Início e Término. Quando uma tarefa é alocada, o membro da equipe preenche o campo Início e, quando termina, preenche o campo Término. Esse fato pode acarretar em um problema, conforme ilustrado na Figura 16, a estimativa em horas de cada

tarefa é colocada no campo duração. Caso uma tarefa seja iniciada em um dia de trabalho e finalizada em outro, ao preencher o campo Término, o MS-*Project* automaticamente preencherá o campo duração como sendo de dois dias, alterando a estimativa real da tarefa.

Para contornar esse problema, quando é alocada uma tarefa que será finalizada no dia seguinte, a mesma é quebrada em duas tarefas onde o tempo em horas de uma será cumprido em um dia, e o da outra, no dia seguinte. O somatório do tempo das duas novas tarefas deverá ser igual ao da original para não alterar no tempo total de estimativa da *Sprint*. A Figura 16 ilustra um caso onde foi necessário quebrar uma tarefa em duas. A tarefa com identificador 87.1 inicialmente era de 2 horas e 30 minutos. Ela foi alocada na quinta-feira no final do expediente do desenvolvedor e não foi possível ser finalizada no mesmo dia, restando meia hora para o dia seguinte.

Sistema	ID História	Modo da	Nome da tarefa	Estimativa	Duração	Início	Término	Nomes dos recursos
Perfil	**87**		**Como administrador, gostaria de cadastrar/excluir cursos de Pós-graduação, unidade acadêmica e Grupos de Pesquisa da UNIFAL-MG**	**2**	**5 dias**	**Qua 04/12/13**	**Ter 10/12/13**	
	87.1		Fazer tela cadastrar/desativar/ativar/alterar curso de pós-graduação - Parte 1		2 hrs	Qui 05/12/13	Qui 05/12/13	Nome do desenvolvedor da tarefa
	87.1		Fazer tela cadastrar/desativar/ativar/alterar curso de pós-graduação - Parte 2		0,5 hrs	Sex 06/12/13	Sex 06/12/13	Nome do desenvolvedor da tarefa
	87.2		Alterar tela para desativar/ativar curso de graduação		1,5 hrs	Qua 04/12/13	Qua 04/12/13	Nome do desenvolvedor da tarefa
	87.3		Fazer tela cadastrar/desativar/ativar/alterar unidade academica		2,5 hrs	Qui 05/12/13	Qui 05/12/13	Nome do desenvolvedor da tarefa
	87.4		Fazer tela cadastrar/desativar/ativar/alterar grupos de pesquisa		2,5 hrs	Ter 10/12/13	Ter 10/12/13	Nome do desenvolvedor da tarefa
	87.5		Pesquisar todas as listas de cursos de graduação, pós-graduação, unidade academica e grupos de pesquisa		1,5 hrs	Seg 09/12/13	Seg 09/12/13	Nome do desenvolvedor da tarefa

Figura 16 – Demonstração de tarefas que foram quebradas

3.3.3.6 Reunião Diária

No tópico Reunião Diária do WBS foram adicionadas tarefas recorrentes com duração de dez minutos para todos os dias de implementação da *Sprint*. Caso ocorra algum evento que influencie no trabalho da equipe, esse evento deverá ser descrito dentro do tópico equivalente ao dia ocorrido. Qualquer fato ocorrido no dia de implementação que possua relevância para o andamento da *Sprint* também foi relatado nesse tópico. A Figura 17 mostra uma anotação realizada no tópico Reunião Diária 1.

	Sistema	ID História	Nome da tarefa	Estimativa	Duração	Início	Término
55			**Reunião Diária**		**12 dias**	**Ter 03/12/13**	**Qui 19/12/13**
56			**Reunião Diária 1**		**10 mins**	**Ter 03/12/13**	**Ter 03/12/13**
57			Alteração da Sprint Backlog		10 mins	Ter 03/12/13	Ter 03/12/13
58			Reunião Diária 2		10 mins	Qua 04/12/13	Qua 04/12/13
59			**Reunião Diária 3**		**10 mins**	**Qui 05/12/13**	**Qui 05/12/13**
61			**Reunião Diária 4**		**10 mins**	**Sex 06/12/13**	**Sex 06/12/13**
63			Reunião Diária 5		10 mins	Seg 09/12/13	Seg 09/12/13
64			Reunião Diária 6		10 mins	Ter 10/12/13	Ter 10/12/13
65			Reunião Diária 7		10 mins	Qua 11/12/13	Qua 11/12/13
66			Reunião Diária 8		10 mins	Qui 12/12/13	Qui 12/12/13
67			Reunião Diária 9		10 mins	Sex 13/12/13	Sex 13/12/13
68			Reunião Diária 10		10 mins	Seg 16/12/13	Seg 16/12/13
69			Reunião Diária 11		10 mins	Ter 17/12/13	Ter 17/12/13
70			Reunião Diária 12		10 mins	Qua 18/12/13	Qua 18/12/13
71			Reunião Diária 13		10 mins	Qui 19/12/13	Qui 19/12/13

Figura 17 – WBS de reuniões diárias

3.3.3.7 Reunião de Revisão de *Sprint*

Depois de finalizado os dias de implementação, foi feito a reunião de revisão de *Sprint* com duração de duas horas. Foi apresentado para a *Product Owner* todo o trabalho desenvolvido pela equipe. O resultado obtido através dessa reunião foi relatado no tópico Reunião de Revisão de *Sprint*, conforme mostrado na Figura 18.

		Sistema	ID História	Modo da Tarefa	Nome da tarefa	Estimativa	Duração	Início	Término
72	✓				**Reunião de Revisão de Sprint**		**2 hrs**	**Sex 20/12/13**	**Sex 20/12/13**
73	✓				**Status das Histórias da Sprint**		**2 hrs**	**Sex 20/12/13**	**Sex 20/12/13**
74	✓				Revisar Historia 94		2 hrs	Sex 20/12/13	Sex 20/12/13
75	✓				Revisar Historia 100		2 hrs	Sex 20/12/13	Sex 20/12/13
76	✓				Revisar Historia 101		2 hrs	Sex 20/12/13	Sex 20/12/13
77	✓				Revisar Historia 110		2 hrs	Sex 20/12/13	Sex 20/12/13
78	✓				Revisar Historia 111		2 hrs	Sex 20/12/13	Sex 20/12/13
79	✓				Revisar Historia 87		2 hrs	Sex 20/12/13	Sex 20/12/13
80	✓				Revisar Historia 96		2 hrs	Sex 20/12/13	Sex 20/12/13
81	✓				Revisar Historia 97		2 hrs	Sex 20/12/13	Sex 20/12/13
82	✓				Revisar Historia 113		2 hrs	Sex 20/12/13	Sex 20/12/13
83	✓				Revisar Historia 114		2 hrs	Sex 20/12/13	Sex 20/12/13
84	✓				Revisar Historia 115		2 hrs	Sex 20/12/13	Sex 20/12/13
85	✓				Revisar Historia 116		2 hrs	Sex 20/12/13	Sex 20/12/13
86	✓				Revisar Historia 95		2 hrs	Sex 20/12/13	Sex 20/12/13

Figura 18 – WBS da Reunião de Revisão de *Sprint*

Para cada *feature* foi criada uma tarefa de revisão. As tarefas relacionadas com as *features* que não passaram no teste da *Product Owner* foram coloridas de vermelho e comentários com o motivo da reprovação foi adicionado.

3.3.3.8 Reunião de Retrospectiva de *Sprint*

Na Reunião de Retrospectiva de *Sprint*, foi realizado um levantamento dos pontos de história comprometidos pela equipe e os pontos de história que foram entregues e aprovados pela *Product Owner*. O total de pontos de história comprometidos foram 37 e foram entregues com aprovação somente 26.

4
Conclusão

Este capítulo apresenta as conclusões obtidas a partir da execução da pesquisa-ação juntamente com a equipe de desenvolvimento de sistemas.

Considerando a definição dada por Wilson de Pádua Paula Filho, onde ele afirma que instanciação de processo é a geração de um WBS do projeto a partir de um WBS do processo e que o mesmo deverá ser completado manualmente e, segundo a primeira abordagem sugerida pelo SPEM, onde a realização do *enactment* poderá ser feita através da instanciação, define-se *enactment* como sendo a parte não automatizada do processo, onde as características e informações específicas de um projeto são adicionadas.

Enactment poderá ser considerado a complementação realizada após a exportação de um processo para uma ferramenta de gerenciamento de projetos. A complementação pode ser considerada como a inserção de tarefas, requisitos e definição de datas e prazos de entregas que devem ser atendidos para atingir ao objetivo do projeto.

Na PRPPG, quando o processo foi importado para o MS-*Project*, o *enactment* foi iniciado com a criação do *Product Backlog*, onde os requisitos específicos do projeto foram adicionados no WBS e finalizado na criação do *Sprint Backlog*, onde informações sobre as estimativas são adicionadas. Com isso, nota-se que o *enactment* pode ser considerado finalizado quando as informações específicas do projeto terminam de serem adicionadas.

No *Scrum* o *enactment* pode ser realizado a qualquer momento, pois qualquer alteração feita no *Product Backlog* implica na adição de novas informações específicas do projeto. A execução é iniciada quando a equipe utiliza o WBS para controle de suas tarefas. A partir da execução é que os requisitos do projeto começam a se transformar em uma entrega.

A ferramenta MS-*Project* é excelente para gerenciamento de projetos que utilizam metodologias tradicionais de gerenciamento, entretanto, para métodos

ágeis o gerenciamento é dificultado devido à riqueza de detalhes oferecida pela ferramenta. As mudanças constantes que acontecem nos métodos ágeis tornam-se muito custosas de serem efetuadas na ferramenta MS-*Project*.

A maioria dos recursos oferecidos pelo MS-*Project* não foram utilizados na PRPPG devido a sua dificuldade de implantação no *Scrum*. Como exemplo, pode ser citado o gráfico de *Gantt* que apesar de ser muito utilizado por gerentes de projetos para inspecionar o andamento do projeto, na PRPPG ele não foi utilizado.

A criação do *Sprint Backlog* no formato de um WBS organiza as tarefas de cada *feature*, deixando a hierarquia e prioridades destas mais claras. Entretanto, efetuar o controle de cada uma das tarefas do *Sprint* diretamente no WBS não foi viável, visto que o controle de datas e duração de itens do *Backlog* pela equipe é mais custoso do que utilizar as maneiras tradicionais do *Scrum*.

Com este estudo foi possível observar que a realização do *enactment* em uma ferramenta de gerenciamento de projetos apresenta algumas dificuldades. Em trabalhos futuros, poderá ser investigado como realizar *enactment* de processos de desenvolvimento de software não ágeis. Com esta abordagem talvez seja possível aproveitar mais os recursos oferecidos pelo MS-*Project*, como a criação do *burndown chart*.

5
Referências Bibliográficas

ABDALA, M. A. D. **Modelagem do Processo de Gerenciamento da Configuração de *Software* para um Ambiente Integrado.** 2006

BECK, K. *et al*. **Agile Manifesto**. 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso em: 08/01/2014.

BENDRAOU, R**.** et al. **Definition of an Executable SPEM 2.0** *14th Asia-Pacific Software Engineering Conference.* **2007.**

BUENO, G. Pearson (Ed.) **Ms-project 2010 & gestão de projetos.** 2011.

DENIS. **WBS (Work Breakdown Structures): Everything you Need to Know.** 2010. Disponível em: < http://www.expertprogrammanagement.com/2010/03/wbs-work-breakdown-structures-everything-you-need-to-know/>. Acesso em: 16/01/2014

EICKMANN, M. **Team Spirit in Scrum.** 2009. Disponível em: <http://www.agile42.com/en/blog/2009/10/19/team-spirit-scrum/>. Acesso em: 09/02/2014.

ENGEL, G. I. **Pesquisa-Ação** - *Educar*, 2000

FAGUNDES, P. B; DETERS. J. I; Santos. S.S. **Comparação entre processos dos métodos ágeis: XP, SCRUM, FDD E ASD em relação ao desenvolvimento iterativo incremental** - *Revista E-Tech: Tecnologias para Competitividade Industrial - ISSN - 1983-1838.* 2008

FEILER, P. H.; HUMPHREY, W. S. ***Software* process development and enactment: Concepts and definitions. In: *Software* Process**, 1993. Continuous *Software* Process Improvement, Second International Conference on the IEEE, 1993.

GENVIGIR, E. C.; BORREGO FILHO, L.F.; SANT'ANNA. N. **Modelagem de Processos de *Software* Através do SPEM-*Software* Process Engineering Metamodel-Conceitos e Aplicação**. III WORCAP. 2003.

GOLANY, B.; SHTUB, A. **Work Breakdown Structure. Handbook of Industrial Engineering: Technology and Operations Management**. Third Edition. 1263-1280.

HAUMER, P. **The Eclipse Process Framework *Composer* - Increasing Development Knowledge with EPFC.** 2006

IEEE ***Software* Engeneering Collection on CD-ROM**. IEEE, New York, 2003.

LOPEZ, O. C. **Introdução ao Microsoft Project** – Apostila, UNISUL 2008

OMG, **SPEM. *Software* and Systems Process Engineering Meta-Model Specification**. 2008.

PAULA FILHO, W. P. **Engenharia de *Software*: Fundamentos, Métodos e Padrões**. 3ª edição. LTC, Rio de Janeiro. 2009.

PRESSMAN, R. S. ***Software* Engineering - A practitioner's approach** McGraw-Hill. 2010

PRESSMAN, R. S. **Engenharia de *Software***. 6ª edição. McGraw-Hill, 2006.

PMI. **A Guide to the Project Management Body of Knowledge: PMBOK® Guide**. Project Management Institute, 2013.

PYLE, A. **Agile WBS**. 2011. Disponível em: <http://agile-pm.pbworks.com/w/page/1526552/Agile%20WBS>. Acesso em: 08/01/2014.

REIS, C. A. L. **Uma Abordagem Flexível para Execução de Processos de *Software* Evolutivos**. 2003

RUIZ-RUBE, I. **Uses and Applications of SPEM Process Models. A Systematic Mapping Study Journal of *Software* maintenance and evolution: research and practice**. 2012.

PORTELA, C. et al**. xSPIDER_ML: Proposal of a *Software* Processes Enactment Language Compliant with SPEM 2.0**. Journal of *Software* Engineering and Applications, No prelo. 2012

QUSEF A.L.A. **Requirements engineering in agile *Software* development**. Journal of Emerging Technologies in Web Intelligence, 2010.

MONTAIN GOAT SOFTWARE. **Scrum Images.** 2005. Disponível em: <http://www.mountaingoatsoftware.com/agile/scrum/images/>. Acesso em: 19/12/2013.

SCHWABER, K. ***Agile project management with Scrum***. O'Reilly Media, Inc., 2004.

SOMMERVILLE, I. ***Engenharia de Software***, 8ª edição, Pearson Addison-Wesley, 2007.

TUFT, B. **Eclipse Process Framework (EPF) *Composer***. Installation, Introduction, Tutorial and Manual. Eclipse Public License V1.0, 2010.

VIANA, A. G. G. **Gerenciamento de projetos em processo ágil de desenvolvimento de *Software***. *Instituto de Educação Tecnológica–IETEC, Belo Horizonte.[2008]*. Disponível em: <http://www.techoje.com.br/site/techoje/artigos_autor/artigos/393>. Acesso em: 15/01/2014.