UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

AENDER DE SOUZA SATURNINO

PROPOSTA DE UM PROCESSO DE GERENCIAMENTO DE REQUISITOS PARA SATISFAÇÃO DA GERÊNCIA DE REQUISITOS DO NÍVEL G DO MPS.BR

CRICIÚMA

2014

AENDER DE SOUZA SATURNINO

PROPOSTA DE UM PROCESSO DE GERENCIAMENTO DE REQUISITOS PARA SATISFAÇÃO DA GERÊNCIA DE REQUISITOS DO NÍVEL G DO MPS.BR

Trabalho de Conclusão de Curso, apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientador: Prof. MSc. Paracelso de Oliveira Caldas

CRICIÚMA

2014

AENDER DE SOUZA SATURNINO

PROPOSTA DE UM PROCESSO DE GERENCIAMENTO DE REQUISITOS PARA SATISFAÇÃO DA GERÊNCIA DE REQUISITOS DO NÍVEL G DO MPS.BR

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia de Software.

Criciúma, 27 de novembro de 2014.

BANCA EXAMINADORA

Prof. Paracelso de Oliveira Caldas - MSc - (UNESC) - Orientador

Prof. Gustavo Bisognin - MSc - (UNESC)

Prof. Matheus Leandro Ferreira - Esp - (UNESC)

AGRADECIMENTOS

Agradeço inicialmente a Deus por ter me concedido a oportunidade de desfrutar os muitos sentidos e experiências, agradeço aos meus pais Yarita e Francisco que me deram a vida, me deram a base de tudo e sempre me motivaram a realizar esta importante etapa em minha vida.

Agradeço a minhas amadas irmãs que sempre sorrindo me motivam a seguir em frente, a minha esposa Sunamita, meu amor, meu porto seguro.

Agradeço ao meu orientador Paracelso que me ensinou muito e acreditou em meu potencial.

Meus amigos em geral, que de uma forma ou de outra fazem parte de mim.

“Eu segurei muitas coisas em minhas mãos, e eu perdi tudo; mas tudo que eu coloquei nas mãos de Deus eu ainda possuo.”

(Martin Luther King)

RESUMO

Com o crescimento do mercado de software aliado a tecnologia, a busca por constantes melhoria afim de oferecer qualidade é cada vez maior. A concepção da necessidade do cliente, a validação, a documentação e a gerência de requisitos são alguns dos processos defendidos pela Gerência de Requisitos. O MPS.BR através do seu nível mais baixo, o G, demonstra cinco resultados esperados para se garantir o mínimo necessário para que a instituição trate de maneira adequada os requisitos de seus projetos. Este trabalho traz um processo de Gerenciamento de Requisitos que busca atender estes resultados, focado principalmente a pequenas e médias empresas.

Palavras-chave: Engenharia de Requisitos, Gerência de Requisitos, Processo de Gerência de Requisitos, Melhoria de Processo de Software Brasileiro.

ABSTRACT

With the growth of the software market along with the technology, the search for constant improvement in order to offer quality is increasing. The design of the customer's need, validation, documentation and requirements management are some of the processes advocated by the Requirements Management. The MPS.BR through its lowest level, the G shows five results expected to ensure the minimum necessary for the institution to treat adequately the requirements of their projects. This work brings a Requirements Management process that seeks to meet these results, focused primarily on small and medium enterprises.

Keywords: Requirements Engineering, Requirements Management, Requirements Management Process, Brazilian Software Process Improvement.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

BID	Banco Interamericano de Desenvolvimento
CEDP	Conselho de Engenheiros para o Desenvolvimento Profissional
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
CMMI-SE/SW	Capability Maturity Model Integration for Systems Engineering/Software Engineering
ECPD	Engineers Council for Professional Development
ER	Engenharia de Requisitos
ERS	Especificação de Requisitos de Software
FAST	Facilitaded Application Specification Techniques
FINEP	Financiadora de Estudos e Projetos
GRE	Gerência de Requisitor
JAD	Joint Application Development
IBGE	Instituto Brasileiro de Geografia e Estatística
IBM	International Business Machines
IEC	International Electrotechnical Commission
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
IPD-CMM	Integrated Product Development Capability Maturity Model
ISO	International Organization for Standardization
MA-MPS	Método de Avaliação para Melhoria de Processo de Software
MCN	Modelo de Negócio Cooperado
MCT	Ministério da Ciência e Tecnologia
MN-MPS	Modelo de Negócio para Melhoria de Processo de Software
MNE	Modelo de Negócio Específico
MPS	Melhoria de Processo de Software
MPS.BR	Melhoria de Processo de Software Brasileiro
MR-MPS	Modelo de Referência para Melhoria do Processo de
NBR	Normas Brasileiras
PMBOK	Project Management Body of
RMM	Requirements Management Maturity
RUP	Rational Unified Process
SEBRAE	Serviço Brasileiro de Apoio às Micro e Pequenas Empresas
SEI	Software Engineering Institute
SOFTEX	Associação para Promoção da Excelência do Software Brasileiro
SW-CMM	Capability Maturity Model for Software
SWOT	Strengths, Weaknesses, Opportunities, Threat
TI	Tecnologia da Informação
UML	Unified Modeling Language
UP	Unified Process

SUMÁRIO

1 INTRODUÇÃO

O último relatório gerado pelo Standish Group em Março/2013 demonstrou que o percentual de falha em projetos de Tecnologia de Informação (TI), em 2012, a cada 100 projetos iniciados somente 39 atingiram o sucesso, 43 sofreram alterações ao longo do processo e 18 fracassaram. Comparando esses dados com os dados do ano 1994, percebemos uma melhora não muito significativa, que naquele ano a cada 100 projetos, apenas 16 atingiam ao sucesso, 53 sofriam alterações e 31 fracassavam.

Diversos autores ao longo do tempo defendem a importância de um correto tratamento dos requisitos de um projeto, desde sua identificação até sua documentação. Eles justificam que por falharem neste item, os números de software que obtiveram sucessos, são tão baixos.

A MPS.BR, em seu nível G, considera a importância de se controlar os requisitos, descreve 5 resultados que a organização deve alcançar no seu projeto implementando algum tipo processo. E o objetivo deste trabalho é justamente desenvolver uma proposta de processo de gerenciamento de requisitos que visa atender os resultados esperados por este nível.

1.1 OBJETIVO GERAL

Desenvolver uma proposta de processo de gerenciamento de requisitos com aderência aos cinco resultados esperados pela Gerência de Requisitos (GRE) do nível G (MPS-BR).

1.2 OBJETIVOS ESPECÍFICOS

1. Entender a importância do levantamento de requisitos durante o projeto de desenvolvimento de software;
2. Analisar critérios de levantamento e avaliação dos requisitos;
3. Identificar os problemas na elicitação de requisitos, fatores de insucesso no desenvolvimento de software;

4. Estudar os cinco resultados esperados pela gerência de requisitos do nível G do MPS.BR.

1.3 JUSTIFICATIVA

A etapa de análise de requisitos que abrange as técnicas de elicitação ou levantamento trata da identificação das regras, das necessidades dos usuários de informações e comunicação destas necessidades no processo de construção do software (GASTALDO e MIDORIKAWA, 2003).

A distinção entre os requisitos não funcionais e funcionais nem sempre é muito clara. Parte da razão advém para o fato de que os requisitos não funcionais estão sempre relacionados a um requisito funcional (CHUNG, 1999). De uma maneira geral, pode-se dizer que um requisito funcional expressa algum tipo de transformação que tem lugar no software, enquanto um requisito não funcional expressa como essa transformação irá se comportar ou que qualidades específicas ela deverá possuir (CYSNEIROS, 2001).

No contexto de Qualidade de Software a gestão de requisitos se torna um procedimento indispensável no desenvolvimento de software. É importante para qualquer empresa ter um processo definido a ser usado como base que permita fazer a gerência dos requisitos (SOFTEX, 2009).

Para Blascheck (2002) muitos erros poderiam ser evitados se as organizações dispusessem de um processo de engenharia de requisitos definido, controlado, medido e apropriado. No entanto, percebe-se que para muitos profissionais de TI esses conceitos não são claros, o que dificulta a ação dos gerentes no sentido de aprimorar seus processos de desenvolvimentos.

Como solução do problema, em dezembro de 2003, a Associação para Promoção da Excelência do Software Brasileiro (SOFTEX) que é uma entidade privada sem fins lucrativos e que promove ações com abrangência nacional visando transformar o Brasil em um centro de excelência mundial na produção e exportação de software, propôs o projeto MPS.BR.

A Melhoria de Processo de Software (MPS) é um modelo de aperfeiçoamento e avaliação do processo de software destinado preferencialmente a pequenas e médias empresas. A Melhoria do Processo de Software Brasileiro (MPS.BR) é um projeto que visa a melhoria do processo de software do país e sua novidade está na estratégia de implementação, criada para a realidade brasileira. A base técnica

utilizada para a construção deste modelo de qualidade foram as normas ISO/IEC 12207, ISO/IEC 15504 além do conteúdo de outros modelos de referência como Capability Maturity Model Integration (CMMI) (WEBER, 2005).

Cada nível de maturidade do MPS.BR estabelece patamares de evolução de processos. Os níveis são definidos em: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado).

Considerando a importância de controlar os requisitos, o MPS.BR traz no seu primeiro nível de maturidade (G) 5 metas, ou resultados esperados, sobre a Gerência de Requisitos (GRE), atividade na qual está baseada na Engenharia de Requisitos, que tem como principais atividades: elicitação, análise e negociação, especificação, validação e gerenciamento de requisitos (SOMMERVILLE, 2003).

1.4 ESTRUTURA DO TRABALHO

A presente pesquisa está estruturada em capítulos da seguinte maneira:

O Capítulo 1 aborda a introdução, onde nela é descrita a definição do problema, objetivo geral, objetivos específicos e a justificativa deste trabalho.

O Capítulo 2 trata do conceito de Engenharia de Software assim como também as fases de desenvolvimento de software, alguns modelos de ciclo de vida, Engenharia de Requisitos e Qualidade de Software

O Capítulo 3 descreve acerca do MPS.BR, seu nível de maturidade G, a Gerência de Requisitos (GRE) e os cincos resultados esperados pelo nível

O Capítulo 4 traz a proposta deste trabalho na qual visa fornecer subsídio para a instituição a escolher qual melhor forma de acordo com sua situação a alcançar os cinco resultados da GRE no nível G.

O Capítulo 5 descreve sobre os trabalhos correlatos na qual me incentivaram a descrever esta proposta.

O Capítulo 6 traz as conclusões obtidas com esta pesquisa.

2 ENGENHARIA DE SOFTWARE

Segundo Pressman (2005), para termos uma definição de software é necessário comparar com outros produtos, a fim de elencar os aspectos que os diferem: ele não possui corpo como um hardware, apenas demonstra resultados do desenvolvimento de um sistema lógico; e apesar de o software ser projetado e produzido por engenharia assim como os outros produtos manufaturados sua principal diferença está na forma de construção; considerando que ele não é físico, não é trivial saber exatamente o que se tem pronto ou quanto ainda falta para finalizá-lo justificando uma das grandes dificuldades na sua construção, que é obter as métricas do projeto.

O software não se desgasta com o passar do tempo diferentemente dos demais produtos, porém, se “deteriora”, pois ao longo da vida sofre modificações, sendo estas modificações inserções de código para correção dos defeitos. Na maioria das vezes sua construção parte do zero, ou seja, não existem componentes que simplesmente combinados formam o software como se fosse um sistema “pré-moldado”.

A primeira definição de engenharia de software segundo literatura foi proposta por Friedrich Bauer em 1969, que defendia o estabelecimento e uso de sólidos princípios de engenharia para se obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais (NATO, 1968).

Por outro lado, Swebok (2011) apresenta outra definição, na qual a engenharia de software é a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção de software.

De acordo com Pressman (2005) a Engenharia de Software é uma área do conhecimento da informática voltada para a especificação, desenvolvimento e manutenção de sistemas de software aplicando tecnologias e práticas de ciência da computação, gerência de projetos e outras disciplinas, objetivando organização, produtividade e qualidade.

O Conselho de Engenheiros para o Desenvolvimento Profissional define engenharia como a aplicação criativa de princípios científicos para projetar ou desenvolver estruturas, máquinas, aparatos, ou processos de manufatura, ou trabalhos utilizando-os isoladamente ou em combinação; ou para construir ou operá-los com pleno conhecimento de sua convenção; ou para prever seu comportamento

sob condições específicas de operação; tudo respeitando uma função prevista, economia de operação e segurança à vida e propriedade. É possível fazer uso dessa definição para engenharia de software substituindo-se “trabalhos” por “trabalhos de software” (ECPD, 1978).

A engenharia de software abrange um conjunto de três elementos – métodos, ferramentas e procedimentos – que proporcionam mecanismos para a construção, planejamento e gerenciamento de um software de alta qualidade e produtividade.

Segundo o Swebok (2011), a Engenharia de Software está subdividida em requisitos de software, testes de software, gerência de engenharia de software, ferramentas e métodos de engenharia de software, qualidade de software, entre outras. No processo há uma integração e uma sequencia coerente de praticas que objetivam a evolução do sistema, onde as atividades de especificação, projeto, implementação e testes são atingidas através da ligação entre as camadas apresentadas na figura 1, ferramentas, métodos, processos e busca constante pela qualidade.

Figura 1 – Camadas de Engenharia de Software

Fonte: Pressman (2003).

As ferramentas de Engenharia de Software são instrumentos que proporcionam um apoio automatizado para a construção de um software. Existem diversas técnicas para sustentar os métodos assim como suas respectivas ferramentas como linguagens de programação. Já os métodos de engenharia de software descrevem formas de “como fazer” ou “como construir” um software, no qual envolvem um amplo conjunto de tarefas que incluem: planejamento e estimativa

de projeto, análise de requisitos de software, projeto da estrutura de dados, algoritmo de processamento, codificação, teste e manutenção (PRESSMAN, 1995).

Um desafio constante da área de Engenharia de Software é melhorar o processo de desenvolvimento de software, cujo objetivo é formar a ligação entre métodos e ferramentas, e mesmo com a constante evolução destes atributos, a entrega de software em prazos estabelecidos nem sempre é alcançada. Existe a necessidade de desenvolver software de forma mais rápida e produtiva, mas com qualidade (COSTA FILHO, 2005).

Qualidade esta, que está diretamente ligada ao processo de desenvolvimento e associada à eficiência das ferramentas utilizadas no processo, ao alto rendimento da equipe de desenvolvimento, a redução de custos e ao desempenho do cronograma estabelecido, evitando a necessidade de uso de recurso adicional.
Segundo Bertani e Borges (2007) para se conseguir cumprir tais objetivos, deve haver alguns comportamentos e ações durante o desenvolvimento do projeto, tais como:

a) A idealização do projeto;
b) Acompanhar os resultados;
c) Utilizar ferramentas unificadas na organização;
d) Monitorar as táticas de testes;
e) Revisar os componentes produzidos pelo processo;
f) Buscar concordância com os padrões de desenvolvimento de software.

Para Rezende (2005) a engenharia de software pode ser conceituada como uma metodologia de desenvolvimento e manutenção de software, com as seguintes características: roteiro dinâmico de soluções tecnológicas; adequação aos requisitos funcionais do cliente e seus respectivos processos; efetivação de padrões de qualidade e produtividade; fundamentação na tecnologia da Informação disponível, viável, oportuna e customizada; planejamento e gestão de atividades, recursos, custos e prazos. Os procedimentos definem a sequência que os métodos serão aplicados, os produtos a serem disponibilizados, o controle de qualidade e avaliação de software. São estes que antecedem e sucedem o software.

2.1 FASES DE DESENVOLVIMENTO DE SOFTWARE

Segundo Pressman (1995) os três elementos fundamentais, ferramentas, métodos e processos passam por três fases genéricas, a saber:

a) Definição: identifica a função do software a ser desenvolvido. Nessa fase acontecem a análise de requisitos e análise e projeto do software.
b) Desenvolvimento: executa o que foi planejado, nessa fase ocorrem os testes e codificação do software.
c) Manutenção: correções, adaptação e melhorias no software.

2.1.1 Fase de definição

Durante esta fase procura-se responder a questão “o que fazer”. O analista pretende levantar quais informações deverão ser processadas e armazenadas pelo sistema, define a interface, as restrições necessárias no projeto e os critérios de validação, tudo isso visando a construção de um sistema de qualidade. Nesta fase comumente abrange três etapas, Análise do Sistema, Planejamento do Projeto de Software e Análise de Requisitos.

Conforme Rezende (2005) é na Análise do Sistema que ocorre a modelagem, fazendo uso de Diagrama de Fluxo de Dados, Dicionário de Dados, entre outros. É construída uma especificação estruturada do projeto através das suas duas principais entradas, a política do cliente e os encargos do projeto. Posteriormente a etapa de Projeto de Software analisa os riscos, os recursos destinados, os custos estimados, e define as tarefas do sistema. Esta etapa também é conhecida como Projeto Lógico (o que fazer) e início do Projeto Físico (como fazer). Por fim, a etapa Análise de Requisitos trata da identificação das necessidades dos usuários, das regras e restrições necessárias.

2.1.2 Fase de desenvolvimento

Durante esta fase procura-se responder a questão “como fazer”. Define-se a maneira que a estrutura de dados e a arquitetura de software serão projetados, como todo o projeto e seus objetivos serão traduzidos em linguagem de programação e também a metodologia aplicada os testes de software. Esta fase

comumente abrange três etapas, Projeto de Software, Codificação e Testes de Software.

O Projeto de Software trata de maneira detalhada o conjunto de requisitos, a estrutura de dados, a arquitetura, o procedimento algorítmico e as características de interface, apresentando-os de maneira textual, tabulares, gráficas ou utilizando outros métodos que os melhor representem. Na etapa de Codificação todas estas informações são implementadas em linguagem de programação convencional ou não procedimental, resultando em instruções que são compreendidas e executadas por um computador. Por fim, os Testes de Software são executados a fim de identificar defeitos, seja de função lógica ou implementar (REZENDE, 2005).

2.1.3 Fase de manutenção

Ao longo do seu ciclo de vida, o software pode vir a ter necessidade de alterações, que passarão por todos os processos de desenvolvimento novamente até ser agregado ao sistema existente. Estas alterações podem ser motivadas por vários motivos: legislação, melhoria, correções de erros, Adaptativa, Perfectiva, Preventiva (SOMMERVILLE, 2003).

2.2 MODELOS DE CICLO DE VIDA

O modelo de ciclo de vida do software descreve todas as atividades desde a criação até o seu desuso ou seu fim. A escolha do modelo contribui para a eficiência do seu desenvolvimento, sua qualidade e esta muito ligada aos riscos inerentes ao processo. Não há um modelo considerado melhor que outro, depende a complexidade do sistema, porém uma aplicação de um modelo de software erroneamente poderá gerar atividades desnecessárias e baixo desempenho (CONNELL, 1996).

O modelo de ciclo de Vida Cascata é o mais antigo e utilizado até hoje, é fácil de ser entendido, explicado e demonstrado, porém desfavorável em casos que os riscos do projeto são altos o suficiente para requerer mais flexibilidade no cronograma. O modelo é composto por uma sequência de fases, cada uma definida em termos de entradas e saídas, no final de cada fase ela é verificada quais tarefas

foram executadas, e quais saídas estão prontas para a próxima fase (SCHMIDT, 2000).

O modelo de Ciclo de Vida Incremental é similar ao modelo anterior, porém as tarefas sequenciais são agrupadas em pequenos ciclos, indicado para quando o projeto apresenta alto risco de sofrer alterações (PRADO, 2007).

Ainda segundo o autor, o Ciclo de Vida Iterativo é dividido em vários ciclos que são repetidos viabilizando um maior refinamento, costuma apresentar implementações mais completas, visto que toda a arquitetura é percorrida várias vezes.

No Modelo de Ciclo de Vida Espiral são permitidas múltiplas interações entre atividades similares do projeto, inicialmente no centro, a evolução é representada por um espiral (lembrando as camadas da casca de um caracol). Cada setor do espiral corresponde a uma fase do desenvolvimento, que poderá ou não seguir o modelo original onde trás: a determinação de objetivos e restrições na qual são estabelecidas estratégias para alcançar os objetivos; avaliação de alternativas para análise de risco; e por fim o desenvolvimento do produto onde equivale ao modelo cascata.

O Modelo Evolucionário quebra o software em fragmentos que poderão ser entregues para o cliente o mais rápido possível. Neste modelo o desenvolvimento é feito em ciclos do modelo cascata e nas fases de operação e manutenção ocorre a sobreposição destas com novas fases.

No Modelo de Ciclo de Vida em V existem avaliações, verificações, validações e testes em todas as etapas do processo de desenvolvimento, um trabalho só pode ser prosseguido quando todos os produtos da fase anterior estiverem em conformidade com as exigências da organização. Propõe revisões nas atividades como um teste antecipado, com objetivo de detectar erros procedentes ao longo do ciclo do desenvolvimento (PRADO, 2007).

2.3 ENGENHARIA DE REQUISITOS

Segundo Zave (1997) é uma das principais áreas de atividade da Engenharia de Software, possui atuação específica, apresenta avanços tecnológicos e meios próprios voltados para pesquisa em terminologia, métodos, linguagens e

ferramentas. Identifica, molda e documenta requisitos de forma iterativa e cooperativa.

É nessa etapa que o engenheiro tenta entender quais são as reais necessidades dos usuários, qual o problema que se deve resolver, solucionar ou dissolver. Onde se começa a identificar as dimensões humanas e sociais relacionadas ao software a ser desenvolvido.

Kotonya e Sommerville (1998) argumentam que a engenharia de requisitos engloba todas as atividades relacionadas à descoberta, documentação e manutenção do conjunto de requisitos de um sistema computacional.

Sommerville (2003) define Engenharia de Requisitos (ER) como um conjunto de atividades para ajudar a equipe de projeto a identificar, controlar e rastrear requisitos e alterações do sistema ao longo do ciclo de desenvolvimento do software.

2.3.1. Requisito

Segundo Dorfmaann e Thayer (1990) e Maciasdzek (2001), requisito de software representa a necessidade do usuário que deve ser atendida por um determinado produto alcançando seu objetivo, atendendo a um contrato, especificação ou a outros documentos formais impostos.

Sommerville (2003) descreve requisito como um conjunto de atividades que o software deve desempenhar, com suas limitações e restrições, além de características não ligadas diretamente a funções desempenhadas por ele. Entender os requisitos com clareza nem sempre é uma tarefa fácil para os engenheiros de software, muitas vezes o cliente não sabe descrever ou não consegue expressar o que realmente precisa, o processo de analisar, identificar, verificar e documentar é extremamente complexo.

De acordo com Davis (1993) em alguns casos o requisito pode ser considerado uma declaração abstrata, de alto nível, ou como uma restrição ou função do sistema, em outro extremo, ele é uma definição detalhada, matematicamente formal. Complementando, Pressman (2001) relata que superar a expectativa e satisfazer a necessidade do cliente torna-se um desafio, os processos de engenharia de software não são a solução de todos os problemas, mas fornece uma sólida abordagem para satisfazer as necessidades levantadas durante todo o processo de desenvolvimento.

2.3.2. Objetivo

A finalidade da ER é cobrir todas as atividades envolvidas em descobrimento (obtenção), documentação e manutenção de um conjunto de requisitos para um sistema baseado em computador. O uso do termo engenharia implica na utilização de técnicas sistemáticas e repetitivas a fim de assegurar que os requisitos dos sistemas sejam completos, consistentes e relevantes (PÁDUA, 2005).

Seu objetivo é definir, descrever funções e restrições dos sistemas e para isso executa atividades como:

a) Estudo da viabilidade do sistema: Avalia-se a necessidade e a importância de seu desenvolvimento, os resultados são registrados em relatórios onde podem conter sugestões sobre custos, prazos e requisitos.

b) Obtenção e Análise de Requisitos: Reúne-se com os usuários para identificar as funções do sistema, os tipos de usuários e entender suas necessidades;

c) Especificação dos Requisitos: Nessa etapa elabora-se a Especificação de Requisitos de Software (ERS) que é uma descrição das funcionalidades do sistema, das interfaces externas, do desempenho, de atributos e de restrições;

d) Validação dos Requisitos: Logo após é feito uma análise para constatar se realmente os requisitos estão de acordo com as necessidades do cliente, evitando retrabalho em etapas posteriores que afetariam no prazo e no custo;

e) Gerenciamento dos Requisitos: A última etapa consiste em gerenciar e controlar as necessidades de alteração dos requisitos e a relação entre eles durante todo o processo de desenvolvimento.

Os engenheiros de software são os responsáveis diretos pelas atividades de Engenharia de Requisitos, porém, são também envolvidos analistas, gerente, clientes e usuários finais. É muito importante a participação destes para que antes do desenvolvimento já seja concluído com muita lucidez e clareza todos os requisitos para que ao final do desenvolvimento o sistema possa realmente atender a todas as necessidades para qual foi destinado (PRESSMAN, 2006).

Segundo o Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE, 2001) um requisito é definido como uma propriedade que o sistema deve apresentar a fim de

resolver algum problema do mundo real, eles podem variar em seu objetivo e no tipo de propriedades que representam, podendo, desta forma, ser classificados em funcionais e não funcionais.

Os requisitos funcionais são aqueles que expressam funções ou serviços que um software deve ou pode ser capaz de executar ou fornecer. As funções ou serviços são, em geral, processos que utilizam entradas para produzir saídas (CYSNEIROS, 2001).

Ainda para o autor, os requisitos não funcionais, são os que colocam restrições sobre o produto em desenvolvimento, sobre o processo de desenvolvimento, e também, especificam restrições externas ao produto, normalmente são determinados pela natureza e tamanho do sistema no qual o software está inserido.

A distinção entre estes dois tipos de requisitos nem sempre é muito clara. Parte da razão advém para o fato de que os requisitos não funcionais estão sempre relacionados a um requisito funcional (CHUNG, 1999). De uma maneira geral, pode-se dizer que um requisito funcional expressa algum tipo de transformação que tem lugar no software, enquanto um requisito não funcional expressa como essa transformação irá se comportar ou que qualidades específicas ela deverá possuir (CYSNEIROS, 2001).

Para Sommerville (2003), os requisitos de software, além de funcionais e não funcionais, ainda podem ser classificados como de domínio. Demonstrados na figura 2, os requisitos não funcionais estão ligados às restrições sobre serviços e funções oferecidas pelo software, podem ser subdivididos em requisitos de produto, de processos organizacionais e agentes externos.

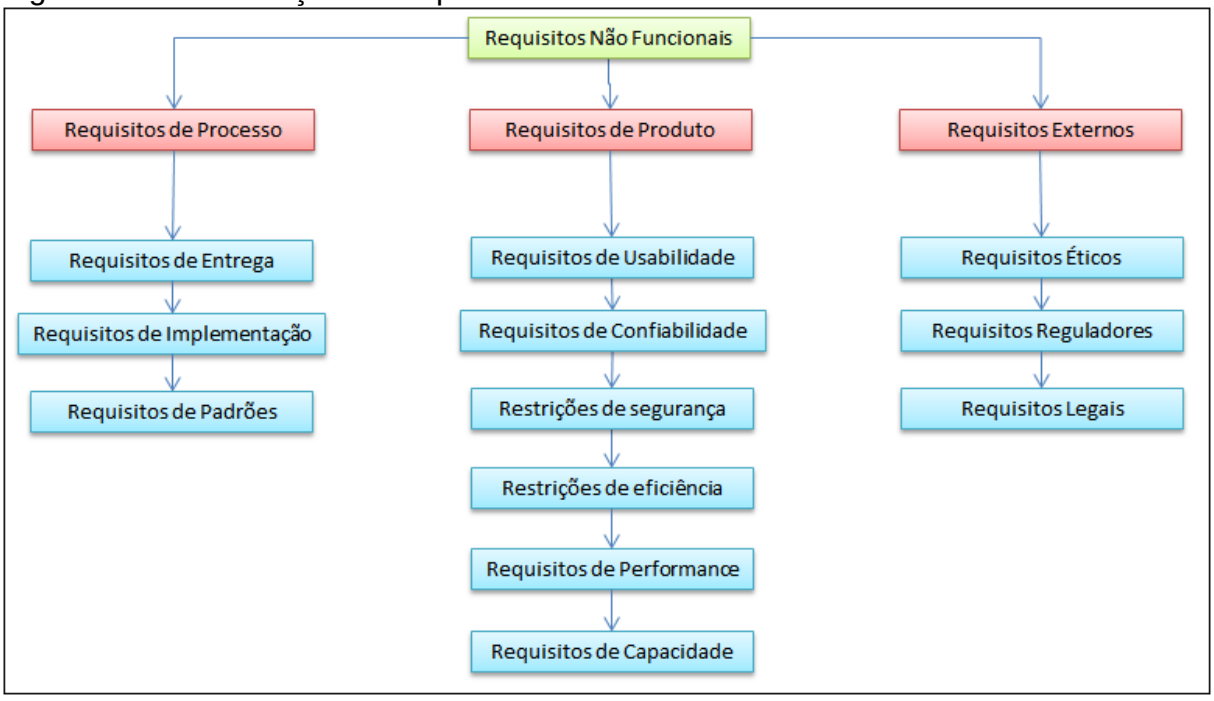
Os requisitos de produto especificam o comportamento do produto, onde encontramos requisitos de desempenho, os requisitos de confiabilidade, os requisitos de portabilidade e os requisitos de usabilidade.

Os requisitos organizacionais são derivados de políticas e procedimentos da organização do cliente e do desenvolvedor. Exemplos incluem padrões de processo que devem ser usados, requisitos de implementação, como a linguagem de programação ou método de projeto usado, e requisitos de entrega que especificam quando o produto e a sua documentação devem ser entregues.

Já os requisitos externos abrangem todos os requisitos derivados de fatores externos ao software e seu processo de desenvolvimento, como os requisitos reguladores, os requisitos legais e os requisitos éticos.

Ainda segundo o autor, os requisitos de domínio são provenientes do domínio de aplicação de software e usam terminologia específica de domínio. Podem ser novos requisitos funcionais, podem restringir os já existentes ou estabelecer como cálculos específicos devem ser realizados. Os engenheiros de software frequentemente encontram dificuldades em compreender como esses requisitos estão relacionados a outros do software pelo fato de serem requisitos especializados. Porém, eles são importantes por refletirem os fundamentos do domínio da aplicação. Sendo assim, se esses requisitos não forem satisfeitos, pode ser difícil fazer o software funcionar satisfatoriamente.

Figura 2 – Classificação de requisitos não funcionais



Fonte: Adapatado de Sommerville (2003).

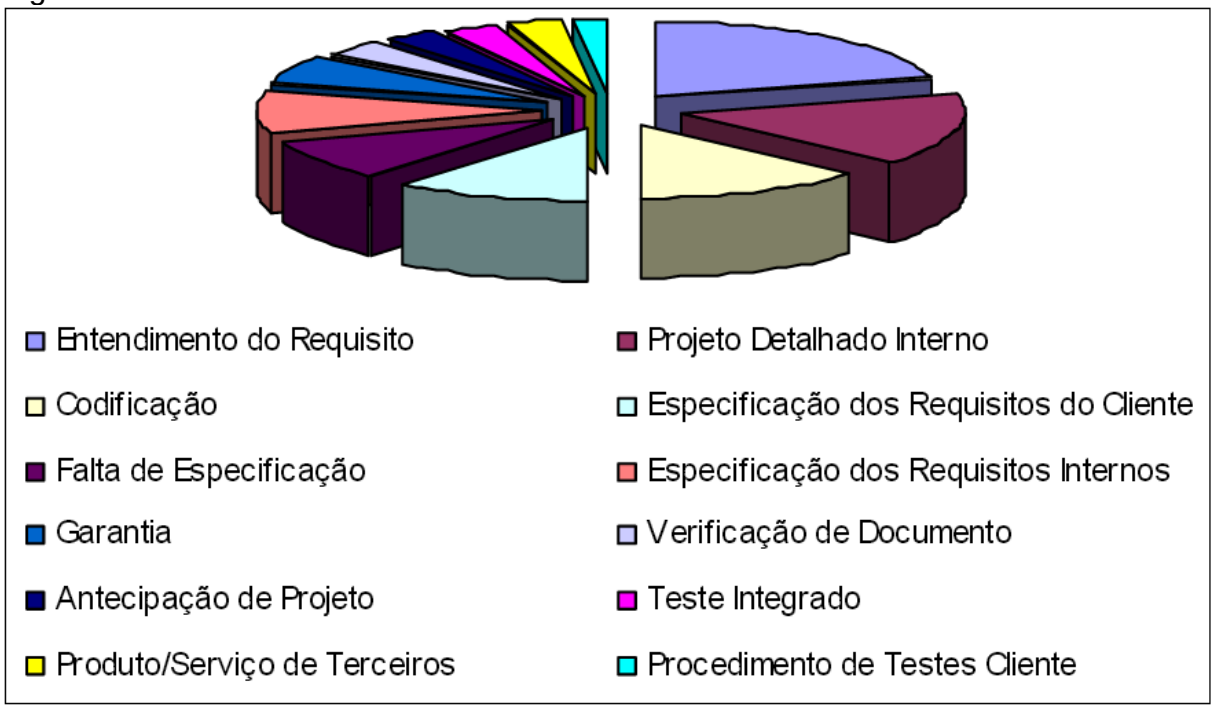
2.3.3. Problemática

No entendimento de Pfleeger (2004), as falhas nas atividades relacionadas aos requisitos como entendimento, documentação e gerenciamento são as principais causas de insucessos nos projetos.

Hofmann (2001) reafirma que os requisitos deficientes ainda são a maior causa de falhas nos projetos de software. Capers Jones descobriu, por meio de pesquisas realizadas com centenas de organizações, que a engenharia de requisitos é deficiente em mais de 75% das empresas (JONES, 1996, apud HOFMANN, 2001, p. 58).

Um estudo realizado por Gastaldo (2003) demonstrou que cerca de 50% do retrabalho referente ao processo de desenvolvimento de software ocorre nas fases iniciais de elicitação, análise e documentação de requisitos conforme demonstra a figura 3. Grande parte das causas do retrabalho é relacionada aos requisitos não funcionais de desempenho. Sua pesquisa revelou que por não serem levados em consideração desde o início do projeto, estes requisitos acabam não sendo atendidos. Em consequência de mudanças de estratégia a fim de inserir um requisito não funcional não projetado, há aumento de custo e prazo de entrega.

Figura 3 – Causas de retrabalho



Fonte: Gastaldo (2003).

De acordo com o último relatório gerado pelo Standish Group em Março/2013 que demonstra o percentual de falha em projetos de (TI), em 2012, a cada 100 projetos iniciados somente 39 atingiram o sucesso, 43 sofreram alterações ao longo do processo e 18 fracassaram, em 1994, quando o instituto fez sua primeira aferição, foi registrado que a cada 100 projetos iniciados apenas 16 atingiram ao

sucesso, 53 sofreram alterações e 31 fracassaram, significa uma melhoria de mais de 100% nos casos de sucesso, porém o estudo ainda demonstra fragilidade na grande maioria dos projetos que não são entregues conforme o planejado (STANDISH GROUP, 2013).

Para Blascheck (2002), dispor de um processo de engenharia de requisitos bem definido, controlado, medido e apropriado poderia evitar muitos erros durante o desenvolvimento, no entanto, percebe-se que para muitos profissionais de TI esses conceitos não são claros, o que dificulta a ação dos gerentes no sentido de aprimorar seus processos de desenvolvimentos.

Sommerville (2003) lembra que o custo para consertar erros de requisitos ainda no estágio de validação, é normalmente muito menor do que consertar erros em estágios posteriores do processo de desenvolvimento. Corrigir problemas de requisitos podem demandar um retrabalho no projeto, implementação e teste do sistema, consequentemente, os custos serão altos. Estima-se que o custo para corrigir erros de requisitos pode ser 100 vezes maior do que corrigir um simples erro de programa.

2.3.4. Melhores práticas em engenharia de requisitos

De acordo com os estudos de Hofmann (2001), projetos de sucesso têm uma correta combinação de conhecimento, recursos e processos. O quadro 1 resume a relação entre estes três elementos, ele apresenta as sugestões de melhores práticas, de acordo com cada uma das áreas foco (conhecimento, recursos ou processo), o custo de execução desta prática (em termos de introdução da prática e de sua aplicação) e os benefícios esperados da realização de tais práticas.

Quadro 1 – Melhores práticas da Engenharia de Requisitos

Área Foco	Melhores Práticas	Custo de Introdução	Custo de Aplicação	Benefícios Chaves
Conhecimento	Envolver clientes e usuários durante a ER	Baixo	Moderado	Melhor entendimento das reais necessidades
	Identificar e consultar todas as origens de requisitos	Baixo e moderado	Moderado	Aprimoramento da cobertura de requisitos
	Atribuir as atividades de ER a gerentes de projetos e membros da equipe habilidosos	Moderado e alto	Moderado	Performance mais previsível
Recursos	Alocar de 15 a 30% do total de esforço do projeto para as atividades de ER	Baixo	Moderado e alto	Manter uma alta qualidade de especificação do início ao fim do projeto
	Prover modelos e exemplos de especificação	Baixo e moderado	Baixo	Aperfeiçoar a qualidade da especificação
	Manter um bom relacionamento com os envolvidos	Baixo	Baixo e moderado	Satisfazer melhor as necessidades dos clientes
Processos	Priorizar requisitos	Baixo	Baixo e moderado	Foco da atenção nas necessidades mais importantes dos clientes
	Desenvolver modelos complementares juntamente com os protótipos	Baixo e moderado	Moderado	Eliminar inconsistências e ambiguidades na especificação
	Manter a matriz de rastreabilidade	Moderado	Moderado	Explicitar a ligação entre requisitos e produto de trabalho
	Usar revisões por pares cenários e "Walkthroughs" para validar e verificar os requisitos	Baixo	Moderado	Especificação mais acurada e alta satisfação do cliente

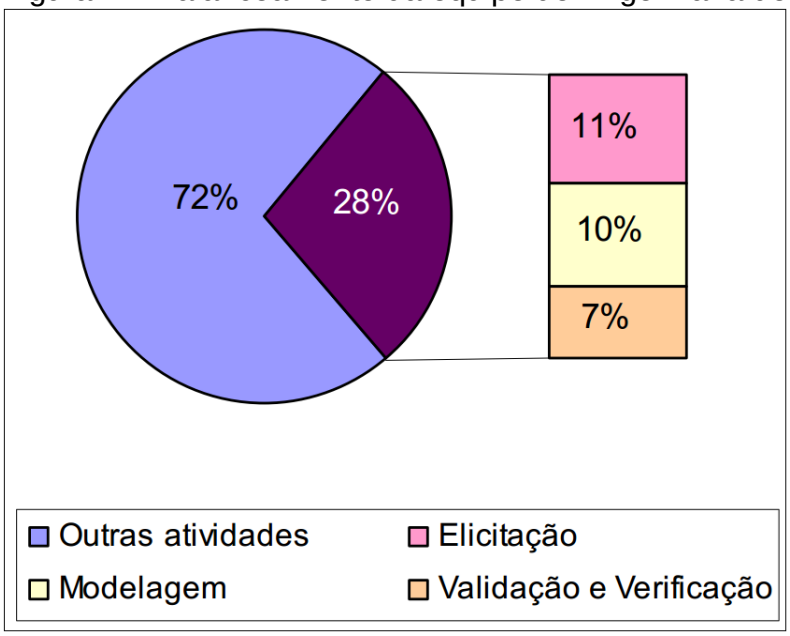
Fonte: Adaptado de Hofmann (2001).

Ainda segundo o estudo de Hofmann (2001), a interação com os envolvidos tem um papel decisivo do início ao fim dos projetos de sucesso em ER. Os times de mais sucesso sempre relacionam seus clientes e usuários neste processo, a participação dos usuários é o fator mais importante para o sucesso. Boas equipes de ER também identificam as fronteiras do domínio da aplicação e dos principais envolvidos para validar seu entendimento do domínio da aplicação. Para tal, eles identificam e consultam todo tipo de fonte de requisitos: examinam artefatos de sistemas, materiais do sistema atual e dos anteriores.

Ainda segundo o autor outros dois fatores também são de grande importância, a escolha dos membros da equipe de requisitos com habilidades no domínio da aplicação e a metodologia a ser utilizada e no processo de ER. Principalmente na escolha de um gerente capacitado e de um especialista do domínio envolvido, projetos considerados de sucesso também alocam uma significativa parcela de recursos para a ER. Conforme apresentado na Figura 4, Hofmann estima que 28% do total de recursos do projeto é alocado para as atividades de ER.

Além disso, a equipe costuma ser distribuída de maneira balanceada: 11% do esforço de projeto para elicitação, 10% para modelagem e 7% para validação e verificação. Para facilitar seu trabalho, normalmente as equipes utilizam modelos e exemplos de documentos de especificação elaborados em projetos anteriores.

Figura 4 – Balanceamento da equipe de Engenharia de Requisitos



Fonte: Hoffmann (2001).

Por fim, Hofmman (2001) destaca que a orientação dos projetos de sucesso é dada por requisitos priorizados pelos envolvidos. Isto permite que a equipe decida quais requisitos devem ser investigados, quando e com que nível de detalhe. Além disso, é indispensável manter a matriz de rastreabilidade dos requisitos, que armazena as relações entre requisitos, ela permite o rastreamento de um requisito de sua origem até sua implementação. Ademais, equipes de sucesso validam e verificam, repetidamente, seus requisitos com múltiplos envolvidos, utilizando técnicas de revisão.

2.3.5 Processo de ER tradicional

Não há uma modelo de processo de ER tradicional considerado efetivo para todas as organizações, cada uma deve desenvolver seu próprio processo baseado no tipo de sistema que desenvolve, sua cultura organizacional, o nível de experiência e habilidades da equipe envolvida. Elencar profissionais que realmente se envolvam na equipe e solicitar consultoria externa a fim de obter uma perspectiva externa sobre o processo pode ser muito produtivo, pois os processos não são transferíveis por completo de uma empresa para outra (SOMMERVILLE 2003).

O autor ainda destaca que um processo de ER considerado completo deve apresentar quais atividades são realizadas, a estrutura ou o agendamento destas,

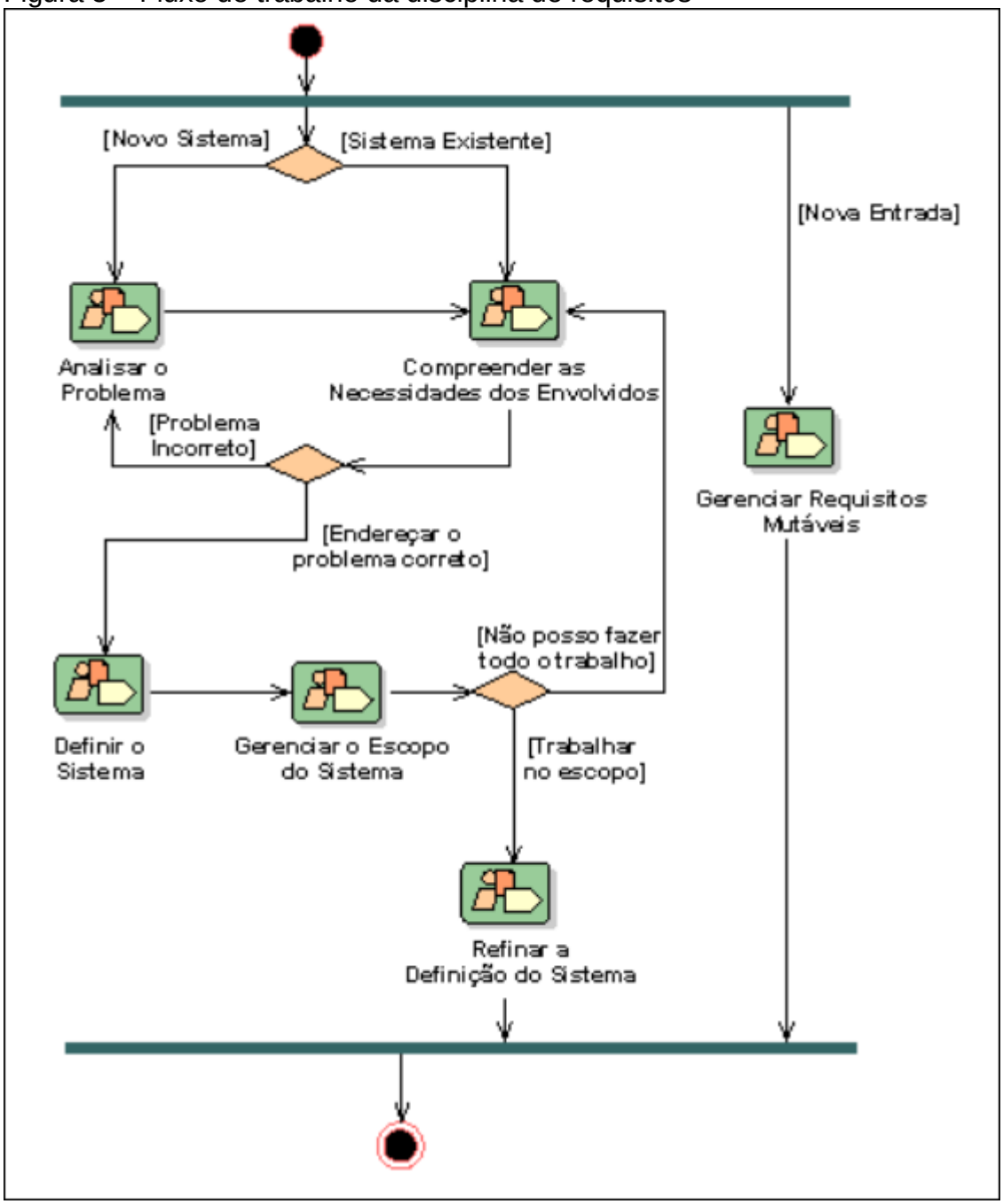
quem é o responsável, as entradas e saídas relacionadas e as ferramentas utilizadas. A maioria das organizações apresentam poucas definições e pouca padronização neste processo, envolvendo apenas saídas de processo, como estrutura de documentos de requisitos.

O Unified Process (UP), ou simplesmente processo unificado, visando à construção de sistemas orientados e a objetos, foi desenvolvido definindo um processo de desenvolvimento de software (LARMAN, 2004). Produto final de três décadas de trabalho, o UP combina as melhores práticas estudadas e tem como base a Unified Modeling Language (UML). As características que o torna realmente único podem ser resumidas em três aspectos: é dirigido a caso de uso, centrado na arquitetura e iterativo e incremental (JACOBSON, BOOCK E RUMBAUGH 2000).

Lançado em 1996, com base no UP o Rational Unified Process (RUP), ou simplesmente Processo Unificado Racional, o processo de engenharia de software oferece suporte às atividades relacionadas ao ciclo de vida do processo de desenvolvimento (KRUCHTEN, 2003). Ele descreve todos os artefatos, atividades e papéis, fornece diretrizes e inclui modelos para a maioria dos artefatos (LARMAN 2004).

A disciplina que trabalha a ER ainda no RUP, exibe as habilidades “chaves” necessárias para a eficiente execução do gerenciamento de requisito, são elas: Analisar o problema demonstrado; Compreender as necessidades dos envolvidos; Definir o sistema; Gerenciar o escopo do sistema; Refinar a definição do sistema e Gerenciar requisitos mutáveis. O fluxo de trabalho é exibido em ordem lógica e sequencial conforme mostra a figura 5.

Figura 5 – Fluxo de trabalho da disciplina de requisitos



Fonte: Damke e Moraes (2008)

2.3.6 Estágios da ER

Para Pressman (2006) a necessidade de um novo negócio ou serviço motiva os esforços da ER, que segundo o autor possui sete estágios: concepção, levantamento, elaboração, negociação, especificação, validação e gestão.

A concepção tem como objetivo estabelecer um entendimento básico do problema e normalmente é feito de forma informal.

No que corresponde ao levantamento de requisitos, a ideia principal consiste em entender a necessidade do cliente, apesar de parecer simples, são três os principais problemas encontrados nessa fase. Os problemas de escopo se referem quando os limites do software não são bem definidos; os problemas de entendimento relacionam-se a uma incerteza ou incapacidade de os clientes expressarem suas necessidades; e por fim, problemas de volatilidade, faz alusão quando os requisitos mudam ao longo desenvolvimento.

Na elaboração as informações obtidas são refinadas e é produzido um modelo técnico das funções, características e restrições do software. No estágio de negociação conciliam-se conflitos entre pedidos de diferentes clientes e usuários, avaliando-se o risco de cada requisito e faz-se uma priorização deles.

O produto final do trabalho dos engenheiros de requisitos se apresenta na especificação, essa etapa serve de base para atividades das etapas seguintes de um modelo de engenharia de software. Pode ser um modelo gráfico, um documento escrito, um modelo matemático formal, um protótipo, uma coleção de cenários de uso ou qualquer combinação desses elementos. O mais relevante é que esteja descrita a função do software, seu desempenho e as restrições que deverão ser levadas em conta durante seu desenvolvimento.

No processo de validação é realizada uma avaliação detalhada do estágio anterior para garantir que todos os requisitos de software foram declarados de modo não ambíguo e que os erros, omissões ou inconsistências tenham sido detectados e corrigidos.

Concluindo o pensamento de Pressman (2006), um conjunto de atividades ajuda a identificar, controlar e rastrear os requisitos, essa é a gestão de requisitos. Rastreá-los e acompanhar as mudanças que ocorrem durante o desenvolvimento é o objetivo. São desenvolvidas tabelas de rastreamento, onde cada requisito recebe um identificador, e nessa tabela são especificados os relacionamentos entre requisitos, assim é possível analisar o impacto que a alteração pode gerar.

2.4 GERÊNCIA DE REQUISITOS

Consiste no processo de gerenciar mudanças nos requisitos do sistema, durante o projeto. Os requisitos de um sistema podem sofrer alterações para refletir as necessidades de mudanças solicitadas pelos envolvidos, surgindo novas

necessidades, identificando novos atributos em funcionalidades já em desenvolvimento, mudanças nas leis e regulamentações.

Estas mudanças devem ser gerenciadas para garantir que estejam dentro do orçamento e do prazo estipulados. As principais atividades da gerência de requisitos são o controle da mudança e a avaliação do impacto da mudança.

Neste sentido, a gerência de requisitos exige que a rastreabilidade dos requisitos seja armazenada, demonstrando as ligações entre os requisitos, as fontes de requisitos. Os projetos do sistema devem ser rastreados e guardados, de maneira que estejam disponíveis no momento de avaliar o impacto e controlar as alterações que os requisitos vierem a sofrer (SOMMERVILLE 2003).

Para Sommerville (2003) a gerência de requisitos controla as mudanças dos requisitos e deve iniciar quando já tiver um esboço das suas especificações, lembra a necessidade de atualizar a documentação sempre que houver alterações. As consequências decorrentes de tais modificações também variam muito, pois levam a alterações no código e na especificação do produto interferindo no seu prazo de entrega, no custo e até mesmo na relação cliente e fornecedor podendo gerar insatisfação e cancelamento do serviço.

Para acompanhar as modificações e evitar grandes prejuízos a gerência de requisitos deve apresentar um planejamento sobre as tarefas relacionadas a atividade. Neste planejamento são definidas atividades para identificar os requisitos e rastreá-los; definir relacionamento entre eles e entre o projeto; analisar impactos de suas alterações e também decidir onde armazenar os requisitos, em ferramentas específicas ou simples planilhas e processadores de textos (Sommerville, 2003).

Segundo Pressman (2005) rastrear requisitos é a capacidade de identificar os componentes do projeto e os resultados gerados pela especificação - rastreamento progressivo, também conhecido como rastreabilidade para frente, o outro tipo é o rastreamento inverso ou rastreabilidade para trás, que é a capacidade de saber quem gerou o produto.

O rastreamento de requisitos possibilita identificar a origem, os relacionamentos e os efeitos que um possui sobre o outro, essa etapa é realizada logo após identificar as características de sistema a serem alteradas, respeitando a ordem de atividades do planejamento.

São três as fases na gerência de requisitos. A fase inicial identifica e analisa o problema do requisito e depois propõe alterações. A segunda avalia a proposta e

juntamente com o rastreamento dos requisitos decide se a mudança é válida ou não. E a última fase implementa a mudança que deverá acontecer tanto na especificação quanto no sistema. O cumprimento de todas as fases da gerência de requisitos, a execução das atividades planejadas e uma boa especificação de requisitos contribuem para o desenvolvimento de software com qualidade.

2.5 QUALIDADE DE SOFTWARE

A NBR ISO 8402 (1994) descreve qualidade como totalidade das características de uma entidade que lhe confere a capacidade de satisfazer às necessidades explícitas e implícitas.

Com uma definição um pouco mais atualizada a ISO/IEC 9126-1 (2003) define como totalidade de características e critérios de um produto ou serviço que exercem suas habilidades para satisfazer as necessidades declaradas ou envolvidas.

Já Pressman (2006) resume sua definição como concordância com os requisitos funcionais e de desempenho, com padrões de desenvolvimento explicitamente documentados e com as características implícitas em todo software desenvolvido profissionalmente.

Portanto, para que um software seja considerado de qualidade ele deve apresentar um alto nível de atendimento, precisão e conformidade em relação aos seus requisitos, refletindo todo seu processo de desenvolvimento. Caso os requisitos documentados não reflitam as reais necessidades, são incompletos, inconsistentes ou não possuem um bom acompanhamento das alterações no período de gerência de requisitos, certamente a qualidade deste software será ameaçada (CARVALHO et al., 2001).

Completando o pensamento, Koscianski (2006) define a qualidade de software como uma área da Engenharia de Software que tem como objetivo garantir a produção de sistemas satisfatórios e que atendam aos custos e cronogramas estabelecidos.

Ao falar de software e de sua produção, estamos falando de produtos e de processos, Leite (2001) destaca que de nada adianta concentrar a atenção só no produto ou só no processo, é necessário que os dois caminhem juntos. A capacidade de gerenciar suas qualidades garantirá o alto nível de satisfação do

produto final. Apenas um bom processo de software não garante que os produtos de software produzidos sejam de boa qualidade, mas é um indicativo de que a organização é capaz de produzir bons produtos de software (KOSCIANSKI & SOARES, 2007).

Para Pádua (2005) a base para desenvolver sistemas com qualidade é possuir uma boa especificação de requisitos, e esta deve apresentar uma documentação precisa, correta, consistente, completa, permitindo alterações ao longo do projeto, mensurando a sua origem. Deve permitindo identificar previamente os impactos das alterações e demonstrar a priorização dos requisitos de acordo com seu grau de importância.

Segundo Machado (2001), a qualidade do processo de software é tão importante quanto a qualidade do produto. A partir da década de 90 houve uma crescente preocupação com a modelagem e melhoria do processo de software. Orgãos como Institute of Electrical and Electronics Engineers (IEEE), Software Engineering Institute (SEI), International Organization for Standardization /International Eletrotechnical Commission (ISO/IEC), Melhoria de Processo de Software Brasileiro (MPS-BR) e Requirements Management Maturity (RMM) produzem padrões, métodos, técnicas e modelos que auxiliam o processo.

Para melhor caracterização dos esforços, são apresentados a seguir alguns modelos criados, que estabelecem uma correlação construtiva, entre a melhoria da qualidade do processo e a consequente melhoria da qualidade do produto.

Conforme Feiler e Humphrey (1993), o modelo de processo é uma representação abstrata da arquitetura, um projeto ou definição do processo de software. Descreve em diferentes níveis de detalhes uma organização dos elementos de um processo e provê definições da maneira como devem ser realizadas a avaliação e a melhoria de processo. Um modelo de processo é composto de grupos de processos e de sub-modelos que variam de acordo com as abordagens de diferentes autores.

O Capability Maturity Model (CMM) foi criada no fim da década de 80 com iniciativa de tornar claros os benefícios reais de técnicas e desenvolver métodos que avaliassem e melhorassem a capacidade de desenvolvimento de software nas organizações. Considerado como uma soma de “melhores práticas”, demostra “o que” deve ser feito e não “como”, propõe avaliar a qualidade dos softwares desenvolvidos pelas empresas (SEI, 2010).

A partir da década de 90, surgiram diversos CMMs, com foco em desenvolvimento de sistemas, engenharia de software, desenvolvimento de produtos e processos. Apesar de esses modelos serem úteis para as organizações, a grande diversificação tornou-se um problema, pois cada modelo tinha quer ser avaliado separadamente, gerando um custo elevado. Pensando na unificação que foi criado o c*apability Maturity Model Integration* (CMMI), que abrange os diversos CMMs (SEI, 2010).

2.5.1 CMMI

O CMMI é um dos principais modelos para melhoria de processos criados pelo Software Engineers Institute (SEI). Criado no ano 2000 é uma integração e evolução de alguns modelos como: o Capability Maturity Model for Software (SW-CMM) para avaliação de fornecedores de sistemas; o System Engineering Capability Model (SE-CMM) direcionado à engenharia de sistemas e o Integrated Product Development Capability Maturity Model (IPD-CMM) para definição de produtos (KOSCIANSKI, 2006).

Alguns objetivos contemplados neste modelo é eliminar inconsistências entre os modelos existentes, implementar melhorias no modelo CMM, estar compatível com a norma ISO/IEC 15504 e contemplar a forma de representação contínua, o modelo permite ainda duas representações, por estágios e contínua.

A representação contínua utiliza os níveis de capacidade para avaliação, preocupa-se com a seleção de uma área de processo particular para melhoria e o nível de capacidade desejado para aquela área de processo, sendo assim se o processo é repetível ou incompleto é importante. Portanto, o nome incompleto é o ponto de partida para a representação contínua. Os níveis de capacidade, que são seis, titulados de 0 a 5, são aplicáveis em organizações em busca de melhoria de processo em áreas de processos individuais, correspondem o meio para incrementar a melhoria dos processos correspondendo a uma determinada área de processo.

Entretanto, na representação por estágios trabalha-se a maturidade da organização em vários níveis. Os Níveis de Maturidade, que são Cinco, enumerados de 1 a 5, são aplicáveis a organizações cujo processo de melhoria deve ser alcançado através de múltiplas áreas de processo, estes níveis correspondem a

base para prever os resultados para os próximos projetos da organização. São seis os níveis de capacidade numerados de 0 a 5.

O quadro 2 ilustra as representações por estágios e contínua que são muito parecidas, as duas contém os mesmos componentes que por sua vez possuem a mesma hierarquia e configuração.

Quadro 2– Comparação entre níveis de maturidade e níveis de capacidade

	Representação contínua Nível de capacidade	**Representação por estágios Nível de maturidade**
Nível 0	Incompleto	Não aplicável
Nível 1	Repetível	Inicio
Nível 2	Gerenciado	Gerenciado
Nível 3	Gerenciado Quantitativamente	Gerenciado Quantitativamente
Nível 4	Gerenciado Quantitativamente	Gerenciado Quantitativamente
Nível 5	Otimizado	Otimizado

Fonte: Software Engineering Institute (2007)

No Brasil, a indústria de software demonstra interesse em melhorar seus processos de software aderindo às avaliações para obtenção do certificado, porém 70% da indústria nacional é constituída por MPEs, destas, poucas possuem disponibilidade financeira para adotar modelos de maturidade, dificultando o destaque da indústria brasileira no mercado internacional (MCT, 2010).

2.5.2 ISO/IEC 12207

A norma NBR ISO/IEC 12207 – Processo de Ciclo de Vida de Software - foi criada pela International Organization for Standardization (ISO) e o International Electrotechical Commission (IEC). Essa norma estabelece os processos, atividades e tarefas a serem aplicadas durante a aquisição, fornecimento, desenvolvimento, operação e manutenção de software. Definindo de forma ampla os processos que guiam a adaptação da sua utilização nos projetos de software em uma empresa.

Para Colenci Neto (2008) como objetivo da norma, a desenvolvedora de software passa a contar com o auxílio dos envolvidos na produção de software de modo a melhor definir suas funções, por meio de processos bem estabelecidos, e assim, proporcionar às empresas que utilizam um entendimento melhor das atividades a serem seguidas nas operações correlacionadas.

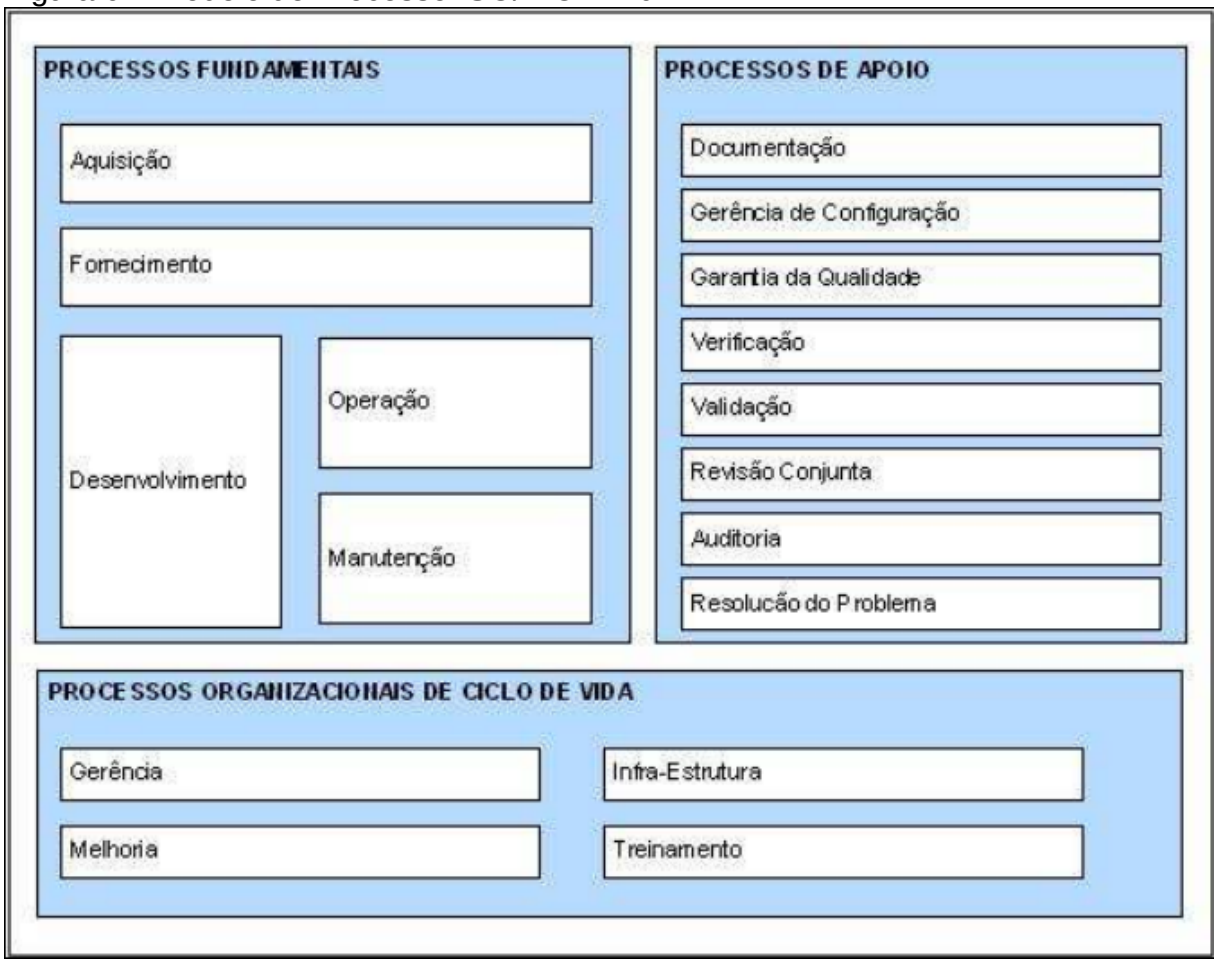
Para Nogueira (2006), a adoção da norma ISO/IEC 12207 pelo desenvolvedor, estabelece os procedimentos de como estruturar e gerenciar o ciclo de desenvolvimento, proporcionando a possibilidade de acompanhamento de todo o processo e, permitindo que o software venha representar mais fielmente, a realidade da empresa modelada, para geração de um sistema customizado, atendendo assim de modo adequado os requisitos e necessidades do cliente.

A ISO/IEC 12207 define e classifica com maior riqueza de detalhes o grupo de processos que compõem um modelo de processo de software, abrangendo todo o ciclo de vida do software desde a concepção até sua descontinuidade (CALSALVARA et al, 2000).

Segundo Calsalvara, et al. (2000), conforme as necessidades de cada organização, os processos dessa norma são agrupados de acordo com seu principal objetivo no ciclo de vida do projeto de forma que sua estrutura seja flexível, modular e adaptável.

A figura 6 ilustra o modelo de processos da norma ISO/IEC 12207 que segundo Calsalvara el al (2000), tem por objetivo fornecer uma estrutura para que todos os envolvidos com o desenvolvimento de software utilizem uma linguagem comum na definição dos processos da organização, o modelo apresentado é composto de três tipos de processos: Processos Fundamentais, Processos organizacionais e Processos de Apoio. (CALSALVARA et al, 2000).

Figura 6 – Modelo de Processo ISO/IEC 12207



Fonte: Calsalvara, et al (2000)

Os Processos Fundamentais abrangem cinco atividades relacionadas ao longo do ciclo de vida do software, atrelados à produção do software: aquisição, atividades para contratar a organização; fornecimento, atividades dos fornecedores do sistema ou serviço que precisa ser adquirido; desenvolvimento, papéis e atividades para o desenvolvimento; operação, ações voltadas para operação do produto e suporte operacional; manutenção, atividades relacionadas a gestão de mudanças, migração e fim do ciclo de vida do software.

A fim de atingir os objetivos de negócio, os Processos Organizacionais auxiliam o desenvolvimento de processos, produtos e recursos utilizados nos projetos. Contemplam este grupo as atividades de gestão, infraestrutura e recursos, melhoria e treinamento. A gestão abrange atividades básicas de gestão de projetos; Infraestrutura e Recursos são meios utilizados ao longo do ciclo de vida; Melhoria descreve atividades onde demonstram onde a organização necessita controlar, medir e melhorar; Treinamento define as atividades de instrução aos usuários.

Os Processos de Apoio são de fundamental importância para a qualidade dos processos fundamentais servindo como ferramenta de auxílio aos processos. Compõem este grupo oito atividades: Documentação, gerência de configuração, garantia da qualidade, verificação, validação, revisão conjunta, auditoria e resolução de problemas.

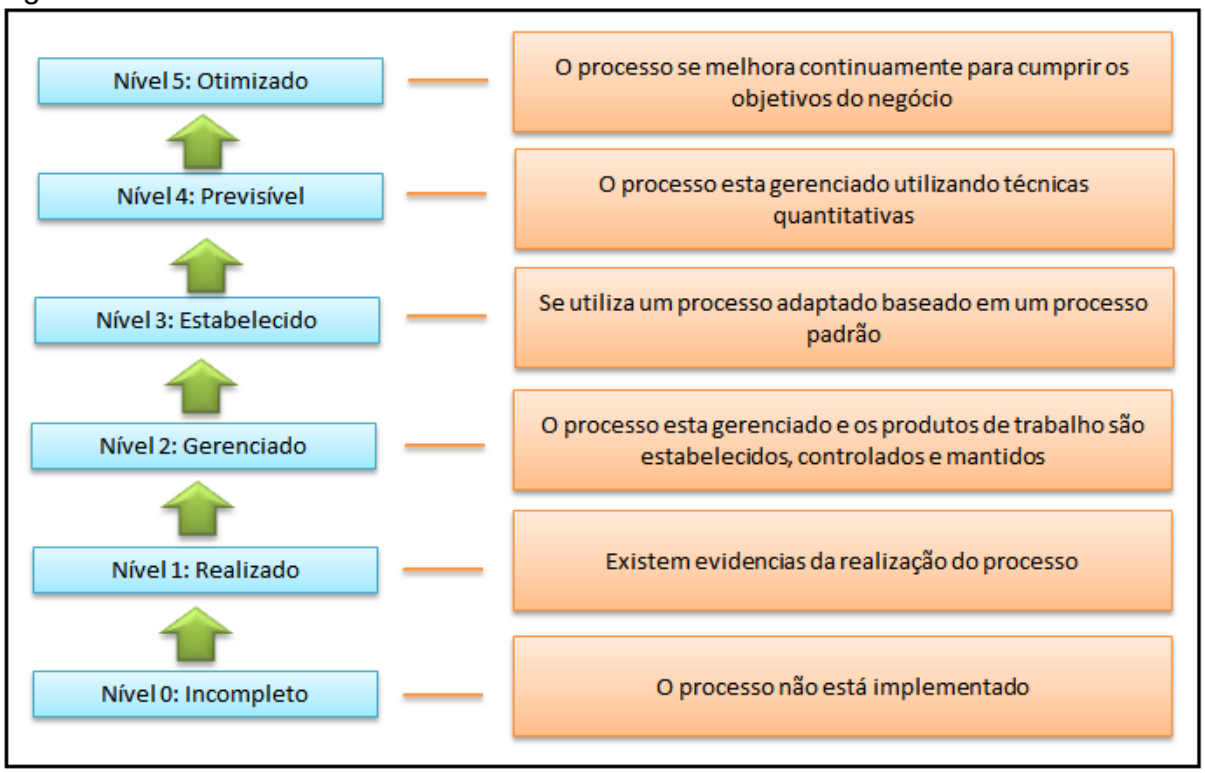
A documentação registra as atividades relacionadas ao longo do ciclo de vida do software. A gestão de configuração define as atividades de gerência, já a garantia de qualidade, utiliza técnicas de revisão, auditoria, verificação e validação para garantir que o software possua os requisitos definidos. A atividade de verificação relaciona a inspeção do software e serviços de acordo com o projeto, enquanto a validação define atividades que comprovam o papel do software de acordo com o projeto. A revisão conjunta envolve todos os envolvidos no software, afim de se obter diferentes pontos de vista. A auditoria envolve atividades que determinam a conformidade com os requisitos, planos e contrato, justificando a necessidade de serem feitas por uma equipe auditora. Por fim a resolução de problemas, que trata inconsistências ou não conformidades que surgem ao longo do processo de vida do software.

2.5.3 ISO/IEC 15504

A ISO/IEC 15504 foi publicada em 2004, com base no projeto Software Process Improvement and Capability Etermination (SPICE), ela define um modelo referência de processos considerados essenciais para a engenharia de software no mundo. A descrição de sua estrutura, conforme demonstrada na figura 7, propõe a avaliação da evolução e da capacidade dos processos em duas dimensões: Na primeira são definidos os processos que devem ser avaliados e na segunda são definidas as escalas utilizadas para a medição da capacidade de processos avaliados (BERGMANN, 2008).

A avaliação de acordo com a ISO/IEC 15504 provê um modelo para medição em desenvolvimento de software. A utilização dos dados reunidos das avaliações pode guiar o processo de melhoria contínua, conduzindo ganhos em produtividade, qualidade de produtos e entregas. Classificação dos processos da ISO/IEC 15504 avaliados em seis níveis, conforme a figura 7.

Figura 7 – Níveis da ISO/IEC 15504



Fonte: Adaptado de Bergmann (2008)

3 MPS-BR

A Melhoria do Processo de Software (MPS) é um modelo de melhoria e avaliação de processo de software. Inicialmente seu foco eram pequenas e médias empresas, normalmente grupos em um Modelo de Negócio Cooperado (MNC), mas também vem sendo aplicado em organizações de grande porte no Modelo de Negócio Específico (MNE), este modelo é a principal base para o projeto a seguir (SOFTEX, 2009).

O MPS.BR é um programa federal para melhoria de processo do software brasileiro que procura incentivar e apoiar a indústria de software nacional na melhoria desenvolvimento do processo de desenvolvimento. Teve início em dezembro de 2003 e é coordenado pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), contando com apoio do Ministério da Ciência e Tecnologia (MCT), da Financiadora de Estudos e Projetos (FINEP) e do Banco Interamericano de Desenvolvimento (BID) (SOFTEX, 2006).

O projeto abrange somente os resultados exigidos para o processo de desenvolvimento e seus atributos, não determinando métodos, técnicas ou

ferramentas que devem ser utilizadas para alcançar estes resultados. Esta característica impõe a academia a exercer a função de gerar conhecimento a respeito de métodos, técnicas e ferramentas que possibilitam alcançar os resultados esperados pelo projeto. Esta é uma das principais motivações da presente proposta.

Para criação desse modelo utilizou-se como base técnica algumas normas e modelo de maturidade como a norma ISO/IEC 12207, a ISO/IEC 15504 e o modelo CMMI. O modelo pretende estabelecer um processo e um método de avaliação, o qual dá sustentação e garante que o MPS.BR está sendo empregado de forma coerente com as suas definições.

O programa possui duas metas a serem alcançadas a médio e longo prazo:

a) meta técnica, visando à criação e ao aprimoramento do Modelo MPS;

b) meta de negócio, visando à disseminação e adoção do Modelo MPS, em todas as regiões do país, em um intervalo de tempo justo, a um custo razoável, tanto em micros, pequenas e médias empresas (foco principal) quanto em grandes organizações privadas e governamentais, com resultados esperados tais como:

(i) criação e aprimoramento do modelo de negócio MN-MPS;

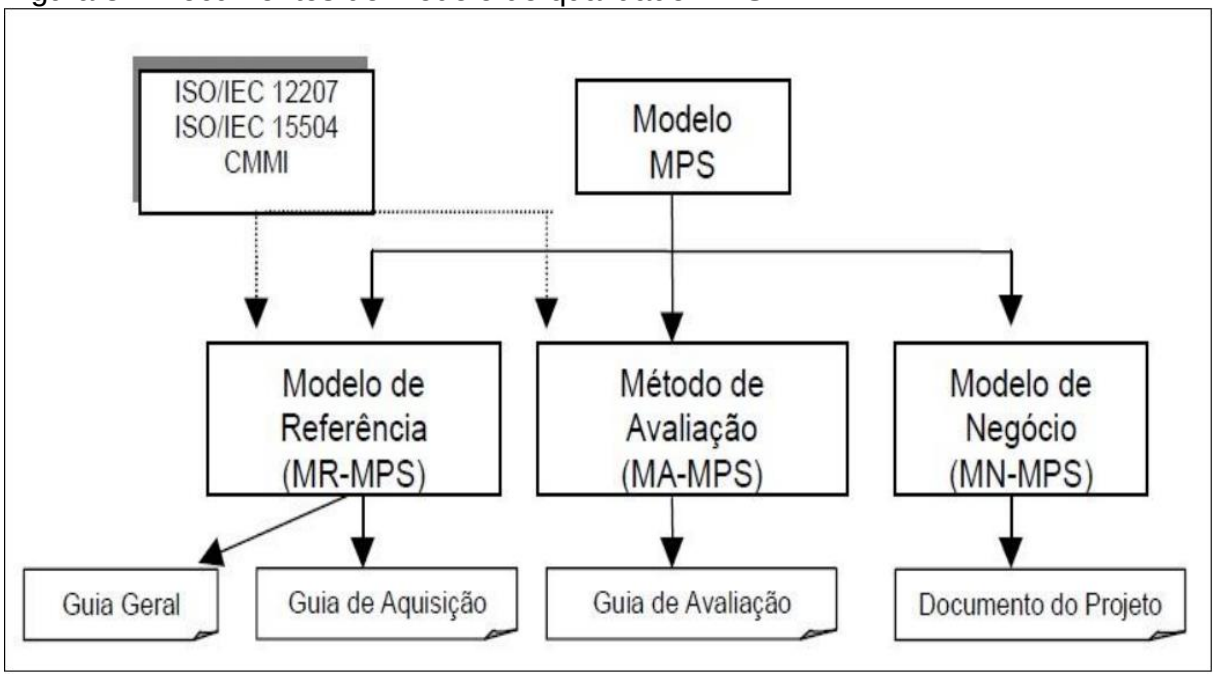
(ii) cursos, provas e workshops MPS;

(iii) organizações que implementaram o Modelo MPS; (iv) organizações com avaliação MPS publicada (tendo prazo de validade de três anos) MA-MPS (EVIDÊNCIAS..., 2012).

O modelo define regras para sua implementação e avaliação. Como mostra a figura 8, o MPS possui três componentes que foram descritos em documentos específicos (SOFTEX, 2009):

a) Modelo de Referência para Melhoria do Processo de Software (MR-MPS);
b) Método de Avaliação para Melhoria de Processo de Software (MA-MPS);
c) Modelo de Negócio para Melhoria de Processo de Software (MN-MPS).

Figura 8 – Documentos de modelo de qualidade MPS.BR



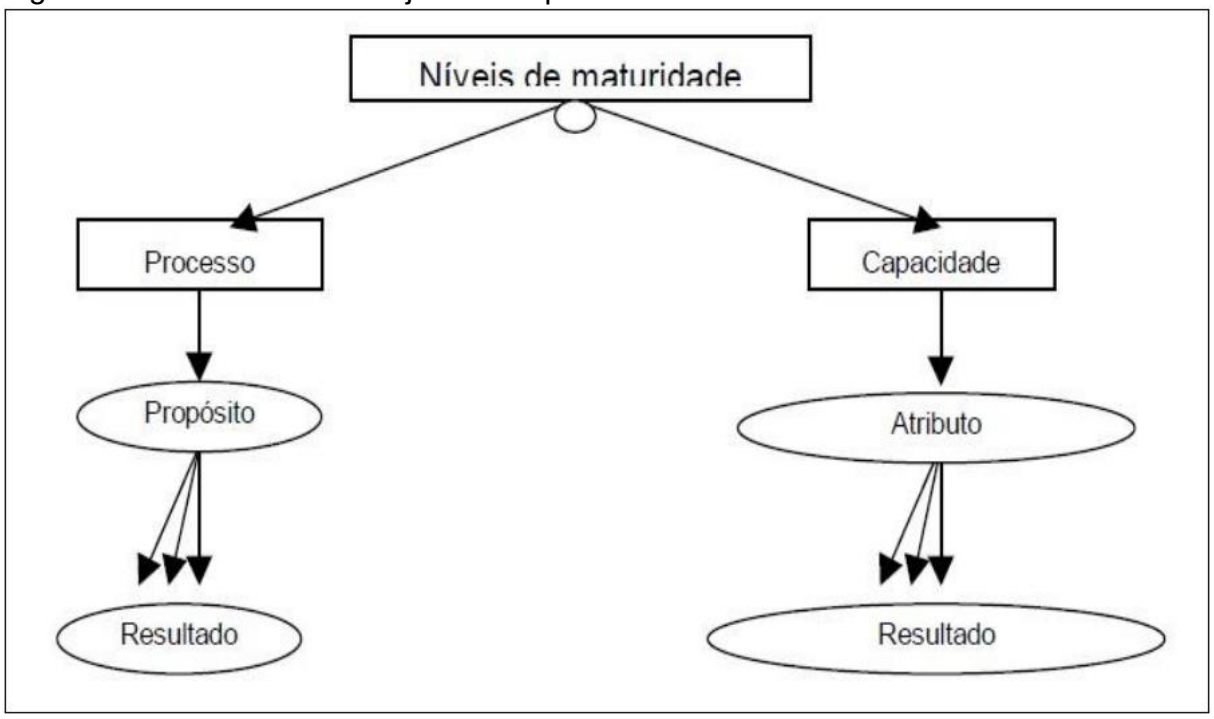
Fonte: SOFTEX (2009)

O Método de Avaliação para Melhoria de Processo de Software (MA-MPS) contém a descrição do processo de avaliação, os requisitos para verificação dos avaliadores e os requisitos para conformidade com o MR-MPS. Este está descrito no Guia de Avaliação (SOFTEX, 2009).

O Modelo de Negócio para Melhoria de Processo de Software (MN-MPS) possui a descrição das regras dos domínios: o domínio do Projeto MPS.BR, coordenado pela SOFTEX; o domínio das Instituições Implementadoras do modelo MPS e Instituições Avaliadoras do Modelo MPS; e o domínio das empresas que queiram utilizar o modelo MPS para melhorar seus processos de software. (SOFTEX, 2009).

É no Modelo de Referência para Melhoria de Processo de Software (MR-MPS) que estão descritos os requisitos que as empresas precisam atender para estar em conformidade com o modelo MPS (SOFTEX, 2009). Modelo este composto por sete níveis de maturidade que são sequencias e cumulativos, cada nível é composto por um conjunto de processos em um determinado nível de capacidade onde o progresso e o atendimento de cada nível de maturidade é obtido somente quando são atendidos todos os requisitos e atributos relacionados àquele nível (SANTANA, TIMÓTEO, VASCONCELOS, 2006). A figura 9 demonstra esta estrutura.

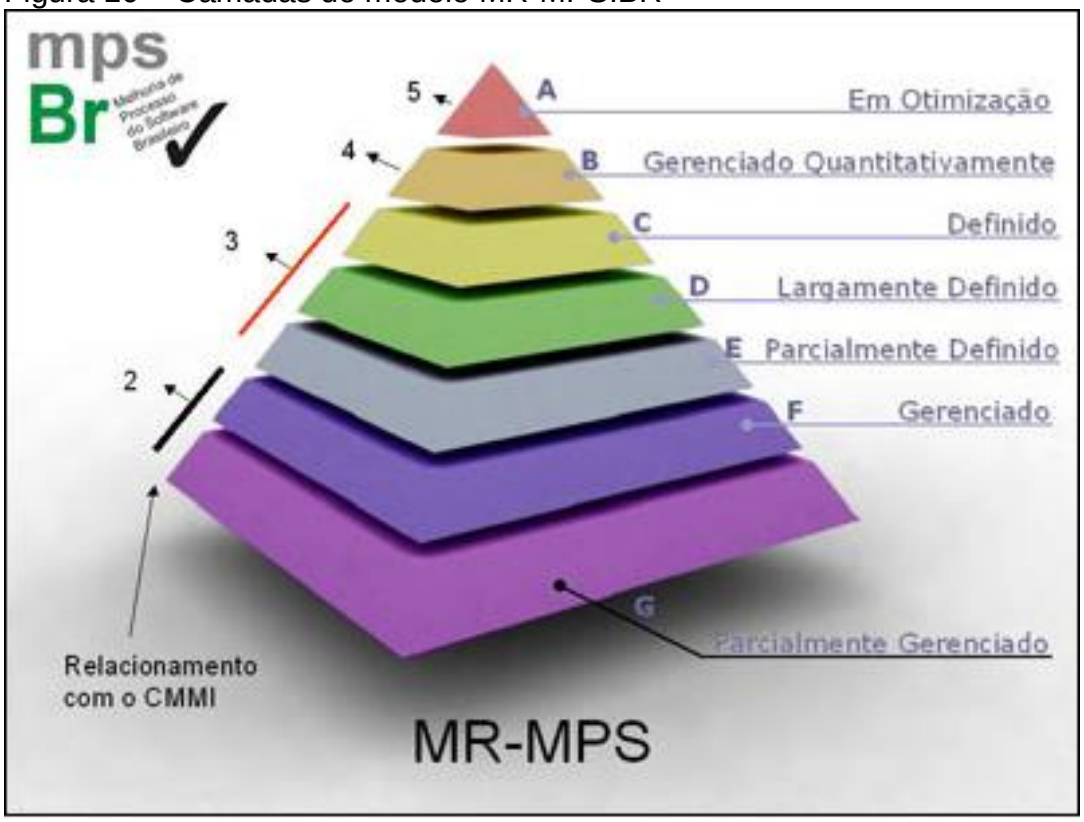
Figura 9 – Estrutura dos conjuntos de processos de cada nível do MPS



Fonte: SOFTEX (2009)

Os sete níveis de maturidade, que estão ilustrados na figura 10, estabelecem patamares de evolução de processos, são descritos do nível mais básico (G) até o mais completo (A).

Figura 10 – Camadas do modelo MR-MPS.BR



Fonte: Evidencias (2008)

Compreende-se que a organização obtém determinado nível de maturidade quando atende aos propósitos de todos os resultados esperados nos processos e atributos de processo do respectivo nível de maturidade, os níveis do MR-MPS-BR são definidos em:

a) **Nível G (**Parcialmente Gerenciado**):** O primeiro nível é o ponto de partida para a implantação de melhoria dos processos de software na organização. Ao alcançar este nível a organização deve focar na criação de mecanismos adequados para o planejamento, monitoramento e controle de projetos e gerenciar os requisitos ao longo do desenvolvimento.

b) **Nível F (**Gerenciado**):** Para estar de acordo com o segundo nível é necessário garantir o gerenciamento do projeto, assegurando a qualidade dos produtos e processos por meio de indicadores quantitativos de desempenho, bem como a gerencia de configuração dos produtos.

c) **Nível E (**Parcialmente Definido**):** O objetivo principal do terceiro nível é a padronização dos processos da organização, auxiliando os recursos

humanos a desprender do conhecimento ou desempenho individual, não dependendo de uma pessoa chave.

d) **Nível D (**Largamente Definido**):** Este nível exige ações mais técnicas para determinadas tarefas, focando na melhoria de processos de engenharia de software. São estabelecidos métodos para validação e verificação de projeto, incluindo revisões e testes como definição de arquitetura e estratégias.

e) **Nível C (**Definido**):** Neste nível há uma maturidade ainda maior em relação a definição do processo, demonstrando capacidade na tomada de decisões em relação a desvios e riscos de projeto.

f) **Nível B (**Gerenciado Quantitativamente): Deste nível em diante, apresentasse um alto nível de maturidade, neste nível o foco é manter um entendimento quantitativo do desempenho dos processos-padrão, possibilitando alcançar os objetivos relacionados a qualidade e desempenho do processo estabelecido para o projeto.

g) **Nível A (**Em Otimização)**:** No nível de maturidade mais completo foca na crescente capacidade competitiva através de inovações, apresentando uma melhora contínua de processos.

Conforme demostra o quadro 3, estes sete níveis do MPS.BR são baseados nos quatro níveis de maturidade da representação por estágio do CMMI (níveis 2 a 5). Os níveis F, C, B e A do modelo MPS correspondem respectivamente aos níveis 2, 3, 4 e 5 do CMMI. Os níveis E e D são níveis intermediários entre os níveis 2 e 3 do CMMI e já o nível G é um nível intermediário entre os níveis 1 e 2 do CMMI (SOFTEX, 2009).

Quadro 3 – Comparação entre os níveis de maturidade do CMMI e do MPS.BR

Comparação entre Níveis de Maturidade CMMI e MPS.BR	
CMMI	**MPS.BR**
5	A
4	B
3	C
--	D
--	E
2	F
--	G
1	Não é definido

Fonte: Bergmann (2008)

A aplicação de algum modelo de maturidade pode beneficiar as organizações na implementação de boas práticas da Engenharia de Software. O MPS.BR fornece as empresas caminhos economicamente viáveis para tais benefícios e permite a agregação de vantagens competitivas no mercado, tanto no local quanto no global.

A transparência oferecida por este modelo evita que o processo tenha uma execução desconhecida, se torna mais claro, independente de qual fase ele se encontre, por ser compreendido por todos na empresa, não fica dependendo do conhecimento ou desempenho individual. Além da clareza outra vantagem do modelo é a existência de um controle em se manter dentro do planejado, independente dos desvios que aconteçam durante o desenvolvimento. Por ser gerenciável, apresenta algumas características que são vitais para a qualidade do processo e do produto, a estabilidade de desenvolvimento dentro do prazo, custo e escopo desejáveis.

3.1 NÍVEL DE MATURIDADE G

Segundo Blaschek (2002), compõem o nível mais básico, o nível G, dois atributos de processos:

a) Gerência de Projetos: Identifica, coordena e monitora as atividades, tarefas e recursos que um projeto necessita para produzir um produto, no contexto dos requisitos e restrições do projeto;

b) Gerência de Requisitos (GRE) - Gerenciam os requisitos e componentes do produto e identificam inconsistências entre esses requisitos a fim de evitar desenvolver sistemas que não estejam de acordo com as necessidades do cliente.

A capacidade do processo de melhoria ou do nível de maturidade possui cinco Atributos de Processo (AP): AP 1.1 (o processo é executado); AP 2.1 (o processo é gerenciado); AP 2.2 (os produtos de trabalho do processo são gerenciados) e AP 3.1 (o processo é definido); AP 3.2 (o processo-padrão está implementado). Estes atributos são baseados nos atributos de processo ISO/IEC 15504 e correspondem às praticas genéricas do CMMI-SE/SW (SOFTEX, 2009). O quadro 4 contém a representação dos níveis de maturidade e seus respectivos atributos.

Quadro 4 – Níveis de maturidade do MPS

Nível	Nome e sigla dos Processos	Atributos de Processo (Capacidade)
A (mais alto)	Inovação e Implantação na Organização – IIO, Análise e Resolução de Causas - ARC	AP 1.1 AP 2.1, AP 2.2, AP 3.1 e AP 3.2
B	Desempenho do Processo Organizacional – DEP, Gerência Quantitativa do Projeto - GQP	AP 1.1 AP 2.1, AP 2.2, AP 3.1 e AP 3.2
C	Gerência de Riscos – GRI, Análise de Decisão e Resolução – ADR	AP 1.1 AP 2.1, AP 2.2, AP 3.1 e AP 3.2
D	Desenvolvimento de Requisitos – DRE, Solução Técnica – STE, Validação – VAL, Verificação –VER, Integração do Produto – ITP, Instalação do Produto – ISP, Liberação do Produto – LIP	AP 1.1 AP 2.1, AP 2.2, AP 3.1 e AP 3.2
E	Treinamento – TER, Definição o Processo Organizacional – DFP, Avaliação e Melhoria do Processo Organizacional – AMP, Adaptação do Processo para Gerência do Projeto – APG	AP 1.1 AP 2.1, AP 2.2, AP 3.1 e AP 3.2
F	Gerência de Configuração – GCO, Garantia da Qualidade – GQA, Medição – MED, Aquisição – AQU	AP 1.1 AP 2.1 e AP 2.2
G (mais baixo)	Gerência de Projeto – GPR, Gerência de Requisitos - GRE	AP 1.1 e AP 2.1

Fonte: SOFTEX (2009)

3.2 GERÊNCIA DE REQUISITOS (GRE)

A MPS.BR traz em seu primeiro nível de maturidade (G), a GRE, na qual considera de suma importância um controle dos requisitos. Ele é baseado na disciplina de Engenharia de Requisitos e possui como principais atividades: elicitação, análise e negociação, especificação, validação e gerenciamento de requisitos. Os cinco resultados esperados pelo GRE são fundamentados nestas atividades e tem como propósitos gerenciar os requisitos do produto e dos componentes do produto e identificar inconsistências entre os requisitos, os planos de projeto e os produtos de trabalho do projeto. (SOMMERVILLE, 2003).

O primeiro resultado esperado, GRE1, tem por objetivo levantar, analisar e validar os requisitos, diante das partes envolvidas, a fim de definir o escopo do projeto, além de gerar um documento com a descrição dos requisitos que será utilizado durante o desenvolvimento. No GRE2, o comprometimento da equipe é obtido através de atividades de análise e validação dos requisitos. As atividades esperadas no GRE3 correspondem a rastreabilidade bidirecional, mapeando a relação entre outros requisitos e com os produtos de trabalho. No GRE4 são revisados produtos e planos de trabalho visando corrigir inconsistências em relação aos requisitos. O foco do GRE5 é gerir as alterações realizadas nos requisitos ao longo do projeto, avaliando impacto e mantendo histórico de mudanças.

Vale ressaltar que as atividades a serem realizadas para se alcançar os objetivos esperados, descritos a seguir, não seguem uma ordem explícita de realização, elas fazem parte de um processo iterativo e incremental onde cada empresa adotara a sua. Uma abordagem mais detalhada sobre cada um dos resultados esperados do nível G para a GRE está descrita nas seções seguintes, nas quais também são abordadas algumas técnicas existentes que permitem alcançar os resultados esperados.

3.2.1 GRE 1 – Os requisitos são entendidos, avaliados e aceitos junto aos fornecedores de requisitos, utilizando critérios objetivos

As atividades de concepção dos requisitos pertencentes ao GRE1 pretendem fazer uso de mecanismos para elicitar, avaliar e validar a fim de garantir uma compreensão compartilhada. Por conseguinte é redigido o documento com a descrição dos requisitos comprovando o entendimento das partes envolvidas. Após

a identificação dos requisitos do produto e dos componentes do produto do projeto é necessário que estes sejam validados utilizando critérios objetivos pelas partes.

A comunidade de Engenharia de Requisitos propõe um conjunto de técnicas que podem ser utilizadas neste processo para alcançar os objetivos esperados.

3.2.1.1 Técnicas de levantamento de requisitos

Para Kendall (2010), manter uma boa comunicação com o cliente, o fornecedor de requisitos é fundamental para assegurar um bom entendimento das necessidades e porventura os requisitos do projeto, aumentando as chances de sucesso do projeto.

A seguir algumas técnicas que auxiliam o levantamento de requisitos e que viabilizam o entendimento necessário à organização para satisfação deste nível.

3.2.1.1.1 Entrevistas

A entrevista é umas das técnicas mais convencionais no processo de levantamento de requisitos, tratando-se de uma conversa entre duas ou mais pessoas com os objetivos bem traçados. É através deste recurso que a estratégia será traçada para as etapas seguintes, baseado nas opiniões expostas, as expectativas e problemas a serem tratados KENDALL (2010).

O autor ainda descreve que as entrevistas devem seguir algumas etapas: Planejamento, condução e registro, podendo ou não ser gravada. Antes de se iniciar a entrevista, devem-se estabelecer perguntas do tipo subjetivas, objetivas e de aprofundamento baseadas nos objetivos, assim como averiguar a pessoa mais adequada a responder cada pergunta.

Segundo Goguen e Linde (1993) a realização dos tipos de perguntas deve ser de forma específica, utilizando-se mais das objetivas e de aprofundamento para obter maior riqueza de detalhes de um assunto específico. O uso desta técnica requer foco e objetividade, a aplicação demasiada poderá resultar em detalhes irrelevantes além de aumentar significativamente o tempo da entrevista, podendo ser causados de acordo com as características dos entrevistados e do nível de aprofundamento das perguntas realizadas.

A documentação gerada deve ser validada tanto pelo engenheiro de requisitos que declara compreender a necessidade do cliente, e também deste último que confirma ao grupo entender as anotações ou representações gráficas de suas informações. Uma boa prática indicada pelo MPS.BR é identificar o validador e o fornecedor dessas informações caso sejam precisos novos questionamentos ou sejam detectadas incoerências pela parte do cliente durante o projeto.

A aplicação desta técnica viabiliza um amplo entendimento dos requisitos, assim como sua respectiva documentação para uma avaliação utilizando critérios previamente definidos.

3.2.1.1.2 Questionários

A aplicação do questionário é indicada em casos que há indisponibilidade física ou há dispersão das pessoas envolvidas, ou até mesmo onde se precisa obter informações de várias pessoas, como se fosse um feedback a respeito de um assunto comum (KENDALL, 2010).

A elaboração de um questionário é mais complexa do que possa parecer, um planejamento adequado visando o conteúdo, formato e ordem das questões podem evitar considerações desnecessárias. Um questionário deve possuir questões claras e sem ambiguidade, ter fluxo definido, prever antecipadamente dúvidas que possam surgir e evitar ser muito extenso, pois pode ocasionar desinteresse das pessoas que irão respondê-lo (PARASURAMAN, 1991).

Ainda segundo o autor, com a entrega do questionário em grupo, obtêm-se as respostas mais rapidamente do que envia-lo por e-mail ou correio. Cabe ao responsável pela pesquisa analisar as respostas e consolida-las, enviando uma cópia com o resumo das informações coletadas para os participantes como consideração pelo tempo dedicado.

A utilização desta técnica permite que o resultado para este nível seja parcialmente atendido, ela permite o entendimento e a documentação, porém não executa a validação pelos fornecedores de requisitos. Para que tal aceitação seja realizada é necessário o envio dos requisitos aos fornecedores várias vezes até que se tenha obtido a confirmação do entendimento em ambos os lados.

3.2.1.1.3Joint Application Development (JAD)

Joint Application Development (JAD) foi desenvolvida pela empresa norte americana International Business Machines (IBM), a técnica trata de uma reunião realizada com as partes interessadas afim de se identificar dúvidas em áreas distintas do projeto.

PRESSMAN (1996) atribui à técnica uma abordagem popular da técnica *Facilitaded Application Specification Techniques* (FAST), onde consta levantar anteriormente o máximo de informações ou expectativas a respeito do que será discutido na reunião. Os participantes devem elaborar uma lista dos objetos do sistema, dos objetos que são usados para que ele execute suas funções, além de uma lista de operações, que manipulam ou interagem com os objetos.

Padua (2001) descreve algumas vantagens da técnica: obtém um maior comprometimento dos usuários com os requisitos; reduz o tempo necessário ao levantamento da especificação dos requisitos; retira do projeto requisitos de valor questionável; esclarece dúvidas de interpretação dos requisitos entre usuários e desenvolvedores; identifica e discute o mais cedo possível, problemas políticos que possam interferir no projeto.

O autor ainda descreve que a técnica precisa ser aplicada com cautela, pois alguns pontos podem comprometer sua eficácia como: o não envolvimento de pessoas que desempenham papéis chaves no processo de uso do produto; A participação de pessoas não comprometidas com o produto e o número excessivo de participantes, além disso, o líder do JAD deve possuir experiência e envolvimento com os participantes.

O que torna a JAD eficiente é um índice de mudança de requisitos baixíssimo e redução de tempo na elicitação de requisitos, muito indicada para pequenos e médios projetos, porém nos grandes podem ser usadas múltiplas sessões JAD para acelerar a definição dos requisitos.

A técnica apresentada atende amplamente os resultados esperados, possibilitando o entendimento dos requisitos. Sua documentação para uma avaliação utiliza os critérios previamente definidos, depois de serem aceitos junto aos fornecedores de requisitos.

3.2.1.1.4 Brainstorming

A técnica Brainstorming, ou simplesmente tempestade de ideias, busca a extração do maior número de opiniões possíveis, reúnem-se um grupo de pessoas, de preferência de setores e competências distintas. Os participantes são motivados a expor seu pensamento através da escrita sobre um tema pretendido, nenhuma ideia deve ser julgada como errada ou absurda, contribuindo assim para que sejam geradas muitas ideias e principalmente não constranger os integrantes.

Nesta técnica é nomeada uma pessoa para registrar todas as ideias em uma lousa ou em papel, à medida que cada ideia é expressa, ela é colocada de forma que todos os participantes possam vê-la. Os participantes também devem ser encorajados a combinar ou enriquecer as ideias de outros e, para isso, é necessário que todas as ideias permaneçam visíveis a todos os participantes.

Após o período de expressão é necessário filtrar as ideias fazendo uma revisão junto ao grupo, os integrantes devem julgar ideia por ideia, considerando as mais valiosas para o grupo e classificando em ordem de prioridade.

O *brainstorming* é uma técnica muito utilizada para criação de novos produtos, ou resolução de problemas, situação na qual, pode ainda não existir um cliente ou fornecedor específico.

A técnica permite que este resultado seja parcialmente atendido, possibilitando o entendimento dos requisitos, documentá-los para uma avaliação utilizando os critérios previamente definidos, porém, dificulta que estes sejam aceitos junto aos fornecedores de requisitos. Para que os fornecedores possam validar os requisitos levantados, a empresa deve escolher um método que possibilite a avaliação do cliente, possivelmente a prototipagem é uma técnica recomendada para essa atividade.

3.2.1.1.5 Etnografia

A etnografia é uma técnica de observação que visa compreender a política organizacional bem como a cultura de trabalho da organização com objetivo de familiarizar-se com o sistema e sua história. O engenheiro de software ou analista é inserido no ambiente de trabalho em que o sistema irá ser utilizado, o trabalho diário é observado e as tarefas reais em que o sistema será utilizado são anotadas.

Descobrir requisitos de sistema implícitos, que refletem os processos reais é o principal objetivo desta técnica, e não os processos formais, onde as pessoas estão envolvidas (GOGUEN & LINDE, 1993).

Para KENDALL (2010), alguns itens são importantes e devem ser executados antes, durante e depois do estudo de observação.

É necessário fazer um planejamento antes do estudo de observação, é indispensável identificar as áreas dos usuários a serem observados**.** Obter a aprovação das gerências apropriadas para executar as observações, identificar os nomes e funções das pessoas chave que estão envolvidas no estudo de observação e explicar a finalidade do estudo.

Durante o estudo é necessário familiarizar-se com o local de trabalho que está sendo observado, identificar os agrupamentos organizacionais e as facilidades manuais e automatizadas. Coletar amostras de documentos e procedimentos escritos que são utilizados em cada processo e acumular informações estatísticas a respeito das tarefas como frequência em que elas ocorrem, estimativas de volumes e tempo de duração para cada pessoa que está sendo observada.

Depois do estudo de observação é documentada toda informação descoberta e observações realizadas, a consolidação do resultado é revisto com as pessoas observadas e com os gerentes de requisitos.

A etnografia permite: que os resultados esperados sejam amplamente atendidos, possibilitando os entendimentos dos requisitos; documentá-los para uma avaliação utilizando os critérios previamente definidos depois serem aceitos junto aos fornecedores de requisitos. A técnica possui algumas desvantagens como, a utilização de muito tempo para a observação e a possibilidade de o analista ser levado ao erro em suas observações. Em geral a etnografia é muito útil e frequentemente usada para complementar descobertas obtidas por outras técnicas.

3.2.1.1.6 Prototipagem

A prototipação é uma técnica que permite a obtenção rápida de sugestões, inovações, revisões ou mudanças e permite captar o *feedback* do usuário a respeito do sistema em desenvolvimento.

Protótipos tem por objetivo explorar aspectos críticos dos requisitos de um produto, implementando de forma rápida um pequeno subconjunto de

funcionalidades deste produto. O protótipo é indicado para estudar as alternativas de interface do usuário; problemas de comunicação com outros produtos; e a viabilidade de atendimento dos requisitos de desempenho (KENDALL, 2010).

Em sua construção, podem ser utilizadas ferramentas de programação que possuam biblioteca de componentes, processadores de texto, ferramentas de manipulação de imagens ou até mesmo lápis e papel.

Alguns dos benefícios da prototipagem são as reduções dos riscos na construção do sistema, pois o usuário chave já verificou o que o analista captou nos requisitos do produto.

A prototipagem permite: que este resultado seja amplamente atendido, possibilitando o entendimento dos requisitos; documentá-los para uma avaliação utilizando os critérios previamente definidos depois serem aceitos junto aos fornecedores de requisitos.

3.2.1.1.7 Pesquisa de Mercado (SWOT)

A Pesquisa de Mercado, Strengths, Weaknesses, Opportunities, Threat (SWOT), Forças, Fraquezas, Oportunidades e Ameaças ou simplesmente análise de competidores foi criada por dois professores de Havard, Kenneth Andrews e Roland Christensen. Consiste em analisar os pontos fortes e fracos além de ameaças e oportunidades, com fins de se projetar um produto que maximize os pontos fortes levantados e minimizar os pontos fracos encontrados nos produtos semelhantes existentes, (FINE, 2009).

Alguns pontos devem ser analisados como verificar se as informações são recentes e/ou isentas, se as fontes devem ser adequadas e se todos os participantes devem conhecer os conceitos envolvidos.

O *SWOT* é uma técnica mais utilizada para criação de novos produtos situação na qual pode ainda não existir um cliente ou fornecedor específico. Resultando o atendimento parcial dos resultados esperados, a análise de mercado possibilita o entendimento dos requisitos, assim como sua documentação para uma avaliação utilizando os critérios previamente definidos, porém, dificulta que estes sejam aceitos junto aos fornecedores de requisitos. Para que os fornecedores possam validar os requisitos levantados, a empresa deve escolher um método que

possibilite a avaliação do cliente. A prototipagem é uma técnica recomendada para essa atividade.

3.2.1.2 Validar Requisitos utilizando critérios objetivos

Com o levantamento dos requisitos finalizado é preciso valida-los, cumprir a 6ª etapa da ER conforme explanado por Pressmam (2006) ou a 3ª etapa segundo Sommerville (2007) tratado anteriormente no ítem 2.3.6, ou seja, é preciso garantir que os requisitos listados atendam às necessidades e expectativas do cliente. Para isso, os requisitos identificados devem ser validados seguindo critérios objetivos, previamente estabelecidos. Atributos como: possuir identificação única, estar claro, não ser ambíguo, ser relevante, ser completo, estar consistente com os demais requisitos, ser implementável, testável e rastreável (IEEE std 830, 1998) ou utilizar outros critérios, devem estar de acordo com as necessidades da organização.

3.2.2 GRE 2 – Obtenção de comprometimento da equipe técnica com os requisitos aprovados

Um comprometimento formal pela equipe técnica deve ser firmado diante dos requisitos aprovados na GRE anterior, para isso o gerente do projeto deve apresentar para a equipe quais foram os requisitos definidos e deixar claro o que deverá ser produzido. Firmar o compromisso da equipe é uma prática muito importante visto que toda a produção das fábricas de software depende do potencial humano e não de maquinários.

A instituição deve utilizar diversos meios de comunicação e registro com sua equipe, dependendo a ocasião, que permitam registro como: um e-mail, uma reunião registrada em ata ou algum outro mecanismo. Novos comprometimentos devem ser firmados caso ocorra alterações nos requisitos, visto que impacta diretamente no cronograma da equipe.

A formalidade desse registro pode mudar de acordo com a politica organizacional da empresa, ou com a metodologia de gerência de projeto e de desenvolvimento de software. Independente da formalidade esse registro com a equipe deve existir, alguns recursos utilizados de registro estão descritos a seguir.

3.2.2.1 Reuniões

A reunião é prática mais utilizada nas empresas, envolvendo todos os participantes do projeto, ela geralmente é organizada pelo gerente do projeto onde são apresentados os requisitos acordados, discute-se ao máximo as informações com finalidade de esclarecer o máximo de dúvidas possíveis. Ao término, a coleta de assinatura dos presentes registra em ata o comprometimento e entendimento dos membros da equipe perante o projeto.

O fato de haver um grande número de envolvidos no projeto reunido permite um melhor esclarecimento e o registro rápido do compromisso são vantagens desta prática, entretanto pode ser oneroso por ter que parar toda a equipe ao mesmo tempo além de necessitar a garantia de disponibilidade dos envolvidos.

3.2.2.2 E-mail

O correio eletrônico também é utilizado como registro de compromisso, através dele o gerente do projeto pode esclarecer o plano do projeto, pode anexar planilhas, escopos, organogramas, ou tratar até mesmo de forma resumida do corpo do e-mail, podendo salvar sua resposta como forma de comprometimento. Por ser menos formal, o e-mail deve conter as informações de forma mais clara possível, e deve haver um lembrete que deve ser respondido como forma de entendimento. Com custo praticamente zero, ele permite a comunicação entre integrantes com indisponibilidade física, recebido de maneira quase instantânea pode trazer muito mais informações que uma ligação telefônica por exemplo.

3.2.2.3 Contratos

Um contrato é um vínculo jurídico entre dois ou mais sujeitos de direito correspondido pela vontade, da responsabilidade do ato firmado, resguardado pela segurança jurídica em seu equilíbrio social, ou seja, é um negócio jurídico bilateral ou plurilateral. É o acordo de vontades, capaz de criar, modificar ou extinguir direitos.

As cláusulas contratuais criam regras entre as partes, que por sua vez são subordinados ao Direito Positivo (princípios e regras que regem a vida social). As cláusulas contratuais não podem estar em desconformidade com a lei, sob pena de

serem nulas. No Brasil, cláusulas consideradas abusivas ou fraudulentas podem ser invalidadas pelo juiz, sem que o contrato inteiro seja invalidado.

Geralmente, contratos são mais usados quando terceiros participam do desenvolvimento, e onde sua “quebra” pode gerar em multa ou outro tipo de ônus.

3.2.3 GRE 3 – A rastreabilidade bidirecional entre os requisitos e os produtos de trabalho é estabelecida e mantida

O propósito da rastreabilidade é manter o controle dos requisitos, desde a fase de obtenção até a sua liberação, analisar como estes se relacionam com os demais requisitos (rastreabilidade horizontal) e com os artefatos ou produtos que são gerados no decorrer do projeto (rastreabilidade vertical).

A rastreabilidade deve ser feita de forma bidirecional, tanto na horizontal quanto na vertical, ou seja, é possível verificar todo o processo tanto “para trás” quanto “para frente”. A rastreabilidade bidirecional na vertical, possibilita o acompanhamento do requisito desde o escopo do projeto, passando pelo documento de requisitos até chegar ao código onde este foi implementado. Já a rastreabilidade bidirecional na horizontal permite analisar todos os requisitos que influenciam naquele que estamos avaliando e ver quais este influencia.

Uma alteração em um requisito pode implicar em sérias modificações nos demais requisitos que estão relacionados àquele. Com os requisitos devidamente mapeados é possível calcular o impacto com uma possível alteração devido sua rastreabilidade (TORANZO, 2002).

O autor ainda propõe uma classificação das informações a serem rastreadas em quatro níveis: ambiental, organizacional, gerencial e de desenvolvimento. O nível de informação ambiental representa as informações do contexto político, econômico e padrão (normas) que são externas à organização e que podem interferir em alguns dos seus sistemas. O nível de informação organizacional representa as informações (recurso, processo, objetivo, regra, etc.) que podem afetar o desenvolvimento dos sistemas da própria organização. O nível de informação gerencial representa alguma restrição, tarefa ou objetivo que são aplicados pela gerência de projetos. Finalmente, o nível de informação de desenvolvimento inclui as informações que representam os diferentes elementos ou artefatos produzidos no desenvolvimento de software, por exemplo, requisito, documento, diagrama e programa.

A rastreabilidade deve proporcionar uma análise prévia no impacto que uma possível mudança nos requisitos irá causar, tal estudo irá viabilizar ou não o projeto. Este resultado estabelece um sistema de rastreamento e que não necessariamente abrange a criação de uma matriz de rastreabilidade específica, contudo deve haver um mecanismo que possibilite a realização da rastreabilidade bidirecional entre os requisitos e os demais produtos de trabalho (SOFTEX, 2009b).

São várias as ferramentas voltadas para rastreabilidade, tanto livres quanto proprietárias, podendo também ser desenvolvidas pela instituição. Embora uma ferramenta seja extremamente importante para auxiliar a rastreabilidade, ela deve apenas servir como auxílio para uma estratégia predefinida e nunca ao contrário. Na próxima Seção são apresentados os tipos de rastreamento existentes.

3.2.3.1 Tipos de rastreamento

Existem diversas possibilidades de realizar um rastreamento de requisitos, levando em consideração: A direção do rastreamento, um requisito pode ser rastreado “para frente” ou “para trás”; A evolução do requisito, onde pode ser rastreado para aspectos que ocorrem “antes” ou “depois” da sua inclusão na especificação de requisitos; Os tipos dos objetos envolvidos onde tem-se um rastreamento interno ou externo.

3.2.3.1.1 Rastreamento para frente e para trás

De acordo com Wiering (1995) a definição de rastreamento para frente é a habilidade de rastrear um requisito para componentes de design ou implementação, já o rastreamento para trás é a habilidade de rastrear um requisito para uma pessoa, instituição, lei, argumento, ou seja, sua fonte.

Indica-se o rastreamento para frente quando um requisito é alterado e é necessário investigar o impacto nos produtos de trabalho. É possível identificar quais classes ou testes sofrem com a alteração do requisito, quais linhas de código precisam ser alteradas ou quais documentos precisam ser revisados e ajustados.

Já o rastreamento para trás deve ser realizado quando ocorre uma mudança e é necessário investigar a informação utilizada para elicitar o requisito alterado. Podem-se rastrear quais as pessoas interessadas na alteração do requisito ou a

partir de quais documentos o requisito foi extraído ou ainda, com quais departamentos da organização o requisito está relacionado.

3.2.3.1.2 Rastreamento pré-especificação e pós-especificação de requisitos

Uma especificação de requisitos é o resultado de um processo de elicitação. O rastreamento de um requisito pode ser realizado de duas formas: localizando informação sobre o processo de elicitação, antes da conclusão da especificação de requisitos, ou localizando informação relacionada ao seu uso, depois do requisito ter sido elicitado e incluído na especificação de requisitos (GOTEL e FINKELSTEIN, 1994).

O rastreamento pré-especificação de requisitos é útil quando ocorre uma mudança num requisito e é necessário localizar as fontes dos requisitos ou as pessoas responsáveis a fim de validar a mudança, descobrir como o requisito foi produzido.

Já o rastreamento pós-especificação de requisitos é útil para localizar o módulo de design no qual o requisito foi alocado ou os procedimentos de teste criados para verificar o requisito, descobrir como ele foi utilizado.

3.2.3.1.3 Extra-rastreamento e intra-rastreamento de requisitos

Durante o processo de elicitação de requisitos ou de elaboração de uma especificação os requisitos são amplamente apurados, derivados e alguns até descartados. A atividade de rastrear os relacionamentos entre os requisitos é chamada de intra-rastreamento de requisitos, também conhecida por rastreabilidade horizontal, a qual é importante identificar quais requisitos serão afetados por alguma mudança na evolução dos requisitos.

Já os relacionamentos entre os requisitos e outros produtos de software são capturados pelo extra-rastreamento de requisitos, ou rastreabilidade vertical, que é importante para a análise e tratamento de mudanças e evolução dos produtos de trabalho. Possibilitando observar através dessa rastreabilidade quais artefatos sofrem por alguma mudança de requisito.

3.2.4 GRE 4 – Revisões em planos e produtos de trabalho do projeto são realizadas visando identificar e corrigir inconsistências em relação aos requisitos

Este resultado possui como objetivo definir um método que descreva como realizar revisões em planos e produtos de trabalho do projeto, visando identificar e corrigir inconsistências em relação aos requisitos. Tais inconsistências devem ser registradas, avaliadas e acompanhadas até que sejam resolvidas.

As principais atividades que envolvem o monitoramento de requisitos são: Gerenciar mudanças nos requisitos acordados; Gerenciar os relacionamentos entre requisitos; Gerenciar as dependências entre o documento de requisitos e outros documentos produzidos ao longo do processo (KOTONYA e SOMMERVILLE, 1998).

O controle das alterações de requisitos no projeto se dá por constatação de novas necessidades ou por constatação de deficiências nos requisitos registrados inicialmente (DORFMAN e THAYER, 1990).

O quadro 5 apresenta o conjunto de atividades de um processo de monitoramento de requisitos, tais atividades visam apoiar a identificação, controle e rastreamento dos requisitos, bem como o tratamento das mudanças nos requisitos (HAZAN e LEITE, 2003).

Quadro 5 – Atividades de um processo de gerência de requisitos

Atividades	Descrição
Receber as solicitações de alterações de requisitos	O grupo de engenharia de requisitos recebe as solicitações de alteração dos requisitos por formulário padronizado ou por meio de um sistema de solicitação de demandas.
Registrar novos requisitos	Os requisitos são registrados em um modelo padrão descritivos ou num meio de um controle sistemático
Analisar impactos das mudanças de requisitos	Uma análise criteriosa deve ser conduzida para avaliar o impacto do requisito a ser incluído, alterado ou excluído sobre cada um dos seus requisitos relacionados, os quais podem ser identificados por meio das matrizes de rastreabilidade. Caso o impacto seja significativo, os requisitos (analisados e relacionados) devem ser revistos.
Elaborar relatório de impacto	Deve ser mantido um histórico de alterações para cada requisito, permitindo uma visão cronológica das principais mudanças nos requisitos.
Notificar os envolvidos	Os envolvidos são um conjunto de pessoas para as quais pode haver um impacto devido a modificações nos requisitos e devem ser notificados.
Coletar métricas	As métricas devem ser utilizadas e coletadas periodicamente para acompanhamento das atividades de Gerência de Requisitos.

Fonte: Hazan e Leite (2003).

Revisões de monitoramento e controle de projeto, inspeções baseadas em critérios explícitos para identificar inconsistências entre os planos, atividades e produtos de trabalho podem ser feitas simultaneamente pelo gerente do projeto. A prática de o monitoramento do projeto com o monitoramento dos requisitos pode trazer agilidade à organização.

Algumas técnicas e práticas sugeridas pelo Project Management Body of Knowledge (PMBOK) ou algumas já utilizadas para a Gerência de Projetos pela empresa podem dar suporte para essa atividade. Como Gerenciamento de integração, Gerenciamento do escopo, Gerenciamento de tempo do projeto, Gerenciamento de custos do projeto, Gerenciamento de recursos humanos, Gerenciamento das comunicações do projeto e Gerenciamento de riscos.

3.2.4.1 Técnicas existentes

A engenharia de requisitos aborda diversas técnicas que auxiliam o processo de revisões de requisitos, algumas delas são descritas a seguir.

3.2.4.1.1 Leitura baseada em perspectiva (PBR)

Esta técnica fornece um conjunto de procedimentos para auxiliar a detecção dos defeitos na inspeção de documento de especificação de requisitos. Para sua aplicação é necessário selecionar diferentes usuários do documento de requisitos, geralmente são utilizadas as figuras do usuário, do projetista e do testador (FREITAS, 2003).

Baseado no uso que cada usuário faz dos requisitos dentro do ciclo de desenvolvimento, são determinadas características da informação em diferentes níveis de importância. Uma vez definidas, os revisores atuam em cada perspectiva e avaliam o documento sob seus diferentes pontos de vista, assumindo posições distintas no ciclo de desenvolvimento.

As diferentes perspectivas cobrem coletivamente todos os aspectos relevantes do documento. Realiza-se uma classificação dos defeitos encontrados evitando duplicação de trabalho, fazendo com que cada revisor busque por defeitos que influenciarão no uso do documento pelo seu respectivo papel. A inspeção através destas perspectivas garante uma completa cobertura do documento, eliminando sobreposição de trabalho e ausência de inspeção em pontos do documento (FREITAS, 2003).

3.2.4.1.2 Revisões formais

As revisões de requisitos são mais relevantes nos estágios iniciais de um projeto, mas podem ser conduzidas a qualquer momento do ciclo de vida do projeto, de acordo com a necessidade. Uma revisão formal de requisitos é uma sessão de trabalho em grupo na qual participam todas as partes interessadas no projeto, são verificados aspectos como precisão, relevância, clareza, probabilidade de execução e conformidade com os padrões da organização.

Os participantes devem ler previamente a documentação de requisitos, para que durante a sessão sejam trabalhadas questões relevantes e mudanças necessárias. Os analistas de negócios precisam revisar cuidadosamente os itens que forem explanados durante a sessão, visto que todas as partes interessadas pelo projeto opinarão livremente.

A sessão de revisão é conduzida juntamente com as diretrizes de uma apresentação formal. É importante criar um registro da reunião ou da sessão para documentar as discussões e decisões acordadas, podendo haver algumas mudanças na documentação dos requisitos ou aos próprios requisitos.

3.2.5 GRE 5 – Mudanças nos requisitos são gerenciadas ao longo do projeto

Ocorrer mudanças nos requisitos são praticamente inevitáveis no processo de desenvolvimento, os requisitos podem ser modificados, adicionados ou retirados do projeto. Este resultado tem como principal objetivo controlar tais alterações, manter um plano com atividades, recursos e responsabilidades definidas são desafios a serem batidos.

Segundo HAZAN e LEITE (2003), a maioria das mudanças ocorre enquanto os requisitos estão sendo elicitados e analisados até sua validação, já os demais podem sofrer alterações durante o processo de desenvolvimento, sendo resultantes da combinação de fatores descritos no quadro 6.

Quadro 6 – Fatores de mudanças de requisitos

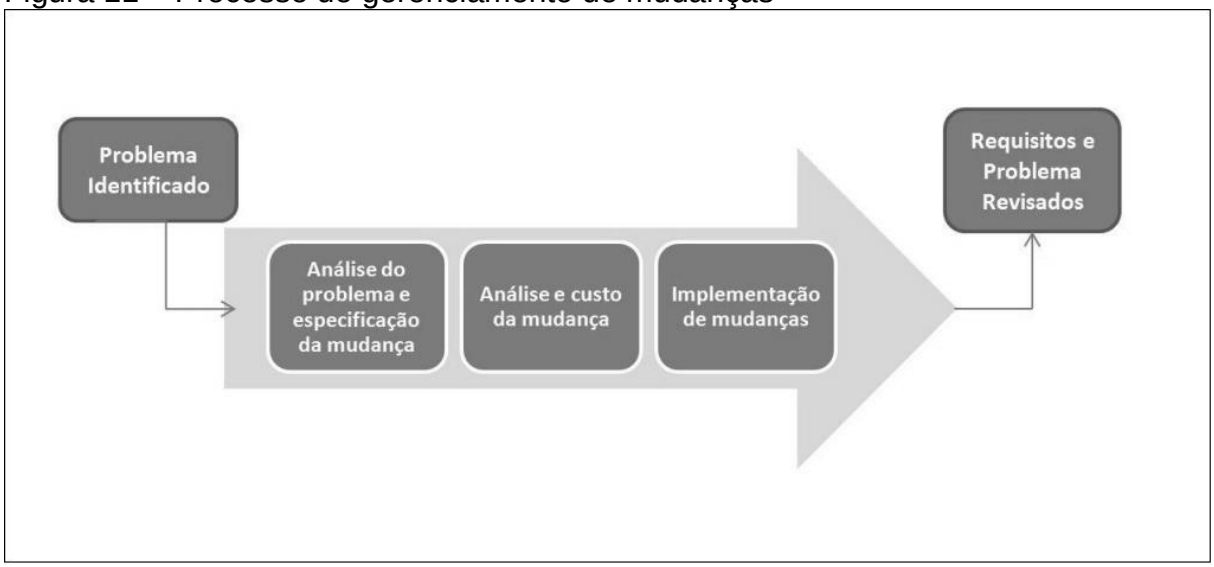
Fator de mudança	Descrição
Erros em requisitos, conflitos e inconsistências.	Conforme os requisitos são analisados e implementados, erros e inconsistências surgem e devem ser corrigidos.
Evolução do conhecimento do cliente	Conforme os requisitos são desenvolvidos, clientes e usuários finais desenvolvem uma melhor compreensão do que desejam.
Problemas técnicos de custo ou cronograma	Problemas podem ser encontrados na implementação dos requisitos. Pode ser muito custoso implementar certos requisitos
Mudanças nas prioridades do cliente	As prioridades do cliente podem mudar durante o desenvolvimento dos sistema como resultado de mudanças no ambiente de negócios.
Mudanças de ambiente	O ambiente no qual o sistema será instalado pode mudar, assim os requisitos devem ser modificados para manter compatibilidade.
Mudanças organizacionais	A organização que pretende usar o sistema pode mudar sua estrutura e processos, resultado em novos requisitos de sistema.

Fonte: Hazan e Leite (2003)

Sommerville (2003), na figura 11, demonstra a realização de alguns estágios que permitem o alcance de um gerenciamento de mudanças eficiente. Inicialmente o

problema ou a necessidade de mudança são identificados. A análise e custo da mudança estimam o impacto e o custo, utilizando as práticas de rastreabilidade de requisitos (GRE3) definem a viabilidade de realizar ou não a alteração do requisito. Por fim, a implementação de mudanças procede com as alterações no projeto e documentação.

Figura 11 – Processo de gerenciamento de mudanças



Fonte: Sommerville (2003).

Ter definido o processo de gerenciamento de mudanças é importante, para toda e qualquer mudança proposta nos requisitos o método definido deve ser aplicado para controlá-las. O principal benefício desta definição ocorre com as mudanças nos requisitos que são realizadas de forma controlada, pois há um processo que monitora as mudanças e exige que essas sejam documentadas.

Este capítulo apresentou uma visão geral a respeito da Gerência de requisitos. Foram apresentados os cinco (5) resultados esperados para GRE do nível G do MPS.BR e diversas técnicas que possibilitam a organização alcançar estes resultados, citando suas principais fases e características.

Uma organização que deseja empregar boas práticas de engenharia de software deve considerar a possibilidade de incluir as atividades indicadas nesse capitulo dentro das suas atividades de desenvolvimento. Para isso, é necessário que estas sejam vistas como atividades tão importantes como as outras atividades do desenvolvimento, mesmo que o custo do investimento nelas seja alto e talvez não tenha um retorno tão imediato.

O recomendável é sempre analisar os requisitos durante o processo de desenvolvimento, e não depois, ou apenas quando existir um problema ou proposta de mudança. No próximo capítulo, tendo como base os estudos realizados nesse capitulo, será apresentada a proposta de GRE que pretende atender as exigências do nível G do MPS.BR.

O estudo agora é voltado para de comparação dessas técnicas no âmbito das MPEs, a fim de capturar as principais características e qualidades que são interessantes e desejáveis no contexto da Engenharia de Requisitos, e que servirão de base para a elaboração da nova proposta de GRE.

4 PROPOSTA DE UM PROCESSO DE GRE QUE SATISFAÇA O NÍVEL G DO MPS-BR

Este capítulo apresenta a proposta do trabalho iniciando com um esclarecimento sobre o domínio das MPE (Seção 4.1). Logo a seguir, apresenta-se algumas técnicas que podem ser utilizadas para atender os cinco resultados esperados pela GRE do MPS.BR nível G (Seções 4.2 a 4.6) e quais estratégias de processos para Implantação da Engenharia de Requisitos podem ser usadas para o sucesso da implantação das técnicas (Seção 4.7). Por fim, a Seção 4.8 é destinada às considerações finais.

É importante ressaltar que a proposta exposta a seguir é baseada nas técnicas citadas no Capitulo 3, é destinada, mas não restrita, às micro e pequenas empresas. De acordo com órgãos competentes, como o SEBRAE (Serviço Brasileiro de Apoio às Micro e Pequenas Empresas) e o IBGE (Instituto Brasileiro de Geografia e Estatística), determina-se micro e pequena empresa por algumas características em comum, tais como: capital da empresa, faturamento e número de empregados.

4.1 GRE 1 – OS REQUISITOS SÃO ENTENDIDOS, AVALIADOS E ACEITOS JUNTO AOS FORNECEDORES DE REQUISITOS, UTILIZANDO CRITÉRIOS OBJETIVOS

Como dito anteriormente (Seção 3.2.1) as atividades de concepção dos requisitos pertencentes ao GRE1 pretendem fazer uso de mecanismos para elicitar, avaliar e validar a fim de garantir uma compreensão compartilhada. Para isso a técnica escolhida pela organização deve permitir: Entender o domínio do cliente de acordo com as atividades listadas a seguir:

a) Registrar os requisitos elicitados e seus fornecedores;
b) Avaliar os requisitos registrados utilizando critérios objetivos;
c) Fornecer informações suficientes para a organização aceitar ou não a produção dos projetos.

4.1.1 Entender o domínio do cliente

Fazer uso de uma técnica que permita o entendimento completo do domínio do cliente é essencial para que haja uma boa comunicação com os fornecedores de requisitos, consequentemente aumentando as chances de sucesso do projeto.

Manter um contato com o cliente facilitando o entendimento real de sua necessidade é uma das premissas para uma boa extração de requisitos, ajudando na compreensão das decisões tomadas durante o desenvolvimento evitando surpresas quando o sistema for construído e entregue.

Segundo Santander e Castro (2002) fazer uso de técnicas que não permitam um contato direto com o cliente até devem ser utilizadas, porém como um meio de extração secundário, de apoio. Utilizar para a GRE1 somente pesquisas ou estudo de mercado tornam-se incompletas pois não mantem um diálogo com o principal fornecedor de requisitos.

4.1.2 Registrar os requisitos elicitados e seus fornecedores

Registrar os requisitos elicitados servirá de comprovação de entendimento e comprometimento de todas as partes envolvidas no projeto, o documento de requisitos é a base para o desenvolvimento de software. Tal registro delimita a

abrangência do software, estabelece funcionalidades, impõe restrições de qualidade, fornece subsídios para o processo de verificação e validação do software construído.

A elaboração de um bom documento de requisitos possibilita estimativas de custos razoavelmente precisas e auxiliará na execução de um cronograma sem mudanças bruscas além de permitir a participação ativa dos usuários na validação do software.

Utilizar alguns critérios definidos previamente pela instituição facilitará na avaliação e na priorização dos requisitos. Registra-los textualmente contendo como mínimo os seguintes itens na sua descrição: Identificador Único, referencia os requisitos; Descrição Textual, explica e contextualiza os requisitos; Nome do Fornecedor, facilita o rastreamento de onde surgiram as informações; Nível de Importância que auxilia na priorização dos requisitos mais importantes.

4.1.3 Validar os requisitos registrados utilizando critérios objetivos

Todos os requisitos que foram anteriormente definidos documento de requisitos devem ser validados, o quadro 7 demonstra as características citadas pela IEEE std 830 (IEEE, 1998) onde é proposto um checklist para verificar a existência de algumas características.

Quadro 7 – Atributos do IEEE std 830

ATRIBUTOS	DESCRIÇÃO
ID. Única	Apresentar uma forma única de referência. Por exemplo: RF001, RNF003
Claro	Um requisito é claro se, e somente se, sua descrição não deixar duvidas, mesmo para aqueles que o lerem pela primeira vez.
Correto	Um requisito é correto se, e somente se, descreve uma ação que o software deve cumprir
Não Ambíguo	Um requisito é não ambíguo se, e somente se, a sua descrição não apresentar mais de uma interpretação
Relevante	Um requisito é relevante se, e somente se, ele está relacionado a uma funcionalidade, desempenho, restrições de design, atributos ou interfaces externas.
Coerente	Coerente refere-se à coerência interna. Se um requisito não concorda com algum documento ou outros requisitos, então não

	é correto.
Nível de Importância	Normalmente, todos os requisitos que dizem respeito a um produto de software não são igualmente importantes. Alguns requisitos podem ser essenciais, especialmente para aplicativos críticos no negócio, enquanto outros podem ser desejáveis.
Completo	Um requisito é completo se apresenta todas as informações necessárias em sua descrição e cumpre todos os itens citados acima. Se possível, é desejável que este especifique as respostas válidas e inválidas para valores de entrada e saída.
Implementável	O requisito deve apresentar características que permitam sua implementação.
Testável	O requisito deve apresentar características que permitam seus testes
Rastreável	O requisito deve apresentar características que permitam sua rastreabilidade horizontal e vertical.

Fonte: IEEE (1998).

Caso algum requisito não contemple algum atributo citado, este deve ser revisto, talvez uma melhor definição ou divisão deste nos demais requisitos possa facilitar o entendimento da importância deste requisito para o projeto. Outros critérios poderão ser estabelecidos pela organização a fim de tornar os requisitos ainda mais bem definidos.

4.1.4 Fornecer informações suficientes para a organização aceitar ou não a produção dos projetos

A etapa de validação dos requisitos indica que estão compreensíveis para todos os envolvidos no projeto, entende-se que o domínio do cliente foi entendido e os requisitos estão registrados de forma correta.

As técnicas a serem utilizadas para o atendimento deste nível devem fornecer no mínimo os seguintes itens para possibilitar o entendimento: Em qual contexto o projeto se insere; qual o escopo do projeto; permitir calcular os recursos necessários para o projeto; as prioridades do projeto; e o prazo para a entrega.

Na Seção seguinte é feito uma avaliação das técnicas já descritas na Seção 3.2.1.

4.1.5 Avaliação das técnicas

O quadro 8 apresenta um resumo da avaliação das técnicas descritas na secção 3.2.1 que podem ser usadas para alcançar os resultados esperados pela GRE1. Cada técnica avaliada em relação aos objetivos é definida em Não Atendido (NA), Parcialmente Atendido (PA) ou Totalmente Atendido (TA) de acordo com a competência para cada item avaliado. A avaliação das técnicas foi baseada no próprio entendimento e estudo realizado durante a execução deste trabalho.

Quadro 8 – Avaliação das técnicas de elicitação em relação aos objetivos da GRE1

Requisitos	Entrevista	Questionários	JAD	Brainstorming	Etnografia	Prototipagem	SWOT
Entendidos	TA	PA	TA	PA	TA	TA	PA
Registrados	TA	TA	TA	TA	TA	PA	TA
Avaliados	TA	TA	TA	TA	TA	TA	TA
Aceitos	TA	TA	TA	TA	TA	TA	TA

Fonte: Próprio Autor.

Na avaliação entendimento do domínio do cliente apenas três técnicas foram classificadas como PA, o questionário, por dificultar a comunicação e limitar as respostas do cliente; o brainstorming e o SWOT por serem técnicas mais voltadas para produtos novos e não exigirem a presença ou a consulta do cliente para o entendimento dos requisitos.

Apenas a prototipagem peca no registro dos requisitos, a prototipagem tem como objetivo gerar telas ou subprogramas para um melhor entendimento do domínio e não para documentação, onde muitas vezes os protótipos gerados são descartados. Já a avaliação dos requisitos não fica comprometida com o uso desta técnica, ela permite que o cliente visualize os requisitos e opine se concorda ou não com eles. Assim todas as técnicas comportam tanto a validação, por critérios objetivos, quanto a aceitação, que após sua validação já demonstra toda sua complexidade, permitindo aceitar ou recusar o projeto.

Considerando os fatores avaliados anteriormente são consideradas aptas na concepção desta proposta as técnicas: Entrevista, JAD e Etnografia que obtiveram em todos os itens avaliados: Totalmente Atendido. As demais, podem ser utilizadas como complementares.

4.2 GRE 2 – OBTENÇÃO DE COMPROMETIMENTO DA EQUIPE TÉCNICA COM OS REQUISITOS APROVADOS

Após a aprovação dos requisitos, o gerente do projeto deve apresentar para a equipe quais foram os requisitos definidos e deixar claro o que deverá ser produzido, garantindo um comprometimento formal da equipe técnica conforme discutido na secção 3.2.2.

Serrano et al (2008), levando em consideração o número de funcionário de uma MPE, sugere que tal apresentação deve ser feita em uma reunião, assim após a exposição das informações, é possível que haja uma discussão e esclarecimento de dúvidas coletivo.

O não comparecimento de algum membro na reunião poderá ser suprido pelo e-mail que garante seu comprometimento através de uma resposta a ser enviada após uma conversa de apresentação com um analista que participou da reunião, ou até mesmo utiliza-lo em projetos com um nível de complexidade menor.

Como citado no ítem 3.2.2, caso empresa necessite de uma formalidade maior ou haja terceirização, um contrato entre as partes pode garantir uma maior segurança para o andamento do projeto.

4.3 GRE 3 – A RASTREABILIDADE BIDIRECIONAL ENTRE OS REQUISITOS E OS PRODUTOS DE TRABALHO É ESTABELECIDA E MANTIDA

Como foi descrito na seção 3.2.3 o objetivo deste resultado é definição de um método que descreva como a organização fará a rastreabilidade bidirecional entre os requisitos e os produtos de trabalho, visando manter o controle dos requisitos e calcular o impacto deles durante o projeto. São diversas as formas na qual o rastreamento pode ser feito, porém, geralmente é utilizada uma ferramenta computacional devido ao grande número de informações e pelo tamanho das matrizes de rastreabilidade.

Tais ferramentas podem ser proprietárias, livres ou criadas pela empresa. A proprietária, geralmente é mais onerosa, porém oferece maior robustez e suporte; A livre que geralmente não possui custo, mas pode apresentar limitações em relação a

funcionalidade e correções de erros; Já a ferramenta própria pode ser desenvolvida de acordo coma as funcionalidades e características necessárias da instituição.

Essa proposta sugere a adoção de uma ferramenta livre que em casos Open Source(Código aberto), permitem a customização, ou no investimento em uma ferramenta própria, exclusiva da instituição, evitando gastos com software proprietários. Embora uma ferramenta seja extremamente importante para auxiliar a rastreabilidade, esta deve apenas servir como auxílio para uma estratégia predefinida e nunca ao contrário. A empresa deve ter clara qual a estratégia que será usada para a rastreabilidade bidirecional e quais artefatos serão rastreados.

Assim, são propostas a seguir as funcionalidades básicas mínimas necessárias para uma ferramenta deste tipo, uma estratégia de rastreabilidade bidirecional, bem como quais os potenciais artefatos que deverão ser rastreados.

4.3.1 Ferramentas de rastreabilidade

A utilização de um campo de rastreabilidade é uma opção para as ferramentas armazenarem as informações necessárias para a rastreabilidade bidirecional vertical e horizontal. Este campo permite ao desenvolvedor localizar onde devem ser feitas alterações e mostra quais partes do sistema podem sofrer influência com as alterações.

Conforme Quadro 9, no subcampo maior superior é registrado os níveis de implementação que vão da aplicação até o código e nos três subcampos menores inferiores estão os requisitos que são implementados pelo serviço e quais se relacionam a este.

Quadro 9 – Demonstração do campo de rastreabilidade

Aplicação / Package / Units, Formulários, Classes: **Nome da aplicação que sofreu alterações / nome dos pacotes onde houveram alterações / nome das unidades, formulários ou classes onde houveram alterações**		
Requisito do Projeto **ID do requisito implementado**	**Impactado por:** ID dos requisitos que impactam este	**Impacta os:** ID dos requisitos que são impactados este

Fonte: Próprio autor.

4.3.2 Rastreabilidade vertical

Para manter o controle dos requisitos, ver como eles se relacionam com os artefatos que são gerados no decorrer do projeto de forma bidirecional, inicialmente é necessário definir quais artefatos rastrear e de qual forma eles se relacionam com os requisitos e entre si.

Fazendo uso do campo de rastreabilidade citado anteriormente tem-se uma demonstração com a evolução nestes aspectos: Escopo, Requisitos, Serviços, Pacotes, *Units*, Classes ou formulários e código fonte.

Inicialmente o documento de requisitos já traz todo o escopo do projeto definido em requisitos, que logo são passados para a implementação no sistema denominado como serviços. No campo de rastreabilidade será definido a qual requisito tal serviço estará relacionado, e com finalidade de facilitar a localização na implementação, é necessário listar quais foram os pacotes que sofreram alterações no campo de rastreabilidade de cada requisito.

A rastreabilidade vertical deve definir quais os produtos que devem ser rastreados e o relacionamento destes com os requisitos como demostra o quadro 10.

Quadro 10 - Matriz de rastreabilidade de requisito X Produtos

	PRD 1	PRD 2	PRD 3	PRD 4	PRD 5	PRD 6
REQ 1		X	X			
REQ 2	X		X			
REQ 3		X			X	
REQ 4				X		
REQ 5					X	
REQ 6			X			X

Fonte: Próprio autor.

4.3.3 Rastreabilidade horizontal

A rastreabilidade bidirecional horizontal visa manter o controle dos requisitos, analisa como eles se relacionam com os demais requisitos. Para isso a empresa deve definir qual a forma de estabelecer o relacionamento entre os requisitos, sendo que para tal tarefa, se propõe a utilização de uma matriz de rastreabilidade conforme quadro 11.

Essa Matriz permite a previsão do impacto de uma mudança, inserção, exclusão de um requisito no sistema, de jeito que todos os requisitos são listados na primeira coluna e na primeira linha, e a intersecção entre elas demonstra a existência ou não de relacionamento entre os requisitos.

Quadro 11 - Matriz de rastreabilidade requisitos X requisitos

	REQ 1	REQ 2	REQ 3	REQ 4	REQ 5	REQ 6	REQ 7	REQ 8	REQ 9
REQ 1		X	X						
REQ 2	X		X						
REQ 3		X			X				
REQ 4				X					
REQ 5					X				
REQ 6			X			X	X	X	X
REQ 7				X					
REQ 8					X				
REQ 9			X			X	X	X	X

Fonte: Próprio autor.

Toranzo (2002) demonstra em seu trabalho que além da existência desta relação, é importante também registrar de forma subjetiva, se esta relação é forte ou fraca. Conforme mostra a figura 12, a descrição da relação dos requisitos que deve

ser realizada depois da matriz registra o impacto tanto para aqueles que o requisito faz referência, quanto para os referenciados.

Figura 12 – Demonstração de impacto de um requisito sobre outro.

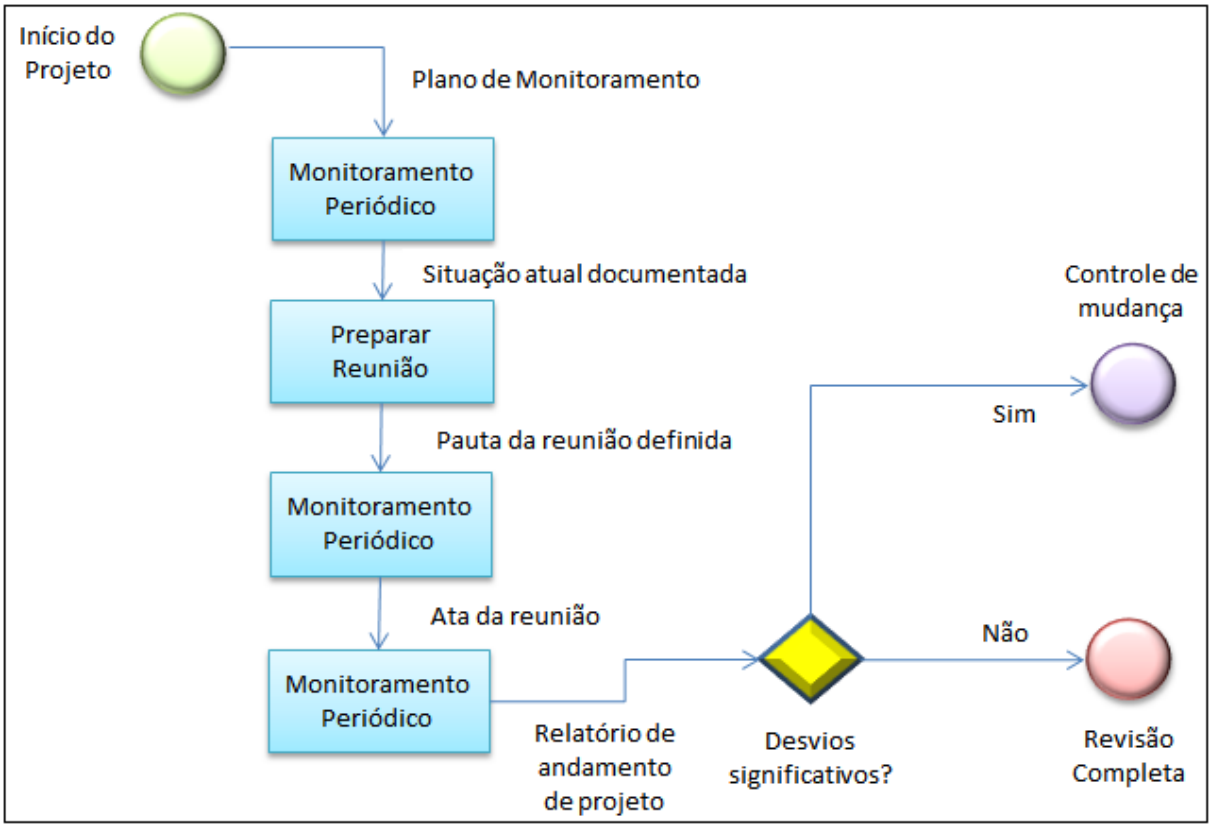
Fonte: Adapatado de Toranzo (2002)

4.4 GRE 4 – REVISÕES EM PLANOS E PRODUTOS DE TRABALHO DO PROJETO SÃO REALIZADAS VISANDO IDENTIFICAR E CORRIGIR INCONSISTÊNCIAS EM RELAÇÃO AOS REQUISITOS

O objetivo da GRE4 é definir um método que descreva como a organização fará revisões em planos e produtos de trabalho do projeto, visando identificar e corrigir inconsistências em relação aos requisitos. Esta identificação é feita controlando a evolução dos requisitos, seja por constatação de novas necessidades, ou por deficiências nos requisitos já registrados.

Conforme a proposta de Hazan e Leite (2003) o fluxograma apresentado na figura 13 demonstra alguns passos necessários para a administração das mudanças, visando a revisão de planos e produtos durante o projeto.

Figura 13 - Plano de revisão



Fonte: Próprio autor.

No monitoramento periódico o responsável irá monitorar os requisitos a fim de identificar desvio, alteração, adição ou exclusão, depois de identificados, serão registrados no histórico do projeto. No estágio seguinte o responsável irá definir a pauta para da reunião de revisão e envia-la para os participantes, além de levar opções de ações corretivas a serem discutidas na reunião.

O foco da reunião de revisão é identificar os desvios ocorridos, porque ocorreram, como corrigir e como evitar que ele volte a acontecer. No término da reunião o responsável deve criar um relatório do andamento do projeto, servindo para identificar o grau de conformidade com o documento de requisitos, o plano de projeto e seu andamento.

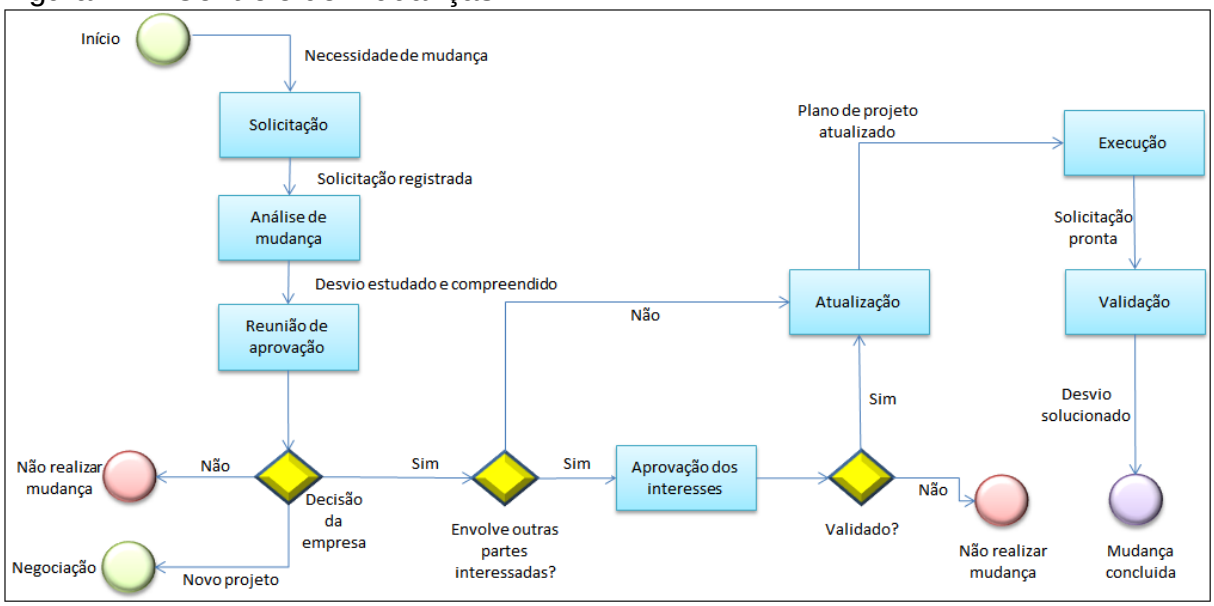
4.5 GRE 5 – MUDANÇAS NOS REQUISITOS SÃO GERENCIADAS AO LONGO DO PROJETO

Como dito anteriormente no ítem 3.2.5 o objetivo deste resultado é estabelecer o uso de mecanismos para gerenciar as mudanças nos requisitos ao longo do projeto, eles auxiliam a empresa em como criar uma solicitação de

mudança e nas decisões a serem tomadas. A análise desta tomada de decisão deve ser baseada no impacto que ela causará, por exemplo, em outros requisitos, cronograma ou custos.

Baseado no trabalho de Sommerville (2003) e nas características das MPEs, com intuito de controlar as mudanças nos requisitos durante o projeto foi criado um fluxograma que apresenta os passos necessários para a administração das mudanças conforme figura 14.

Figura 14 – Controle de mudanças.



Fonte: Próprio autor.

Inicialmente qualquer mudança que modifique algum requisito ou que possa alterar o custo, prazo, tamanho e/ou esforço do projeto é considerado um desvio e este deve ser tratado até sua conclusão. Na identificação de um desvio uma solicitação de mudança deve ser realizada, e precisa ser registrada no histórico da empresa, meio por onde ela será comunicada aos interessados.

Seguindo pela análise de mudança, um estudo para conhecer o impacto e a viabilidade deverá ser realizado. Tal estudo leva em consideração todos os aspectos que podem ser afetados com a solicitação de mudança, requisitos diretos e indiretos, projeto, custos, prazo, verificando se tal mudança será viável ou não.

A reunião de aprovação das mudanças indicadas terá como pauta a análise de mudança, ela deve ser feita pelos responsáveis de dentro da empresa na qual decidem se a alteração deve ou não ser feita, e dependendo da complexidade se deverá ser encarada como um novo projeto. Tal complexidade, justifica a

necessidade de que a empresa tenha sua opinião formada diante da solicitação de mudança antes do cliente, inclusive na hora da negociação.

Havendo a aprovação dos interessados no projeto em realizar as mudanças, o comprometimento pode ser firmado por assinaturas em ata, por e-mail ou por algum outro mecanismo formal definido pela organização.

Todas as alterações de escopo, custo, prazo e esforço, cronograma, plano de projeto, documento de requisitos, matriz de rastreabilidade e todos os ativos de processo impactados devem ser atualizados.

Só então as alterações são implementadas e testadas de acordo com a metodologia adotada usualmente pela empresa. Por fim a validação encerra o processo de solicitação de mudanças, nela o desvio solicitado deve estar totalmente implementado e testado satisfazendo totalmente os requisitos solicitados pelo desvio.

5 TRABALHOS CORRELATOS

Neste capítulo serão abordados os trabalhos correlatos que foram encontrados durante a pesquisa com abordagem no desenvolvimento e aplicação de processos e também levaram em conta o MPS.BR.

5.1 UMA SOLUÇÃO SISTÊMICA PARA A GERÊNCIA DE ATIVOS NO CONTEXTO DA IMPLEMENTAÇÃO DOS NÍVEIS DO MPS.BR

No trabalho realizado por Jesus (2009) verifica falta de uma ferramenta homologada pela SOFTEX para cooperar na implementação do modelo de qualidade MPS.BR, e como tal solução visa fornecer a catalogação de ativos e boas práticas para todos os níveis do MPS.BR. Para isto foi feito um estudo sobre o processo de software e as normas e modelos de qualidade com foco principal no modelo MPS.BR que foi desenvolvido para a realidade nacional.

Como conclusão do trabalho o mesmo desenvolveu o Siga-MPS.BR, que é uma solução para a catalogação das boas práticas. Demonstrou também a utilização desta ferramenta em certificação oficial para analisar a implementação tida como base para o modelo MPS.BR, que proporciona uma melhor gestão do conhecimento

dos ativos organizacionais usados como parâmetros para a execução de tal atividade.

5.2 MAPEAMENTO DOS PROCESSOS DE DESENVOLVIMENTO ÁGEIS EM RELAÇÃO AO MODELO DE MELHORIA DO PROCESSO DE SOFTWARE DO BRASIL (NÍVEL G)

Neste trabalho Santos (2009) fez uma análise dos resultados alcançados utilizando métodos ágeis de desenvolvimento de software para certificação do nível G do modelo MPS.BR. Teve como objetivo do trabalho um guia para que as empresas possam se basear para adquirir a certificação do nível G.

Na conclusão do trabalho o mesmo verificou que as metodologias ágeis não atingem todos os objetivos necessários para certificação do nível G pelo fato de que alguns requisitos do modelo MPS.BR não deixam de forma clara a necessidade ou não do registro de informações da execução de tal tarefa.

5.3 MAPEAMENTO DOS REQUISITOS DA GERÊNCIA DE PROJETOS DO NÍVEL E DO MPS.BR APLICANDO AS PRÁTICAS DO SCRUM

No trabalho realizado por Rodrigues (2012) ele realizou uma pesquisa para verificar o relacionamento da metodologia ágil Scrum com os resultados esperados pelo nível E do MPS.BR. Apresentando um mapeamento e avaliação dos requisitos da Gerência de Projetos do nível E do modelo de qualidade Melhoria do Processo de Software Brasileiro aplicando-se as práticas da metodologia ágil de desenvolvimento Scrum.

Ao final das avaliações é apresentada a análise das mesmas e a conclusão, definindo se com a metodologia ágil Scrum é possível atender os requisitos da Gerência de Projetos do nível E do modelo de qualidade Melhoria do Processo de Software Brasileiro.

6 CONCLUSÃO

Este trabalho de pesquisa me proporcionou um vasto ganho de conhecimento em relação aos conceitos de Engenharia de Software, Engenharia de Requisitos, Qualidade de Software e principalmente o MPS.BR com o aprofundamento no seu nível mais básico de aplicação, o G. Onde traz diversos resultados esperados para que a instituição atinja através da execução de um processo de gerência de requisitos

Ao longo dos anos pesquisas demonstram que a quantidade de projetos de software que chegam ao sucesso, contemplando todos os requisitos iniciais, é a minoria do todo.

Com o decorrer dos anos, o aumento na qualidade de software tem sido relevante com a procura das instituições para certificação nas ISO/IEC 12207, ISO/IEC 15504 e especialmente o MPS.BR.

O modelo MPS.BR foi desenvolvido a partir da necessidade de aplicar um modelo de processo eficiente mas que fosse adequado às pequenas e médias empresas. Com a certificação para o nível G onde abrange a Gerência de Requisitos já ajudou muitas empresas a melhorarem a qualidade de seu software.

O estudo apresentou diversos autores que defendem há muitos anos a necessidade de um bom processo de gerencia de requisitos, através dele a empresa conseguirá entender bem a necessidade do cliente, verificar, validar, documentar e gerenciar seus requisitos ao longo do software.

Foram analisadas várias técnicas quem visam fornecer um subsídio maior desde a fase de elicitação até a gerência ao longo do ciclo de vida de um software.

Entende-se que os resultados obtidos neste trabalho de pesquisa encontram-se a importância da certificação ao nível G do MPS.BR, principalmente para as pequenas e médias empresas que ainda possuem um número muito pequeno de empresas certificadas em território nacional e principalmente por ser o principal foco do programa. A proposta de processo apresentada neste trabalho pode fornecer a muitas empresas subsídios para um aperfeiçoamento na gerência de requisitos, diretamente ligada a qualidade de software apresentada no final do projeto.

Sugere-se como trabalhos futuros a aplicação de uma ferramenta (E-FLOW PROJECT ou CONTROLA) em um estudo de caso baseado nesta proposta.

Também sugere-se a avaliação desta proposta para o atingimento de outro nível do MPS.BR que também trata a GRE.

REFERÊNCIAS

ATAÍDES, Adriana da Costa. **UM MÉTODO PARA ACOMPANHAMENTO E CONTROLE DA IMPLANTAÇÃO DO CMMI.** 2006. 125 f. Dissertação (Mestrado) - Universidade de Brasília, Brasília, 2006. Disponível em: <http://repositorio.bce.unb.br/bitstream/10482/6733/1/ADRIANA%20DA%20COSTA%20A TA%C3%8DDES.pdf>. Acesso em: 10 out. 2013.
Calsalvara, A., C. A. F. Machado, S. S. Reinehr, e R. C. Burnett. “Norma NBR ISO/IEC 12207.” 2000

BERGMANN, T, O. **Implantação do MPS.BR Nível G**. TCC – Bacharel, UFSC, Florianópolis, 2008.

BERTANI, Jôber Ferreira, BORGES, Andréia da Silva. **Critérios para avaliação de uma ferramenta de Engenharia de Requisitos**: Um Experimento Prático com o E-FLOW PROJECT e CONTROLA. Monografia, Universidade Católica de Salvador, Bahia, 2007

BLASCHEK, J. R. **Gerência de Requisitos**: O principal problema dos projetos de Software. Developer’s Magazine, Rio de Janeiro, Agosto, 2002.

CARVALHO, A. E. S. et al. **Uma Estratégia para Implantação de uma Gerencia de Requisitos Visando a Melhoria Dos Processos de Software**. In: Anais do WER01 - Workshop em Engenharia de Requisitos, 2001, Buenos Aires, 2001.

CHUNG, L., NIXON, B., YU, E. and MYLOPOULOS,J. **Non-Functional Requirements in Software Engineering**. Kluwer Academic Publishers, 1999.

COLENCI NETO, Alfredo. **Proposta de um modelo de referência para desenvolvimento de software com foco na certificação do MPS.BR**. 2008. 179 f. Tese (Doutorado) - Universidade de São Paulo, São Carlos, 2008. Disponível em: <http://www.teses.usp.br/teses/disponiveis/18/18140/tde-29012009-093822/publico/AlfredoColenciNeto.pdf>. Acesso em: 21 nov. 2013.

CONNELL, Steve. **Rapid Development: Taming Wild Software Schedules**. Microsoft Press. 1996.

COSTA FILHO, Edes Garcia da et al. **Padrões e Métodos Ágeis: agilidade no processo de desenvolvimento de software**. Disponível em: <http://subversion.assembla.com/svn/TCC_Agil/9673.pdf>. Acesso em: 07 maio 2013.

CYSNEIROS, L. M. **Requisitos Não Funcionais**: Da Elicitação ao Modelo Conceitual. *PUC, Rio de Janeiro*, 2001.

DAVIS, A. M. e HICKEY A, M. (2002) - **Requirements Elicitation and Elicitation Technique Selection**: A Model for Two Knowledge-Intensive Software Development Processes - Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS’03) IEEE.

DORFMAANN, M e THAYER, R. **Software Requirements** - a management perspective - Systemand Software Requirements Engineering. Institute of Electrical and Electronics Engineers, Inc., 1990.

ENGINEERS' COUNCIL FOR PROFESSIONAL DEVELOPMENT. (1978). **Engineering education and accreditation report**, 1977. New York, The Council.

EVIDÊNCIAS sobre o desempenho das empresas que adotaram o modelo MPS-SW desde 2008. SOFTEX. IMPS, 2012. Disponível em: <http://www.softex.br/mpsbr/_livros/arquivos/Softex_iMPS_2012_Portugues.pdf>. Acesso em: 25 maio 2013.

FEILER, P; HUMPHREY, H; **Software process development and enactment**: Concepts and definitions. IEEE Computer Society Press, 1993.

FREITAS, M. E. et al. **Inspeção de Documentos de Requisitos Baseado em Técnica de Leitura PBR**: Experiência Prática no CPqD. In: Anais do Simpósio Brasileiro de Qualidade de Software, Brasília, 2003.

FINE, L. G. **The SWOT Analysis**: Using your Strength to overcome Weaknesses, Using Opportunities to overcome Threats, 1 ed. Booksurge llc, 2009.

GASTALDO, D.L. e MIDORIKAWA, E.T. **Processo de Engenharia de Requisitos Aplicado a Requisitos Não-Funcionais de Desempenho** – Um Estudo de Caso. 2003.

GOGUEN, J. A. e LINDE, C. **Techiniques for Requirements Elicitation**. In: Proceedings, Requirements Engineering 1993, IEEE CS Press, 1993.

GOTEL, O; FINKELSTEIN, A. **An Analysis of the Requirements Traceability Problem. In**: Proceedings of the First International Conference Requirements Engineering; Colorado Springs, IEEE CS Press, 1994.

HAZAN, C. e LEITE, J. C. S. P. **Indicadores para a Gerência de Requisitos**. In: Anais doWER03 - Workshop em Engenharia de Requisitos, 2003, Piracicaba, 2003, pp 285-301. IEEE Std 830-1998.

HEUMANN, Jim. **The Five Levels of Requirements Management Maturity**. Rational Edge. 2003. Disponível em: <http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/feb03/ManagementMaturity_TheRationalEdge_Feb2003.pdf>. Acesso em: 22 dez. 2013.

HOFMANN, H.F. e LEHNER, F. **Requirements Engineering as a Success Factor in Software Projects**. Vol. 18. 4 vols. 2001.

IEEE, **Recommended practice for software requirements specifications**, Institute of Electrical and Electronics Engineers, 1998.

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James; **UML** – Guia do Usuário. Tradução Fábio Freitas da Silva. Rio de Janeiro: Campus, 2000.

JESUS, Anderson Jones Silva de. **UMA SOLUÇÃO SISTÊMICA PARA A GERÊNCIA DE ATIVOS NO CONTEXTO DA IMPLEMENTAÇÃO DOS NÍVEIS DO MPS.BR** Trabalho. 2009. 79 f. Dissertação (Bacharel) - Universidade Federal Do Pará, Belém, 2009. Disponível em: <http://www.spider.ufpa.br/projetos/siga_mpsbr/tcc.pdf>. Acesso em: 19 nov. 2013.

KENDALL, **Systems Analysis and Design**, 8 ed., Pearson, 2010.

KOSCIANSKI, A; et tal. **Qualidade de Software**: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software. São Paulo: Novatec Editora, 2006.

KOTONYA, SOMMERVILLE. **Requirements Engineering**: Processes and Technique, 1 ed, Wiley. 1998.

KRUCHTEN, P. **The Rational Unified Process**: Na Introduction, 3 ed, Addison Wesley, 2003.

LARMAN, C. **Agile e Iterative Development**: A Manager's guide. Addison Wesley, 2004.

LEITE, J. C. S. P. **Gerenciando a Qualidade de Software com Base em Requisitos**. In: Qualidade de Software: Teoria e Prática. 1ª. Edição. Prentice Hall. p 238-246, 2001.

MACAULAY, L., **Requirements Engineering**, 2 ed. Springer, 1996.

MACHADO, M. **Métricas e Qualidade de Software**. Departamento de Informática – Universidade Federal do Espírito Santo, 2001.

MACIASZEK, L. A. **Requirements Analysis and System Design** - Developing Information Systems with UML, Addison Wesley, 378p, 2001.

MCCONELL, Steve. **Rapid Development**: Taming Wild Software Schedules. Microsoft Press. 1996.

MCT. **Ministério da Ciência e Tecnologia**, Disponível em: <www.mct.gov.br>, Acessado em: 18 de Jul. 2014.

NATO, **Software Engineering Conference**, Garmisch, Germany, 1968.

NOGUEIRA, M. O. **Qualidade no Setor de Software Brasileiro**, Tese de Doutorado, COPPE/UFRJ, abril 2006.

PÁDUA, W. **Engenharia de Software**: Fundamentos, Métodos e Padrões. 2. ed. Rio de Janeiro: LTC, 2003.

PARASURAMAN, A. Z. **Marketing Research**. 1 ed. Addison Wesley, 1991.

PATRÍCIO, Nathalia Sautchuk, **Engenharia de Requisitos em Software para E-Learning**. Dissertação(Mestrado). Escola Politécnica da Universidade de São Paulo. São Paulo, 2013

PETERS, J. F; et al. **Engenharia de Software**. Tradução Ana Patrícia Garcia. Rio de Janeiro: Campus, 2001.
PFLEEGER, S. L. **Engenharia de Software**: Teoria e Prática, Prentice Hall do Brasil, 2ª Edição, 2004.

PRADO, Moacir de Sousa. **Engenharia de Software**. Disponível em http://www2.dem.inpe.br/ijar/CicoloVidaSoftPrado.html, acessado em novembro de 2007.

PRESSMAN, R. S. **Engenharia de Software**. 6 ed, McGraw Hill, 2006.
REZENDE, Denis Alcides. **Engenharia de Software e Sistemas de Informação**. 3° Rio de Janeiro: Brasport, 2005. 313 p.

RODRIGUES, A. R. dos S. **Mapeamento dos requisitos da Gerência de Projetos do nível E do MPS.BR aplicando as práticas do scrum**. UNESC, criciúma, 2012.

SANTOS, Rafael Liu. **Emprego de Test Driven Development no Desenvolvimento de Aplicações.** 2010. 108 f. Dissertação (Bacharelado) – Universidade de Brasília, 2010

SCHMIDT, Michael E.C. Implementing the IEEE Software Engineering Standards, SAMS, 2000

SEI. **Pagina Oficial CMMI**, Disponível em: <www.sei.cmu.edu/cmmi/index.cfm>, Acessado em: 17 de Jun. 2014.

SERPRO, Disponível em: <http:// http://www.serpro.gov.br/>. Acessado em: 18 de Setembro de 2010.

SERRANO, M. et al. **Uma Proposta para Avaliação de Equipes de Requisitos. In**: Anais do WER08 - Workshop em Engenharia de Requisitos, 2008, Barcelona, 67-75, 2 SERRANO, M. et al. Uma Proposta para Avaliação de Equipes de Requisitos. In: *Anais do WER08 - Workshop em Engenharia de Requisitos,* 2008, Barcelona, 67-75, 2008.008.

SOMMERVILLE. Ian. **Engenharia de Software.** Addison Wesley, São Paulo, 2003.

SOMMERVILLE, I. (2005) - **Innovation in Requirements Engineering** - Integrated Requirements Engineering: A Tutorial – IEEE p16 - 23

SOARES, Felipe S. Furtado et al. **Adoção de SCRUM em uma Fábrica de Desenvolvimento Distribuído de Software**. Recife: 2007. Disponível em: <http://200.17.137.110:8080/schisto/publicacoes/i-wdds/adocao-de-scrum-em-uma-fabricade-desenvolvimento.pdf>. Acesso em: 20 out. 2011.

SOFTEX (Org.). MPS.BR - Melhoria de Processo do Software Brasileiro: **Guia de implementação - Parte 1: Fundamentação para Implementação do Nível G do MR-MPS.** Campinas, 2009. 31 p. Disponível em: <http://www.softex.br/mpsbr/_guias/guias/MPS.BR_Guia_de_Implementacao_Parte_1_2009. pdf >. Acesso em: 23 jun. 2013.

SOFTWARE ENGINEERING INSTITUTE. **Improving processes for acquiring better products and service. CMMI® for Acquisition**, Version 1.2, November 2007. Technical Report CMU/SEI-2007-TR-017 - ESC-TR-2007-017.

STANDISH GROUP. **Report Chaos**, The Standish Group 2013. Disponível em: http://www.projectsmart.co.uk/docs/chaos-report.pdf. Acesso em: 17 set. 2014.

SWEBOK. Guide to the software Engineering Body of Knowledge. Disponível em:http://www.computer.org/portal/web/swebok/html/ch1. Acesso em: 21 de set. 2013

TORANZO, M. A. **Uma Proposta para Melhoria de Requisitos de Software**. Tese de Doutorado, UFPE, setembro 2002.

YOURDON, **Declínio e queda dos analistas e programadore**, 3 ed, Makron Books, 1995.

WEBER, Kival Chaves et al. **Modelo de Referência e Método de Avaliação para Melhoria do Processo de Software - versão 1.0 (MR-MPS e MA-MPS). In: IV, 4., 2005, Florianópolis.** Modelo de Referência e Método de Avaliação para Melhoria do Processo de Software - versão 1.0 (MR-MPS e MA-MPS). Florianópolis: Softex, 2005. p. 1 - 14. Disponível em: <http://golden.softex.br/portal/softexweb/uploadDocuments/MR_MAMPS.pdf>. Acesso em: 26 jun. 2013.

WIERING, R. J. **An introduction to requirements traceability**. Faculty of Mathematics and Computer Science, University of Vrije, Amsterdam, Setembro 1995.

ZAVE, P. **Classification of Research Efforts in Requirements Engineering – ACM Computing Surveys,** Vol. 29, No 4, 1997.

APÊNDICE(S)

APÊNDICE A – Artigo

Proposta de um processo de gerenciamento de requisites para satisfação da Gerência de Requisitos do nível G do MPS.BR

Aender de S. Saturnino[1] Prof Msc. Paracelso de O. Caldas[2]

[1]Acadêmico de Bacharelado em Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC)
Caixa Postal 3167 – Criciúma – SC – Brasil

[2]Professor de Bacharelado em Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC)
Caixa Postal 3167 – Criciúma – SC – Brasil

aender_ss@hotmail.com, poc@unesc.net;

Abstract. *With the growth of the software market along with the technology, the search for constant improvement in order to offer quality is increasing. The design of the customer's need, validation, documentation and requirements management are some of the processes advocated by the Requirements Management. The MPS.BR through its lowest level, the G shows five results expected to ensure the minimum necessary for the institution to treat adequately the requirements of their projects. This work brings a Requirements Management process that seeks to meet these results, focused primarily on small and medium enterprises.*

Resumo. *Com o crescimento do mercado de software aliado a tecnologia, a busca por constantes melhoria afim de oferecer qualidade é cada vez maior. A concepção da necessidade do cliente, a validação, a documentação e a gerência de requisitos são alguns dos processos defendidos pela Gerência de Requisitos. O MPS.BR através do seu nível mais baixo, o G, demonstra cinco resultados esperados para se garantir o mínimo necessário para que a instituição trate de maneira adequada os requisitos de seus projetos. Este trabalho traz um processo de Gerenciamento de Requisitos que busca atender estes resultados, focado principalmente a pequenas e médias empresas.*

1. INTRODUÇÃO

O último relatório gerado pelo Standish Group em Março/2013 demonstrou que o percentual de falha em projetos de Tecnologia de Informação (TI), em 2012, a cada 100 projetos iniciados somente 39 atingiram o sucesso, 43 sofreram alterações ao longo do processo e 18 fracassaram. Comparando esses dados com os dados do ano 1994, percebemos uma melhora não muito significativa, que naquele ano a cada 100 projetos, apenas 16 atingiam ao sucesso, 53 sofriam alterações e 31 fracassavam.

Diversos autores ao longo do tempo defendem a importância de um correto tratamento dos requisitos de um projeto, desde sua identificação até sua

documentação. Eles afirmam que por falharem neste item, os números de software que obtiveram sucessos, são tão baixos.

A MPS.BR, em seu nível G, considera a importância de se controlar os requisitos, descreve 5 resultados que a organização deve alcançar no seu projeto implementando algum tipo processo. E o objetivo deste trabalho é justamente desenvolver uma proposta de processo de gerenciamento de requisitos que visa atender os resultados esperados por este nível.

A etapa de análise de requisitos que abrange as técnicas de elicitação ou levantamento trata da identificação das regras, das necessidades dos usuários de informações e comunicação destas necessidades no processo de construção do software (GASTALDO e MIDORIKAWA, 2003).

A Melhoria de Processo de Software (MPS) é um modelo de aperfeiçoamento e avaliação do processo de software destinado preferencialmente a pequenas e médias empresas. A Melhoria do Processo de Software Brasileiro (MPS.BR) é um projeto que visa a melhoria do processo de software do país e sua novidade está na estratégia de implementação, criada para a realidade brasileira. A base técnica utilizada para a construção deste modelo de qualidade foram as normas ISO/IEC 12207, ISO/IEC 15504 além do conteúdo de outros modelos de referência como Capability Maturity Model Integration (CMMI) (WEBER, 2005). Cada nível de maturidade do MPS.BR estabelece patamares de evolução de processos. Os níveis são definidos em: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado).

Considerando a importância de controlar os requisitos, o MPS.BR traz no seu primeiro nível de maturidade (G) 5 metas, ou resultados esperados, sobre a Gerência de Requisitos (GRE), atividade na qual está baseada na Engenharia de Requisitos, que tem como principais atividades: elicitação, análise e negociação, especificação, validação e gerenciamento de requisitos (SOMMERVILLE, 2003).

2. ENGENHARIA DE SOFTWARE

De acordo com Pressman (2005) a Engenharia de Software é uma área do conhecimento da informática voltada para a especificação, desenvolvimento e manutenção de sistemas de software aplicando tecnologias e práticas de ciência da computação, gerência de projetos e outras disciplinas, objetivando organização, produtividade e qualidade.

Segundo o Swebok (2011), a Engenharia de Software está subdividida em requisitos de software, testes de software, gerência de engenharia de software, ferramentas e métodos de engenharia de software, qualidade de software, entre outras. No processo há uma integração e uma sequencia coerente de praticas que objetivam a evolução do sistema, onde as atividades de especificação, projeto, implementação e testes são atingidas.

Um desafio constante da área de Engenharia de Software é melhorar o processo de desenvolvimento de software, cujo objetivo é formar a ligação entre métodos e ferramentas, e mesmo com a constante evolução destes atributos, a entrega de software em prazos estabelecidos nem sempre é alcançada. Existe a necessidade de desenvolver software de forma mais rápida e produtiva, mas com qualidade (COSTA FILHO, 2005).

Segundo Pressman (1995) os três elementos fundamentais, ferramentas, métodos e processos passam por três fases genéricas, a saber:

d) Definição: identifica a função do software a ser desenvolvido. Nessa fase acontecem a análise de requisitos e análise e projeto do software.

e) Desenvolvimento: executa o que foi planejado, nessa fase ocorrem os testes e codificação do software.

f) Manutenção: correções, adaptação e melhorias no software.

3. ENGENHARIA DE REQUISITOS

É nessa etapa que o engenheiro tenta entender quais são as reais necessidades dos usuários, qual o problema que se deve resolver, solucionar ou dissolver. Onde se começa a identificar as dimensões humanas e sociais relacionadas ao software a ser desenvolvido.

Sommerville (2003) define Engenharia de Requisitos (ER) como um conjunto de atividades para ajudar a equipe de projeto a identificar, controlar e rastrear requisitos e alterações do sistema ao longo do ciclo de desenvolvimento do software.

3.1 Problemática

Hofmann (2001) reafirma que os requisitos deficientes ainda são a maior causa de falhas nos projetos de software. Capers Jones descobriu, por meio de pesquisas realizadas com centenas de organizações, que a engenharia de requisitos é deficiente em mais de 75% das empresas (JONES, 1996, apud HOFMANN, 2001, p. 58).

Um estudo realizado por Gastaldo (2003) demonstrou que cerca de 50% do retrabalho referente ao processo de desenvolvimento de software ocorre nas fases iniciais de elicitação, análise e documentação de requisitos conforme demonstra a figura 3. Grande parte das causas do retrabalho é relacionada aos requisitos não funcionais de desempenho. Sua pesquisa revelou que por não serem levados em consideração desde o início do projeto, estes requisitos acabam não sendo atendidos.

4. GERÊNCIA DE REQUISITOS

Consiste no processo de gerenciar mudanças nos requisitos do sistema, durante o projeto. Os requisitos de um sistema podem sofrer alterações para refletir as necessidades de mudanças solicitadas pelos envolvidos, surgindo novas necessidades, identificando novos atributos em funcionalidades já em desenvolvimento, mudanças nas leis e regulamentações.

Estas mudanças devem ser gerenciadas para garantir que estejam dentro do orçamento e do prazo estipulados. As principais atividades da gerência de requisitos são o controle da mudança e a avaliação do impacto da mudança.

Neste sentido, a gerência de requisitos exige que a rastreabilidade dos requisitos seja armazenada, demonstrando as ligações entre os requisitos, as fontes de requisitos. Os projetos do sistema devem ser rastreados e guardados, de maneira que estejam disponíveis no momento de avaliar o impacto e controlar as alterações que os requisitos vierem a sofrer (SOMMERVILLE 2003).

Para Sommerville (2003) a gerência de requisitos controla as mudanças dos requisitos e deve iniciar quando já tiver um esboço das suas especificações, lembra a necessidade de atualizar a documentação sempre que houver alterações. As consequências decorrentes de tais modificações também variam muito, pois levam a alterações no código e na especificação do produto interferindo no seu prazo de entrega, no custo e até mesmo na relação cliente e fornecedor podendo gerar insatisfação e cancelamento do serviço.

5. QUALIDADE DE SOFTWARE

Completando o pensamento, Koscianski (2006) define a qualidade de software como uma área da Engenharia de Software que tem como objetivo garantir a produção de sistemas satisfatórios e que atendam aos custos e cronogramas estabelecidos.

Para Pádua (2005) a base para desenvolver sistemas com qualidade é possuir uma boa especificação de requisitos, e esta deve apresentar uma documentação precisa, correta, consistente, completa, permitindo alterações ao longo do projeto, mensurando a sua origem. Deve permitindo identificar previamente os impactos das alterações e demonstrar a priorização dos requisitos de acordo com seu grau de importância.

5.1 MPS.BR

A Melhoria do Processo de Software (MPS) é um modelo de melhoria e avaliação de processo de software. Inicialmente seu foco eram pequenas e médias empresas, normalmente grupos em um Modelo de Negócio Cooperado (MNC), mas também vem sendo aplicado em organizações de grande porte no Modelo de Negócio Específico (MNE), este modelo é a principal base para o projeto a seguir (SOFTEX, 2009).

O MPS.BR é um programa federal para melhoria de processo do software brasileiro que procura incentivar e apoiar a indústria de software nacional na melhoria desenvolvimento do processo de desenvolvimento. Teve início em dezembro de 2003 e é coordenado pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), contando com apoio do Ministério da Ciência e Tecnologia (MCT), da Financiadora de Estudos e Projetos (FINEP) e do Banco Interamericano de Desenvolvimento (BID) (SOFTEX, 2006).

Para criação desse modelo utilizou-se como base técnica algumas normas e modelo de maturidade como a norma ISO/IEC 12207, a ISO/IEC 15504 e o modelo CMMI. O modelo pretende estabelecer um processo e um método de avaliação, o qual dá sustentação e garante que o MPS.BR está sendo empregado de forma coerente com as suas definições.

Modelo este composto por sete níveis de maturidade que são sequencias e cumulativos, cada nível é composto por um conjunto de processos em um determinado nível de capacidade onde o progresso e o atendimento de cada nível de maturidade é obtido somente quando são atendidos todos os requisitos e atributos relacionados àquele nível (SANTANA, TIMÓTEO, VASCONCELOS, 2006).

Nível G (Parcialmente Gerenciado)**:** O primeiro nível é o ponto de partida para a implantação de melhoria dos processos de software na organização. Ao alcançar este nível a organização deve focar na criação de mecanismos adequados para o

planejamento, monitoramento e controle de projetos e gerenciar os requisitos ao longo do desenvolvimento.

6. PROPOSTA

Apresenta-se algumas técnicas que podem ser utilizadas para atender os cinco resultados esperados pela GRE do MPS.BR nível G e quais estratégias de processos para Implantação da Engenharia de Requisitos podem ser usadas para o sucesso da implantação das técnicas.

É importante ressaltar que a proposta exposta é destinada, mas não restrita, às micro e pequenas empresas. De acordo com órgãos competentes, como o SEBRAE (Serviço Brasileiro de Apoio às Micro e Pequenas Empresas) e o IBGE (Instituto Brasileiro de Geografia e Estatística), determina-se micro e pequena empresa por algumas características em comum, tais como: capital da empresa, faturamento e número de empregados.

6.1 GRE 1 – OS REQUISITOS SÃO ENTENDIDOS, AVALIADOS E ACEITOS JUNTO AOS FORNECEDORES DE REQUISITOS, UTILIZANDO CRITÉRIOS OBJETIVOS

6.1.1 Entender o domínio do cliente

Fazer uso de uma técnica que permita o entendimento completo do domínio do cliente é essencial para que haja uma boa comunicação com os fornecedores de requisitos, consequentemente aumentando as chances de sucesso do projeto.

6.1.2 Registrar os requisitos elicitados e seus fornecedores

Registrar os requisitos elicitados servirá de comprovação de entendimento e comprometimento de todas as partes envolvidas no projeto, o documento de requisitos é a base para o desenvolvimento de software. Tal registro delimita a abrangência do software, estabelece funcionalidades, impõe restrições de qualidade, fornece subsídios para o processo de verificação e validação do software construído.

6.1.3 Validar os requisitos registrados utilizando critérios objetivos

Todos os requisitos que foram anteriormente definidos documento de requisitos devem ser validados, o quadro 7 demonstra as características citadas pela IEEE std 830 (IEEE, 1998) onde é proposto um checklist para verificar a existência de algumas características.

6.1.4 Fornecer informações suficientes para a organização aceitar ou não a produção dos projetos

As técnicas a serem utilizadas para o atendimento deste nível devem fornecer no mínimo os seguintes itens para possibilitar o entendimento: Em qual contexto o projeto se insere; qual o escopo do projeto; permitir calcular os recursos necessários para o projeto; as prioridades do projeto; e o prazo para a entrega.

6.1.5 Avaliação das técnicas

O quadro 1 apresenta um resumo da avaliação das técnicas descritas na secção 3.2.1 que podem ser usadas para alcançar os resultados esperados pela GRE1. Cada técnica avaliada em relação aos objetivos é definida em Não Atendido (NA),

Parcialmente Atendido (PA) ou Totalmente Atendido (TA) de acordo com a competência para cada item avaliado. A avaliação das técnicas foi baseada no próprio entendimento e estudo realizado durante a execução deste trabalho.

Quadro 1. Avaliação das técnicas de elicitação em relação aos objetivos da GRE1

Requisitos	Entrevista	Questionários	JAD	Brainstorming	Etnografia	Prototipagem	SWOT
Entendidos	TA	PA	TA	PA	TA	TA	PA
Registrados	TA	TA	TA	TA	TA	PA	TA
Avaliados	TA	TA	TA	TA	TA	TA	TA
Aceitos	TA	TA	TA	TA	TA	TA	TA

6.2 GRE 2 – OBTENÇÃO DE COMPROMETIMENTO DA EQUIPE TÉCNICA COM OS REQUISITOS APROVADOS

Após a aprovação dos requisitos, o gerente do projeto deve apresentar para a equipe quais foram os requisitos definidos e deixar claro o que deverá ser produzido, garantindo um comprometimento formal da equipe técnica.

Serrano et al (2008), levando em consideração o número de funcionário de uma MPE, sugere que tal apresentação deve ser feita em uma reunião, assim após a exposição das informações, é possível que haja uma discussão e esclarecimento de dúvidas coletivo.

6.3 GRE 3 – A RASTREABILIDADE BIDIRECIONAL ENTRE OS REQUISITOS E OS PRODUTOS DE TRABALHO É ESTABELECIDA E MANTIDA

O objetivo deste resultado é definição de um método que descreva como a organização fará a rastreabilidade bidirecional entre os requisitos e os produtos de trabalho, visando manter o controle dos requisitos e calcular o impacto deles durante o projeto. São diversas as formas na qual o rastreamento pode ser feito, porém, geralmente é utilizada uma ferramenta computacional devido ao grande número de informações e pelo tamanho das matrizes de rastreabilidade.

Tais ferramentas podem ser proprietárias, livres ou criadas pela empresa. A proprietária, geralmente é mais onerosa, porém oferece maior robustez e suporte; A livre que geralmente não possui custo, mas pode apresentar limitações em relação a funcionalidade e correções de erros; Já a ferramenta própria pode ser desenvolvida de acordo coma as funcionalidades e características necessárias da instituição.

6.3.1 Rastreabilidade vertical

Para manter o controle dos requisitos, ver como eles se relacionam com os artefatos que são gerados no decorrer do projeto de forma bidirecional, inicialmente é necessário definir quais artefatos rastrear e de qual forma eles se relacionam com os requisitos e entre si.

Fazendo uso do campo de rastreabilidade citado anteriormente tem-se uma demonstração com a evolução nestes aspectos: Escopo, Requisitos, Serviços, Pacotes, *Units*, Classes ou formulários e código fonte.

6.3.2 Rastreabilidade horizontal

A rastreabilidade bidirecional horizontal visa manter o controle dos requisitos, analisa como eles se relacionam com os demais requisitos. Para isso a empresa deve definir qual a forma de estabelecer o relacionamento entre os requisitos, sendo que para tal tarefa, se propõe a utilização de uma matriz de rastreabilidade.

6.4 GRE 4 – REVISÕES EM PLANOS E PRODUTOS DE TRABALHO DO PROJETO SÃO REALIZADAS VISANDO IDENTIFICAR E CORRIGIR INCONSISTÊNCIAS EM RELAÇÃO AOS REQUISITOS

O objetivo da GRE4 é definir um método que descreva como a organização fará revisões em planos e produtos de trabalho do projeto, visando identificar e corrigir inconsistências em relação aos requisitos. Esta identificação é feita controlando a evolução dos requisitos, seja por constatação de novas necessidades, ou por deficiências nos requisitos já registrados.

Conforme a proposta de Hazan e Leite (2003) o fluxograma apresentado na figura 1 demonstra alguns passos necessários para a administração das mudanças, visando a revisão de planos e produtos durante o projeto.

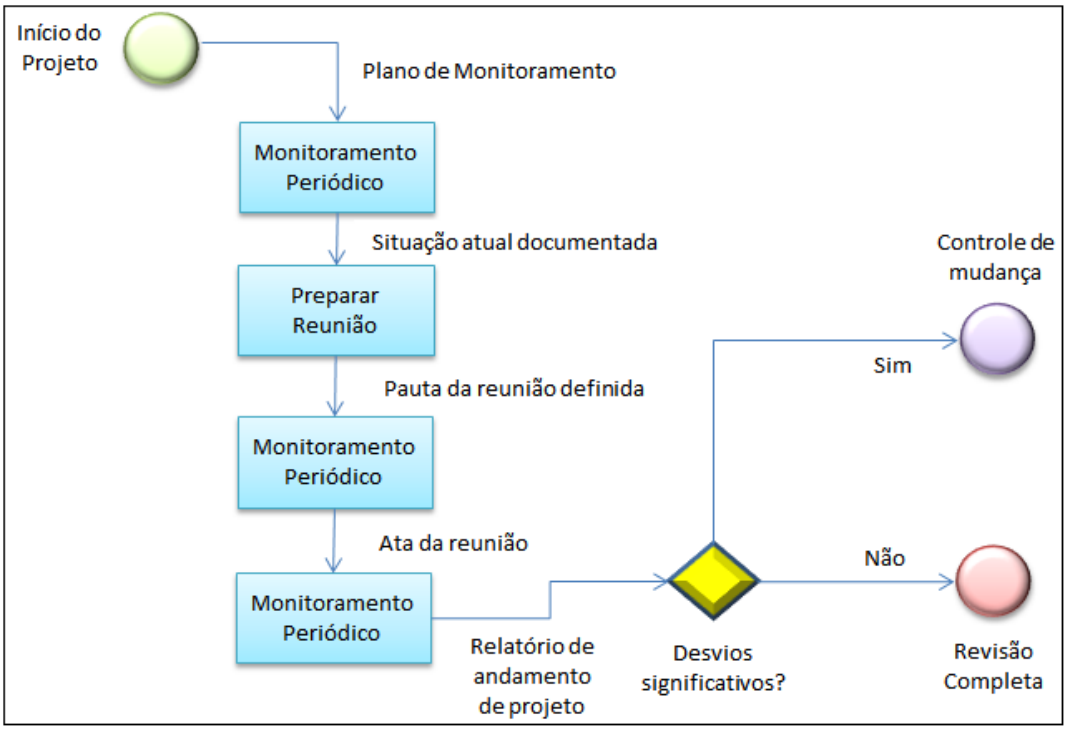


Figura 1 - Plano de revisão
Fonte: Próprio autor.

No monitoramento periódico o responsável irá monitorar os requisitos a fim de identificar desvio, alteração, adição ou exclusão, depois de identificados, serão registrados no histórico do projeto. No estágio seguinte o responsável irá definir a

pauta para da reunião de revisão e envia-la para os participantes, além de levar opções de ações corretivas a serem discutidas na reunião.

O foco da reunião de revisão é identificar os desvios ocorridos, porque ocorreram, como corrigir e como evitar que ele volte a acontecer. No término da reunião o responsável deve criar um relatório do andamento do projeto, servindo para identificar o grau de conformidade com o documento de requisitos, o plano de projeto e seu andamento.

6.5 GRE 5 – MUDANÇAS NOS REQUISITOS SÃO GERENCIADAS AO LONGO DO PROJETO

Baseado no trabalho de Sommerville (2003) e nas características das MPEs, com intuito de controlar as mudanças nos requisitos durante o projeto foi criado um fluxograma que apresenta os passos necessários para a administração das mudanças conforme figura 2.

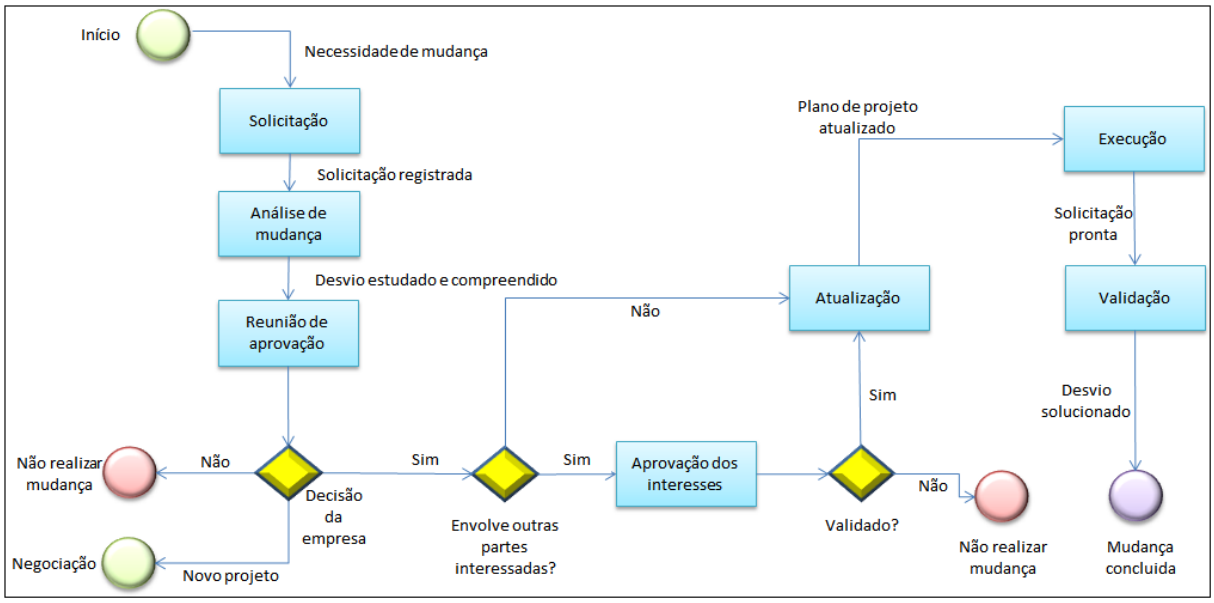


Figura 2 – Controle de mudanças.
Fonte: Próprio autor.

7. CONCLUSÃO

Ao longo dos anos pesquisas demonstram que a quantidade de projetos de software que chegam ao sucesso, contemplando todos os requisitos iniciais, é a minoria do todo.

Com o decorrer dos anos, o aumento na qualidade de software tem sido relevante com a procura das instituições para certificação nas ISO/IEC 12207, ISO/IEC 15504 e especialmente o MPS.BR.

O modelo MPS.BR foi desenvolvido a partir da necessidade de aplicar um modelo de processo eficiente mas que fosse adequado às pequenas e médias empresas. Com a certificação para o nível G onde abrange a Gerência de Requisitos já ajudou muitas empresas a melhorarem a qualidade de seu software.

O estudo apresentou diversos autores que defendem há muitos anos a necessidade de um bom processo de gerencia de requisitos, através dele a empresa conseguirá

entender bem a necessidade do cliente, verificar, validar, documentar e gerenciar seus requisitos ao longo do software.

Foram analisadas várias técnicas quem visam fornecer um subsídio maior desde a fase de elicitação até a gerência ao longo do ciclo de vida de um software.

Entende-se que os resultados obtidos neste trabalho de pesquisa encontram-se a importância da certificação ao nível G do MPS.BR, principalmente para as pequenas e médias empresas que ainda possuem um número muito pequeno de empresas certificadas em território nacional e principalmente por ser o principal foco do programa. A proposta de processo apresentada neste trabalho pode fornecer a muitas empresas subsídios para um aperfeiçoamento na gerência de requisitos, diretamente ligada a qualidade de software apresentada no final do projeto.

REFERENCES

COSTA FILHO, Edes Garcia da et al. Padrões e Métodos Ágeis: agilidade no processo de desenvolvimento de software. Disponível em: <http://subversion.assembla.com/svn/TCC_Agil/9673.pdf>. Acesso em: 07 maio 2013.

GASTALDO, D.L. e MIDORIKAWA, E.T. Processo de Engenharia de Requisitos Aplicado a Requisitos Não-Funcionais de Desempenho – Um Estudo de Caso. 2003.

HOFMANN, H.F. e LEHNER, F. Requirements Engineering as a Success Factor in Software Projects. Vol. 18. 4 vols. 2001.

KOSCIANSKI, A; et tal. Qualidade de Software: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software. São Paulo: Novatec Editora, 2006.

KOTONYA, SOMMERVILLE. Requirements Engineering: Processes and Technique, 1 ed, Wiley. 1998.

PRESSMAN, R. S. Engenharia de Software. 6 ed, McGraw Hill, 2006.

REZENDE, Denis Alcides. Engenharia de Software e Sistemas de Informação. 3° Rio de Janeiro: Brasport, 2005. 313 p.

SANTANA, Célio A.; TIMÓTEO, Aline L.; VASCONCELOS, Alexandre M. L.. Mapeamento do modelo de Melhoria do Processo de Software Brasileiro (MPS.BR) para empresas que utilizam Extreme Programming (XP) como metodologia de desenvolvimento. Recife: 2006. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/sbqs/2006/009.pdf>. Acesso em: 22 nov. 2013.

SOMMERVILLE. Ian. Engenharia de Software. Addison Wesley, São Paulo, 2003.

SWEBOK. Guide to the software Engineering Body of Knowledge. Disponível em:http://www.computer.org/portal/web/swebok/html/ch1. Acesso em: 21 de set. 2013