UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
COORDENAÇÃO DE INFORMÁTICA
CURSO SUPERIOR EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JORGE ROBERTO ROSA PEREIRA

DESENVOLVIMENTO DE UM SOFTWARE PARA MÉTRICAS EM RASTREABILIDADE DE REQUISITOS DE SOFTWARE

PROPOSTA DE TRABALHO DE DIPLOMAÇÃO

CORNÉLIO PROCÓPIO
2011

JORGE ROBERTO ROSA PEREIRA

DESENVOLVIMENTO DE UM SOFTWARE PARA MÉTRICAS EM RASTREABILIDADE DE REQUISITOS DE SOFTWARE

Proposta de Trabalho de Diplomação, apresentado à disciplina Trabalho de Diplomação, do curso Superior em Análise e Desenvolvimento de Sistemas da Coordenação de Informática – COINF – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para a obtenção do título de Analista de Sistemas.

Orientador: Prof. Dr. Elias Canhadas Genvigir

CORNÉLIO PROCÓPIO

2011

RESUMO

PEREIRA, Jorge Roberto Rosa. **Um Software para Métricas em Rastreabilidade de Requisitos de Software**. 2011. Proposta de Trabalho de Diplomação (Analise e Desenvolvimento de Sistemas), Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2011.

Durante o processo de desenvolvimento de software ter o conhecimento sobre qual proposto o software será criado é essencial, na Engenharia de Software essa atividade é atribuída a Engenharia de Requisitos. Entre os estudos da Engenharia de Requisitos temos a rastreabilidade, que possui como meta a definição dos relacionamentos entre artefatos e requisitos, produzidos durante o processo de desenvolvimento de software. Associada diretamente a qualidade de um software, a rastreabilidade de requisitos é uma das mais custosas atividades dentro do processo de desenvolvimento e a constante melhoria na qualidade da rastreabilidade é de certa forma essencial. Este trabalho propõe métricas para a rastreabilidade de requisitos, as métricas de uma forma geral proporcionam recursos e informações quantitativas para gerenciamento e controle do processo de desenvolvimento de software melhorando significativamente a qualidade do produto e do processo. Especificamente as métricas para rastreabilidade de requisitos utilizam informações sobre ligações entre artefatos e requisitos, apresentando como uma organização mantém a mensuração sobre o relacionamento dos requisitos, proporcionando, por exemplo, uma análise simplificada de que todos os relacionamentos requeridos por uma dependência entre os requisitos são realmente atendidos. Com um modelo de métricas para rastreabilidade de requisitos definido, também é desenvolvida uma arquitetura de software que possui como objetivo a construção de um software para Rastreabilidade de Requisitos em um ambiente WEB, fazendo uso de tecnologias de código aberto. Ao final do trabalho são apresentados os resultados de estudos experimentais, que tiveram como objetivo a aplicação do software frente a projetos que não fazem uso das facilidades do software proposto.

ABSTRACT

PEREIRA, Jorge Roberto Rosa. **A Software for Metrics in Software Requirements Traceability**. 2011. 151 f. Proposal of Work Graduation (Analysis and Development of Systems), Federal Technological University of Paraná. Cornélio Procópio, Paraná, Brazil, 2011.

During the process of software development have knowledge about which proposed software will be created is essential in software engineering that activity is assigned to Requirement Engineering. Among the studies of the Requirements Engineering we have traceability, which has a target to define the relationships between requirements and artifacts produced during the process of software development. Directly associated with software quality, traceability of requirements is one of the most expensive activities within the process of development and constant improvement in the quality of traceability is somehow essential. This paper proposes metrics for the traceability of requirements metrics generally provide quantitative information and resources for managing and controlling the process of software development significantly improving product quality and process. The particular metrics used for requirements traceability information about links between requirements and artifacts showing how an organization maintains the measure on the relationship of requirements by, for example, a simplified analysis that all the relationships required by a dependency between requirements are actually met. With a model of metrics for traceability of requirements defined, it's also developed a software architecture that has as objective the construction a software for Requirements Traceability in a WEB environment, using open source technologies. At the end of the paper presents the results of experimental studies that were aimed at applying the software forward to projects that do not use the facilities of the proposed software.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
GPL	General Public License
GQM	Goal Question Metrics
HTML	HiperText Markup Language
IDE	Integrated Development Environment
JDBC	Java Database Connectivity
JPEG	Joint Photographic Experts
JSP	JavaServer Pages
MPS. BR	Melhoria do Processo de Software Brasileiro
MVC	Model View Controller
PDF	Portable Document Format
PNG	Portable Network Graphics
RUP	Rational Unified Process
UML	Unified Modeling Language
XML	Extensible Markup Language

LISTA DE ACRÔNIMOS

ACM	Association for Computing Machinery
IEEE	*Institute of Electrical and Electronic Engineers*
SWEBOK	*Guide to the Software Engineering Body of Knowledge* (Corpo de Conhecimento da Engenharia de Software).
UTFPR	Universidade Tecnológica Federal do Paraná

1. INTRODUÇÃO

Com a evolução da tecnologia, pequenas, médias e grandes organizações dependem de complexos softwares associados a soluções de problemas e (ou) realização de tarefas. Eles proporcionam uma maior praticidade e segurança para o domínio em que o mesmo será alocado.

A fim de suprir a melhoria da qualidade e da produtividade, e manter uma boa relação custo-benefício, foram criados processos de software, ou seja, um conjunto de tarefas padronizadas que norteia o desenvolvimento de software para a qualidade, definindo um roteiro de passos para elaboração de um software (PRESSMAN, 2005).

A partir dos processos de software foi criada então a engenharia de software, aplicação de uma abordagem sistemática, disciplinada e quantificável, para o desenvolvimento, operação e manutenção do software; isto é, a aplicação da engenharia ao software (IEEE, 1993). O processo de software é adaptativo, por isso a engenharia de software deve ser realizada por pessoas criativas e com amplos conhecimentos, somente assim será criado um processo de software que irá suprir as necessidades do seu mercado (PRESSMAN, 2005).

Com o intuito de promover a profissionalização da engenharia de software, uma colaboração entre o IEEE (Institute of Electrical and Electronics Engineers) e a ACM (Association for Computing Machinery), de 1993 a 2000, articularam um comitê para coordenar todas as atividades relacionadas à engenharia de software, o SWECC (Software Engineering Coordinating). Em 1998, o SWECC deu inicio a projeto SWEBOK, acrônimo do inglês *Guide to the Software Engineering Body of Knowledge,* documento com a finalidade de servir como referência a assuntos pertinentes a área de engenharia de software.

Em 2004, o SWEBOK classificou a Engenharia de Software em dez áreas de conhecimento. A primeira área definida pelo SWEBOK trata dos requisitos de software. Em grande parte dos processos de desenvolvimento, os requisitos de software são definidos nas fases iniciais do projeto, especificando atributos e propriedades do sistema, e o que deve ou não ser desenvolvido.

A fim de ajudar os engenheiros de software a compreender melhor o problema que eles vão trabalhar para resolver, surgiu então a Engenharia de Requisitos.

A engenharia de requisitos contempla um conjunto de atividade que direcionam ao entendimento da regra de negócio que o software irá impactar o retorno que o cliente espera do software e da interação final do usuário com o software (PRESSMAN, 2005).

Dentre as atividades que compõem a engenharia de requisitos pode-se dividi-las em quatro áreas: Elicitação, Análise, Documentação e o Gerenciamento (GENVIGIR, 2009). Este trabalho contempla a área de gerenciamento de requisitos, ou seja, a área que abrange toda compreensão e o controle de mudanças nos requisitos de sistema.

Devido a essas mudanças que podem ocorrer ao longo da vida dos sistemas, dentro da gestão de requisitos deve haver uma rastreabilidade dos mesmos. A atividade de rastreabilidade de requisitos visa controlar a evolução e agregação dos requisitos ao projeto de software (GENVIGIR, 2009).

Por isso, durante o estágio de gerenciamento de requisitos, há a necessidade de uma identificação única para cada requisito, para que posteriormente possa ser feita uma referência cruzada de requisito para requisito, facilitando assim a rastreabilidade.

Mas apesar da importância em se fazer uma política de rastreamento, nem todas as organizações disponibilizam recursos para esse propósito. E dentre as organizações que fazem uso da rastreabilidade, poucas se preocupam com a qualidade dos seus requisitos rastreados. Um elemento chave para a qualidade dos requisitos rastreados é a medição.

Usamos medidas para um melhor entendimento daquilo que criamos ou fazemos, facilitando e melhorando também a qualidade dos produtos ou sistemas que construímos (SOMMERVILLE, 2003).

Uma análise minuciosa sobre as métricas para rastreabilidade de requisitos de software aponta a falta de preocupação com a qualidade da relação de requisito para requisito, ou seja, as métricas propostas obtêm informações da rastreabilidade para determinar a qualidade dos requisitos, enquanto que a qualidade da rastreabilidade em si não é avaliada.

Neste trabalho serão desenvolvidas e estudadas modelos de métricas para a rastreabilidade desses requisitos, tendo como objetivo proporcionar uma reversão neste quadro.

1.1. VISÃO GERAL DO TRABALHO

A gestão de requisitos auxilia a equipe de projeto a identificar, controlar e rastrear os requisitos e suas mutações ao longo do projeto (PRESSMAN, 2005).

Os relacionamentos são estabelecidos entre requisitos e artefatos de software usando-se elos (GENVIGIR, 2009), melhor descrito na sessão 2.1.1 que trata a respeito da rastreabilidade de requisitos.

Primordiais para criação da rastreabilidade, os elos são responsáveis para a criação do relacionamento entre os artefatos. Os artefatos podem ser modelos, documentos, código fonte, sequências de testes, requisitos ou executáveis, ou seja, elementos do projeto que podem ser rastreados.

A melhoria da visibilidade e qualidade da rastreabilidade desses requisitos pode ser feitas através de métricas. O uso de métricas de software de uma maneira geral, proporciona uma maior visibilidade a custos, cronograma, auxilia gerentes de projetos em tomadas de decisões, identificando e resolvendo problemas antes que se tornem crises.

No entanto, as atuais métricas de software na literatura não abordar globalmente o pleno ciclo de vida do software ou fornecem detalhes suficientes. (COSTELLO; LIU, 1995), o mesmo acontece com as métricas para rastreabilidade de requisitos de software.

Mesmo com todos os benefícios que as métricas proporcionam, quando definidas de formas abusivas e inconsistentes, os indicadores que as métricas ofereceriam acarretaria grandes problemas. Ou seja, sem uma abordagem de métrica sadia não será possível coletar dados significativos (COSTELLO; LIU, 1995).

Durante o ciclo de vida de um projeto podem ser identificadas três categorias de métricas: produto e processo, progresso e recursos. Esse conjunto de métricas é normalmente necessário para a cobertura total de uma abordagem de métrica para qualquer processo de desenvolvimento de software (COSTELLO; LIU, 1995).

Métricas de produto e processo são usadas para medir os atributos da documentação (eletrônico e (ou) papel), código fonte, características das atividades, métodos, práticas e mudanças utilizadas no desenvolvimento de software. Já métricas de progresso são capazes de identificar e indicar a capacidade da organização para aderir a cronogramas. Métricas de recurso fornecem indicadores a

quantidade de desenvolvimento, integração, testes e (ou) apoio, recursos e pessoais disponíveis e usados (COSTELLO; LIU, 1995).

O conceito de aplicação de métricas para produtos e processos de software ao longo do ciclo de vida do software não são novos. Em particular, essas aplicações de métricas não cobrem adequadamente todo o ciclo de vida de um processo de desenvolvimento de software.

Geralmente são utilizadas definições de métricas pobres ou escritas em um nível alto, fazendo que grande parte dos dados recolhidos não forneça informações necessárias para identificar e resolver os problemas potencialmente graves (COSTELLO; LIU, 1995).

O foco deste trabalho está em definir e implementar um modelo de métricas para mensurar especificamente a rastreabilidade, usando dados sobre os elos e outras técnicas de representação.

Objetivos mais precisos sobre a solução adotada são tratados na próxima seção.

1.2. OBJETIVOS DO TRABALHO

O trabalho tem por objetivo apresentar um modelo para métricas em rastreabilidade de requisitos de software e a implementação deste modelo. Tal objetivo é atingido através da execução das seguintes atividades:

- Definir modelos para métricas em rastreabilidade de requisitos de software, usando informações de elos e atributos dos requisitos rastreados, baseados nos padrões de métricas estudados;
- Desenvolver um software para o modelo escolhido;
- Executar estudos experimentais para coleta de dados sobre o software;
- Realizar comparativo entre o modelo desenvolvido em relação a outras técnicas e projetos existentes;

Padrões de métricas para rastreabilidade de requisitos de software será estudado, será definido um modelo baseado neste estudo e posteriormente um software para esse modelo é construído a fim de demonstrar sua viabilidade. Ao final, será realizado um estudo experimental e apresentados resultados sobre o uso deste modelo.

1.3. ORGANIZAÇÃO DO TRABALHO

O capítulo 2 apresenta a rastreabilidade de requisitos de software juntamente com as métricas, apresentando conceitos de engenharia de requisitos e rastreabilidade de requisitos, bem como abordagens e pesquisas para métricas em geral, e por fim as métricas para rastreabilidade de requisitos de software juntamente com o modelo proposto neste trabalho.

O capítulo 3 descreve as justificativas para a realização do trabalho. São apresentadas também as soluções para os problemas e as vantagens em se mensurar a rastreabilidade de requisitos de software.

O capítulo 4 apresenta as tecnologias utilizadas durante a realização deste trabalho.

No capítulo 5 é apresentado à metodologia utilizada, modelos e estratégias que serão usadas durante todo o processo de desenvolvimento do trabalho. Na subseção 5.1 é descrito o modelo arquitetônico e organizacional utilizado durante todo o desenvolvimento do trabalho.

No capítulo 6 é descrito o método de pesquisa e o processo de experimentação a ser aplicadas, formas de estudo de caso com objetivos de melhorar a abrangência e a competência deste trabalho.

O capítulo 7 mostra o cronograma de atividades e suas previsões para realização.

.

2. RASTREABILIDADE DE REQUISITOS DE SOFTWARE E MÉTRICAS

Neste Capítulo é apresentado o conceito e a importante utilização das métricas dentro da engenharia de software e o impacto das mesmas sobre a gerência de projetos, com foco na abordagem de métricas para rastreabilidade de requisitos.

São discutidas também algumas atividades que utilizam um processo de medição dentro da engenharia de software e técnicas existentes. Porém, a atenção maior será para as métricas dentro da engenharia de requisitos, na atividade de rastreabilidade, onde serão levantados os elementos para representar relacionamentos, elos de rastreabilidade e suas propriedades e por fim as métricas para a rastreabilidade desses requisitos.

Também será tratado neste capítulo o modelo proposto, sendo desta forma contextualizado o foco da pesquisa desenvolvida neste trabalho.

2.1. ENGENHARIA DE REQUISITOS

Em um processo de software a engenharia de requisitos é uma das primeiras atividades e é de suma importância para o sucesso de um projeto de software. Na figura 1, é possível visualizar todas as entradas do processo de engenharia de requisitos e a atividade de gerenciamento ocorrendo ao longo do processo.

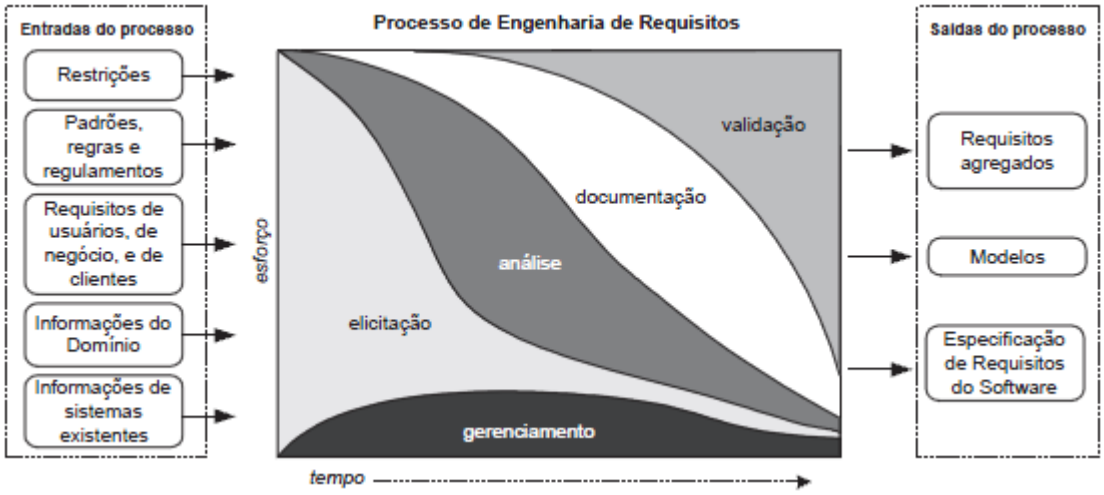

Figura 1 – Processo de engenharia de requisites.

Fonte: GENVIGIR (2009)

Por ser uma atividade inicial, o foco da engenharia de requisitos é definir toda a descrição do que o software proposto deve fazer para satisfazer as necessidades informadas pelo cliente (PETERS, JAMES F, 2001).

A engenharia de requisitos trata também de dimensionar o proposto para qual o software será criado, ou seja, levantando os requisitos e os conhecendo adequadamente será o necessário para as fases seguintes do processo de desenvolvimento de software.

A fim de demonstrar em qual nível da engenharia de software se encontra a rastreabilidade, a figura 2, apresenta sua alocação.

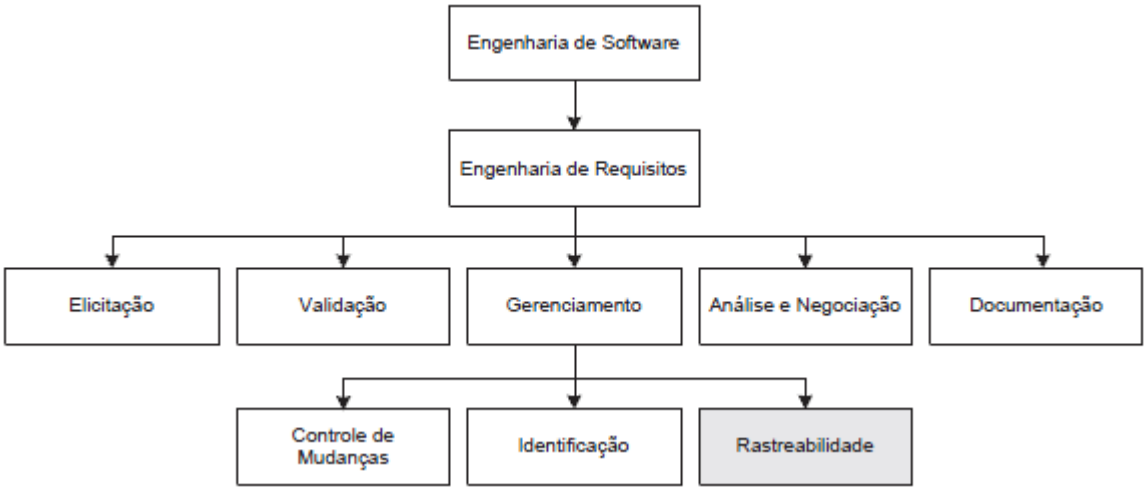

Figura 2 – Alocação da rastreabilidade dentro da engenharia de software.

Fonte: GENVIGIR (2009)

Dentre as diversas atividades que compõem a engenharia de requisitos, é no gerenciamento de requisitos que se encontra a rastreabilidade de requisitos e por fim suas métricas. As métricas para rastreabilidade de requisitos é o foco principal deste trabalho e será explorada detalhadamente nas próximas seções.

2.1.1. RASTREABILIDADE DE REQUISITOS

Em projetos de software não é novidade que requisitos mudam, e para que esse software atenda a novas mudanças é necessária uma compreensão detalhada e rastreada de todos os requisitos e funcionalidades.

A rastreabilidade esta associada ao processo de produção de software, aos requisitos e a capacidade de estabelecer vínculos entre esses requisitos e outros artefatos (modelos, documentos, código fonte) que os satisfaçam (GENVIGIR, 2009).

Em resumo existem duas classificações gerais para os tipos de rastreabilidade:

- Rastreabilidade horizontal e vertical;
- Pré e pós-rastreabilidade;

A rastreabilidade horizontal trata diferentes versões ou variações de requisitos em uma determinada fase do ciclo de vida, já a vertical trata dos requisitos produzidos pelo processo ao longo do ciclo de vida.

A figura 3 mostra uma melhor visão para rastreabilidade horizontal e vertical.

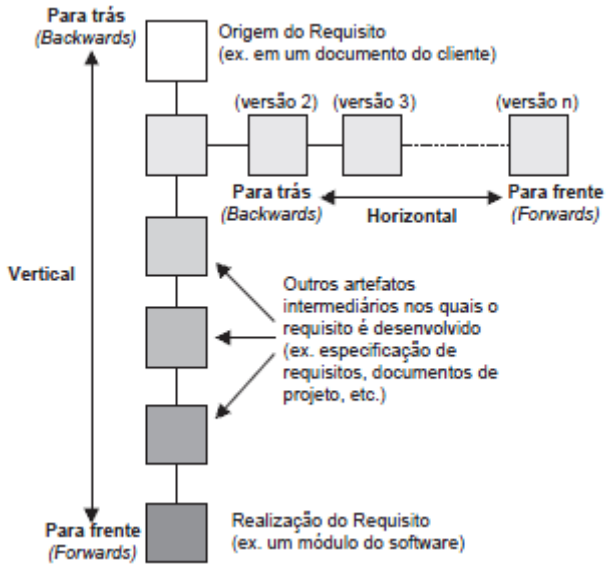

Figura 3 – Tipos de rastreabilidade (horizontal e vertical).

Fonte: GENVIGIR (2009)

Já a pré rastreabilidade esta ligada aos requisitos antes de serem incluídos no documento de requisitos e a pós-rastreabilidade é o contrário, trata dos requisitos após serem incluídos na especificação. A figura 4 ilustra melhor a pré e pós-rastreabilidade.

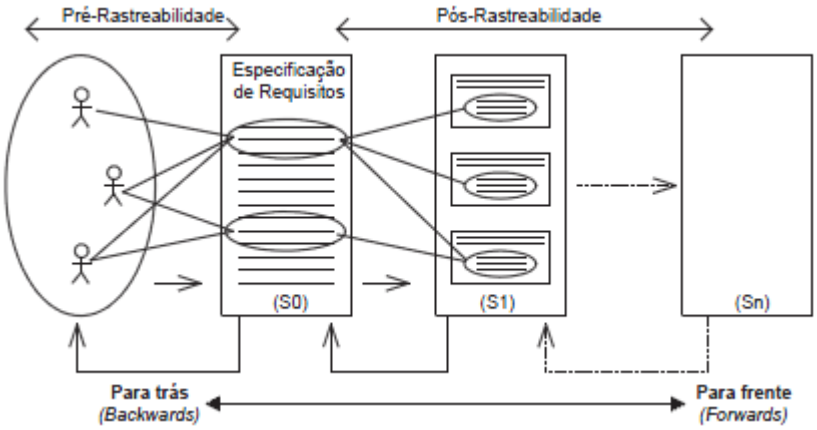

Figura 4 – Tipos de rastreabilidade (pré e pós-rastreabilidade)

Fonte: GENVIGIR (2009)

Para manter e representar os relacionamentos de rastreabilidade utiliza-se o recurso elo (do inglês *link*). A figura 5 ilustra os elementos básicos de um elo e como ele representa visualmente.

Figura 5 – Elementos básicos de um elo.

Fonte: GENVIGIR (2009)

A qualidade dos elos entre os requisitos rastreados é um item que pode ser útil para outras atividades da engenharia de requisitos, e atualmente é pouco estudado ou melhorado.

Na próxima seção será abordado o estudo de métricas e especificamente para rastreabilidade de requisitos, a fim de melhorar a qualidade desta importante atividade da engenharia de requisitos.

2.2. MÉTRICAS

Na maioria dos empreendimentos técnicos, as medições e as métricas auxiliam a entender o processo usado para desenvolver um produto. O processo é medido, num esforço para melhorá-lo. O produto é medido, num esforço para aumentar sua qualidade (PRESSMAN, 1995).

Métricas de software referem-se a uma ampla variedade de medidas, por exemplo, dentro do contexto de gerenciamento de projetos a maior preocupação é com as métricas de produtividade e de qualidade, ou seja, medidas do resultado d a produtividade no desenvolvimento de software, como uma função do esforço aplicado e do resultado que é produzido.

Um software de modo geral, é medido por muitas razões:

- Atingir prazos e orçamentos previstos;
- Motivar equipe;
- Indicar a qualidade do produto;
- Avaliar a produtividade das pessoas que produzem o produto;
- Avaliar os benefícios (produtividade e qualidade) de novos métodos e ferramentas de software;
- Formar estimativas;
- Identificar oportunidades de melhorias – medir impacto de técnicas e ferramentas;

Mesmo com todas as inúmeras razões para a medição, infelizmente é comum iniciar-se o desenvolvimento do software sem que seja feita uma avaliação do real tamanho e complexidade. Em grande parte dos casos, define-se uma estimativa baseada em experiências anteriores de quanto tempo é necessário para todo o processo de desenvolvimento. Algumas razões podem ser apontadas para esse tipo de procedimento:

- Falta de comprometimento da gerência;
- Pressa em estimar e desenvolver rapidamente o software;
- Pouco conhecimento sobre medições e métricas;
- Resistência em adaptar a novos métodos;

- Gerentes de projetos acham custoso coletar informações para o cálculo das métricas;

Medição ou mensuração é o processo pelo qual números ou símbolos são associados aos atributos das entidades do mundo real, com o objetivo de descrevê-la de acordo com um conjunto de regras claramente definidas (FENTON e PFLEEGER, 1998). Uma entidade pode ser um artefato, como por exemplo, uma especificação de requisitos, os atributos são as propriedades desse artefato, como por exemplo, a qualidade da especificação de requisitos ou tamanho.

Medição de software tem por objetivo obter valores para atributos de um produto ou de um processo de software (SOMMERVILLE, 2003).

Mensuração de software é a quantificação de uma característica, tanto as finais do produto as dos processos envolvidos em sua concepção e construção (VAZQUEZ, SIMÕES e ALBERT, 2008). Sendo assim, métricas de software referem-se a qualquer tipo de medida de software, processo ou documentação relacionada.

Através de uma métrica ou da combinação delas, é obtido os indicadores que fornecem compreensão do processo, do projeto ou do produto de software (PRESSMAN, 1995). Indicador é uma variável definida nos resultados de um processo ou a ocorrência de uma condição específica (IEEE, 1990).

O propósito para mensuração de software é coletar, analisar e relatar os dados relativos aos produtos desenvolvidos e aos processos desenvolvidos na organização e em seus projetos, de forma a apoiar os objetivos organizacionais (MPS. BR, 2007).

2.2.1. CLASSIFICAÇÃO DAS METRICAS

Semelhante as medições do mundo físico, as métricas de software podem ser divididas em diversas categorias: medidas diretas e indiretas (PRESSMAN, 1995) e métricas de controle ou preditivas (SOMMERVILLE, 2003).

Medidas diretas são aquelas que incluem custo, esforço, quantidade de linhas de código produzidas e total de defeitos registrados. São fáceis de serem reunidas e avaliadas. Já as medidas indiretas são aquelas obtidas a partir de outras métricas. São mais difíceis de serem avaliadas, uma vez que analisa a funcionalidade, qualidade, complexidade, eficiência, confiabilidade, manutenibilidade do software, entre outras (KLAUCK, 2010).

Métricas de controle geralmente são associadas a processos de software. São dados quantitativos, como por exemplo, esforço e tempo dedicado ao processo. Métricas preditivas são associadas aos produtos de software, como por exemplo, tamanho do código, complexidade do software, número de atributos e classes (SOMMERVILLE, 2003).

O domínio das métricas de software pode ser dividido em: métricas da produtividade, métricas da qualidade e métricas técnicas. Métricas da produtividade concentram-se na saída do processo de engenharia de software, métricas da qualidade permite indicar o nível de resposta do software às exigências implícitas, e por fim as métricas técnicas concentram-se nas características do software PRESSMAN (1995).

Pela figura 6 observa-se uma segunda divisão em categorias: métricas orientadas ao tamanho (ou medidas diretas do software), métricas orientadas a função (ou medidas indiretas do software) e métricas orientadas a seres humanos (ou pessoas).

As métricas orientadas a seres humanos compilam informações sobre a maneira segundo a qual as pessoas desenvolvem software de computador e percepções humanas sobre a efetividade das ferramentas e métodos, por isso não são universalmente aceitas para se medir o processo de desenvolvimento de software PRESSMAN (1995).

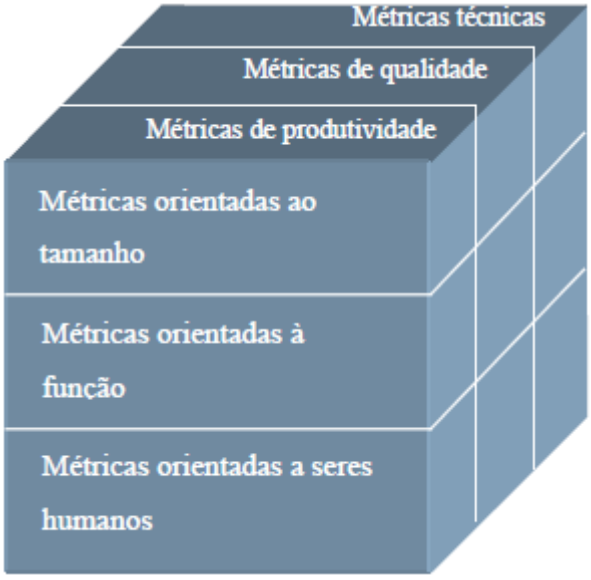

Figura 6 – Divisão das métricas em categorias.

Fonte: PRESSMAN (1995)

Várias propostas têm sido feitas para métricas de processo e projeto, e ainda assim elas seguem melhorando e se aperfeiçoando continuamente. Já as métricas para rastreabilidade não recebem atenção privilegiada.

Na próxima seção serão contextualizadas os padrões, propostas e pesquisas existentes em métricas para rastreabilidade de requisitos de software, bem como um novo modelo de métricas baseado nos estudos e propostas apresentados.

2.2.2. MÉTRICAS PARA RASTREABILIDADE DE REQUISITOS

Ainda vem se estudando a lentos passos as métricas para rastreabilidade de requisitos de software, e as abordagens existentes mostraram que grande parte delas obtém informações da rastreabilidade para determinar a qualidade dos requisitos, sendo que a qualidade da rastreabilidade não é mensurada.

Este trabalho basicamente aborda o estudo de métricas para rastreabilidade entre dois autores, HULL ET AL (2002) e COSTELLO; LIU (1995).

O conceito de métricas é útil em relação ao fluxo de requisitos, e para determinar os aspectos úteis de mensuração é necessário distinguir entre dois tipos de métricas, as métricas de camadas e as métricas globais.

As métricas de camadas são medições relativas a uma única fase do desenvolvimento, por exemplo, apenas para a camada dos requisitos do sistema. Já as métricas globais abrangem várias fases de desenvolvimento (HULL ET AL., 2002).

Para esses dois tipos de métricas, existem três dimensões de rastreabilidade:

- Largura;
- Profundidade;
- Crescimento;

A dimensão por largura representa a métrica do tipo cobertura, é utilizada para medir o progresso do processo que cria a rastreabilidade em uma única camada. Avalia a extensão na qual os requisitos são cobertos pelas camadas adjacentes, acima, abaixo ou ao lado quando olhando para qualificação.

A dimensão de profundidade foca no número de camadas que a rastreabilidade estende para cima ou para baixo a partir de um determinado nível, o que faz dela uma métrica global.

Intimamente relacionada com o impacto de mudanças, a dimensão por crescimento, examina a maneira que alguns requisitos, em níveis mais baixos, estão relacionados a um único requisito em um nível mais alto. A figura 7 ilustra quatro situações, que posteriormente será discutida.

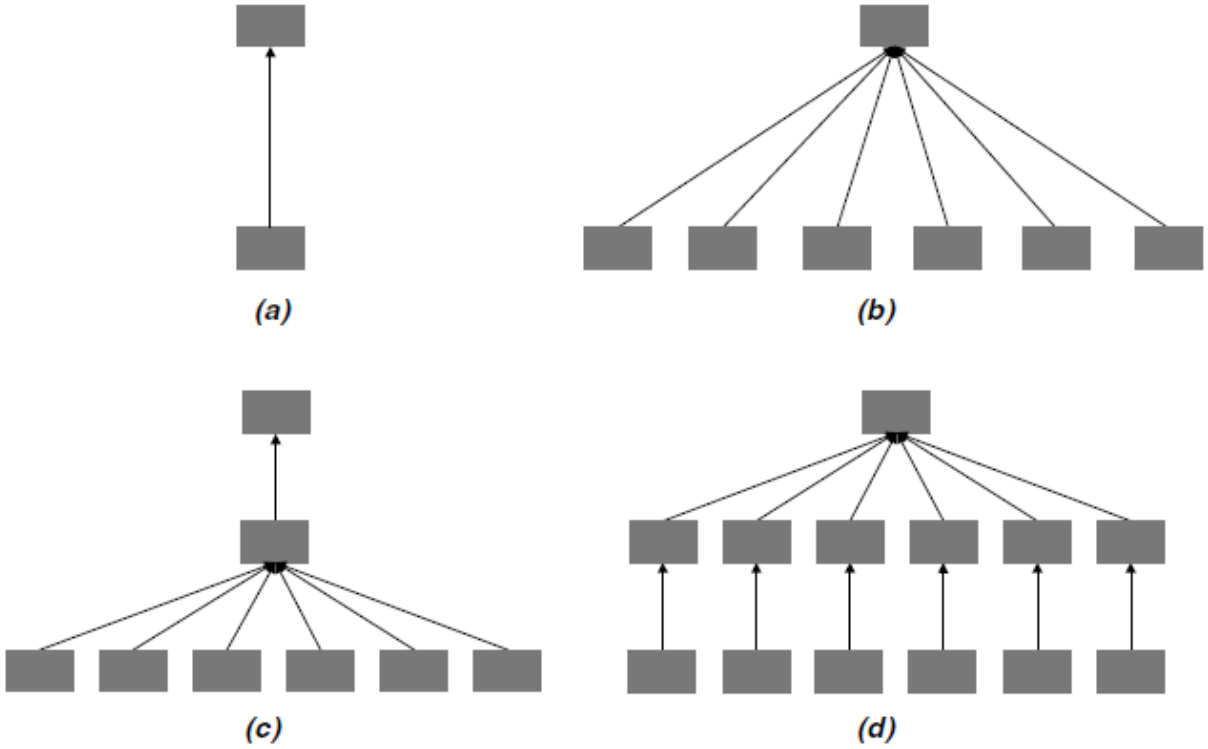

Figura 7 – Dimensão por crescimento.

Fonte: HULL ET AL (2002)

No caso (a) um único requisito é satisfeito por apenas um requisito, pode-se dizer então que o fator de crescimento desse requisito é de apenas um. Em (b) seis requisitos satisfazem um único requisito, com isso, o fator de crescimento neste caso é de seis. Uma breve análise entre esses dois casos pode ser feita da seguinte forma:

- No caso (b) os requisitos podem ter sido mal elaborados e necessita de decomposições;
- O requisito (b) pode ser inerentemente mais complexo do que o requisito (a), portanto necessita de uma atenção especial;
- O requisito (a) possui menor impacto a mudanças do que o requisito (b);

Naturalmente, um desequilíbrio em um nível pode ser abordado no próximo nível para baixo. Por exemplo, nos casos (c) e (d), o fator de crescimento dos dois são idênticos, porém, o requisito superior em (c) estava em um nível muito alto e os requisitos do meio em (d) estavam num nível baixo demais.

A verificação do fator de crescimento dos requisitos entre as camadas é algo que só amadurece com a experiência. No entanto, examinar o equilíbrio entre as necessidades de crescimento dos requisitos é essencial.

Por isso, HULL ET AL (2002) define mais dois elementos de métricas usados para o equilíbrio entre as camadas de requisitos:

- Balanço;
- Mudança Latente;

O balanço é focado na observação da distribuição de fatores de crescimento para requisitos individuais entre duas camadas, examinando aqueles estão no exterior da distribuição.

O objetivo consiste em identificar os requisitos, que tem alguma anormalidade, tanto alta quanto baixa, no fator de crescimento, e submetê-los a um tratamento especial.

Como mostra a figura 8, fazendo uso do elemento de métrica de balanço e com uma análise da taxa de crescimento em relação ao número de requisitos.

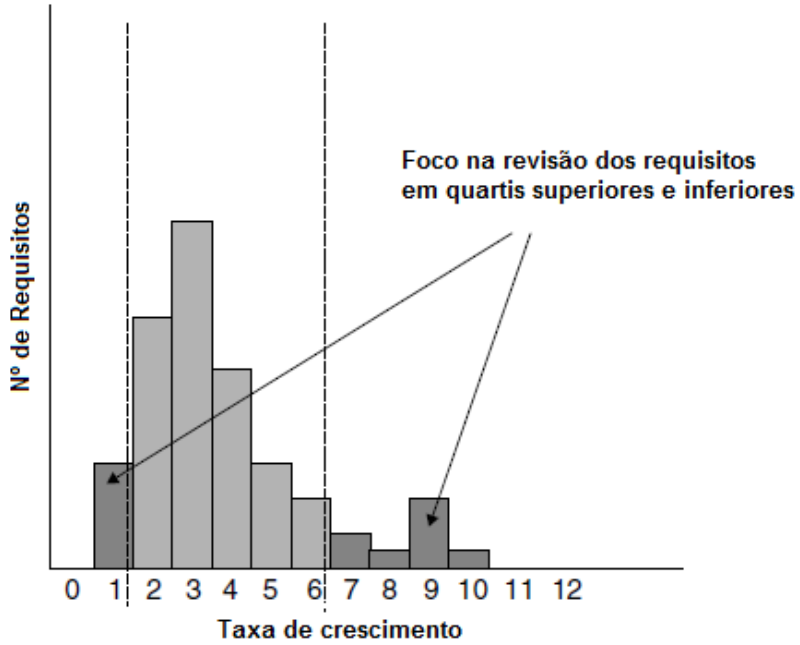

Figura 8 – Distribuição da freqüência de crescimento.

Fonte: *Modificado de* HULL ET AL (2002)

Analisando brevemente a figura 8, pode-se dizer que, a maioria dos requisitos se situam entre o fator de crescimento 2 e 6. Os requisitos com taxa de crescimento de apenas 1 ou superior a 6, são os requisitos que devem ser identificados e receber atenção especial.

A análise realizada na figura 8 foi sobre o fator de crescimento para baixo, ou seja, examinando o número de requisitos que fluem para saídas de outros. Na figura 9 é realizada a análise de balanceamento maneira oposta, ou seja, de baixo para cima, tendo em vista o fator de crescimento para cima.

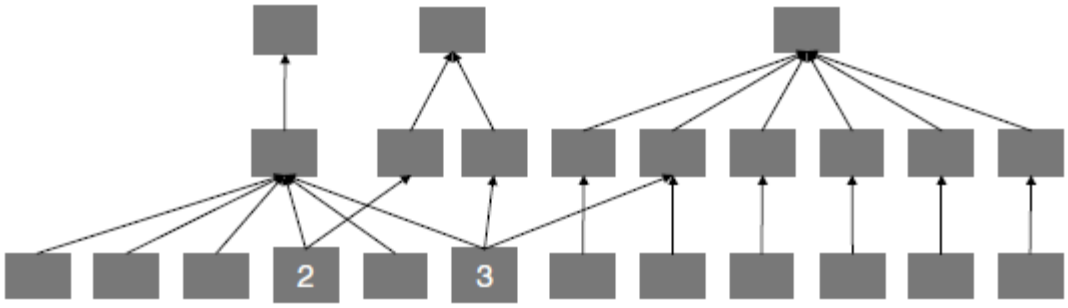

Figura 9 – Criticidade de requisitos.

Fonte: HULL ET AL (2002)

Tendo em conta que a rastreabilidade é um relacionamento de muitos para muitos, pode-se dizer que na figura 9 os dois requisitos (2 e 3) tem ligação com múltiplos requisitos, aumentando assim o fator de crescimento tornando-os talvez mais importantes, que outros, devendo por isso ser dada uma atenção especial aos mesmos.

Por fim, o último elemento de métrica definido por HULL ET ALL (2002) é a mudança latente. Esse elemento refere-se às mudanças não aparentes, ou que ainda não se manifestaram.

Ou seja, quando um pedido de alteração é disparado contra um requisito, todos os requisitos que se encontram no rastreamento são movidos para um estado suspeito até que a equipe de engenheiros possa verificar o verdadeiro impacto. Com isso, é apresentada uma cascata de mudanças latentes no sistema.

Em tais circunstâncias, seria altamente desejável para acompanhar o progresso e estimar os trabalhos conseqüentes. A figura 10 ilustra a complexidade do impacto da mudança.

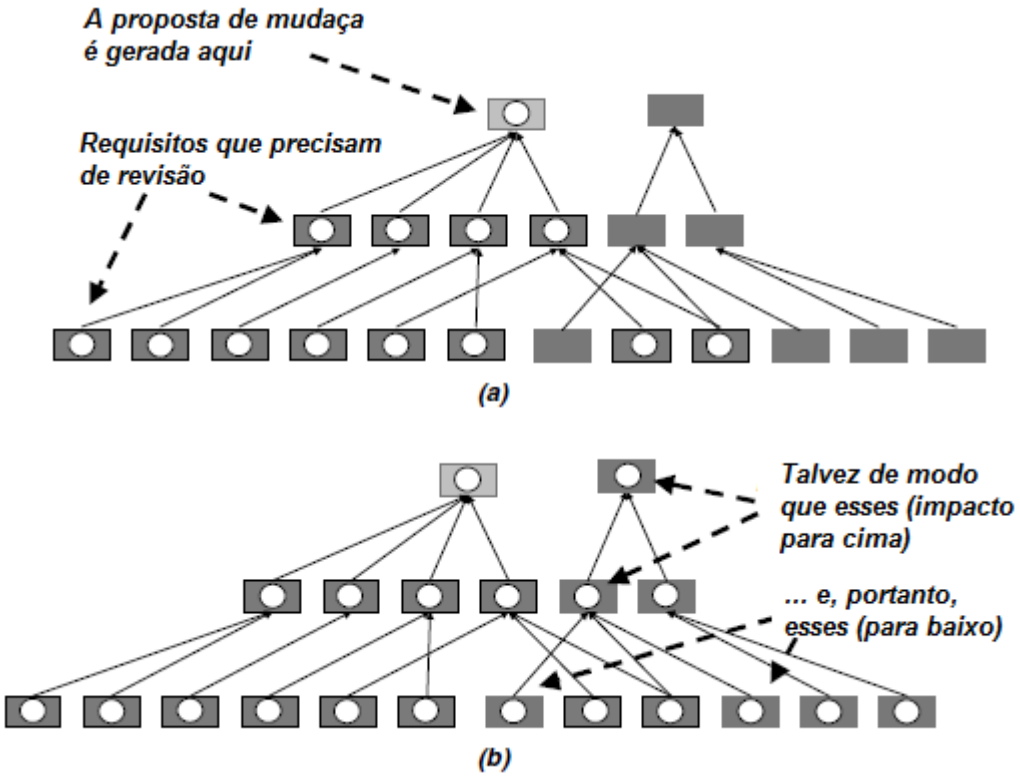

Figura 10 – Mudança latente resultante de uma proposta de mudança.

Fonte: *Adaptado de* HULL ET AL (2002)

Analisando a figura 10, a mudança latente de alguns requisitos fica clara em diversos aspectos, a solicitação de mudança é gerada em um requisito de nível mais alto, onde com isso dá inicio a cascata de requisitos com possíveis mudanças.

Por isso, quando uma solicitação de mudança é levantada, todos os outros requisitos para baixo e para cima são marcados como suspeitos.

A parte (a) mostra a dimensão do impacto usando a técnica de rastreabilidade para baixo, onde os requisitos marcados com o circulo branco estão sujeitos a alterações, ou seja, são requisitos com características de mudança latente.

Já a parte (b) analisa o impacto de mudanças usando a rastreabilidade para cima. Isso ocorre, pois mudanças em uma condição de nível baixo podem causar repactuação nos níveis mais elevados. Com isso neste exemplo, todos os requisitos são passiveis a alterações.

O estado de alteração pode ser medido em números de requisitos que ainda se encontram em estado suspeito. Com isso os engenheiros podem decrescer o numero de requisitos suspeitos com avaliações realizadas requisito por requisito.

A quantidade de mudanças em um sistema, portanto, vai do máximo (quando uma nova proposta de mudança é apresentada), e diminuindo com o tempo e obviamente o trabalho dos engenheiros, como mostra a figura 11.

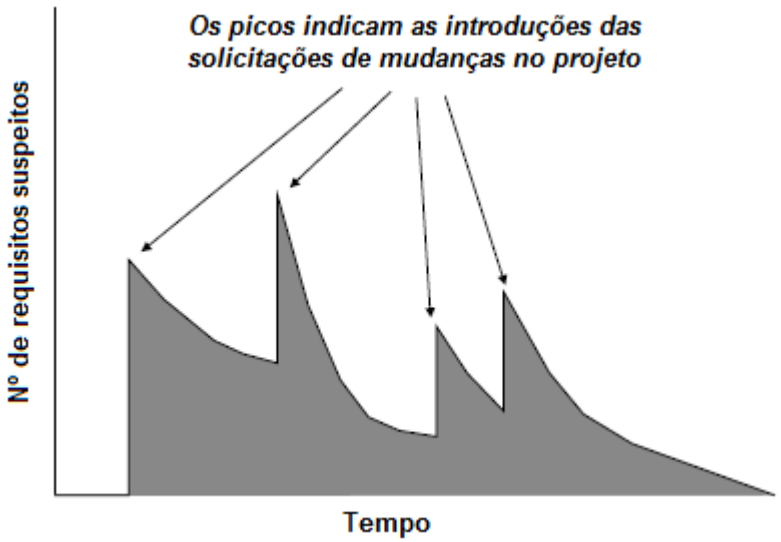

Figura 11 – Progresso no processo de mudança.

Fonte: *Adaptado de* HULL ET AL (2002)

O estudo do modelo proposto por HULL ET AL (2002) aponta padrões de métricas focados na quantidade e qualidade dos requisitos rastreados, porém, outra discussão é estudada nesta mesma seção.

Similar aos padrões de métricas definidos por HULL ET AL (2002), os autores COSTELLO; LIU (1995) aborda as métricas para rastreabilidade de requisitos com foco na cobertura e abrangência dos requisitos rastreados.

Dessa forma, COSTELLO; LIU (1995) divide as métricas para rastreabilidade de requisitos de software em cinco tipos:

- Cobertura de próximo nível (COV);
- Profundidade plena e alta cobertura (DHCOV);
- Estatística de Vinculação;
- Rastreabilidade inconsistente (ITM);
- Rastreabilidade indefinida (UTM);

Cada um dos tipos se aplica tanto para rastreabilidade ascendente e descendente.

As métricas do tipo COV se divide em dois subtipos, cobertura de próximo nível acima (COVup) e cobertura de próximo nível abaixo (COVdown). As métricas COVup incluem o número de requisitos que serão rastreados consistentemente para o próximo nível acima, já as métricas do tipo COVdown trata dos requisitos rastreados consistentemente para baixo.

A figura 12 ilustra um exemplo de relatório de métricas, fazendo uso da rastreabilidade de próximo nível (COV) para cima (COVup) e para baixo (COVdown).

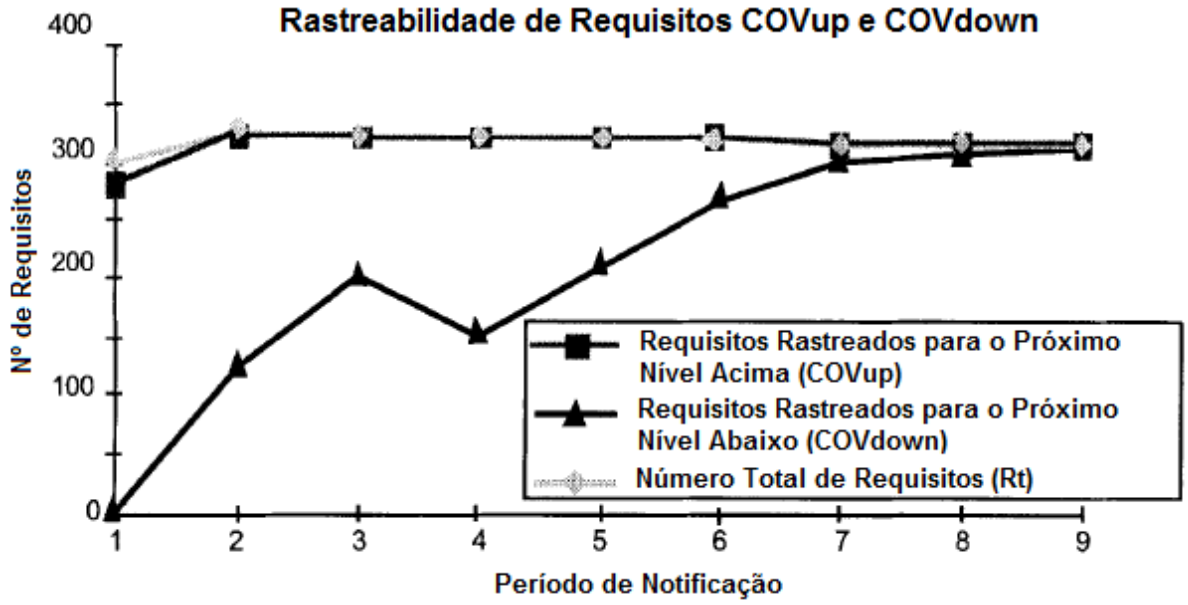

Figura 12 – Relatório de Métricas – Rastreabilidade de Próximo Nível.

Fonte: *Adaptado de* COSTELLO; LIU (1995)

Naturalmente as métricas do tipo COV tratam o número de requisitos rastreados consistentemente para o próximo nível em ambos os sentidos.

Um segundo tipo de métrica definido por COSTELLO; LIU (1995) trata das Métricas DHCOV.

Semelhante às métricas COV, as métricas do tipo DHCOV avaliam a rastreabilidade em níveis de especificação mais altos (HCOV) e mais baixos (DCOV), tanto é que, para sistemas que possuem três ou mais níveis de especificações os tipos COV e DHCOV serão idênticos.

O terceiro tipo de métrica é a estatística de vinculação, onde seu propósito é medir a complexidade dos dados da rastreabilidade. Com ela é possível uma contagem dos requisitos de baixo e alto nível para o qual cada requisito de uma especificação é rastreado.

Com o intuito de exemplificar uma análise de medição da vinculação entre os requisitos, a figura 13 mostra a relação ou o vínculo entre o número de requisitos quanto ao número de ligações (elos) entre eles.

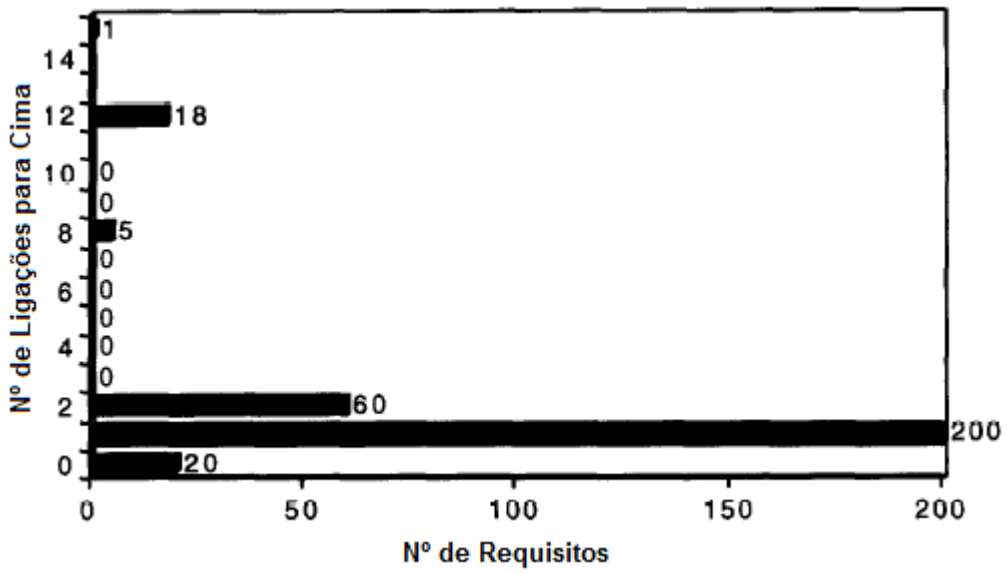

Figura 13 – Histograma da vinculação entre os requisitos rastreados.

Fonte: *Adaptado de* COSTELLO; LIU (1995)

Analisando a figura 13, pode-se dizer que grande parte dos requisitos (260) possuem ligações para 1 ou 2 requisitos de nível superior, analisando dessa maneira, pode-se dizer que a rastreabilidade entre esses requisitos é de certa forma razoável. Porém, cerca de 20 requisitos não possuem ligação com qualquer outro requisito de nível superior em um todo.

O que demanda uma maior atenção dos engenheiros de software são os requisitos que possuem números de ligações anormais em relação aos requisitos de nível superior, como por exemplo:

- 5 requisitos rastreados possuem ligação com 8 requisitos de nível mais elevado;
- 18 requisitos com um nível de ligação para com 12 requisitos para cima;
- Apenas 1 requisito com 15 ligações entre requisitos superiores;

O quarto tipo de métrica trata da rastreabilidade inconsistente, como o próprio nome diz, inclui o número de requisitos em uma especificação que tenham pelo menos um elo inconsistente ascendente (ITMup) e descendente (ITMdown).

Por fim, a rastreabilidade indefinida, esse tipo inclui o número de requisitos em uma especificação que não tenham nenhum elo ascendente (UTMup) e descendente (UTMdown), ou seja, sejam ligações indefinidas entre os requisitos.

Como descrito anteriormente, as métricas para rastreabilidade de requisitos apresentadas são muito semelhantes, porém, alguns fatores são tratados por um autor e não definido especificamente pelo outro.

Por isso, este trabalho apresentará um novo modelo para mensuração da rastreabilidade de requisitos. O modelo criado é baseado nos principais padrões de métricas para rastreabilidade de requisitos de software, tendo como característica uma junção dos padrões definidos por HULL ET AL (2002) e COSTELLO; LIU (1995) de forma que ambos se concluem como mostra a figura 14.

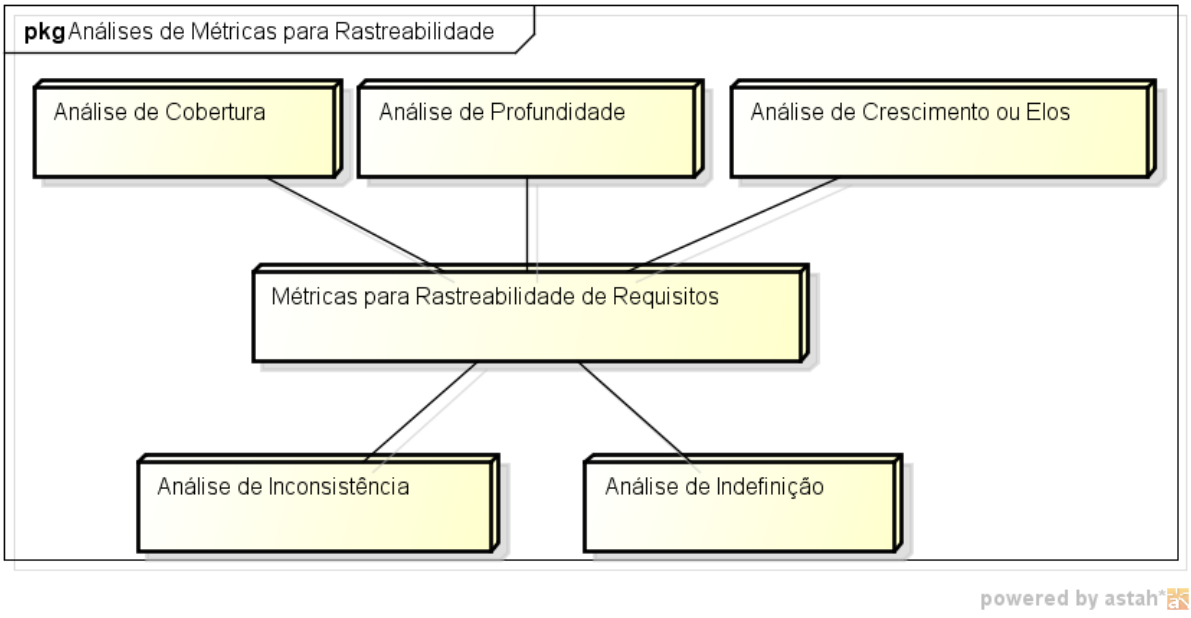

Figura 14 – Modelo de métricas para rastreabilidade de requisitos.

Como visto na figura 14, o modelo proposto neste trabalho é composto de cinco análises, onde cada uma das análises trata da particularidade de um determinado padrão de métrica.

A análise de cobertura é baseada na junção dos padrões de largura e cobertura de próximo nível (COV), definidos por HULL ET AL (2002) e COSTELLO; LIU (1995) respectivamente. Esta análise implica no quão abrangente está a

rastreabilidade entre os requisitos da especificação. Esta analise pode ser trabalhada quantitativamente numa relação entre requisitos rastreados para com os requisitos da especificação, por exemplo, num resultado desta análise poderia definir que 80 requisitos de um total de 100 requisitos da especificação foram ou estão rastreados.

A análise de profundidade trabalha baseada também nos padrões de Profundidade, definido por HULL ET AL (2002), com o padrão de Profundidade plena e alta cobertura (DHCOV), definido por COSTELLO; LIU (1995). Esta análise trabalha na quantidade de requisitos rastreados para cima ou para baixo.

Baseada também na junção de dois padrões de métricas, Crescimento e Estatística de Vinculação, respectivamente propostos por HULL ET AL (2002) e COSTELLO; LIU (1995), a análise de Crescimento ou Elos trata da avaliação do crescimento dos elos ou ligações entre cada requisito da especificação, de modo quantitativo para cima e para baixo.

Já a Analise de Inconsistência é baseada unicamente no padrão de Rastreabilidade Inconsistente (ITM) definido por COSTELLO; LIU (1995). Basicamente trabalha na identificação de elos inconsistentes entre os requisitos da especificação.

Por fim, a Análise de Indefinição trata da identificação dos requisitos que não possuem elos entre nenhum dos requisitos da especificação. Esta análise é baseada no padrão de Rastreabilidade Indefinida (UTM), definido por COSTELLO; LIU (1995).

De uma maneira geral, o modelo proposto faz uso dos elos definidos pela rastreabilidade para definir a mensuração entre os relacionamentos dos requisitos. Todas as análises definidas neste trabalho determinam e apresentam indicadores que proporcionam um maior entendimento para todos os envolvidos no projeto, e a melhoria da qualidade na rastreabilidade dos requisitos.

3. JUSTIFICATIVAS

Uma das primeiras atividades a serem realizadas no inicio de um projeto de software é a engenharia de requisitos, onde todas as necessidades do sistema se transformarão em requisitos para posteriormente satisfazer o propósito para qual o software será criado.

Primordial para o sucesso nessa fase é a rastreabilidade dos requisitos levantados, ou seja, com todos os requisitos rastreados possivelmente a chance de sucesso será grande, visto que toda a equipe de desenvolvimento saberá de todos os possíveis impactos e mudanças no software, tornando em longo prazo um software passivo a fáceis mudanças caso necessário.

Para que a garantia da qualidade dos requisitos rastreados seja segura, esse trabalho define um modelo de métricas para rastreabilidade de requisitos.

Desta forma será implementado esse modelo, na qual irá assegurar que todo o processo de rastreabilidade de requisitos seja mensurado. Com isso possibilitará indicadores, relatórios, gráficos e outras ferramentas que vise à melhoria nesta atividade da engenharia de requisitos.

Como descrito anteriormente, toda a abordagem deste modelo de métricas para rastreabilidade de requisitos de software é baseada nos elos que interligam os artefatos em toda a cadeia rastreada.

Além do modelo de métricas para rastreabilidade de requisitos proposto neste trabalho, poderá também ser usadas adaptações de outros tipos de métricas, como por exemplo, métricas de processo e produto, a fim de melhorar o modelo de medição para esse fim.

4. TECNOLOGIAS UTILIZADAS

Neste capítulo será descrito todas as tecnologias e ferramentas que serão utilizadas para realização deste trabalho, desde as fases iniciais de concepção até as fases finais.

Para a modelagem de dados UML será utilizada a IDE *JUDE*, em sua versão gratuita essa ferramenta é costumeira no ambiente universitário durante todo o período de graduação. Desenvolvida em Java, sua interface é rápida, descomplicada e intuitiva, também é possível com ela exportar no formato Java, HTML ou em forma de imagem: PNG ou JPEG.

Será usado a ferramenta *DBDesigner 4* para modelagem do banco de dados, a escolha dessa ferramenta foi pelo fato de ser um software livre sob a licença GPL, oferecer suporte ao banco de dados que será trabalhado, o PostgreSQL e sua interface ser bem elaborada e de uso simplório.

O ambiente de desenvolvimento integrado (IDE) será a ferramenta *NetBeans 6.x*, total ambiente Java, esta ferramenta é gratuita e de código aberto a IDE é multiplataforma além de fornecer inúmeras outras ferramentas alem de desenvolvimento.

Para o desenvolvimento da aplicação Web foi escolhida a tecnologia Java JSP (*JavaServer Pages*), os motivos que levaram a escolha do JSP foi pela afinidade com a linguagem de programação Java, na qual o JSP foi baseado, e por ser uma tecnologia aberta.

Juntamente com o desenvolvimento JSP, será utilizada a API Java Servlet, na qual trabalhou basicamente como uma classe Java. Dinamicamente os Servlets irão processar as requisições e respostas gerando conteúdo em HTML onde os mesmos irão interagir com os clientes.

Será utilizada também a linguagem de marcação HTML para renderização das páginas web.

O sistema também fará uso da técnica avançada de manipulação de javascript e XML, o AJAX.

Para o mapeamento objeto/relacional será utilizado o Hibernate. De uma maneira objetiva, o funcionamento do Hibernate acontece da seguinte forma: todos os objetos do banco de dados são transformados em um grafo de objetos definido pelo desenvolvedor em suas classes internas.

Usando o Hibernate será possível acelerar a velocidade no desenvolvimento e proporcionar facilidades no mapeamento dos atributos entre a base dados e a aplicação, mediante o uso de arquivos *XML*.

Para a comunicação com o banco de dados relacional PostgreSQL, será utilizado o conjunto de classes e interfaces do Java, o JDBC.

Para a geração e manipulação de gráficos será utilizada a API gratuita do Java, o *JFreeChart*, consistente e bem estruturada ela visa facilitar a construção de gráficos matemáticos, com ela é possível a visualização em 2D e 3D, e também exporta em diversos formatos como imagens e PDF.

Em conjunto com a API JFreeChart será utilizada uma outra biblioteca gratuita para criação de gráficos, o *CeWolf*, um framework para desenvolvimento de gráficos jsp/servlet utilizando o padrão MVC. Baseado totalmente em JFreeChart ele aproveita todos os mecanismos de desenho dos gráficos do JFreeChart porém com um design e layout mais aprimorado.

Para geração de relatórios será utilizado o framework open-source Java *JasperReports*. O *JasperReports p*ermite dinamicamente a geração de relatórios em diversos formatos como: PDF, HTML, XLS, CSV e XML.

5. METODOLOGIA

A escolha de um modelo de processo irá definir um conjunto de atividades, tarefas, marcos e produtos que serão necessários para um bom projeto arquitetural. Proporcionando qualidade e definindo estratégias durante todo o processo de desenvolvimento deste trabalho.

Os modelos processo foram originalmente propostos para colocar ordem no caos do desenvolvimento de software. A história tem indicado que esses modelos convencionais têm trazido certa dose de estrutura útil para o trabalho de engenharia de software. (PRESSMAN, 2005).

Durante o desenvolvimento deste trabalho, será utilizado o RUP (*Rational Unified Process*). Criado pela empresa Rational, que posteriormente foi adquirida pela IBM, a escolha do processo RUP foi feita devido a ele ser um processo adaptativo sendo possível dimensionar corretamente o processo para o projeto em questão, não sendo necessário utilizar todos os artefatos existentes. A afinidade e a prática com o RUP um dos quesitos que também influenciaram na decisão.

Seu fluxo de processo é interativo e incremental dando sensação de evolução constante, o que é essencial no desenvolvimento de software. Dentre os benefícios desta abordagem, tem também o *Feedback*[1], o que é importantíssimo, podendo fazer correções no projeto a tempo.

Para este trabalho o RUP será adaptado para as seguintes atividades obrigatórias:

- Levantamento de requisitos;
- Análise de sistema;
- Projeto;
- Desenvolvimento;
- Testes;

Essas atividades se concentraram nas quatro fases que regem o RUP, como mostra a figura 7.

[1] *Feedback:* em português, retorno de informação, é o procedimento que consiste no provimento de informação à uma pessoa sobre o desempenho, conduta, eventualidade ou ação executada por esta, objetivando orientar, reorientar e/ou estimular uma ou mais ações de melhoria, sobre as ações futuras ou executadas anteriormente (Babylon Dictionary, 2011).

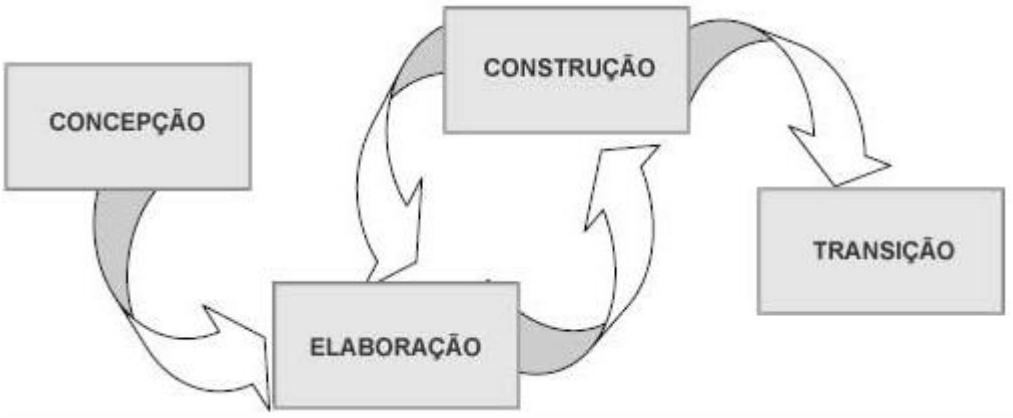

Figura 15 – Fases do RUP.

Fonte: IBM (2011)

A primeira atividade é o levantamento de requisitos, em alguns casos, uma conversa casual onde é realizada uma série de questões sobre o software proposto pelo cliente. Esta fase serve para compreender: o domínio da aplicação, o contexto do negócio, o problema a ser resolvido e as necessidades dos usuários.

As atividades para levantamento de requisitos de software pode ser divida em cinco áreas de esforço:

- Reconhecimento do problema;
- Avaliação e síntese;
- Modelagem;
- Especificação;
- Revisão;

Todo o escopo do software é definido no reconhecimento do problema. Já na avaliação e síntese estabelece as restrições do sistema (escopo negativo). Os diagramas e o documento de especificação de requisitos serão criados na modelagem. E por fim, a revisão de todas as áreas da atividade de levantamento de requisitos, se necessário, podendo nela também ocorrer alterações.

A segunda atividade, análise de sistema, é a atividade onde será construída toda a diagramação do sistema utilizando a notação UML (Unified Modeling Language). Com descrito anteriormente, será utilizada a ferramenta JUDE para a elaboração dos diagramas da UML. Nesta fase serão realizados os seguintes

diagramas: diagrama de Caso de Uso, diagrama de Seqüencia e diagrama de Classe.

A etapa de projeto é a terceira atividade a ser realizada, nela será possível definir as estruturas de dados, baseados nos indicadores colhidos nas atividades anteriores.

A quarta atividade será o desenvolvimento, seguido por testes unitários. Como descrito anteriormente o ambiente de desenvolvimento será a IDE NetBeans sob a plataforma JEE com a linguagem de programação Java, mais detalhes na seção 4.

Ao final dessas atividades é realizada a bateria de testes, a última atividade antes da entrega e implantação.

Teste é um conjunto de atividades que podem ser planejadas antecipadamente e conduzidas sistematicamente. Pode também ser definido um roteiro, onde um conjunto de passos e técnicas será descrito para a realização do teste (PRESSMAN, 1995).

Com isso, será utilizada uma estratégia de testes, que fornecerá um roteiro contendo os passos para condução do teste.

5.1. ARQUITETURA

A complexidade em se projetar e desenvolver sistemas complexos e de grande porte, fez com que engenheiros de software fizessem uso de disciplinas, a fim de obter resultados de baixo custo e maior qualidade, foi criada então a arquitetura de software.

A estrutura dos componentes de um programa/sistema, seus inter-relacionamentos, princípios e diretrizes ao longo do projeto, são guiadas pela arquitetura de software (GARLAN, 1995).

A idéia da arquitetura de software é que, um software complexo possa ser descrito através de subsistemas ou componentes sendo esses posteriormente inter-relacionados, facilitando e padronizando o processo de desenvolvimento de software.

Para este trabalho será usada a arquitetura MVC (*Model-View-Controller)*, como o próprio nome diz, o sistema é dividido em três perspectivas, ou seja, os componentes: Modelo (*Model*), Visão (*View*) e Controle (*Controller*).

O componente de modelo define o núcleo e toda a regra de negócio do sistema, ou seja, neste trabalho será onde ficarão todas as classes e métodos.

Já o componente de visão define a saída de dados e apresentação ao usuário, neste trabalho a camada de visão terá todas as páginas de listagem JSP e HTML.

A camada de controle é encarregada de tratar as entradas e comandos de usuários e gerenciá-la entre os componentes de visão e modelo, neste trabalho a camada de controle terá todas as páginas JSP contendo entrada de dados. Uma ilustração da arquitetura MVC é mostrada na figura 8.

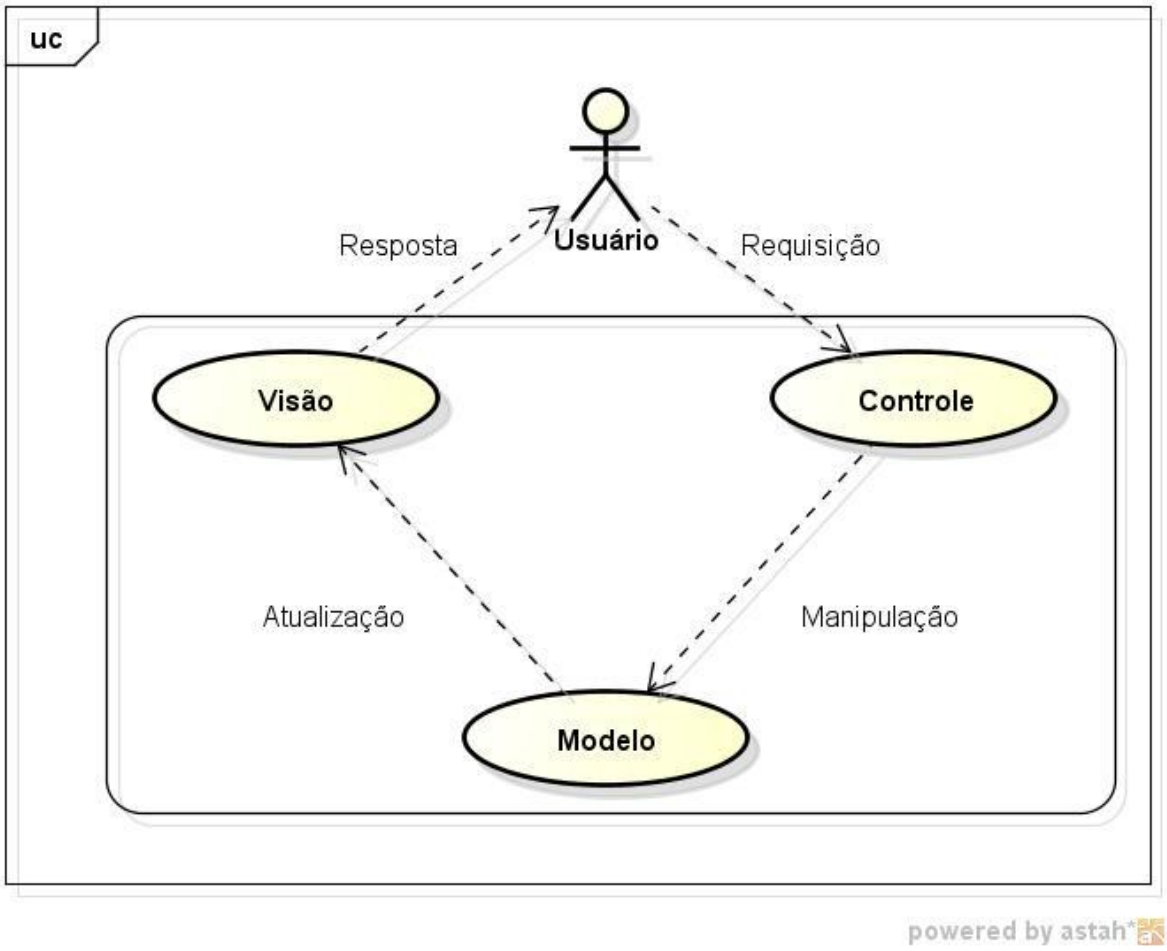

Figura 16 – Arquitetura MVC.

6. MÉTODO DE PESQUISA

Serão realizados estudos experimentais, em forma de estudo de caso, com o objetivo de avaliar a abrangência da plataforma e dos padrões de métricas estabelecidos. Tais estudos serão realizados utilizando-se como referência o modelo GQM (Goal Question Metrics) (BASILI, 1994).

A idéia básica do GQM é derivar métricas de software a partir de perguntas e objetivos, o paradigma do GQM foi proposto com uma abordagem orientada os objetivos e metas, sendo composto por três níveis:

- Conceitual;
- Operacional;
- Quantitativo;

O nível conceitual (*Goal*) define as metas do experimento, fazendo a seguinte pergunta: “Quais são as metas ou objetivos?” facilita na formação desse nível.

Já o operacional (*Question*) define as questões a serem usadas para caracterizar o que vai ser avaliado: “*Quais questões se deseja responder?*”.

E por fim o nível quantitativo (*Metrics*) definindo as métricas que irão compor o grupo de dados associados às questões com o fim de responder as elas de forma quantitativa, respondendo a pergunta “*Quais métricas poderão ajudar?*” auxilia na composição das métricas.

A Figura 9 apresenta o modelo GQM (GENVIGIR, E. C., 2010).

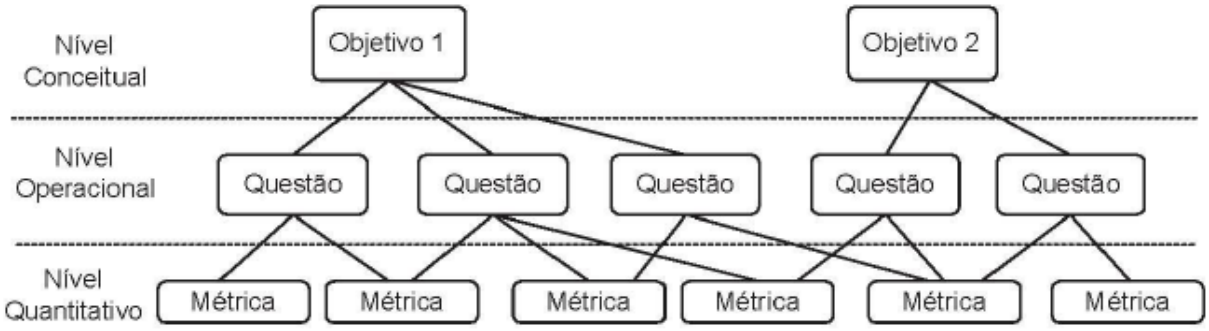

Figura 17 – Níveis conceituais do GQM.

Fonte: GENVIGIR, E. C., 2010 (Adaptado de BASILI et al. (1994))

O processo de experimentação a ser aplicado será um refinamento do modelo GQM apresentado por SOLINGEN e BERGHOUT (1999). Esse processo é composto pelas atividades de Definição, Planejamento, Interpretação e de Coleta de Dados, como mostra a Figura 10.

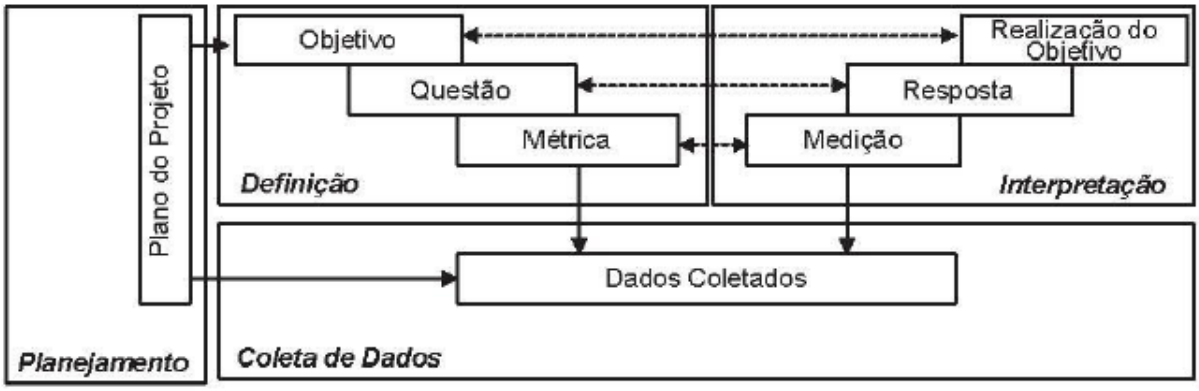

Figura 18 – Processo de experimentação.

Fonte: GENVIGIR, E. C., 2010 (Adaptado de SOLIGEN e BERGHOUT (1999))

A fase de Definição inclui os três níveis conceituais do GQM em que são descritos os objetivos, as questões e as métricas. O principal resultado desta atividade é fornecer o conhecimento e a direção geral para a realização do experimento.

O Planejamento determina a fundamentação do experimento. Definem-se o contexto, as hipóteses, os instrumentos, identificam-se as variáveis e a selecionam-se os métodos de análise.

A Coleta de Dados define como os dados serão coletados e armazenados para uso posterior na fase de Interpretação. E, por fim, a Interpretação considera 19

as formas de apresentação dos dados coletados e a elaboração dos resultados obtidos através da análise estatística (GENVIGIR, E. C., 2010).

7. CRONOGRAMA

As atividades do cronograma terão inicio na primeira semana de Junho, sendo previsto o período de sete meses.

Tabela 1 - Cronograma

Atividades	JUN	JUL	AGO	SET	OUT	NOV	DEZ
Pesquisa Bibliográfica							
Redigir Monografia							
Análise de Requisitos							
Análise e Projeto							
Implementação							
Testes							

REFERÊNCIAS

COSTELLO, R.; LIU, D. **Metrics for requirements engineering**. Journal of Systems and Software, 1995.

FENTON, N, PFLEEGER, S. **Software Metrics: A Rigorous and Practical Approach**. 2ed. Boston, MA: PWS Publishing, 1998.

GENVIGIR, E.C. **Um modelo para rastreabilidade de requisitos de software baseado em generalização de elos e atributos.** 2009. 203p. Dissertação (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2009.

HULL, E.; JAKSON, K.; DICK, J. **Requirements engineering**. London: Spring Verlag, 2002.

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS – IEEE. **Standard glossary of software engineering terminology**. New York, NY, USA, 1990.

MENDES, A. **Arquitetura de Software: Desenvolvimento orientado para arquitetura**. 1ed. Rio de Janeira, RJ: Editora Campus, 2002.

PETERS, JAMES F. **Engenharia de Software**. 1ed. Rio de Janeiro, RJ: Editora Campus, 2001.

PRESSMAN, R.S. **Engenharia de Software**. 1ed. Rio de Janeiro, RJ: Makron Books, 1995.

_____. 3ed. Rio de Janeiro, RJ: Makron Books, 1995.

SOMMERVILLE, I. **Engenharia de Software**. 6ed. Nacional: Addison Wesley Editor, 2003.

_____. 8ed. Nacional: Addison Wesley Editor, 2007.

VAZQUEZ, C.E; SIMÕES, G.S; ALBERT, R.M. **Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de Software**. São Paulo, SP: Érica, 2008.